Programozási

I.

1985-1994.
Általános iskolások országos versenye 1991-94

Nemes Tihamér
Nemzetközi Informatikai Tanulmányi Verseny 1985-94

Szerkesztette: Zsakó László

Az általános iskolások országos számítástechnikai versenyének feladatsorait

Zsakó László (ELTE)

vezetésével az ELTE munkatársai állították össze:

Horváth László ***Pap Gáborné*** ***Temesvári Tibor*** ***Szlávi Péter***

A Nemes Tihamér OKSzTV feladatsorait

Ada-Winter Péter
Hanák Péter (BME)
Zsakó László (ELTE)

vezetésével az NJSzT Országos Versenybizottsága mindenkori tagjai dolgozták ki. Közülük sok éven át

Frigó József (BME) ***Gulyás László (ELTE)*** ***Horváth László (ELTE)***
Hunyady István (BME) ***Jánosi Tibor (BME & EMT)*** ***Kovács László (BME)***
Mohay Tamás (BME, Oracle Hungary) ***Orbán Anna (Államigazgatási Főiskola)***
Puskás Zsolt (BME) ***Reé Balázs (BME)*** ***Szabadhegyi Csaba (ELTE)***
Székely Jenő (ELTE) ***Szlávi Péter (ELTE)*** ***Tagányi György (BME)***
Temesvári Tibor (ELTE) ***Török Turul (ELTE)***

vett részt a közös munkában. Nem látó (vak) tanulók számára a feladatszövegeket Braile-írással

Helfenbein Henrik (ELTE)

állította elő.

A kiadvány a Neumann János Számítógép-tudományi Társaság által kiadott *Programozási versenyfeladatok tára 1* című kötet alapján készült.

ISBN 978-963-489-036-2

A kiadvány „A felsőoktatásba bekerülést elősegítő készségfejlesztő és kommunikációs programok megvalósítása, valamint az MTMI szakok népszerűsítése a felsőoktatásban” (EFOP-3.4.4-16-2017-006) című pályázat keretében készült 2018-ban.

Eötvös Loránd Tudományegyetem
Informatikai Kar

SZÉCHENYI 2020
MAGYARORSZÁG KORMÁNYA
Európai Unió
Európai Szociális Alap
BEFEKTETÉS A JÖVŐBE

Előszó

Magyarországon több kezdeti próbálkozás után 1985-ben szervezte meg az első országos középiskolai programozási versenyt Nemes Tihamér Nemzetközi Informatikai Tanulmányi Verseny (NT NITV) néven a Neumann János Számítógép-tudományi Társaság (NJSZT)[1]. A 9 évig kétfordulós versenyen 1989-ig egy korcsoportban, 1990-től külön korcsoportban indulhatnak a 14-15, ill. a 16-19 éves tanulók (azaz a középiskolák I.-II., ill. III.-V. osztályos diákjai). Az 1993/94-es tanévtől a versenyt három fordulóban rendezzük (az iskolai forduló és az országos döntő közé iktattunk egy regionális-megyei fordulót). A verseny első tíz helyezettje – a többi országos középiskolai tanulmányi versenyhez hasonlóan – érettségi, illetve felvételi kedvezményben részesül, és közülük válogatjuk ki az 1989 óta ugyancsak évente megrendezett Nemzetközi Informatikai Diákolimpia magyar résztvevőit is.

A Nemes Tihamér verseny, szerénytelenség nélkül megállapíthatjuk, népszerűvé vált a diákok körében: az utóbbi években 260-300 magyarországi középiskola kb. 2500-2500 I.-II., illetve III.-V. osztályos diákja vett részt a verseny első, iskolai fordulójában. Közülük kb. 600 diák jutott tovább a második fordulóba, majd kb. 180 a harmadikba. Említést érdemel, hogy a környező országokban élő, magyar anyanyelvű vagy magyarul jól beszélő diákok közül is egyre többen érdeklődnek a verseny iránt, illetve vesznek részt a versenyen; az Erdélyi Magyar Műszaki Tudományos Társaság szervezésében évente kb. 500 diák indul.

1991 óta különböző szervezési formákban az általános iskolások is résztvehetnek országos programozási jellegű versenyen. Két évig a minden megyére kiterjedő versenyt 3, illetve 4 fordulóban rendezték meg (iskolai, területi, megyei, országos), majd a következő két évben a verseny csak néhány megyében folytatódott, s az országos döntő elmaradt. Sokak igényét elégítettük ki azzal, hogy a Nemes Tihamér NITV-t e korosztállyal bővítettük, a megrendezésre jelentkező megyéket (regionális versenybizottságokat) feladatokkal láttuk el, s megszerveztük az országos döntőt is.

Az NJSZT Országos Versenybizottsága sokak kívánságát teljesíti most azzal, hogy önálló kötetekben megjelenteti az eddigi versenyfeladatokat. Reméljük, hogy ezzel is segíteni tudjuk a leendő versenyzőket a felkészülésben, a tanáraikat pedig a számítástechnikai foglalkozások és szakkörök megtartásában.

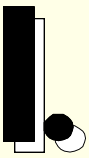
Ebben a kötetben az 1985-1994 közötti Nemes Tihamér versenyek, valamint a Nemes Tihamér verseny előtti országos általános iskolás versenyek programozási feladatait ismertetjük. Minden egyes feladat után a javító tanároknak szóló megoldási és értékelési útmutatót is közöljük. A versenyekre készülő diákoknak természetesen azt javasoljuk, hogy mielőtt a közölt megoldást és értékelést elolvasnák, saját maguk, önállóan próbálják megoldani a kitűzött feladatot. Nem törekedtünk szöveghűségre: ahol az eredeti megfogalmazást pontatlannak vagy hibásnak találtuk, módosítottunk a szövegen. Reméljük, hogy ezzel sikerült csökkenteni a hibás feladatok számát, de nem garantálhatjuk, hogy már nincsenek rossz vagy félreérthető megfogalmazások és megoldások.

[1] Régi nevén: Nemes Tihamér Országos Középiskolai Számítástechnikai Tanulmányi Verseny (NT OKSzTV)

Tartalom

I.

I. Versenyfeladatok



1. Általános iskolások országos számítástechnikai versenye

1991. Első forduló

1. feladat: (11 pont)

Egy trafikban kétféle tollat lehet vásárolni, az egyik A, a másik pedig B forintba kerül. Készíts programot, amely megmondja, hogy ha C forintért vásároltunk tollat, akkor mennyit vehettünk az első, és mennyit a második fajtából! Add meg azt is, hogy egyértelműen maghatározható-e az eredmény, s ha nem, akkor az összes megoldást megadja!

2. feladat: (6 pont)

1901. január 1. hétfői nap volt. Írj programot, amely ez alapján meghatározza, hogy egy tetszőleges év (1901 után) január 1. milyen napra esik!

3. feladat: (12 pont)

Készíts programot, amely beolvas egy egész számot 1 és 99 között, majd kiírja a számot betűkkel! (Például, ha a beolvasott szám 36, akkor azt írja ki, hogy "harminchat".)

4. feladat: (8 pont)

Feladatunk az első 20 négyzetszám kiírása, amelyre háromféle megoldást készítettünk. Melyik megoldás (megoldások) helyes, miért?

A:
```
10 FOR I=1 TO 20
20  PRINT I*I
30 NEXT I
```

B:
```
10 FOR K=1 TO 20
20  K=K*K: PRINT K
30 NEXT K
```

C:
```
10 K=0
20 FOR J=0 TO 19
30  K=K+2*J+1
40  PRINT K
50 NEXT K
```

5. feladat: (18 pont)

Ismerjük egy labdarúgó bajnokság eredményeit mérkőzésenként (pl. 1. csapat, 3. csapat: 3:0). Készíts programot, amely táblázatot készít a bajnokságról a következő formában:

győzelmek száma	döntetlenek száma	vereségek száma	lőtt gólok száma	kapott gólok száma	pontszám

Győzelemért 2, döntetlenért 1, vereségért pedig 0 pont jár. A táblázatot nem kell rendezni!

Elérhető összpontszám: 55 pont

1991. Második forduló

1. feladat: (7 pont)

Készíts programot, amely kiírja a háromjegyű Armstrong számokat! Ezek olyan számok, amelyek számjegyei köbeinek (az A köbe: A*A*A) összege megegyezik magával a számmal!

2. feladat: (11 pont)

A LOGO programnyelv három utasításának a jelentése a következő:

`REPEAT darabszám [utasítások]`	a zárójelbe tett utasításokat darabszámszor megismétli.
`FORWARD lépésszám`	lépésszám hosszan vonalat húz az aktuális helyzetből kiindulva az aktuális irányban.
`LEFT szög`	az aktuális irányt balra, fokokban a megadott szöggel megváltoztatja.

Itt egy program ezen a nyelven:

```
REPEAT A [FORWARD B LEFT C]
```

Milyen síkidomot rajzol, ha az A,B,C változóknak rendre a következő értékeket adjuk?

1. A=3, B=100, C=120
2. A=4, B=100, C=90
3. A=360, B=1, C=1

3. feladat: (13 pont)

Egy lánctalpas jármű lánctalpait be, illetve kikapcsolhatjuk. A LÁNC(I,1) utasítás az I. lánctalpat bekapcsolja, a LÁNC(I,0) pedig kikapcsolja. A baloldali lánctalp sorszáma 1, a jobboldalié pedig 2. A FÉNY változó értéke akkor -1, ha a jármű balról érzékel több fényt, akkor 1, ha jobbról, illetve 0, ha balról és jobbról azonos erősségű fény jön. Írj olyan programot, amely a járművet a fényforrás felé vezeti!

4. feladat: (12 pont)

Egy H hosszúságú rudat bevonunk N réteggel egyenletesen (a végein is ugyanolyan vastagon, mint az oldalán). D(I) jelenti az I. bevonás utáni átmérőt, D(0) a kezdeti átmérő. Keress hibákat a következő programrészletben, amely az egyes felhasznált anyagok térfogatát határozza meg!

```
100 V=D(0)*D(0)/4*PI*H
110 FOR I=1 TO N
120   U=D(I)*D(I)/4*PI*H
130   PRINT V-U
140   V=V-U
150   H=H+D(I)
160 NEXT I
```

5. feladat: (12 pont)

Írd át gyorsabbra (kevesebb, illetve egyszerűbb műveletet végzőre) a következő programrészletet!

```
100 S=0
110 FOR I=1 TO N
120   S=S+A(I)/N
130 NEXT I
140 P=0
150 FOR I=1 TO N
160   P=P+(A(I)-S)^2/N
170 NEXT I
```

6. feladat: (15 pont)

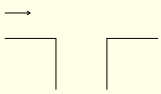
Készíts programot, amely elvégzi a Bongó-sorsolást, majd egy játékos számai alapján megmondja, hogy mennyit nyert az illető! A Bongón 7 számjegyet sorsolnak véletlenszerűen, s a nyereség attól függ, hogy a tippelt számjegyek közül hátulról visszafelé mennyi egyezik meg folyamatosan a sorsolt szám számjegyeivel! Ha az utolsó kettő megegyezik, akkor a nyeremény 70 forint, ha az utolsó három, akkor 700, ha az utolsó négy, akkor 7000, ...

Elérhető összpontszám: 70 pont

1991. Harmadik forduló

Feladat: (50 pont)

Készíts BASIC programot, amely egy útkereszteződés forgalmát szimulálja! Az autók a két útra véletlenszerűen érkezzenek, az úton a végéig haladjanak (ott eltűnnek), a kereszteződés forgalmát jelzőlámpák irányítsák! A kereszteződés mindkét útja egyirányú, irányt változtatni nem lehet.



Elérhető összpontszám: 50 pont

1991. Negyedik forduló

1. feladat: (45 pont)

Írj számrendszerek közötti átalakítást végző programot, amelyben a felhasználó a következők közül választhat (pozitív egész számok átalakítására):

- 10-es számrendszerbeli szám kiírása 2-es számrendszerben,
- 2-es számrendszerbeli szám kiírása 10-es számrendszerben,
- 16-os számrendszerbeli szám kiírása 2-es számrendszerben,
- 2-es számrendszerbeli szám kiírása 16-os számrendszerben.

2. feladat: (25 pont)

Készíts programot, amely beolvassa egy ember nevét, majd kiírja a monogramját! (Vigyázz a különleges esetekre, pl.: Kiss Szabó Zsolt monogramja: K. Sz. Zs.) A név részei, illetve a monogram betűi között egy-egy szóköz van.

3. feladat: (25 pont)

Készíts rajzoló programot! A programot billentyűzetről irányítjuk. Induláskor a kurzor álljon a képernyő közepén, a J(obbra), B(alra), L(e), F(el) billentyűk egyszeri lenyomása jelentsen a megfelelő irányban egy egységnyi elmozdulást! Így ezen billentyűk többszöri lenyomásával tetszőleges alakzatok rajzolhatók! Ha a rajzolás során kilépnénk a képernyőről, akkor lépjünk be az ellenkező oldalon!

4. feladat: (45 pont)

Készíts programot, amely megadja, hogy egy évben hányadikára esik húsvét!

A húsvét meghatározásának szabálya:

Húsvét a tavaszi napéjegyenlőség (március 21.) utáni első holdtölte utáni első vasárnap, illetve hétfő. A holdtöltek egymástól 29 és fél napra vannak.

A program olvassa be, hogy az év első napja a hét mely napjára esik, az évszámot, illetve, hogy hányadikán van az első holdtölte, s aznap délelőtt vagy délután, s ezek alapján írja ki húsvétvasárnap, illetve -hétfő dátumát! Például 1991. január 1. kedd volt, az első holdtölte: január 30.-án délelőtt volt (a következők: 02.28. délután, 03.30. délelőtt), tehát húsvét az ezutáni első vasárnap, illetve hétfő: 03.31, 04.01.

Elérhető összpontszám: 180 pont

1992. Első forduló

1. feladat: (15 pont)

Készíts programot, amely az A(N) és a B(N) vektorokban számjegyenként a helyiértékük növekvő sorrendjében tárolt N+1-jegyű egész számokat összeadja, az eredmény a C(N+1)-ben keletkezik. (Például N=1, A(0)=5, A(1)=6, B(0)=3, B(1)=7 esetén az eredmény C(0)=8, C(1)=3, C(2)=1, mert 65+73=138.)

2. feladat: (28 pont)

Készíts programot, amely a grafikus képen egy kurzort képes mozgatni a kurzormozgató billentyűk (←,↑,→,↓) segítségével! Az aktuális pozíción megjelenítendő kurzor ábráját az 1,2,3,4 billentyű bármikori lenyomásával lehessen kiválasztani! Az egyes számokhoz tartozó kurzorképek:

`1: ◊;` `2: ↖;` `3: ×;` `4: +`

3. feladat: (22 pont)

Egy sakktáblára (8 sora és 8 oszlopa van) elhelyezünk egy bástyát, egy futót és egy huszárt. Készíts programot, amely megadja, hogy üti-e valamelyik bábu a másikat!

A bástya a vele azonos sorban vagy oszlopban levőket üti, a futó a vele azonos átlóban levőket, a huszár pedig a tőle bármely irányban L-alakban levőket, pl.:

H		
		*

4. feladat: (21 pont)

Mit rajzolnak a következő, LOGO nyelvű programok? (Induláskor a kurzor a képernyő közepén áll és felfelé néz.)

`LEFT f, RIGHT f`	balra, illetve jobbrafordulás helyben f fokkal,
`FORWARD h`	előrelépés h egységgel, közben rajzol,
`BACK h`	hátralépés h egységgel, közben rajzol,
`REPEAT db [utasítások]`	utasítások db-szeri megismétlése)

```
A. RIGHT 150
   REPEAT 3 [FORWARD 10 RIGHT 120]
   FORWARD 20 BACK 20 RIGHT 60 FORWARD 20

B. REPEAT 2 [RIGHT 90 FORWARD 10 BACK 10 LEFT 90 FORWARD 10]
   RIGHT 90 FORWARD 10

C. REPEAT 2 [LEFT 90 FORWARD 5 LEFT 90 FORWARD 10 BACK 20
             FORWARD 10 RIGHT 90 BACK 5 LEFT 90]
```

5. feladat: (14 pont)

Az A(6) vektor az óra aktuális állását tartalmazza (pl. A(1)=6, A(2)=5, A(3)=4, A(4)=3, A(5)=2, A(6)=1 azt jelenti, hogy most 12 óra, 34 perc, 56 másodperc a pontos idő). Készítettünk egy programrészletet, amely az így tárolt pontos időt 1 másodperccel megnöveli, azonban a program nem mindig működik helyesen. Írd le milyen hibajelenségeket tapasztalhatsz használatakor! (Csak tartalmi hibákat keress!)

```
1000 I=1
1010 A(I)=A(I)+1
1020 IF I=1 OR I=3 OR I=5 THEN 1050
1030 IF A(I)<=6 THEN 1100
1040 I=I+1: GOTO 1010
1050 IF A(I)<=9 THEN 1100
1060 A(I)=0: I=I+1: GOTO 1010
1100 ...
```

Elérhető összpontszám: 100 pont

1992. Második forduló

1. feladat: (24 pont)

Készíts programot, amely az A(N) és a B(N) vektorokban számjegyenként a helyiértékük növekvő sorrendjében tárolt N+1-jegyű egész számokat összeszorozza, az eredmény a C(2*N+1)-ben keletkezik. (Például N=1, A(0)=5, A(1)=6, B(0)=3, B(1)=7 esetén az eredmény C(0)=5, C(1)=4, C(2)=7, C(3)=4, mert 65*73=4745.)

2. feladat: (18 pont)

Mit rajzolnak a következő, LOGO nyelvű programok? (Induláskor a kurzor a képernyő közepén áll és felfelé néz.)

`LEFT f, RIGHT f`	balra, illetve jobbrafordulás helyben f fokkal,
`FORWARD h`	előrelépés h egységgel, közben rajzol,
`PENUP, PENDOWN`	toll felemelése, leengedése a papírra
`REPEAT db [utasítások]`	utasítások db-szeri megismétlése)

```
A. LEFT 30
   REPEAT 3 [REPEAT 2 [FORWARD x RIGHT 60 FORWARD x
                       RIGHT 120] RIGHT 120]

B. REPEAT 4 [REPEAT 360 [FORWARD 1 RIGHT 1] PENUP RIGHT 90
             FORWARD 10 LEFT 90 PENDOWN]
```

3. feladat: (16 pont)

Készíts verselemző programot, amely egy beírt sor szótagjainak hangrendjét adja meg. Egy szótag magas hangrendű, ha a magánhangzója magas (e, é, i, í, ö, ő, ü, ű), illetve mély hangrendű, ha a magánhangzója mély (a, á, o, ó, u, ú).

4. feladat: (18 pont)

A következő programrészlet egy fizikai feladat megoldására szolgál, azonban a futása nagyon lassú. A feladat ismerete nélkül alakítsd át úgy, hogy gyorsabban fusson! Az átalakítás elve: a ciklus belsejében kevesebb műveletet végezzen, valamint a szorzásnál gyorsabb az összeadás, a hatványozásnál pedig a szorzás.

```
100 G=9.81
110 FOR T=0 TO N
120   V(T)=G*T
130   S(T)=1/2*G*T^2
140 NEXT T
```

5. feladat: (10 pont)

Egy számítógéppel irányított kisautót elhelyeztünk egy labirintusban, s a labirintusból való kijutáshoz egy programot készítettünk. A megoldás elve egy nagyon régi módszer, haladjunk úgy, hogy a bal kezünk végig a falat éri, így egy utat biztosan nem próbálunk kétszer.

```
Ciklus amíg nem ért ki
   Fordulj balra
   Ciklus amíg fal van előtted
      Fordulj jobbra
   Ciklus vége
   Menj előre 1 egységnyit
Ciklus vége
```

Ezzel a programmal elindítjuk az autót és egyes labirintusoknál azt tapasztaljuk, hogy mégsem jut ki belőlük. Milyen felépítésűek ezek a labirintusok?

6. feladat: (14 pont)

Az alábbi BASIC programrészlet az A$ változó alapján állít elő egy szöveget a B$-ban.

RIGHT$(T$, i)	a T$ szöveg jobb szélső i db karaktere
LEFT$(T$, i)	a T$ szöveg bal szélső i db karaktere
MID$(T$, k, i)	a T$ szöveg középső i db karaktere a k.-tól kezdve
LEN(T$)	a T$ szöveg karaktereinek száma
CHR$(i)	az ASCII kódtábla i. jele

```
1000 S=1: B$=LEFT$(A$,1): A$=A$+CHR$(1)
1010 IF B$=A$ THEN 1200
1020 FOR I=2 TO LEN(A$)
1030   IF RIGHT$(B$,1)=MID$(A$,I,1) THEN S=S+1: GOTO 1090
1040   IF S>2 THEN B$=B$+CHR$(0)+CHR$(S)
1050   IF S=2 THEN B$=B$+RIGHT$(B$,1)
1060   B$=B$+MID$(A$,I,1): S=1
1090 NEXT I
1100 ...
```

A. Mi lesz B$-ban az 1100-as sorban, ha A$ tartalma kezdetben

A1. "Általános iskola"

A2. "Fallal bekerített terület"

A3. "13333333-szor ismételd meg."

B. Fogalmazd meg tömören, mit csinál a fenti program!

Elérhető összpontszám: 100 pont

1992. Harmadik forduló

1. feladat: (20 pont)

Készíts programot, amely egy a képernyőn kijelölt tetszőleges háromszöget megrajzol, majd vízszintes vonalakkal besatíroz (a háromszög belsejében a képernyő minden második sorába egy vízszintes vonalat húz). A háromszög három pontjának koordinátáit be kell olvasni!

2. feladat: (40 pont)

Készíts programot, amely beolvas egy többsoros szöveget, a képernyőn a bal- és a jobbmargó oszlopsorszámát, majd kiírja a szöveget a képernyőn sorokra tagolva a bal- és a jobbmargó között úgy, hogy ha egy szó nem férne ki teljesen egy sorba, akkor azt már a következő sorban kezdi! A programnak lehessen adni margóhoz igazítás parancsot: balmargóhoz, jobbmargóhoz, középre, mindkét margóhoz igazítást.

3. feladat: (20 pont)

Készíts naptárprogramot, amely beolvassa, hogy 1901.01.01. milyen napra esett, majd egy tetszőleges ezutáni év tetszőleges hónapjára kiírja annak a hónapnak a naptárját (melyik napja a hét milyen napjára esik)! Felhasználhatod, hogy a szökőévek a 4-gyel osztható évek, de amelyek 100-zal is oszthatók, azok közül csak a 400-zal oszthatók.

Elérhető összpontszám: 80 pont

1993. Első forduló

1. feladat: (14 pont)

Az alábbi feladatok megoldására írj egy BASIC nyelvű kifejezést vagy egy ciklussal számold ki a kifejezés értékét!

A. Y= 2 tizedesjegyre kerekítése X-nek,

B. Y= X Ft-ból N év múlva mennyi lesz a kamatokkal együtt, ha egy évre K% a kamat,

C. Y= A és B közötti egész véletlenszám, ha RND(0) 0 és egy közötti valós véletlenszámot jelent, ami 0 lehet, de 1 nem,

D. Y= X számjegyeinek összege.

2. feladat: (20 pont)

Mit rajzol a következő, LOGO nyelvű program? (Induláskor a kurzor a képernyő közepén áll és felfelé néz.) Milyen geometriai alakzatokat ismersz fel? Rajzold le és írj mellé méreteket is!

`LEFT f, RIGHT f`	balra, illetve jobbrafordulás helyben f fokkal,
`PENUP`	ettől kezdve mozgás közben nem rajzol,
`PENDOWN`	ettől kezdve rajzol,
`FORWARD h`	előrelépés h egységgel,
`BACK h`	hátralépés h egységgel,
`REPEAT db [utasí-tások]`	utasítások db-szeri megismétlése

```
ELSO N M X Y   ← eljárásnév és paraméterei
  PENUP REPEAT M [MASODIK N X Y LEFT 90 FORWARD Y RIGHT 90]
END  ← eljárás vége

MASODIK N X Y
  REPEAT N [HARMADIK X Y FORWARD X] BACK N*X
END

HARMADIK X Y
  PENDOWN
  REPEAT 2 [FORWARD X RIGHT 30 FORWARD Y/2 RIGHT 120
            FORWARD Y LEFT 120 FORWARD Y/2 RIGHT 150]
  PENUP
END
```

3. feladat: (20 pont)

Egy programrészletet írtunk, amely képes racionális számok összeadására, illetve szorzására. Racionális számok mindig felírhatók két egész szám hányadosaként, de mivel ilyen számpár minden racionális számhoz végtelen sok van, ezek közül a legkisebb párt választjuk, azaz a számlálót és a nevezőt egyszerűsítjük. Asz,An jelöli az egyik szám számlálóját, s nevezőjét, Bsz,Bn, illetve Csz,Cn pedig a másik kettőét. Ekkor C=A+B, illetve C=A*B így írható:

```
Összeadás(A,B,C):
  Csz= Asz*Bn+Bsz*An
  Cn = An*Bn
  X  = Lnko(Csz,Cn)
  Csz= Csz/X
  Cn = Cn/X
Eljárás vége.
```

```
Szorzás(A,B,C):
  Csz= Asz*Bsz
  Cn = An*Bn
  X  = Lnko(Csz,Cn)
  Csz= Csz/X
  Cn = Cn/X
Eljárás vége.
```

A számláló és a nevező egyszerűsítése ugyanis azt jelenti, hogy osszuk el őket a legnagyobb közös osztójukkal.

A két algoritmus legtöbbször helyes eredményt ad, de a részeredmények kiszámításakor időnként túlcsordulást tapasztalunk. Alakítsd át a két algoritmust úgy, hogy az egyszerűsítésekből, amit lehet, azt még a közbülső szorzások előtt végezzen el!

4. feladat: (20 pont)

Olyan programot írtunk, amellyel a képernyő egy pontjából kiindulva véletlenszerűen bolyongunk, azaz az aktuális pontból a nyolc szomszédja valamelyikére lépünk. A bolyongás addig tart, amíg a véletlenszerű lépkedés során vissza nem térünk az eredeti helyre. A megoldó BASIC programrészlet (RND(N) 1 és N közötti véletlenszámot jelent):

```
100 X=RND(100): Y=RND(100): A=X: B=Y
110 IF X=A AND Y=B THEN 200
120 R=RND(3)-2
130 S=RND(3-2)
140 IF R=0 OR S=0 THEN 120
150 LINE X,Y,X+R,Y+S: REM szakaszrajzolás (X,Y) és (X+R,Y+S)
között
160 X=A+R
170 Y=Y+S
180 GOTO 110
200 ...
```

A. Egészítsd ki a programot azzal, hogy a képernyőről ne tudjon kilépni!

B. A program több hibát tartalmaz, melyek ezek?

5. feladat: (26 pont)

4 játékos magyar kártyával játszik, a 32 lapot egyenlően elosztották. Az I. játékos kezében levő J. kártyalapot az A$(I,J) szöveg típusú tömbelem tartalmazza, pl. ilyen értékei lehetnek: "piros hetes", "zöld hetes", "zöld ász". Készíts olyan BASIC programrészletet (a beolvasást nem kell megírni), amely e tömb alapján megadja mindegyik játékosra, hogy melyik színből hány lapja van, valamint, hogy hány ász van nála!

Elérhető összpontszám: 100 pont

1993. Második forduló

1. feladat: (20 pont)

4 játékos magyar kártyával játszik, a 32 lapot egyenlően elosztották. Az I. játékos kezében levő J. kártyalapot az A$(I,J) szöveg típusú tömbelem tartalmazza, pl. ilyen értékei lehetnek: "piros hetes", "zöld kilences", "makk ász". A játékban az egyes lapok pontértéke a következő: hetes=7, nyolcas=8, kilences=9, tízes=10, alsó=2, felső=3, király=4, ász=11. Írj programrészletet, amely a négy játékos lapjai alapján kiszámolja a kezükben levő lapok pontértékét!

2. feladat: (30 pont)

A BKV új jegyrendszerre tér át a budapesti autóbuszokon. Ezentúl minden jegyen 2 lyukasztás lesz, de a lehetséges lyukasztásokat úgy akarják kiosztani, hogy ha az utas a jegyet fordítva teszi be a lyukasztógépbe, akkor is egyértelműen felismerhető legyen.

Ezért például a következő két lehetőségből csak az egyik fordulhat elő:

1	2	3
4	5	6
7	8	9

két "azonos értékű" lyukasztása

O	O	3
4	5	6
7	8	9

1	O	O
4	5	6
7	8	9

Készíts programrészletet, amely az összes jó lyukasztást megadja (kiírja, hogy mely két helyen kell a jegyet kilyukasztani)!

3. feladat: (25 pont)

A LOGO programnyelven írtunk két eljárást:

```
TO mitcsinal
  RIGHT 90                      - jobbra fordul 90 fokot
  IF NOT fal? THEN FORWARD 1    - ha nem fal van előtte, akkor elő re megy 1 lépést
                   mitcsinal      és végrehajtja a mitcsinal eljárást
  LEFT 90                       - balra fordul 90 fokot
  FORWARD 1                     - előre megy 1 lépést
END

TO mitkeres
  IF NOT fal? THEN FORWARD 1    - ha nem fal van előtte, akkor előre megy 1 lépést
                   mitkeres       és végrehajtja a mitkeres eljárást
               ELSE LEFT 90     - különben balra fordul 90 fokot és
                   mitcsinal      végrehajtja a mitcsinal eljárást
END
```

Mi a *mitkeres*, illetve a *mitcsinal* eljárások feladata?

4. feladat: (25 pont)

Egy asztalon egy piros, egy zöld és egy kék kockát helyeztünk el tetszőlegesen, akár egymás mellett, akár egymáson. Egy robotot szeretnénk utasítani, hogy tegye őket úgy egymásra, hogy alul legyen a piros, fölötte a zöld és legfelül a kék kocka! A robot a következő három utasítást tudja végrehajtani:

- Fogd meg a ... színű kockát!
- Tedd a ... színű kockára!
- Tedd az asztalra!

Megjegyzés: Ha a robot nem felül levő kockát fog meg, akkor a rajta levő leesik az asztalról, s a robot nem tudja végrehajtani a feladatot.

A kockák állapotát egy 3*3-as táblázat írja le, néhány lehetséges esete:

a kockák egymás mellett:

	1	2	3
1			
2			
3	piros	kék	zöld

a robot célja:

	1	2	3
1	kék		
2	zöld		
3	piros		

piros a kéken, a zöld mellettük:

	1	2	3
1			
2	piros		
3	kék	zöld	

A robot utasításait így kezdjük:

```
Ha Tábla(1,1) nem üres
    akkor Fogd meg a Tábla(1,1) színű kockát!
          Tedd az asztalra!
Ha ...
```

A. Folytasd az utasításokat!

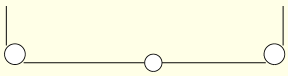
B. Fogalmazz meg egy részcélt, aminek elérése után a robot már nagyon könnyen megoldhatja a feladatot!

Elérhető összpontszám: 100 pont

1993. Harmadik forduló

1. feladat: (60 pont)

Készíts BASIC programot, amellyel biliárdozni lehet! A biliárdasztal képe:



Egyetlen golyót helyezhetünk el az asztalon, s azt valamilyen irányban (a nyolc szomszéd hely valamelyike felé, azaz vízszintesen, függőlegesen vagy átlósan) elindíthatjuk.

A golyó az asztal széléről visszapattan, a O-val jelölt lyukakon leeshet, a középen álló bábut (X) pedig feldöntheti.

Kövessük nyomon a golyó mozgását, amíg le nem esik, vagy fel nem dönti a bábut!

A program kérdezze meg, hogy hova tegyük induláskor a golyót és az merre mozogjon, utána rajzolja ki az asztalt, s rajzolja a golyó mozgását!

2. feladat: (40 pont)

A BKV új jegyrendszerre tér át a budapesti autóbuszokon.

Ezentúl minden jegyen 2 lyukasztás lesz, de a lehetséges lyukasztásokat úgy akarják kiosztani, hogy ha az utas a jegyet fordítva teszi be a lyukasztógépbe, akkor is egyértelműen felismerhető legyen.

Ezért például a következő két lehetőségből csak az egyik fordulhat elő:

1	2	3
4	5	6
7	8	9

két "azonos értékű" lyukasztása

O	O	3
4	5	6
7	8	9

1	O	O
4	5	6
7	8	9

Készíts programot, amely beolvassa, hogy mely két helyen kell kilyukasztani a buszjegyet, s kirajzolja a jegyet normál és fordított állásban is! (Ha a beolvasott két szám az 1 és a 2, akkor a fenti két rajz jelenjen meg!)

Elérhető összpontszám: 100 pont

1994. Első forduló

1. feladat: (26 pont)

Mit rajzolnak a következő, LOGO nyelvű programok? (Induláskor a kurzor a képernyő közepén áll és felfelé néz.) Milyen geometriai alakzatokat ismersz fel? Rajzold le és írj mellé méreteket is!

`LEFT f, RIGHT f`	balra, illetve jobbrafordulás helyben f fokkal,
`FORWARD h`	előrelépés h egységgel,
`REPEAT db [utasítások]`	utasítások db-szeri megismétlése

```
ELSOKEP A    ← eljárásnév és paraméterei
  REPEAT 4 [FORWARD A RIGHT 90]
END  ← eljárás vége

MASODIKKEP X Y
  REPEAT 2 [FORWARD X RIGHT 90 FORWARD Y RIGHT 90]
END

HARMADIKKEP B C
  REPEAT 2 [FORWARD B RIGHT C FORWARD B RIGHT 180-C]
END

NEGYEDIKKEP D E F
  REPEAT 2 [FORWARD D RIGHT F FORWARD E RIGHT 180-F]
END
```

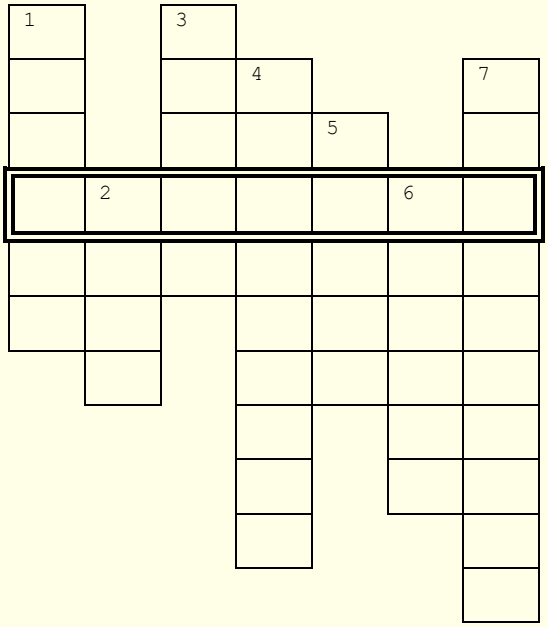
2. feladat: (25 pont)

Mit csinál az *Első*, *Második*, *Kiszámol*, *Harmadik* nevű algoritmus? (A bennük használt jegy(X,i) függvény az X 10-es számrendszerbeli szám hátulról számított i. számjegyét adja meg.)

```
Első(X):
  j:=jegy(X,1)
  Ha j=0 vagy j=2 vagy j=4 vagy j=6 vagy j=8
     akkor  Ki: "IGEN" különben Ki: "NEM"
Eljárás vége.

Második(X):
  Ha X>=10 akkor X:=Kiszámol(X)
  Ha X=0 vagy X=3 vagy X=6 vagy X=9 akkor  Ki: "IGEN"
                                   különben Ki: "NEM"
Eljárás vége.

Kiszámol(X):
  S:=0
  Ciklus i=1-től X számjegyei számáig
    S:=S+jegy(X,i)
  Ciklus vége
  Ha S<10 akkor  Kiszámol:=S
         különben Kiszámol:=Kiszámol(S)
Eljárás vége.

Harmadik(X):
  j:=jegy(X,1)
  Ha j=0 vagy j=5 akkor Ki: "IGEN" különben Ki: "NEM"
Eljárás vége.
```

3. feladat: (24 pont)

Ha a keresztrejtvény oszlopait megfejted, a duplán keretezett sorban, vízszintesen egy neves magyar származású matematikus-számítástechnikus nevét kapod. Töltsd ki a keresztrejtvényt! Add meg a kitalálandó személy keresztnevét is!



1. Egy állat, amely az egyik programozási nyelvnek is központi alakja.
2. Egy másik állat, egyben a számítógép bemeneti eszköze.
3. A számítástechnikában gyakori angol szó, jelentése: bemenet.
4. Számítógép gyártó cég.
5. Egy programozási nyelv.
6. Számítógép hálózati operációs rendszer.
7. Háttértár típus.

4. feladat: (25 pont)

Készíts algoritmust (programot), amely beolvas egy-egy szöveg típusú változóba két db maximum 10 értékes számjegyet tartalmazó 2-es számrendszerbeli számot, összeadja őket a 2-es számrendszer összeadási szabálya alapján, majd kiírja a képernyőre. Az eredmény is maximum 10 értékes számjegyet tartalmazzon (túlcsordulás lehet, amit jelezned kell)!

Elérhető összpontszám: 100 pont

1994. Második forduló

1. feladat: (40 pont)

Hamupipőke gonosz mostohájától azt a feladatot kapta, hogy különböző színű lencséket válogasson szét. Az egyező színűeket azonos tálkába kell tennie. Előre nem tudja, hogy hányféle színű lencse lesz, ez csak feldolgozás közben derül ki. Készíts programot, amellyel segíthetsz neki.

A program olvassa be egyenként az egyes lencsék színeit, majd írja ki, hogy összesen hányféle lencse volt, s milyen színűből mennyit kell kiválogatni! Az utolsó beírt szín után üres sort kell írni.

Példa:

a beolvasott lencsék színei: fehér, fehér, sárga, fehér, sárga

az eredmény: összesen 2-féle lencse fordult elő
fehér lencse: 3 db
sárga lencse: 2 db

2. feladat: (20 pont)

Az ősember még csak egyes számrendszerben tudott számolni, azaz annyi jelet (vonalat) húzott, amennyi a szám lenne. Ezt a számrendszert módosítjuk úgy, hogy a nullát is kezelhessük benne.

Minden számot az értékénél eggyel több vonallal ábrázolunk, például:

```
0 -  |
1 -  ||
2 -  |||
...
```

Készíts programot, amely két szöveg típusú változóba beolvas egy-egy így ábrázolt pozitív egész számot, majd tényleges számmá alakításuk nélkül kiszámolja az összegüket, a különbségüket és a szorzatukat. Ellenőrizd, hogy az elsőből kivonható-e a második (az eredmény sem lehet negatív szám)!

Például: ||| + || = |||| (2+1=3), ||| - || = || (2-1=1),
||| * || = ||| (2*1=2).

3. feladat: (40 pont)

Készíts programot, amely megtanítja számítógépedet a LOGO programozási nyelv néhány utasítására! A program először olvassa be a végrehajtandó utasításokat (egy sorban egy utasítást adhatunk, az utasítások száma nem lehet több 100-nál), majd törölje le a képernyőt és hajtsa végre a beolvasott utasításokat!

Az utasításokat egy üres sor zárja, értelmetlen utasítást a program ne hajtson végre!

A LOGO nyelvben egy teknőcöt utasítunk mozgásra, forgásra, s mozgása közben a hasára erősített tollal rajzolhat. Kezdetben a teknőc a képernyő közepén áll, a tolla lent van a papíron (azaz rajzolhat vele), s a képen felfelé néz.

Az utasítások a következők lehetnek:

Előre: a helyétől az adott irányban előre lép 10 lépést, ha a tolla lent van, akkor közben vonalat húz,

Hátra: a helyétől az adott irányban hátra lép 10 lépést, ha a tolla lent van, akkor közben vonalat húz,

Jobbra: helyben elfordul jobbra 90 fokkal,

Balra: helyben elfordul balra 90 fokkal,

TollFel: a tollát felemeli a papírról, ettől kezdve nem rajzol, csak mozog,

TollLe: a tollát leteszi a papírra, ettől kezdve nemcsak mozog, hanem rajzol is.

Elérhető összpontszám: 100 pont

Az általános iskolás korúak országos versenye az első két év után szervezési problémák miatt megszűnt, 1993-ban és 1994-ben csak budapesti versenyként létezett.

Az általános iskolák, valamint az időközben megerősödött 6-, illetve 8-osztályos gimnáziumok egyre jelentősebb igénye miatt az 1994/95-ös tanévben az NJSZT Országos Versenybizottsága és Regionális Versenybizottságai úgy döntöttek, hogy e tanévtől a Nemes Tihamér OKSzTV-t egy harmadik korosztállyal, az általános iskolás korúakkal bővítik.

2. Nemes Tihamér
Nemzetközi Informatikai Tanulmányi Verseny

1985. Első forduló

Első-ötödik osztályosok

A programok, programrészletek HT-1080Z számítógépre készültek.

1. feladat: (4 pont)

Mi a funkciója az alábbi szubrutinnak, ha M1, M2 és M3 átadott, illetve átvett paraméterek?

```
500 IF M1<M2 OR M1<M3 THEN IF M2>M3 AND M2>M1 THEN
       S=M1: M1=M2: M2=S ELSE S=M1: M1=M3: M3=S
510 IF M2<M3 THEN S=M2: M2=M3: M3=S
520 RETURN
```

2. feladat: (5 pont)

Mit csinál a következő program?

```
10 INPUT N: IF N<2 OR N<>INT(N) THEN 10
20 FOR I=2 TO N
30   IF N/I<>INT(N/I) THEN 70
40   PRINT I
50   N=N/I
60   IF N/I=INT(N/I) THEN 50
70 NEXT I
80 STOP
```

3. feladat: (6 pont)

A következő programrészlet két növekvően rendezett számsorozat közös elemeit írja ki. Mi a feltétele annak, hogy egy elemet se írjon ki többször?

```
100 I=1: J=1
110 IF I>N OR J>M THEN 150
120   IF A(I)=B(J) THEN PRINT A(I)
130   IF A(I)<B(J) THEN I=I+1 ELSE J=J+1
140 GOTO 110
150 ...
```

4. feladat: (6 pont)

Az alábbi két program mit ír ki, mikor áll meg, és van-e különbség a kapott eredményekben? A válaszokat indokold!

1. program

```
20 K%=0: L%=0
30   K%=K%+L%
40   L%=2*L%
50   PRINT K%,-K%-1
60 GOTO 30
```

2. program

```
110 K%=1: L%=1: J%=-(K%+L%)
120   PRINT K%,J%
130   L%=2*L%
140   K%=K%+L%
150   J%=-(K%+1)
160 GOTO 120
```

5. feladat: (6 pont)

Milyen hibák vannak a következő rendező szubrutinban?

```
100 DIM A(N)
...
200 REM RENDEZÉS
210 FOR I=N TO 1 STEP -1
220 FOR J=1 TO I
230  IF A(J)>A(J+1) THEN X=A(I): A(I+1)=A(I): A(I)=A(I+1)
240 NEXT J
250 NEXT I
260 ...
```

6. feladat: (8 pont)

A következő szubrutin szövegek rendezett sorozatába beilleszt egy új szöveget; helyesen működik. Milyen kezdőértékadást kell alkalmazni ahhoz, hogy a szubrutin akkor is helyesen működjék, amikor a rendezett sorozatban még egyetlen elem sincs?

```
1000 REM BEILLESZTÉS (X$)
1010 N=N+1: T$(N)=X$
1020 B=0: A=T(B)
1030 IF A=-1 THEN 1070
1040 IF X$<=T$(A) THEN 1070
1050   B=A: A=T(B)
1060 GOTO 1030
1070 T(B)=N: T(N)=A
1080 RETURN
```

7. feladat: (10 pont)

Mit csinál a következő szubrutin? Mi a helyes működés feltétele? Indokold meg a 60-as és a 70-es sort! Miért szerepel a 110-es sorban H+1?

```
50 H1=X2-X1: H2=Y2-Y1
60 I1=ABS(H1): I2=ABS(H2)
70 IF I1>I2 THEN H=I1 ELSE H=I2
80 X=X1: Y=Y1
90 IF H=0 THEN SET(X,Y): RETURN
100 S1=H1/H: S2=H2/H
110 FOR I=1 TO H+1
120   SET(X,Y)
130   X=X+S1: Y=Y+S2
140 NEXT I
150 RETURN
```

8 feladat: (15 pont)

Zoli mérnök barátjától 16 nyomógombos, ún. hexadecimális billentyűzetet kapott ajándékba. "A billentyűk állapotát két függvényhívással, INP(60%)-kal és INP(61%)-kal kérdeztetheted le" - magyarázta a mérnök.

Talán tudod, INP inputot, bevitelt jelent, 60 és 61 pedig perifériacímek. Ha a lekérdezés pillanatában bármelyik billentyű le van nyomva, a neki megfelelő bit 1 lesz a függvénye előállította szavas érték alsó byte-jában; a felső byte mindenképpen 0 lesz:

függvényhívás értéke	7.	6.	5.	4.	3.	2.	1.	0. bitje
INP(60%)	"7"	"6"	"5"	"4"	"3"	"2"	"1"	"0"
INP(61%)	"F"	"E"	"D"	"C"	"B"	"A"	"9"	"8"

Zoli olyan szubrutint írt, amely bármelyik billentyű leütésére a neki megfelelő decimális számot állítja elő az A% változóban:

```
1000   B%=INP(61%)*256%+INP(60%)
1010   IF B%=0% THEN 1000
1020   A%=-1%
1030 IF B%<>0% THEN B%=B%/2%: A%=A%+1: GOTO 1030
1090 RETURN
```

Később Zoli a szubrutint úgy akarta módosítani, hogy az a decimális érték helyett a megfelelő karakter (0..9,A..F) ASCII-kódját tegye A%-ba. E célból a rutinba a következő sort szúrta be:

```
1040 A%=A%+ASC("0")
```

és kicsit csodálkozott, mert nem egészen az történt, amit várt. Amikor pedig a szubrutint alaposabban kipróbálta, észrevette, hogy az "furcsán" működik, ha az "F" billentyűt és egy vagy több másikat egyszerre üt le. Átgondolva a dolgot, az 1020-as és 1030-as sort így módosította:

```
1020   A%=15%
1030 IF B%>0% THEN B%=B%*2%: A%=A%-1: GOTO 1030
```

Válaszolj a következő kérdésekre!

1. Mit csinál a szubrutin, amíg nem ütnek le billentyűt?
2. Mit csinál a szubrutin az 1040-es sor hatására?
3. Javítsd ki úgy a fenti szubrutint, hogy az valóban a leütött karakter ASCII-kódját tegye az A% változóba!
4. Mit tesz a szubrutin első változata az A% változóba, ha egyszerre két vagy több billentyűt ütnek le?
5. Milyen furcsaságot tapasztalt Zoli, amikor az "F" billentyűvel egyszerre ütött le egy vagy több másikat? Mi a jelenség magyarázata?
6. Mutasd meg, hogy az új 1020-as és 1030-as sorok hatására a szubrutin minden esetben jól fog működni!

Elérhető összpontszám: 60 pont

1985. Második forduló

Első-ötödik osztályosok

Darazsak:

A gázok kiterjedését az I. gimnáziumi Fizika tankönyv az alábbi módon modellezi:

"Rajzold le két egymásba nyíló, teljesen egyforma szoba alaprajzát! Vágj ki papírból 6 darazsat, számozd meg őket 1-től 6-ig, és tedd mindet az egyik szobába! 'Röpködjenek' a darazsak teljesen rendszertelenül az egyik szobából a másikba! A véletlenszerű átröppenést dobókocka segítségével modellezd! Amelyik darázs számát dobod, az repül át a másik szobába. Jegyezd fel minden dobás után, hány darázs van a szobákban! Legalább 30 dobás után számold meg, hányszor adódott a (6,0), hányszor az (5,1), ..., hányszor a (0,6) állapot!"

Készíts a feladat számítógépes megoldására olyan programot, amely legfeljebb 16 darázs esetén szemlélteti a darazsak röpködését és eloszlását!

Jelölések:

D - Darazsak száma
D(I) - az I-edik darázs melyik szobában van (1 vagy 2) (I=1,...,D)
T - az aktuális idő (T=1,2,...)
X - mennyi darázs van az adott (T) időpillanatban az 1. szobában
S(X) - hányszor volt x darab darázs az 1. szobában (X=0,...,D)

A feladatot az alábbi, jól elkülönített (és önállóan értékelhető) részekből építsd fel:

1., Szimulációs lépés (a fenti változók és tömbök kezelése)

2., A darazsak megjelenítése a képernyőn:

- a képernyőn látható legyen T és X értéke
- a képernyőn jelenjen meg a két szoba, és bennük a darazsak (pl. egy-egy ponttal jelképezve)

3., A program felhasználójával való kapcsolat:

- megadható legyen a darazsak száma (D) és kezdetben az első szobában levő darazsak száma
- a felhasználó menet közben bármikor beavatkozhasson(!), és ilyenkor a következőket teheti:
  - befejezi a program használatát
  - összegző oszlopdiagramot kér (hisztogram) (lásd 5. részfeladat)
  - üzemmódot változtat:
    - a. minden röptetés után a program egy billentyű lenyomására vár
    - b. folyamatos röpködés
    - c. megjelenítés nélkül "gyorsan" végzi a szimulációs lépéseket

4., Grafikon készítése a darazsak számáról az 1. szobában

- a grafikonon mindig az utolsó 50 időegység állapota látszódjon (ennek a legegyszerűbb megoldása az, ha az 50. oszlop után újra az elsőbe kezd rajzolni a program, és jelzi azt, hogy melyik az aktuális oszlop)
- ez a grafikon mindig elfér a képernyő megfelelő részén (D≤16!)

5., Összegző oszlopdiagram (hisztogram)

- szemléltesd azt, hogy hányszor volt az 1. szobában 0,1,...,D darab darázs
- ez a grafikon nem biztos, hogy elfér a képernyőn (gondolj például 1000 kísérletre), így ilyenkor gondoskodnod kell a megfelelő kicsinyítésről úgy, hogy azért használd ki a képernyő minél nagyobb részét
- a grafikon megnézése után legyen mód a szimuláció folytatására.

Elérhető összpontszám: 100 pont

1986. Első forduló

Első-ötödik osztályosok

A programok, programrészletek HT-1080Z számítógépre készültek.

1. feladat: (11 pont)

1.1-1.6-ig: add meg a helyes válasz(ok) betűjelét!

1.7-1.10-ig: a csoportokban "kakukktojások" vannak: egy-egy fogalom más, mint a többi; add meg ezek betűjelét!

1.1. Mi az asszembler? (1 pont)
- A: egy nyelv
- B: egy program
- C: egy számítógép
- D: egy híres számítástechnikus
- E: egy számítógépgyártó cég

1.2. Mely állítás(ok) igaz(ak) a hexadecimális számrendszerre? (1 pont)
- A: számjegyeit 0..9-cel és A..F-fel jelöljük
- B: alapszáma 16
- C: egy hexadecimális szám mindig H-val kezdődik
- D: két hexadecimális számjeggyel 255-ig lehet elszámlálni
- E: két hexadecimális számjeggyel 65535-ig lehet elszámlálni

1.3. Mi az EPROM? (1 pont)
- A: elektromosan programozható, írható és olvasható tár
- B: mikrokazetta adattároláshoz
- C: speciális eszközzel törölhető és programozható, csak olvasható tár
- D: elektromosan törölhető tár
- E: mágneslemez adattároláshoz

1.4. Mi a szoftver? (1 pont)
- A: a számítógéphez használható programok összessége
- B: az operációs rendszer
- C: a fordítóprogramok összessége
- D: a számítógép gépi berendezéseinek összessége
- E: a számítógéphez adott gyári programok

1.5. Mely állítás(ok) igaz(ak) a bit fogalmára? (1 pont)
- A: a kettes számrendszer egy számjegye
- B: a számítógép méretének egysége
- C: hexadecimális számjegy
- D: a számítógéppel ábrázolható legkisebb szám
- E: az információ egysége

1.6. Mi a megszakítás (interrupt)? (2 pont)
- A: a számítógép egyes részeinek kikapcsolása
- B: külső jel hatására a futó program végrehajtása felfüggesztődik és egy másik programrész indul el
- C: külső jel hatására a számítógép elölről kezdi működését
- D: a BASIC program megállítása STOP utasítással
- E: a BASIC program megállítása END utasítással

1.7. (1 pont)
A: PRINT
B: INPUT
C: INKEY$
D: PEEK
E: INP

1.8. (1 pont)
A: GOSUB
B: IF
C: RETURN
D: ON GOTO
E: GOTO

1.9. (1 pont)
A: regiszter
B: rekesz
C: jelzőbit
D: szubrutin
E: verem

1.10. (1 pont)
A: LOG
B: EXP
C: TAB
D: TAN
E: SIN

2. feladat: (6 pont)

Zsolti terepasztalt épített. A vonatok és a terepasztal (váltók,jelzők) állapotának lekérdezéséhez számítógépet használ. Az alábbi szubrutin meghívásakor az X és Y változókban olyan, 255-nél nem nagyobb számok vannak, amelyeknek egyes bitjei vagy bitcsoportjai a terepasztal és a vonatok állapotára jellemző információkat hordoznak.

Határozd meg az egyes bitek szerepét!

```
20 PRINT "ELSŐ VONAT";X AND 7;"CM/S"
30 PRINT "MÁSODIK VONAT";Y AND 7;"CM/S"
40 PRINT: PRINT "VÁLTÓK"
50 FOR I=3 TO 7: PRINT TAB(3);I-2;":";
60   IF (X AND INT(2^I))/INT(2^I)=1 THEN PRINT "kitérő" ELSE
PRINT "egyenes"
70 NEXT I
80 PRINT: PRINT "JELZŐK"
90 FOR I=3 TO 7: PRINT TAB(3);I-2;":";
100  IF (Y AND INT(2^I))/INT(2^I)=1 THEN PRINT "tilos" ELSE
PRINT "szabad"
110 NEXT I
120 RETURN
```

3. feladat: (5 pont)

Értsd meg az alábbi programot és válaszolj a kérdésekre!

```
10 INPUT A
20 INPUT N
30 IF N<1 OR N<>INT(N) OR LEN(STR$(INT(A)))+N>10 THEN
PRINT"H": GOTO 20
40 N=10^N: V=INT(N*A)
50 IF N*A-V>=.5 THEN V=V+1
60 A=V/N
```

3.1. Mit csinál a program?

A: semmit, hibajelzéssel megáll
B: kerekíti A-t N tizedesjegyre
C: kerekíti A-t N+1 tizedesjegyre
D: levág A-ból N tizedesjegyet
E: valami mást

3.2. A számolás során lehet-e túlcsordulás?

A: nem, mert nem működik
B: igen, ha A túl nagy
C: igen, ha N túl nagy
D: soha, mert védve van túlcsordulás ellen
E: egyéb esetekben igen

4. feladat: (4 pont)

Egy szakácskönyvben ezt olvastuk:

"GRÍZGALUSKA KÉSZÍTÉSE"
Hozzávalók: 1 tojás, 5 dkg gríz, só.
1 tojásfehérjét felverünk habnak. Belekeverjük a sárgáját, csipet sót és 5 dkg grízt. Forró vízben addig főzzük, amíg a grízszemek üvegesek nem lesznek."

Ez a leírás egy algoritmus, amely több részalgoritmusból áll, amelyek további részalgoritmusokból állnak, s.í.t. Eleminek tekintett algoritmusokból összetett algoritmusokat úgy hozhatunk létre, hogy

1. felsoroljuk a részalgoritmusait,
2. megismétlünk bizonyos részalgoritmusokat,
3. választunk részalgoritmusok között.

Az algoritmusok műveleteket hajtanak végre tárgyakkal, tárgyakon (ezeket a számítástechnikában gyakran adatoknak nevezzük). Algoritmusok és adatok viszonya többféle lehet:

A. Az algoritmuson kívül létező adatokat – az algoritmus szempontjából – külső adatoknak hívjuk.

B. Az algoritmus működése során létrejövő és az algoritmus befejeztével megszűnő adatokat belső adatoknak nevezzük.

C. Adatok lehetnek bemenő vagy kimenő paraméterei egy algoritmusnak.

4.1. Milyen fogalmat takarnak az ételreceptben szereplő nevek?

Fogalom: belső adat (BA), külső adat (KA), bemenő paraméter (BP), kimenő paraméter (KP), algoritmus (A).

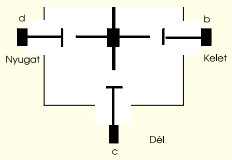
Név: tojáshab, belekeverjük, tojás, víz, grízgaluska, üvegesek nem lesznek.

4.2. Mondj példát az ételreceptből ismétlést tartalmazó algoritmusra!

5. feladat: (9 pont)

Zolit mérnök barátja ezúttal botkormánnyal lepte meg. "Rohannom kell!" – mondta köszönés helyett is a mérnök. – "Én csináltam négy kapcsolóból, itt van a rajza. A kipróbálásához írtam egy kis programot. Lehet, hogy rossz, nem volt időm futtatni. Sok sikert!"

Segíts Zolinak, magyarázd meg a 'hosszú várakozás' és a 'rövid várakozás' szerepét a programban, és javítsd ki a hibákat!



A botkormány a jelzett négy irányba mozgatható, egyszerre csak egy érintkező záródhat. Az a, b, c és d jelű kimenetek illesztőegységen keresztül csatlakoznak a számítógéphez, állapotukat az INP(60) utasítással lehet megtudni: a zárt érintkező 0-t, a nyitott pedig 1-et jelez az alábbi pozíciókban (a 7-4. bitek értéke mindig 1):

```
       7. 6. 5. 4. 3. 2. 1. 0. bit
       1  1  1  1  d  c  b  a
```

```
 10 REM pergésgátlás
 20 A = INP(60): IF A = 255 THEN GOTO 20
 30 FOR I = 1 TO 40: NEXT I
 40 IF INP(60) <> A THEN GOTO 20
 50 GOSUB 1000
 60 REM hosszú várakozás
 70 FOR I = 1 TO 150: NEXT I
 80 IF INP(60) <> A THEN GOTO 20
 90 GOSUB 1000
100 REM rövid várakozás
110 FOR I = 1 TO 20: NEXT I
120 IF INP(60) <> A THEN GOTO 20
130 GOSUB 1000
140 GOTO 110
1000 REM a kód értelmezése
1010 A = A AND 31
1020 IF A = 14 THEN PRINT "D"; : RETURN
1030 IF A = 11 THEN PRINT "É"; : RETURN
1040 IF A = 13 THEN PRINT "K"; : RETURN
1050 IF A =  8 THEN PRINT "N"; : RETURN
1060 RETURN
```

6. feladat: (8 pont)

Mi lesz végrehajtás után az A, B, C változók értéke? (A ':=' jelölés az értékadást jelöli.)

```
A:=0; B:=1; C:=1
Ciklus amíg B<=N
  A:=A+1
  C:=C+2
  B:=B+C
Ciklus vége
```

7. feladat: (11 pont)

Az A(N,N) mátrixban egy fekete-fehér képet tárolunk, képpontonként. A(I,J)=0, ha a képpont teljesen fehér; 255, ha fekete; a közbülső értékek a szürke egyenletesen sötétedő árnyalatait jelentik.

Mit tesznek a képpel a következő transzformációk (N páratlan szám)?

```
7.1. 100 FOR I=0 TO N: FOR J=0 TO N
     110   B(I,J)=A(I,J)*A(I,J)/255
     120 NEXT J: NEXT I

7.2. 100 FOR I=0 TO N/2: FOR J=0 TO N/2
     110   C(I,J)=(A(I*2,J*2)+A(I*2+1,J*2)+
                   A(I*2,J*2+1)+A(I*2+1,J*2+1))/4
     120 NEXT J: NEXT I

7.3. 100 FOR I=0 TO N: FOR J= 0 TO N
     110   D(I,J)=A(I/3,J/3)
     120 NEXT J: NEXT I
```

8. feladat: (6 pont)

Egy programnyelv 3 utasításának jelentése a következő:

REPEAT darabszám [utasítások]	- a zárójelbe tett 'utasítások'-at 'darabszám'-szor megismétli
FORWARD lépésszám	- 'lépésszám' hosszan vonalat húz az aktuális helyzetből kiindulva az aktuális irányban
LEFT szög	- az aktuális irányt balra, a fokokban megadott 'szög'-gel megváltoztatja

Itt van egy program ezen a nyelven:

```
REPEAT A [FORWARD B LEFT C ]
```

Mi történik, ha az A, B, C változóknak rendre az alábbi értékeket adjuk?

	8.1.	8.2.	8.3.	8.4.	8.5.
A	3	4	360	3	5
B	100	100	1	10	100
C	120	90	1	240	144

9. feladat: (6 pont)

Milyen bemenő adatokkal működik a következő programrészlet a leggyorsabban, illetve a leglassabban (N>2 és állandó)?

```
100 I=N
110 IF I<2 THEN 190
120 CS=0: J=1
130 IF J=I THEN 170
140 IF A(J)>A(J+1) THEN X=A(J):A(J)=A(J+1):A(J+1)=X:CS=J
150 J=J+1
160 GOTO 130
170 I=CS
180 GOTO 110
190 ..
```

Elérhető összpontszám: 66 pont

1986. Második forduló

Első-ötödik osztályosok

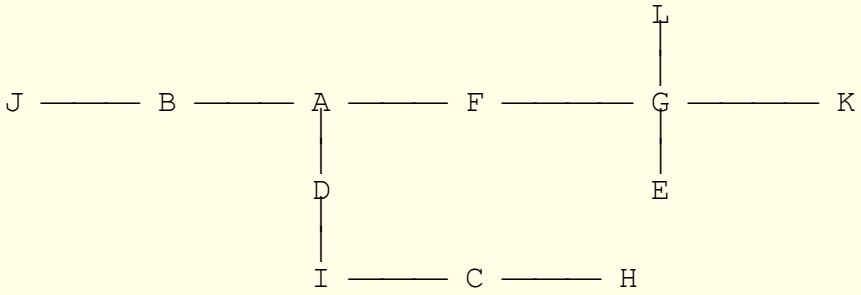
Folyóiratküldő:

Tervezz és írj programot a Neumann János Számítógéptudományi Társaság tagjainak nyilvántartására, amely számítástechnikai szakfolyóiratok szétküldésére és a szállítási útvonalak meghatározására szolgál.

Tárolandó adatok:

- név (max. 25 betű, ékezetes betűk nem szerepelnek)
- város (számuk max. 16, jelük egy betű az (A,P) tartományban)
- kért újság (vagy a Számítástechnika vagy a Mikroszámítógép Magazin rendelhető).

Az újságokat vasúton szállítják. A vasúti összeköttetéseket a felhasználó adja meg. A hálózat városokból, mint csomópontokból és ezeket összekötő vonalszakaszokból áll. A hálózat megadásának és tárolásának módját te határozd meg. Egy példa a lehetséges összeköttetésekre:



Megoldandó részfeladatok:

A. A tagok adatainak nyilvántartása:

1. Az adatbázis létrehozása legalább 30 tagra
2. Új tagok felvétele
3. Tag törlése
4. Tag adatainak módosítása
5. A tagok névsor szerinti listája minden hozzá tartozó adattal
6. A tagok listája újságonként külön-külön, ezen belül városok szerint ábécé sorrendben

B. Szállítással kapcsolatos feladatok:

1. A vasúti hálózat modelljének létrehozása a hozzá tartozó kezelőfunkciókkal. Ezek a funkciók a választott beviteli módtól függenek, így ezeket is neked kell meghatároznod.
2. Újságonként külön-külön lista készítése arról, hogy melyik városba mennyi újságot kell küldeni, és ebből hányat kell továbbítani a következő állomás(ok)ra. Feltételezzük, hogy mindkét újságot ugyanott, az "A" városban nyomtatják, és hogy "A" városból minden városba el lehet jutni és csakis egy útvonalon.
3. A feladat ugyanaz, mint az előző pontban, de a két újságot bármelyik két különböző városban nyomtathatják, és néhány város több úton is elérhető.

Kiegészítések:

- A listákat képernyőre kell írni.

- Az adatbázisok háttértárolását nem kell megoldani.
- A program a lehető legjobban legyen dokumentálva, feleljen meg az általános formai követelményeknek, és legyen barátságos a felhasználóval.
- A program első sora egy REM sor legyen, és tartalmazza a következő adatokat: név, iskola, helység.

Elérhető összpontszám: 70 pont

A verseny végeredménye:

1. Erdei Zsolt	Árpád Gimnázium, Budapest
2. Huszár Péter	Széchenyi István Gimnázium, Sopron
3. Paller Gábor	Teleki Blanka Gimnázium, Budapest
4. Mazán Zsolt	Szilády Áron Gimnázium, Kiskunhalas
5. Makay Géza	Ságvári Endre Gimnázium, Szeged
6. Bélteky Zsolt	Hermann Ottó Gimnázium, Miskolc
7. Jánossy Zoltán	Lovassy László Gimnázium, Veszprém
8. Novák István	Berze Nagy János Gimnázium, Gyöngyös
9. Mamrovics László	Árpád Gimnázium, Budapest
10. Pallagi László	Vörösmarty Mihály Gimnázium, Érd

1987. Első forduló

Első-ötödik osztályosok

1. feladat: (12 pont)

Melyek a bemenő adatok, s milyennek kell lenniük, hogy a következő programrészlet jól működjön? A program egy számsorozatot úgy ír ki, hogy először a benne szereplő negatív értékeket írja, majd pedig a nemnegatívakat! Mi a szerepe az F1, F2 változóknak és a B(N) vektornak?

```
110 F1=0: F2=0: X=0
120 FOR I=1 TO N
130  IF A(I)>=0 THEN B(I)=F2: F2=I
                ELSE B(I)=F1: F1=I: IF B(I)=0 THEN X=I
140 NEXT I
150 B(X)=F2: X=F1
160 IF X<>0 THEN PRINT A(X): X=B(X): GOTO 160
```

2. feladat: (5 pont)

Egy veremautomata a következőképpen működik: ha számot kap a bemenetén, azt egy verembe teszi, ha műveleti jelet, akkor azt a verem tetején levő számokkal elvégzi, majd az eredményt a verembe teszi (pl. A-B a következőképpen néz ki az automata nyelvén: A B -). A veremautomata bemenetén a következő sorozatot kapja (a számokat egymástól és a jelektől szóköz választja el):

```
1 3 + 2 5 4 * 8 - 6 / + -
```

Add meg, mi lesz a verem állapota az egyes jelek (adat vagy művelet) érkezése után, illetve a feldolgozás végén!

3. feladat: (12 pont)

Mit csinálnak a következő rekurzív programok (X az angol ABC betűinek sorozata)?

A.
```
F(X):
   Ha X üres akkor eredmény=:0
   különben Ha (első(X) (A,E,I,O,U)-valamelyike)
              akkor  eredmény:=1+F(elsőnkívüli(X))
            különben eredmény:=F(elsőnkívüli(X))
  Függvény vége.
```

B.
```
F(X):
   Ha X üres akkor eredmény:=IGAZ
   különben eredmény:=(első(X) (A,E,I,O,U)-valamelyike)
                 ÉS F(elsőnkívüli(X))
  Függvény vége.
```

C.
```
F(X):
   Ha elsőnkívüli(X) üres akkor eredmény:=első(X)
   különben Ha első(X)>F(elsőnkívüli(X))
              akkor  eredmény=:első(X)
            különben eredmény:=F(elsőnkívüli(X))
  Függvény vége.
```

Megjegyzés: Az első(X) függvény megadja az X karaktersorozat első tagját. Az elsőnkívüli(X) függvény megadja az X karaktersorozat tagjait a 2.-tól a végéig. az ÉS a szokásos logikai "és" művelet.

4. feladat: (9 pont)

Mik a hibák a következő programban? Adj meg egy olyan bemenő adatot, amelyre helyesen működik!

```
10 INPUT "KÉREK EGY EGÉSZ SZÁMOT! (>2)";N
15 IF N<3 OR N<>INT(N) THEN 10
20 K=0: L=0
30 FOR I=2 TO SQR(N)
40   IF N/I=INT(N/I) THEN K=K+1
50   IF K=1 THEN L=I
60 NEXT I
70 PRINT "VALÓDI OSZTÓK SZÁMA=";K*2
80 PRINT "LEGKISEBB VALÓDI OSZTÓ=";L
90 STOP
```

5. feladat: (12 pont)

A következő LOGO nyelvű program, adott x és f értékre, egy ábrát rajzol a képernyőre (az egyes utasításokat szóköz választja el egymástól).

```
FORWARD 2*x RIGHT f FORWARD x LEFT f/2 BACK x LEFT f/2 BACK x
```

Az utasítások jelentése (a LOGO-ban egy teknőcnek nevezett kurzor segítségével rajzolhatunk, amely az aktuális irányba előre vagy hátra képes lépni, valamint jobbra vagy balra tud fordulni):

FORWARD hossz - előre lép és vonalat húz
BACK hossz - hátra lép és vonalat húz
RIGHT fok - jobbra fordul helyben
LEFT fok - balra fordul helyben

Mit kell módosítani, ha azt szeretnénk elérni, hogy a kapott ábra az eredeti tükörképe legyen

A: az eredeti irányban a kezdőponton át húzott egyenesre,

B: az eredeti irányra merőleges, a kezdőponton át húzott egyenesre,

C: a kezdőpontra?

A végrehajtandó utasítások száma nem változhat! (Próbálj többféle megoldást adni!)

6. feladat: (5 pont)

A következő programrészletek A és B egész számok (B≠0) hányadosának egészrészét határozzák meg és helyezik el az X változóban. Egyik sem működik tetszőleges egész számokra. Milyen feltételek mellett helyesek az egyes megoldások (a B≠0 feltételen kívül)?

A:
```
Osztás:
     Ciklus amíg B>1
        Ha A páros és B páros akkor A:=A/2: B:=B/2
     Ciklus vége
     X:=A
  Eljárás vége.
```

B:
```
Osztás:
     X:=0
     Ciklus amíg abs(A)≥abs(B)
        A:=A-B: X:=X+1
     Ciklus vége
  Eljárás vége.
```

7. feladat: (10 pont)

A következő gépi kódú program az A és a B regiszterben vár egy-egy egész számot. Mely esetben fejezi be a végrehajtást az IGEN címkénél, és mely esetben a NEM címkénél (A,B>0)?

```
CIKLUS:
  CP 0  ; hasonlítsuk össze A-t 0-val!
  JP Z,IGEN  ; ugrás, ha A=0 volt
  JP M,NEM   ; ugrás, ha A<0 volt
  BIT0,A  ; A legkisebb helyiértékű bitje 1-es?
  JP NZ,PTLANA ; ugrás, ha 1-es volt
  BIT0,B  ; B legkisebb helyiértékű bitje 1-es?
  JP NZ,PTLANB ; ugrás, ha 1-es volt
  SRLA  ; A bitjeinek eggyel jobbra léptetése
        ; a legmagasabb helyiértékre 0 lép be
  SRLB  ; B bitjeinek eggyel jobbra léptetése
        ; a legmagasabb helyiértékre 0 lép be
  JP CIKLUS  ; ugrás a CIKLUS címkére
PTLANA:
  BIT0,B  ; B legkisebb helyiértékű bitje 1-es?
  JP Z,NEM   ; ugrás, ha 0-s volt
  SUBB  ; A:=A-B
  JP CIKLUS  ; ugrás a CIKLUS címkére
PTLANB:
  SRLA  ; A bitjeinek eggyel jobbra léptetése
        ; a legmagasabb helyiértékre 0 lép be
  JP CIKLUS  ; ugrás a CIKLUS címkére
IGEN:   ...
  ...
NEM: ...
```

8. feladat: (8 pont)

Egy tetszőleges F(X) folytonos függvény zérushelyét keressük E pontossággal az [A,B] intervallumon (abs(F(X))≤E tulajdonságú X-et keresünk). Milyen elégséges feltételeknek tegyen eleget az F(X) függvény az A és a B pontban, hogy az [A,B] intervallumban biztosan legyen legalább egy zérushelye és a programrészlet ezek közül egyet megtaláljon?

A:
```
K:=(A+B)/2
  Ciklus amíg abs(F(K))<=E
     Ha F(K)<0 akkor A:=K különben B:=K
     K:=(A+B)/2
  Ciklus vége
  Ki: K,F(K)
```

B:
```
K:=A-F(A)*(B-A)/(F(B)-F(A))
  Ciklus amíg abs(F(K))<E
     Ha F(K)*F(A)<0 akkor B:=K különben A:=K
     K:=A-F(A)*(B-A)/(F(B)-F(A))
  Ciklus vége
  Ki: K,F(K)
```

9. feladat: (7 pont)

Írd át gyorsabb működésűre a következő programrészletet! (A program BASIC nyelvű legyen és egy sorban csak egy utasítás lehet.) A program H0 magasságból leeső M tömegű test helyzeti és mozgási energiáját számolja méterenként.

```
100 INPUT M,H0
110 DIM X(H0),Y(H0)
120 G=9.81
130 FOR H=H0 TO 0 STEP -1
140  EH=M*G*H
150  EM=M*G*H0-M*G*H
160  X(H)=EM
170  Y(H)=EH
180 NEXT H
```

10. feladat: (15 pont)

Pisti mikroszámítógépéhez kapott egy "egeret". Ez egy cigarettásdoboz méretű eszköz, amely az asztalon szabadon tologatható, és elmozdulását egy vezetéken keresztül közli a számítógéppel. Az egér állapota az INP(63) utasítással kérdezhető le. Pisti írt egy rövid programot:

```
10 PRINT INP(63);
20 GOTO 10
```

A program elindítása után Pisti várt egy kicsit, majd lassan elmozdította az egeret először jobbra, majd balra, azután előre, végül hátra. Az irányváltások előtt picit mindig várt. A képernyőn a következő számsorozat jelent meg:

```
!<── az egér áll ──>!<──────── jobbra mozog ─────────>!<───── áll ──────>!
 240 240 240 240 240 242 243 241 240 242 243 241 240 242 242 242 242 242

!<────── balra mozog ──────>!<── áll ──>!<──────── előre mozog ────────>!
 240 241 243 242 240 241 243 243 243 243 251 255 247 243 251 255 247 243

!<──── áll ────>!<────────── hátra mozog ──────────>!<──── áll ────>!
 243 243 243 243 247 255 251 243 247 255 251 243 247 247 247 247 247
```

Pisti ezután a következő programot írta (egyes részeket neked kell megírnod!):

```
10 X=10: Y=10: X0=0: Y0=0
20 E=INP(63): E1=E-16*INT(E/16): Y1=INT(E1/4): X1=E1-4*Y1
30 X9=X0*10+X1
40 IF feltétel1 THEN X=X+1: GOTO 60: REM jobbra
50 IF feltétel2 THEN X=X-1: REM balra
60 Y9=Y0*10+Y1
70 IF feltétel3 THEN Y=Y+1: GOTO 90: REM előre
80 IF feltétel4 THEN Y=Y-1: REM hátra
90 PLOT X,Y: REM kirajzolja az (X,Y) koordinátájú pontot
100 X0=X1: Y0=Y1:GOTO 20
```

A: Mit jelentenek az egyes bitek?

B: Mi a szerepe az (X,Y), (X1,Y1), (X9,Y9) változópároknak?

C: Mit csinál a program és mi jelenik meg a képernyőn?

D: Add meg a feltétel1, feltétel2, feltétel3, feltétel4 helyére beírandó, X9-től, vagy Y9-től függő logikai kifejezéseket1

Elérhető összpontszám: 95 pont

1987. Második forduló

Első-ötödik osztályosok

Üzenetközvetítő:

A Zabfeldolgozó és Forgalmazó Vállalat – a továbbiakban ZAFFO – székházának portáján egy olyan számítógép van, mint előtted. Ez a gép üzenetközvetítőként működik. Minden vezető beosztású dolgozó, valamint a rendszer működéséért felelős programozó a nevével, a 6 osztály többi dolgozója pedig a nevével és osztályának azonosítójával jelentkezik be. A vállalatnak nincs két azonos nevű dolgozója. A dolgozó kiírathatja a neki szóló leveleket, leveleket küldhet és törölhet. A következő levéltípusok vannak:

Fajta	Címzett	Küldő
körlevél	minden dolgozó	vezérigazgató, illetve az MSZMP, a KISZ és a szakszervezet titkára
körlevél	igazgatók, titkárok	vezérigazgató
körlevél	osztályvezetők	vezérigazgató, igazgatók
körlevél	egy osztály dolgozói	vezető dolgozók
levél	egy dolgozó	bárki

A különböző levéltípusokat a következő szabályok alapján törölhetjük:

A. egy olvasás után törlendő,
B. adott dátumig mindig közlendő,
C. visszavonásig közlendő – a levél küldője vonhatja vissza,
D. a levelet kapó törölheti.

A törlési szabályokat a levél küldője határozza meg. Nem minden fajta levélre alkalmazható bármelyik törlési szabály, az értelmetlen kapcsolatokat a programnak ki kell zárnia.

A program funkciói:

A. a napi dátum beírása (ezt minden reggel Kiss Attila programozó végezheti),
B. levél küldése,
C. az összes levél összes adatának kiírása (ezt csak Kiss Attila programozó kérheti),
D. a dolgozónak szóló levelek kiírása,
E. a levél törlése (Kiss Attila bármely üzenetet törölhet).

(Javasoljuk, hogy az egyes részfeladatokat ilyen sorrendben oldd meg!)

A ZAFFO vezető dolgozói:

Nagy Lajos	vezérigazgató
Kiss Benedek	gazdasági igazgató
Szabados Erika	kereskedelmi igazgató
Kun Ilona	az áruforgalmi osztály vezetője
Forint Ede	a pénzügyi osztály vezetője
Talpas Egon	a könyvelési osztály vezetője

Debreceni Attila	a személyzeti osztály vezetője
Tokos Ibolya	a termelési osztály vezetője
dr. Antal Frigyes	a műszaki osztály vezetője
Koszorú Gyula	MSZMP titkár
Kezes Aranka	KISZ titkár
Petrovics Etelka	szakszervezeti titkár

Készítsd el a program tervét: a szükséges adatszerkezet leírását, a fontosabb változókat, a program algoritmusát, majd a feladatot megoldó BASIC programot! A program tagolt, áttekinthető, egyszerű legyen! A programnak a dátum helyességét nem kell ellenőriznie! A háttértáron való tárolást nem kell megoldanod! A levél szövege nem lehet hosszabb, mint ami elfér egy szöveg típusú változóban!

A munka végeztével be kell adni a programtervet, a program listáját kinyomtatva, a programot kazettán vagy lemezen.

Elérhető összpontszám: 90 pont

A verseny végeredménye:

1. Ertner Péter	Kossuth Lajos Gimnázium, Nyíregyháza
2. Drasny Gábor	Fazekas Mihály Gimnázium, Budapest
3. Rátkai István	Fazekas Mihály Gimnázium, Budapest
4. Rimányi Richárd	Révai Miklós Gimnázium, Győr
5. Kovács László	I. István Gimnázium, Budapest
6. Boros Péter	Radnóti Miklós Gimnázium, Szeged
7. Makó Balázs	Földes Ferenc Gimnázium, Miskolc
8. Dunay Rezső	Lovassy László Gimnázium, Veszprém
9. Kucsma István	Irinyi János Szakközépiskola, Kazincbarcika
10. Habony Zsolt	Földes Ferenc Gimnázium, Miskolc

1988. Első forduló

Első-ötödik osztályosok

1. feladat: (6 pont)

Az (x - 10.0 ** i) * (x - 1.0) = 0.0 egyenlet gyökeit az

```
a :=1.0;
b :=-(10.0 ** i + 1.0);
c :=10.0 ** i ;
x1:=(-b - SQRT(b ** 2 - 4.0 * a * c))/(2.0 * a);
x2:=(-b + SQRT(b ** 2 - 4.0 * a * c))/(2.0 * a);
```

utasításokkal akarjuk kiszámíttatni az i=1,2,...,25 értékekre, mégpedig egy olyan számítógépen, amelyen a valós számokat lebegőpontos alakban ábrázolják: a mantissza 24 bites bináris szám (az egyik bit az előjel), a 2-es alapú kitevő pedig 8 bites, ugyancsak bináris szám (az egyik bit itt is az előjel). A programban a ** a hatványozás jele.

Ha x1-et és x2-t kiíratjuk, milyen i értékek esetén nem kapjuk meg a helyes megoldást?

2. feladat: (15 pont)

Az alábbi program az m természetes számhoz megkeresi annak a leghosszabb [i,j] zárt intervallumnak a hosszát, amelyre az i,i+1,...,j természetes számok összege pontosan m (1≤i≤j).

```
     hossz := 0;
[1]  WHILE m > az első hossz szám összege
     REP hossz := hossz + 10 ENDREP;
[2]  WHILE m < az első hossz szám összege
     REP hossz := hossz - 1 ENDREP;
[3]  WHILE (m - az első hossz szám összege) MOD hossz<>0
     REP hossz := hossz - 1 ENDREP.

  az első hossz szám összege:
     IF hossz MOD 2 = 0
     THEN hossz DIV 2 * (hossz+1)
     ELSE (hossz +1) DIV 2 * hossz
     ENDIF.
```

A programban hossz (=j-i+1) és m egész számokat jelölnek, DIV egészosztás hányadosát, MOD egészosztás maradékát adja eredményül, WHILE ... REP ... ENDREP elöl vizsgált ismétlés, az első hossz szám összege a kiszámított értéket átadó (a függvényeljárásokhoz hasonló) algoritmus.

2.1. Mi a szerepe az [1]-gyel, [2]-vel, ill. [3]-mal jelölt ismétléseknek?

2.2. Hányszor kell végrehajtani a [3] ismétlést, 4095 illetve 4096 esetén?

2.3. Mutasd meg, hogy a program biztosan befejeződik (azaz nincs benne végtelen ciklus)!

2.4. Mutasd meg, hogy ha m 2 nemnegatív hatványa, akkor a maximális intervallumhossz éppen 1!

2.5. Hogyan lehetne gyorsabbá tenni a programot a meglévő jó tulajdonságok megőrzésével?

3. feladat: (8 pont)

Egy dobozban fekete és fehér kávészemek vannak. Véletlenszerűen kiveszünk két szemet. Ha ezek azonos színűek, akkor helyettük egy feketét rakunk vissza. (Erre a célra korlátlan mennyiségű fekete kávészem áll a rendelkezésünkre.) Ha viszont különböző színűeket húzunk, akkor egy fehéret dobunk vissza a dobozba.

3.1. Hogyan változik a fekete, ill. a fehér kávészemek száma a folyamat során?

3.2. A kávészemek számának milyen lényeges tulajdonsága nem változik meg az algoritmus végrehajtása közben?

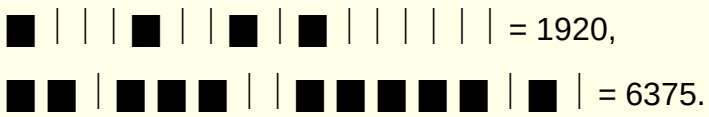
3.3. Mitől függ, hogy milyen színű kávészem marad a dobozban?

4. feladat: (9 pont)

A Sakál üzletház pénztárgépeihez vonalkódolvasó készülékeket vásárolt. Néhány hónap alatt kiderült, hogy a készülékek nem a nemzetközileg elfogadott vonalkódot, hanem valami mást ismernek fel. Bit Benő elhatározta, hogy megfejti a kódot. A készülék gépkönyvében erről a következőket találta:

A. A vonalkódban a bináris 0-kat egyszeres, az 1-eseket kétszeres vastagságú sötét vonal jelöli. A sötét vonalakat legfeljebb kétszeres vastagságú fehér csíkok választják el egymástól.

B. Egy-egy árucikk azonosítója 1000 ... 6999 közötti decimális szám; minden számjegyet 4 sötét vonallal kell kódolni. Példák:



C. A kódolvasó ceruzát a papírra kell tenni, és a vonalakra lehetőleg merőleges irányban folyamatos kézmozgással kell végighúzni.

D. Az árucikk száma nem függ attól, hogy a kódolvasó ceruzát elölről hátrafelé, vagy fordítva húzzuk-e végig a kódolt számsoron.

4.1. Mit állapított meg Benő, mi a számjegyek kódja?

4.2. Hogyan ismerhető fel a cikkszám mindkét irányból?

5. feladat: (6 pont)

Mit csinál a következő program, és milyen bemenő adatokra működik rosszul (feltéve, hogy a$-ba számjegyeket, b-be pedig pozitív egész számot írunk)?

```
10 INPUT "Számjegyeket kérek";a$
20 INPUT "Pozitív egész számot kérek";b
30 IF LEN(a$) < 3 OR b >= LEN(a$) THEN 10
40 PRINT MID$(a$,1,1);".";MID$(a$,2,b-2);
50 c$=MID$(a$,b,1)
60 IF MID$(a$,b+1,1) > "5" THEN c$=CHR$(ASC(c$)+1)
70 PRINT c$;"E";LEN(a$)-1
```

MID$(x$,p,h) az x$ karaktersorozat p-edik pozícióján kezdődő, h hosszúságú karaktersorozatot állítja elő.

6. feladat: (7 pont)

Adott egy 0-kból és 1-esekből álló számsorozat, meghatározandó a csupa 1-esekből álló szakaszok hossza. Pl. a 000111001100 sorozatban van egy 3 és egy 2 hosszúságú 1-es szakasz. Két algoritmust készítettünk a megoldásra, mindkettő az A(N) vektort használja. (Feltesszük, hogy N>1 természetes szám, és az A(N) vektor csak 1 és 0 elemeket tartalmaz.)

```
A: I:=2: H:=0
  Ciklus amíg I<+N
    Ha A(I-1)=0 és A(I)=1
       akkor H:=0
             Ciklus amíg A(I)=1
               I:=I+1: H:=H+1
             Ciklus Vége
             Ki: H
    különben I:=I+1
  Ciklus vége
```

```
B: I:=1: H:=A(1)
  Ciklus amíg I<N
    Ha A(I)=0 és A(I+1)=1 akkor H:=0
    Ha A(I+1)=1 akkor H:=H+1
    Ha A(I)=1 és A(I+1)=0 akkor Ki: H
    I:=I+1
 Ciklus vége
```

6.1. Milyen esetben működnek különbözően és mi ez a különbség?

6.2. Milyen esetben hibás valamelyik és mi a hiba?

7. feladat: (7 pont)

Mit rajzol a következő COMAL nyelvű program, ha feltesszük, hogy a rajzolás során végig a képernyőn maradunk?

```
0010 DIM X(5), Y(5)
0020 FOR I := 1 TO 4 DO
0030   INPUT X(I), Y(I)
0040 ENDFOR I
0050 X(5) := X(1); Y(5) := Y(1)
0060 INPUT A
0070 SETGRAPHIC 0
0080 ALFA := 0
0090 RAJZOLÁS
0100 ALFA := A
0110 RAJZOLÁS
0120 STOP
```

```
0200 PROC RAJZOLÁS
0210    PENUP
0220    SETXY X(1) * COS(ALFA) + Y(1) * SIN(ALFA),
             -X(1) * SIN(ALFA) + Y(1) * COS(ALFA)
0230    PENDOWN
0240    FOR I := 2 TO 5 DO
0250       SETXY X(I) * COS(ALFA) + Y(I) * SIN(ALFA),
                 -X(I) * SIN(ALFA) + Y(I) * COS(ALFA)
0260    ENDFOR I
0270 ENDPROC RAJZOLÁS
```

SETXY x,y jelentése: húzz egyenest az aktuális ponttól (x,y)-ig, majd vedd (x,y)-t az aktuális pontnak. SETGRAPHIC 0 grafikus üzemmódba vált át, PENUP felemeli, PENDOWN pedig leteszi a tollat.

8. feladat: (9 pont)

Egy n*m-es tömbben tároljuk az ország térképét. Minden (x,y) koordinátájú pontban M(x,y) jelenti az adott pont tengerszint feletti magasságát. A BASIC programrészlet megpróbálja megkeresni az ország legmagasabb pontjának koordinátáit, mindig a legmeredekebb növekedés irányában haladva.

Milyen esetekben nem találja meg, és mi történik ilyenkor?

```
100 x=RND(n): y=RND(m): REM 1 és n, ill. m közötti véletlen
egész
110 x0=x: y0=y
120 FOR i=x-1 TO x+1: IF i<1 OR i>n THEN 160
130 FOR j=y-1 TO y+1: IF j<1 OR j>m OR (i=x AND j=y) THEN 150
140     IF M(i,j)>=M(x0,y0) THEN x0=i: y0=j
150  NEXT j
160 NEXT i
170 IF x0<>x OR y0<>y THEN x=x0: y=y0: GOTO 110
180 PRINT x,y
```

9. feladat: (8 pont)

Egy lánctalpas játékautóra 2 fényérzékelőt szereltünk, amelyek egymással 10-15 fokos szöget zárnak be. A jármű bal-, ill. jobb oldali lánctalpát külön is, egyszerre is működtethetjük. A járműnek adható parancsok a következők:

OUT 10,szám	a szám legkisebb helyiértékű bitje a bal oldali, a következő pedig a jobb oldali lánctalp mozgatására használható (1 = elindul, 0 = megáll);
INP(11)	értéke a bal oldali fényérzékelő által mért fényerősség (0-255 közötti egész);
INP(12)	értéke a jobb oldali fényérzékelő által mért fényerősség (0-255 közötti egész).

A fényerősség mért értéke attól függ, hogy milyen messze van az érzékelő a fényforrástól, és milyen szöget zár be vele. Készítettünk a játékautóhoz egy BASIC programot:

```
 10 INPUT e: IF e<=0 THEN 10
 20   x=INP(11)-INP(12)
 30   IF x>e THEN 100
 40   IF x<-e THEN 200
 50   OUT 10,3
 60   IF ABS(INP(11) - INP(12))<=e THEN 60
 70   OUT 10,0
 80 GOTO 20
100   OUT 10,2
110   IF INP(11)-INP(12)>e THEN 110
120   OUT 10,0
130 GOTO 20
200   OUT 10,1
210   IF INP(12) - INP(11)>e THEN 210
220   OUT 10,0
230 GOTO 20
```

9.1. Hogyan mozog a játékautó a padlón, ha az álló fényforrást a fényérzékelőkkel egy magasságba helyezzük el?

9.2. Mi az e változó szerepe?

9.3. Álló, illetve mozgó fényforrás mellett milyen esetben lehetnek gondok a program működésével?

Elérhető összpontszám: 75 pont

1988. Második forduló

Első-ötödik osztályosok

Mini-LOGO:

Készíts egy grafikus nyelvhez értelmező programot BASIC vagy PASCAL nyelven! A program legyen képes beolvasni az ezen a nyelven írt programszöveget (forrásprogramot), majd értelmezni és végrehajtani azt! Egy programsorban az utasítást a paraméterétől és egyik utasítást a másiktól szóközzel kelljen elválasztani! Forrásprogram végét a V utasítás jelzi (lásd lejjebb), ennek beolvasása indítja el – automatikusan – az értelmezést. A forrásprogram végrehajtása során keletkezett ábra annak lefutása után is maradjon látható mindaddig, amíg a felhasználó nem jelzi (egy tetszőleges billentyű lenyomásával), hogy az értelmező alapállapotba kerülhet, vagyis újabb grafikus program befogadására álljon készen! A forrásprogram elindulása előtt maga az értelmező gondoskodjon a képernyő letöréséről!

Az értelmezendő utasítások és leírásuk:

M <hossz> (Menj)	A ceruzát, az aktuális helyről, az aktuális irányba mozdítsa el <hossz>-nyit! (1 egység legyen a gép legkisebb grafikus egysége!) Ha kell, akkor a mozgás nyoma látszódjon a képernyőn! (lásd N,R parancsok!)
F <szög> (Fordulj)	Az aktuális helyen maradva az új haladási irány legyen a régi irány és <szög> összege! (A <szög> előjeles mennyiség, pozitív irány az óra járásának megfelelő irány.)
N (Nerajzolj)	Az utasítás kiadása után a következő R utasításig a ceruza mozgásának nyoma ne látszódjon!
R (Rajzolj)	A következő N utasításig lehessen a képernyőn követni a ceruza mozgását!
T (Törölj)	Hatására törlődjön a képernyő, de a ceruza aktuális helyzete (hely, irány, és hogy rajzol-e vagy sem) ne változzon!
H (Haza)	Hatására a ceruza "kerüljön" az alapállapotba → hely:= képernyő közepe, irány:= függőlegesen felfelé, és rajzoló üzemmódban legyen! (A hazamozgás során természetesen rajzoljon, ha rajzoló üzemmódban van!)
V (Vége)	Ez az utasítás jelezze a forrásprogram végét az értelmező számára! Ezután a beírt program azonnal hajtódjon végre!
I <szám> (<utasítás-sor>) (Ismételd)	Ez az utasítás tegye lehetővé, hogy az <utasítás-sor>-ban leírt teendők <szám>-szor végrehajtódjanak!
E <név> (<utasítás-sor>) (Eljárás)	Ezzel az utasítással tudjunk az itt felsorolt utasításokból definiálni <név> névvel egy új, a többivel egyenrangú utasítást! (Ezeket aktivizálni a <név> leírásával lehessen!)
D <név> (Dobdel)	Ezzel lehessen a már feleslegessé vált, ÁLTALUNK definiált eljárásokat törölni, hogy ezáltal hely szabaduljon fel az újabbak számára!

FONTOS!

- A kész program mellé be kell adni a program tervét! (Fontosabb szubrutinok és a változók leírása!)
- ELŐSZÖR a vezérlő programrészt írd meg, majd – ajánljuk, hogy – az utasításokat a fenti sorrendben valósítsd meg!
- Bármilyen gépet és bármilyen nyelvet is használsz, a rajzoláshoz a programodban csak az egy pont rajzolásához és a képernyő törléséhez szükséges utasítást használhatod fel! (Ebből következik, hogy a szakaszrajzoló rutint is neked kell megírni!) Ha az általad használt gépen vagy nyelvben nincs grafika, akkor a legkisebb egység nálad – természetesen – a betűméret lesz!

Bővítési lehetőségek:

- Ha az eddigi feladatokat megoldottad, akkor próbáld meg úgy módosítani a programodat, hogy jelezze a felhasználónak a szintaktikus hibákat, majd a futás közbeni hibákat is!
- Ha ezt is túl kevésnek találod, akkor az eddigiek elkészítése után nekiláthatsz az utasítások neveinek bővítéséhez oly módon, hogy az egybetűs parancs alatt zárójelben levő szavakkal lehessen utasításokat beírni.

Mintaprogramok:

```
1. M 20 F 60 M 20 F 60 M 20 V
2. T I 10 ( R M 20 F 18 N M 20 F 18 ) V
3. E TETO ( F 45 M 60 F 90 M 60 ) TETO V
```

Elérhető összpontszám: 84 pont

A verseny végeredménye:

1. Biczó Tibor	Zrínyi Miklós Gimnázium, Zalaegerszeg
2. Gárdonyi Gergely	József Attila Gimnázium, Budapest
3. Nagy Bálint	Ságvári Endre Gimnázium, Budapest
4. Czabala Péter	Kun Béla Gimnázium, Leninváros
5. Borha Zoltán	Árpád Gimnázium, Budapest
6. Károlyi Zoltán	Radnóti Miklós Gimnázium, Budapest
Tuba Imre	Vajda János Gimnázium, Keszthely
8. Boros Péter	Radnóti Miklós Gimnázium, Szeged
9. Fehér Csaba	Fazekas Mihály Gimnázium, Debrecen
Árvai László	Zalka Máté Szakközépiskola, Miskolc

1989. Első forduló

Első-ötödik osztályosok

1. feladat: (3 pont)

Milyen számot ír ki a képernyőre a következő program? Indokold meg!

```
10 I=0
20 GOSUB 50
30 PRINT I
40 STOP
50 GOSUB 60
60 GOSUB 70
70 GOSUB 80
80 GOSUB 90
90 GOSUB 100
100 GOSUB 110
110 I=I+1
120 RETURN
```

2. feladat: (8 pont)

Hol vannak hibák az alábbi BASIC szubrutinban? Javítsd ki! Milyen állítást fogalmazhatsz meg a változók tartalmáról és a T$ vektor rendezettségéről a megjelölt helyeken (a helyes változatban)? A szubrutin a T$() N elemű vektort rendezné növekvőn.

A DO WHILE ... LOOP ciklus magja mindaddig végrehajtódik, amíg a WHILE mögé írt feltétel teljesül.

```
1000 REM BESZÚRÁSOS RENDEZÉS:
1010 FOR I=2 TO N
1020    IS$=T$: J=I-1
   <------------------------------------------------------------ 1.
1030    DO WHILE J>=1 OR IS$<T$(J)
1040       T$(J)=T$(J+1): J=J-1
   <------------------------------------------------------------ 2.
1050    LOOP
1060    T$(J)=IS$
   <------------------------------------------------------------ 3.
1070 NEXT I
1080 RETURN
```

3. feladat: (6 pont)

Mit számít ki az alábbi program, ha az x változó (0<x<1) legfeljebb 2 tizedes törtjegyet tartalmaz? Milyen feltételek teljesülése esetén fejeződik be a végrehajtás? Mit ír ki a program x=0.3 esetén?

```
i:=0; w:='0.';
kiír('x:='); beolvas(x);
r[0]:=kerekít(x*100);
repeat
  i:=i+1;
  r[i]:=r[i-1]*2;
  if r[i]>=100 then begin r[i]:=r[i]-100; w:=w+'1'; end
               else w:=w+'0';
  j:=0;
  while (r[i]<>0) and (r[j]<>r[i]) and (j<i) do j:=j+1;
until not ((i<30) and (j=i));
kiír('Végeredmény:  ',w);
```

4. feladat: (6 pont)

Mit csinál az alábbi algoritmus? Az algoritmus egy T$(N,2)-es táblázattal és a K1$, K2$ változókkal dolgozik. Az index függvény egy szöveg karakterei alapján 1 és N közötti egész számot generál. (az 'amíg' ciklus akkor áll le, ha az amíg mögé írt feltétel már nem teljesül.)

```
Eljárás:
  K:=index(K1$): KK:=0
  Ha T$(K,1)≠"" akkor
     KK:=K
     Ciklus
        Ha K<N akkor K:=K+1 különben K:=1
     amíg KK≠K és T$(K,1)≠""
     Ciklus vége
  Elágazás vége
  Ha KK=K akkor Ki: "baj!!!"
        különben T$(K,1):=K1$: T$(K,2):=K2$
Eljárás vége.
```

5. feladat: (6 pont)

Az alábbi algoritmus egy utcai automata pénzvisszaadásának menetét írja le úgy, hogy a lehető legkevesebb számú érmét adja vissza. Az FT() vektor írja le, hogy az automata milyen érméket kezel, értéke (20,10,5,2,1).

```
Pénzvisszaadás(ÖSSZEG):
  Ciklus I=1-től 5-ig
     Ha ÖSSZEG≥FT(I) akkor DB:=INT(ÖSSZEG/FT(I))
                           Ki: DB;" db. ";FT(I);" Ft-os"
                           ÖSSZEG:=ÖSSZEG-DB*FT(I)
     Elágazás vége
  Ciklus vége
Eljárás vége.
```

Módosítsd a visszaadás algoritmusát, ha azt is tudjuk, hogy az automatának melyik pénzérméből hány darabja van, amit egy D() vektorban tárolunk!

6. feladat: (7 pont)

Mit rajzol az alábbi, LOGO nyelven írt program, ha teknőcünk kezdetben a képernyő közepén tartózkodik és felfelé néz? Mi a GVONAL eljárás szerepe? Mit tartalmaz a paramétere?

```
RIGHT 90 ÁBRA 30
```

Az ÁBRA és a GVONAL eljárások LOGO definíciója:

```
TO ÁBRA :X
  IF :X>9 THEN GVONAL :X LEFT 180 ÁBRA :X*2/3 LEFT 180
  GVONAL :X
END

TO GVONAL :Y
  REPEAT 180 [LEFT 1 FORWARD :Y*3.141592654/180]
END
```

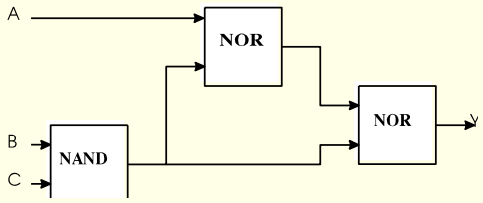
Az utasítások jelentése:

`RIGHT fok, LEFT fok`	jobbra, balra fordulás fok fokkal
`FORWARD lépés`	elmozdulás előrelépés hosszan
`REPEAT db [utasítások]`	az utasítások megismétlése db-szer
IF	a THEN ág a sor végéig tart

Az utasításokat egymástól, illetve a paraméterüktől is szóköz választja el. Az eljárások paraméterei elé :-ot kell tenni, az eljárásdefiníció kezdetét TO, végét az END alapszó jelzi.

7. feladat: (8 pont)

Egy elektronikus áramköröket gyártó vállalatnak digitális áramkörök hibás kapuit kell megkeresnie. Előzetes tapasztalatok alapján feltételezhető, hogy az ábrán látható áramkörben legfeljebb egy hiba van, és az úgy jelentkezik, hogy valamelyik kapu kimenete a bemeneteitől függetlenül állandóan logikai 0-t ad ki (0-ba ragadt). Ennek megállapításához a G1, G2, G3 kapukból álló áramkör A, B, C bemeneteit a számítógép 63-as kimenetének 0., 1., 2. bitjére kötöttük, az áramkör Y kimenetét pedig a 64-es bemenet 0. bitjére:



Az egyes kapuk igazságtáblázata (NOR =nem vagy, NAND= nem és):

NOR	0	1
0	1	0
1	0	0

NAND	0	1
0	1	1
1	1	0

A következő programmal próbáljuk felfedni a hibákat:

```
10 REM HIBAKERESÉS
20 FOR I=1 TO 3
30  READ X,Y,H$
40  OUT 63,X          : REM X KIKÜLDÉSE A 63-AS KIMENETRE
50  IF (INP(64) AND 1)<>Y THEN 80: REM A 64-ES BEMENETEN NEM Y JÖTT
60 NEXT I
70 H$="NINCS 0-BA RAGADT KIMENET"
80 PRINT H$
90 STOP
100 DATA ?,?,"G1 HIBÁS"
110 DATA ?,?,"G2 HIBÁS"
120 DATA ?,?,"G3 HIBÁS"
```

Milyen számok kerüljenek a DATA sorokban levő kérdőjelek helyére?

8. feladat: (8 pont, problémánként 1-1 pont)

Egy liftet vezérlő program algoritmusát láthatod az alábbiakban. Gondold végig, hogyan működik ez a lift (ezt nem kell leírnod)! Sorolj fel minél több kritikus (kellemetlen) helyzetet, amelybe a lift utasa kerülhet az algoritmus miatt! A HOVA() vektor 32 elemű. (Hívni a liftet valamelyik emeletről lehet, parancsot adni pedig a fülkében.)

```
Liftvezérlő program:
  SZINT:=0: UTASÍTÁSSZÁM:=0
  Ciklus
     Várj amíg nincs HÍVÁS
     Jegyezd föl(HÍVÓGOMB)
     Ciklus amíg UTASÍTÁSSZÁM>0
        Menj oda(HOVA(UTASÍTÁSSZÁM))
        Ajtót nyiss
        Várj amíg eltelik 10 másodperc
        Ajtót csukj
        UTASÍTÁSSZÁM:=UTASÍTÁSSZÁM-1
     Ciklus vége
  Ciklus vége
Program vége.

Menj oda(x):
  Ciklus amíg SZINT<>X
     Ha SZINT>X akkor  Menj le eggyel: SZINT:=SZINT-1
              különben Menj föl eggyel: SZINT:=SZINT+1
     Ha van PARANCS akkor Jegyezd föl(PARANCSGOMB)
     Ha van HÍVÁS akkor Jegyezd föl(HÍVÓGOMB)
  Ciklus vége
Eljárás vége.

Jegyezd föl(GOMB):
  Ciklus I=UTASÍTÁSSZÁM-tól 1-ig -1-esével
     HOVA(UTASÍTÁSSZÁM+1):=HOVA(UTASÍTÁSSZÁM)
  Ciklus vége
  UTASÍTÁSSZÁM:=UTASÍTÁSSZÁM+1; HOVA(1):=GOMB
Eljárás vége.
```

A Várj amíg ..., a Menj ..., az Ajtót ... tevékenységek végrehajtására, a PARANCS-, illetve HÍVÓGOMB érzékelésére, valamint a PARANCS és HÍVÁS logikai értékek beállítására a lift közvetlenül képes, azokat nem kell definiálnunk.

Elérhető összpontszám: 52 pont

1989. Második forduló

Első-ötödik osztályosok

Áramkörszimuláció:

Készíts programot egy digitális áramkör működésének szimulálására! Az áramkör működésében feltételezzük, hogy minden alapelem kimenetén egységnyi idő múlva jelenik meg a bemeneti jelkombináció hatása, és ez azonnal megjelenik a következő alapelemek bemenetén is. (Azaz az információ a vezetékeken 0, a kapukon egységnyi idő alatt halad át.)

Az áramkör alapelemei a következők lehetnek: INVERTER (NOT), AND, OR, NAND, NOR, EXOR kapuk. (Az inverternek 1, a többi kapunak pedig 2 bemenete van.) Az áramkörben használható kapuk száma maximum 10 lehet. Kezdetben minden kapu minden bemenetén valamilyen határozott érték (például 0) van.

Az áramkör leírását úgy adjuk meg, hogy az alapelemek bemeneteihez, illetve kimenetéhez rendelünk egy csomópont-azonosító számot, s az azonos csomóponthoz tartozó be-, illetve kimeneteket tekintjük összekötöttnek (lásd az ábrát).

A szimulálandó hálózatról, áramkörről tegyük fel, hogy véges időn belül kialakul valamilyen stabil kimeneti jelkombináció (például ha N kapu van az áramkörben, akkor ez a korlát legyen N^2)!

Az egyes kapuk működésének leírása:

NOT	
0	1
1	0

OR	0	1
0	0	1
1	1	1

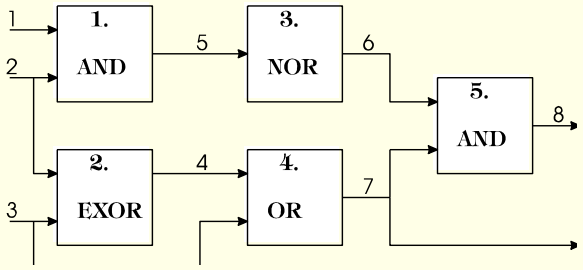
AND	0	1
0	0	0
1	0	1

EXOR	0	1
0	0	1
1	1	0

NOR	0	1
0	1	0
1	0	0

NAND	0	1
0	1	1
1	1	0

Példa (programod kipróbálásához célszerű ennél egyszerűbb áramköröket választanod):
Az áramkör rajza :



A példaáramkör leírása:

1. AND: 1, 2, 5 2. EXOR: 2, 3, 4 3. NOT: 5, 6 4. OR: 4, 3, 7 5. AND: 6, 7, 8	Az áramkör bemenetei: 1, 2, 3. Az áramkör kimenetei: 7, 8
Bemeneti kombinációk:	Kimenetek:
1,1,1 0,0,0 0,0,1	1,0 0,0 1,1

Feladatok:

A: Add meg, hogyan ábrázolod a memóriában az áramkört!

B: Olvasd be az áramkör leírását tetszőleges, általad választott módon, s ebből építsd fel az áramkört az általad megválasztott adatszerkezet alapján!

C: Olvass be bemeneti jelkombinációkat és az áramkör működésének szimulálásával határozd meg az ezekre adott kimenetet! (A kimenetet akkor tekintjük késznek, amikor az áramkör minden alapelemének kimenete változatlan marad.)

D: Állítsd elő automatikusan az összes bemeneti jelkombinációt, és az áramkör működésének szimulálásával határozd meg az ezekre adott kimenetet!

E: Ellenőrizd, hogy az áramkör leírását helyesen adták-e meg!

A lehetséges hibák:

- egy kapu kimenetét sehova sem kötötték, és az nem is az áramkör kimenete,
- egy kapu bemenetét sehova sem kötötték, és az nem is az áramkör bemenete,
- két (vagy több) kapu kimenetét összekötötték,
- van olyan kapu a hálózatban, amelyet a bemenetről nem lehet elérni,
- az áramkör valamelyik kimenete nem valamelyik kapu kimenete,
- az áramkör valamelyik bemenetét nem kötötték egy kapu bemenetére sem,
- a korlátnak állított időegységszám után sem alakul ki stabil kimenete az áramkörnek.

Elérhető összpontszám: 100 pont

A verseny végeredménye:

1. Szabó Dániel	Árpád Gimnázium, Budapest
2. Biczó Tibor	Zrínyi Miklós Gimnázium, Zalaegerszeg
3. Ladányi József	Árpád Gimnázium, Budapest
4. Dedesi Péter	Földes Ferenc Gimnázium, Miskolc
5. Oravecz Róbert	Földes Ferenc Gimnázium, Miskolc
6. Maróti Miklós	Radnóti Miklós Gimnázium, Szeged
7. Késmárki Mátyás	Katona József Gimnázium, Kecskemét
8. Berendi Péter	Fazekas Mihály Gimnázium, Budapest
9. Szentesi Áron	Árpád Gimnázium, Budapest
10. Tornyi Lajos	Fazekas Mihály Gimnázium, Budapest

1990. Első forduló

Első-második osztályosok

1. feladat: (4 pont)

A következő algoritmus egy kifejezés zárójelezésének helyességét ellenőrzi. Milyen hibajelenségek tartoznak az egyes HIBAx utasításokhoz?

```
Ellenőrzés(X$):
  DB:=0
  Ciklus I=1-től X$ hosszáig
     Y$:=X$ I. betűje
     Ha Y$="("  akkor   DB:=DB+1
              különben  Ha Y$=")" akkor DB:=DB-1
     Ha DB<0 akkor HIBA1
  Ciklus vége
  Ha DB>0 akkor HIBA2
Eljárás vége.
```

2. feladat: (15 pont)

A következő két algoritmus egy labirintusból vezet ki:

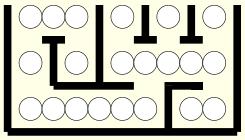
```
A-Labirintus:
  Ciklus amíg nem ért ki
    Balrafordulás
    Ciklus amíg fal van előtte
      Jobbrafordulás
    Ciklus vége
    Előrelépés
  Ciklus vége
Eljárás vége.
```

```
B-Labirintus(X,Y):
  Ciklus amíg nem ért ki
    Ha szabad(X,Y-1)
       akkor Y:=Y-1
    különben ha szabad(X-1,Y) akkor X:=X-1
  Ciklus vége
Eljárás vége.
```

Az ábrán egy labirintus látható, amelynek kijárata a bal felső sarokban van.

Másold le két példányban, és jelöld meg benne x-szel azokat a helyeket, amelyekről elindulva előfordulhat, hogy a fenti A és B algoritmusok nem vezetnek ki a labirintusból!

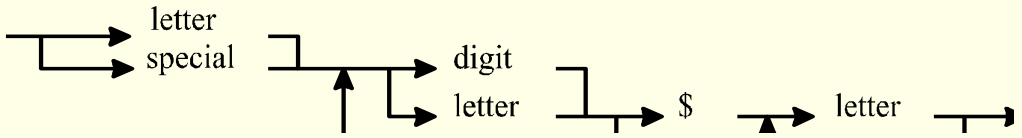
Milyen tulajdonságú labirintusokból, illetve pontokból vezet ki biztosan A, illetve B?



3. feladat: (10 pont)

Az alábbi szintaxisábrák alapján döntsd el, hogy szintaktikailag melyik karaktersorozat helyes és melyik hibás! Magyarázd meg a válaszaidat!

```
'letter':   bármely kisbetű 'a' és 'z' között,
'digit' :   bármely számjegy '0' ' és '9' között,
'special':  az aláhúzás (_) vagy a hátratört-vonal (\).
```



a) april$ b) _apr$a c) apRIL
e) \\apr$a d) \apr$aa$a f) apr4$aa

4. feladat: (13 pont)

A következő kártyakeverő programban a megjelölt helyen 4 különböző programrész állhat:

```
FOR I:=1 TO 32 DO LAP[I]:=I;
FOR I:=1 TO 32 DO
  BEGIN
     (* *)
     KIRAJZOL(J)
  END
```

A négy beilleszthető rész a következő:

```
(*1*) J:=VÉLETLEN(32);

(*2*) REPEAT
         J:=VÉLETLEN(32)
      UNTIL LAP[J]>0;
      LAP[J]:=0;

(*3*)K:=VÉLETLEN(32);J:=LAP[K];
     LAP[K]:=LAP[I];LAP[I]:=J;

(*4*) K:=VÉLETLEN(33-I)+I-1;
      J:=LAP[K];
      LAP[K]:=LAP[I];LAP[I]:=J;
```

A program egy 32 lapból álló, megkevert pakli lapjait rajzolja ki. Az egyes kártyalapokat az 1, 2, ..., 32 számok képviselik, a KIRAJZOL eljárás egy lapot rajzol ki a képernyőre, a VÉLETLEN(SZÁM) függvény pedig egy 0-nál nagyobb és SZÁM-nál nem nagyobb véletlen egész számot állít elő.

Szabályosnak akkor számít a keverés, ha minden lap egyszer és csak egyszer fordul elő, mégpedig bármelyik helyen azonos eséllyel.

A: Melyik algoritmus kever szabályosan és melyik nem?

B: A szabálytalanul keverő változatokról írd le, hogy mit csinálnak!

C: Tudsz-e valamit mondani arról, hogy a (*2*) változat a FOR-ciklus magjának egy-egy végrehajtása során hányszor hívja meg a VÉLETLEN() függvényt?

5. feladat: (7 pont)

Adott a következő BASIC nyelvű program:

```
10 DIM D(300): FOR I=1 TO 300: D(I)=0: NEXT I
20 INPUT "HÁNY SZÁM LESZ" ; N
30 FOR I=1 TO N
40   INPUT J: IF J<>INT(J) OR J<1 OR J>300 THEN PRINT
"HIBÁS!": GOTO 40
50   D(J)=D(J)+1
60 NEXT I
70 FOR I=1 TO 300
80   IF D(I)>0 THEN FOR J=1 TO D(I): PRINT I: NEXT J
90 NEXT I
```

Mit csinál a program a beadott számokkal (egy mondatban, a lényeget)?

6. feladat: (8 pont)

Számítógéppel titkosítottak egy értelmes magyar SZÓ-t. A program lefutása után közölték velünk az előállított KÓD-ot, de a használt KULCS-ot nem. A kódoláskor használt algoritmus a következő volt:

```
Kódolás:
  KÓD(0):=KULCS
  Ciklus I=1-től HOSSZ-ig
     KÓD(I):=SZÓ(I) művelet KÓD(I-1)
  Ciklus vége
Eljárás vége.
```

Az 1-től HOSSZ-ig indexelt SZÓ tömb a kódolandó, a 0-tól HOSSZ-ig indexelt KÓD tömb pedig a kódolt szót tartalmazza.

A KÓD (1-től HOSSZ-ig) ez lett: CJNKL. Az alábbi igazságtáblával megadott műveletet a számítógép az operandusokon bitenként végzi el (A és B az operandusok, C az eredmény).

Művelet:

A	0	1	0	1
B	0	0	1	1
C	0	1	1	0

A használható betűk és kódjaik:

-	0000	A	0001	B	0010	C	0011
D	0100	E	0101	F	0110	G	0111
H	1000	I	1001	J	1010	K	1011
L	1100	M	1101	N	1110	O	1111

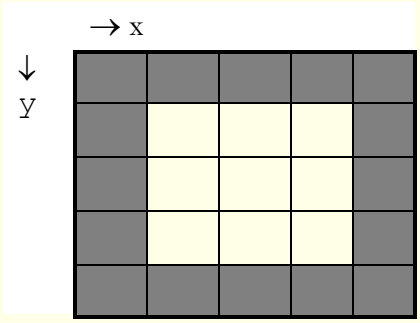
Mi volt az eredeti, értelmes magyar szó (megjegyzés: az első betű kitalálásához az algoritmus nem ad segítséget), és mi volt a KULCS? Magyarázd meg, hogyan jöttél rá a megoldásra!

Elérhető összpontszám: 57 pont

Harmadik-ötödik osztályosok

1. feladat: (15 pont)

Az alábbi algoritmusok a képernyő egy tetszőleges vonallal határolt részének befestésére alkalmasak. Melyik eljárás milyen sorrendben fest be egy bekerített 5x5-ös négyzetet a közepéről kiindulva, ha a szélei már be vannak festve (9 befestetlen pontja van,)?




```
A: Festés(x,y):
     Pontrajz(x,y)
     Ha üres(x-1,y) akkor Festés(x-1,y)
     Ha üres(x,y-1) akkor Festés(x,y-1)
     Ha üres(x+1,y) akkor Festés(x+1,y)
     Ha üres(x,y+1) akkor Festés(x,y+1)
   Eljárás vége.
```

```
B:Festés(x,y):
     Pontrajz(x,y): Sorba(x,y)
     Ciklus amíg Sor nem üres
       Sorból(x,y)
       Ha üres(x-1,y) akkor Pontrajz(x-1,y): Sorba(x-1,y)
       Ha üres(x,y-1) akkor Pontrajz(x,y-1): Sorba(x,y-1)
       Ha üres(x+1,y) akkor Pontrajz(x+1,y): Sorba(x+1,y)
       Ha üres(x,y+1) akkor Pontrajz(x,y+1): Sorba(x,y+1)
     Ciklus vége
   Eljárás vége.
```

```
C: Festés(x,y):
     Pontrajz(x,y): Verembe(x,y)
     Ciklus amíg Verem nem üres
       Veremből(x,y)
       Ha üres(x-1,y) akkor Pontrajz(x-1,y): Verembe(x-1,y)
       Ha üres(x,y-1) akkor Pontrajz(x,y-1): Verembe(x,y-1)
       Ha üres(x+1,y) akkor Pontrajz(x+1,y): Verembe(x+1,y)
       Ha üres(x,y+1) akkor Pontrajz(x,y+1): Verembe(x,y+1)
     Ciklus vége
   Eljárás vége.
```

Sor: Olyan adatszerkezet, amelynek a végére lehet új elemet betenni, illetve az elején állót lehet kivenni.

Verem: Olyan adatszerkezet, amelynek a végére lehet új elemet betenni, illetve az utoljára betettet lehet kivenni.

2. feladat: (6 pont)

Minek a közelítő értékét határozza meg a következő programrészlet?

```
100 S=0
110 FOR I=1 TO 1000
120  X=2*RND(1)-1: Y=2*RND(1)-1
130  IF X*X+Y*Y<1 THEN S=S+1
140 NEXT I
150 PRINT S/250
```

3. feladat: (11 pont)

Az a kijelentés, hogy "az egyszerű mondat alanyból és állítmányból áll", formális szabályokkal például így írható le:

```
(1)   <egyszerű mondat> ::= <alany> <állítmány>
(2)   <egyszerű mondat> ::= <állítmány> <alany>
```

Megadunk néhány további szabályt is:

```
(3)  <alany>       ::= <névelő> <főnév>
(4)  <alany>       ::= <jelző> <főnév>
(5)  <alany>       ::= <főnév>
(6)  <állítmány>   ::= <jelző>
(7)  <állítmány>   ::= <jelző> vagyok

(8)  <névelő>      ::= a
(9)  <főnév>       ::= Pista
(10) <főnév>       ::= tengeralattjáró
(11) <főnév>       ::= fiú
(12) <jelző>       ::= okos
(13) <jelző>       ::= sárga
```

E szabályok alapján az alábbi mondatok közül melyek helyesek és melyek hibásak? Válaszodban sorold föl azokat a szabályokat, amelyek alapján az egyes mondatok elfogadhatók!

A: Pista okos
B: Pista okos fiú
C: a tengeralattjáró sárga
D: sárga a tengeralattjáró
E: okos vagyok

4. feladat: (10 pont)

A következő algoritmus egy kifejezés zárójelezésének helyességét ellenőrzi. A kifejezésben gömbölyű és szögletes zárójelek is lehetnek. Milyen hibajelenségek tartoznak az egyes HIBAx utasításokhoz?

```
Ellenőrzés(X$):
  DB1:=0: DB2:=0
  Ciklus I=1-től hossz(X$)-ig
     Y$:=X$ I. betűje
     Ha Y$="(" akkor DB1:=DB1+1: Verembe(Y$)
     Ha Y$="[" akkor DB2:=DB2+1: Verembe(Y$)
     Ha Y$=")" akkor Ha DB1=0 akkor HIBA1
                     különben DB1:=DB1-1: Veremből(Z$)
                              Ha Z$="[" akkor HIBA2
                     Elágazás vége
```

```
     Ha Y$="]" akkor Ha DB2=0 akkor HIBA3
                     különben DB2:=DB2-1: Veremből(Z$)
                              Ha Z$="(" akkor HIBA4
                     Elágazás vége
  Ciklus vége
  Ha DB1>0 akkor HIBA5
  Ha DB2>0 akkor HIBA6
Eljárás vége.
```

5. feladat: (8 pont)

Számítógéppel titkosítottak egy értelmes magyar SZÓ-t. A program lefutása után közölték velünk az előállított KÓD-ot, de a használt KULCS-ot nem. A kódoláskor használt algoritmus a következő volt:

```
Kódolás:
  KÓD(0):=KULCS
  Ciklus I=1-től HOSSZ-ig
     KÓD(I):=SZÓ(I) művelet KÓD(I-1)
  Ciklus vége
Eljárás vége.
```

Az 1-től HOSSZ-ig indexelt SZÓ tömb a kódolandó, a 0-tól HOSSZ-ig indexelt KÓD tömb pedig a kódolt szót tartalmazza.

A KÓD (1-től HOSSZ-ig) ez lett: HIKJFG.

Az alábbi igazságtáblával megadott műveletet a számítógép az operandusokon bitenként végzi el (A és B az operandusok, C az eredmény).

Művelet:

A	0	1	0	1
B	0	0	1	1
C	0	1	1	0

A használható betűk és kódjaik:

-	0000	A	0001	B	0010	C	0011
D	0100	E	0101	F	0110	G	0111
H	1000	I	1001	J	1010	K	1011
L	1100	M	1101	N	1110	O	1111

Mi volt az eredeti, értelmes magyar szó (megjegyzés: az első betű kitalálásához az algoritmus nem ad segítséget), és mi volt a KULCS?

Magyarázd meg, hogyan jöttél rá a megoldásra!

6. feladat: (16 pont)

A következő kártyakeverő programban a megjelölt helyen 4 különböző programrész állhat:

```
FOR I:=1 TO 32 DO LAP[I]:=I;
FOR I:=1 TO 32 DO
  BEGIN
     (* *)
     KIRAJZOL(J)
  END
```

A négy beilleszthető rész a következő:

```
(*1*)J:=VÉLETLEN(32);
```

```
(*2*)REPEAT
       J:=VÉLETLEN(32)
     UNTIL LAP[J]>0;
     LAP[J]:=0;
```

```
(*3*)K:=VÉLETLEN(32);J:=LAP[K];
     LAP[K]:=LAP[I]; LAP[I]:=J;
(*4*) IF I<32 THEN BEGIN
          K:=VÉLETLEN(32-I)+I;
          J:=LAP[K]; LAP[K]:=LAP[I]; LAP[I]:=J;
        END
        ELSE J:=LAP[I];
```

A program egy 32 lapból álló, megkevert pakli lapjait rajzolja ki. Az egyes kártyalapokat az 1, 2, ..., 32 számok képviselik, a KIRAJZOL eljárás egy lapot rajzol ki a képernyőre, a VÉLETLEN(SZÁM) függvény pedig egy 0-nál nagyobb és SZÁM-nál nem nagyobb véletlen egész számot állít elő. Szabályosnak akkor számít a keverés, ha minden lap egyszer és csak egyszer fordul elő, mégpedig bármelyik helyen azonos eséllyel.

A: Melyik algoritmus kever szabályosan, és melyik nem?

B: A szabálytalanul keverő változatokról írd le, hogy mit csinálnak!

C: Tudsz-e valamit mondani arról, hogy a szabályosan keverő változatok a FOR-ciklus magjának egy-egy végrehajtása során hányszor hívják meg a VÉLETLEN() függvényt?

7. feladat: (7 pont)

Adott a következő BASIC nyelvű program:

```
10 DIM D(300): FOR I=1 TO 300: D(I)=0: NEXT I
20 INPUT "HÁNY SZÁM LESZ" ; N
30 FOR I=1 TO N
40   INPUT J: IF J<>INT(J) OR J<1 OR J>300 THEN PRINT
"HIBÁS!": GOTO 40
50   D(J)=D(J)+1
60 NEXT I
70 FOR I=1 TO 300
80   IF D(I)>0 THEN FOR J=1 TO D(I): PRINT I: NEXT J
90 NEXT I
```

Mit csinál a program a beadott számokkal (egy mondatban, a lényeget)?

8. feladat: (14 pont)

A FONTOSKODÓ vállalat nyilvántartásában szerepel ez a 2 táblázat:

Alkalmazottak

azon	név	munkakör	főnöke	fizu	oszt
544	Álmos	szállító	545	7000	2
545	Éber	osztvez	550	9800	2
546	Élő	osztvez	550	15000	3
547	Dolgos	munkás	546	11000	3

550	Főfő	igazgató	NULL	20000	4
551	Buzgó	munkavéd	550	10500	2

	osztályok	
kód	osztnév	hely
2	szállítás	Pécs
3	termelés	Pécs
4	igazgatás	Győr

Lekérdezésükre használható a SELECT utasítás:

```
SELECT oszlopnév, oszlopnév, ...
   FROM táblázatnév, táblázatnév, ...
   WHERE logikai kifejezés;
```

A logikai kifejezésben állhat konstans (pl. "Pécs"), oszlopnév (pl. hely) vagy zárójelben egy újabb, beágyazott SELECT utasítás.

A lekérdezés azokat a sorokat (két táblázat esetén sorpárokat) választja ki a táblázat(ok)ból, amelyekre a logikai kifejezés teljesül. Minden kiválasztott sorból, illetve sorpárból csak a megnevezett oszlopokba eső adatokat kapjuk eredményül. A nem beágyazott SELECT utasítás eredménye kiíródik a képernyőre. Például

```
SELECT osztnév FROM osztályok WHERE hely = "Pécs" OR
                                    hely = "Győr";
```

kiírja a pécsi és a győri osztályok nevét.

A: Milyen parancs írja ki

A1: a 10000-nél többet kereső osztályvezetők nevét és fizetését?

A2: a Pécsett dolgozók nevét és osztályuk nevét?

B: Mit ír ki?

```
SELECT név
  FROM alkalmazottak
  WHERE főnöke = (SELECT azon
                  FROM alkalmazottak
                  WHERE név = "Főfő";
```

Elérhető összpontszám: 87 pont

1990. Második forduló

Első-második osztályosok

Metró:

Készíts programot egy metróállomás (ami nem végállomás) forgalmának szimulálására! A feladatot négy, egyre bővülő lépésben oldd meg!

A program kövesse nyomon a metróállomáson történő eseményeket, avatkozzon közbe, ha szükséges, és jelezze ki az egyes helyeken (lépcsők előtt, lépcsőkön, peronon, szerelvényekben) várakozók számát, a lépcsők állapotát, a beavatkozásokat! A működési paramétereket (TA, TE, ML stb.) a program a billentyűzetről olvassa be!

A. feladat:

A metrószerelvények két irányba mehetnek, az utasok egy közös peronról szállnak fel, illetve a metrókocsikból ide szállnak ki (mozgólépcső ebben a részfeladatban még nincs). A peronon, illetve a metrókocsikban tetszőleges számú utas tartózkodhat.

Megoldandó részfeladatok:

1. Az állomásra véletlenszerűen érkeznek az utasok, mindegyikük a két lehetséges irány valamelyike felé szeretne menni. Az irányt érkezéskor véletlenszerűen kell meghatároznia a programnak!

2. Az érkezők beállnak a megfelelő irányba menő metróra várók közé.

3. A metrószerelvények TE időközönként érkeznek (a két irányból nem feltétlenül egyszerre), TA ideig állnak az állomáson. TE, TA az időegység (l. alább) egész számú többszöröse.

4. Az érkező metrószerelvényben véletlenszerűen meghatározott számú utas van, s ezek egy része száll ki.

5. A metrószerelvény minden várakozót elvisz, aki abba az irányba akar menni. (A TA idő elég a le-, illetve felszállóknak.)

6. A leszállók elhagyják a peront.

B. feladat:

Az állomáson 2 mozgólépcső van, az egyik felfelé, a másik lefelé viszi az utasokat. Az utasok fent egy váróterembe érkeznek, a mozgólépcsők a várótermet kötik össze a lenti peronnal. Az előzőkhöz hasonlóan a váróteremben is tetszőleges számú utas tartózkodhat. A lépcső ML hosszúságú (azaz 2*ML utas fér rá).

Megoldandó részfeladatok:

7. Az állomásra érkezők a lefelé menő mozgólépcsőnél sorban állnak.

8. A lépcsőre egy időegység alatt maximum 2 ember léphet; ha van várakozó utas, az rá is lép a lépcsőre.

9. A lépcső egy időegység alatt egy lépcsőfoknyit halad lefelé (a másik lépcső felfelé).

10. A leszállók beállnak a felfelé menő lépcsőnél várakozók sorába.

11. Aki felért, az elhagyja a metróállomás területét.

C. feladat:

Az állomás helyiségeinek befogadóképessége adott, a leszállás és a felszállás időt vesz igénybe. A peronon egyszerre PDB ember fér el, a fenti váróteremben pedig VDB. A felfelé menő lépcső külön kijárathoz vezet, így a kifelé igyekvők nem növelik meg a fenti váróterem telítettségét. Egy szerelvényben maximum MDB ember tartózkodhat.

Megoldandó részfeladatok:

12. Ha a fenti váróterem megtelt, akkor az újonnan érkezők elmennek (pl. más járművel utaznak).

13. A metrószerelvény annyi utast visz el, amennyi még belefér a kocsikba (a többiek megvárják a következő szerelvényt).

14. Ha a peron a lefelé menők miatt megtelt, akkor a szimuláció álljon le!

15. Egy időegység alatt a metrószerelvényből SZ számú utas tud ki-, illetve beszállni. Amíg a kiszállók nem szálltak ki, addig senki sem száll be.

16. Ha a peronon nem férnének el a kiszállók, akkor a kocsikban maradnak.

D. feladat:

Az állomáson 3 mozgólépcső van, amelyekből kettő mindig működik, a harmadik pedig bekapcsolható valamelyik irányban, ha szükséges, illetve kikapcsolható, ha már nincs rá szükség.

Megoldandó részfeladatok:

17. A program felhasználója bármikor kezdeményezhesse a harmadik mozgólépcső elindítását, illetve leállítását (vészleállítás, esti kikapcsolás)!

18. A program szükség esetén automatikusan indítsa el a harmadik lépcsőt (ha a peronon vagy a váróteremben már sok utas tartózkodik).

19. Ha a peron lent tele van, akkor a program állítsa le egy időre a lefelé menő lépcső(ke)t! (A 14. pont emiatt érvényét veszti.)

Értékelési szempontok:

1. A programban megvalósított funkciók megbízható működése.

2. A megvalósított funkciók száma, sokrétűsége.

3. A program belső szerkezete: logikus tagolás, általános célú eljárások, bővíthetőség ...

4. A program olvashatósága: beszédes elnevezések, áttekinthető tördelés, magyarázatok.

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

Szóelválasztás:

Készíts programot, amely a billentyűzetről vagy egy szöveges állományból (file-ból) bevitt, szóköz vagy újsor karakterekkel tagolt, helyes magyar szavakat kötőjelekkel elválasztva (szótagolva) ír ki a képernyőre! A programnak csak az alábbi elválasztási szabályokat kell betartania; ne találj ki új szabályokat! A felsoroltak közül próbálj meg minél többet alkalmazni, de úgy, hogy a program működőképes legyen!

A program alapváltozata minden beolvasott szót külön sorban írjon ki a képernyőre; ha több lehetséges elválasztás közül nem tudja kiválasztani a helyeset, akkor egymás után írja ki az összes változatot! Ha van időd, fejleszd tovább a kezelői felületet úgy, hogy a beolvasott szöveg egyetlen változatban a képernyő tetején jelenjen meg, több lehetséges elválasztás esetén pedig a képernyő alján ajánlja fel kiválasztásra az egyes változatokat! Pl. fö-lül, föl-ül, me-gint, meg-int.

A bevitt szövegben az elválasztás helyét egyenlőségjellel (=) megjelölhetjük; ilyenkor a szót alkotó szórészeket mint önálló szavakat kell elválasztani a szótagolás szabályai szerint.

A program az elválasztás szempontjából ne tegyen különbséget kis- és nagybetűk között!

Az ékezetes magánhangzók jelölése: a', e', i', o', o:, o'', u', u:, u''.

Az egyjegyű hosszú mássalhangzók: bb, cc, dd, ff, gg, hh stb.

A többjegyű rövid mássalhangzók: cs, dz, dzs, gy, ly, ny, sz, ty, zs.

Az (egyszerűsítetten kettőzött) többjegyű hosszú mássalhangzók: ccs, ddz, ddzs, ggy, lly, nny, ssz, tty, zzs.

Elválasztási szabályok (Forrás: A magyar helyesírás szabályai, 1984.)

Az egyszerű szavak elválasztásának alapja a szótagolás.

1. Minden szó annyi szótagú, ahány magánhangzó van benne. Magánhangzó önmagában is alkothat szótagot. A szótag vagy magánhangzóval kezdődik, vagy – az elsőt kivéve – egyetlen rövid mássalhangzóval. Az egyetlen magánhangzót tartalmazó, valamint a háromnál nem több hangból álló szavakat nem választjuk el. Pl. bál, György, sztrájk, te-ker-vé-nyes, trom-bi-ta (de: Leó, szia, apu).

2. A két magánhangzó között lévő egyetlen mássalhangzót – akár egy-, akár többjegyű a betűje – a következő szótagba visszük át. Pl. be-tű, ba-tyu, bo-dza, apá-mé, né-gyet, bri-dzsel.

3. A két magánhangzó között lévő kétféle rövid mássalhangzót jelölő (azaz nem kettőzött) betűk közül – akár egy-, akár többjegyűek – az elsőt az előző szótagban hagyjuk, a másodikat pedig átvisszük a következőbe. Így járunk el akkor is, ha a két mássalhangzó közül az első hosszú (azaz kettőzött betű jelöli). Pl. am-per, fosz-fát, mor-zsa, ta-risz-nya, tal-pa-lat-nyi, bron-zot, töl-gyes, hall-gat, könny-től.

4. Ha a két magánhangzó közötti mássalhangzó hosszú (azaz kettőzött betű jelöli), ennek egyik jegyét az előző szótagban hagyjuk, másik jegyét pedig átvisszük a következőbe. Pl. ber-reg, mil-li-mé-ter, em-ber-rel, hal-lak, job-ban, víz-zé.

5. Ha a két magánhangzó közötti hosszú mássalhangzót egyszerűsítetten kettőzött többjegyű betű jelöli, ennek elválasztásakor mind az előző szótag végén, mind a következő szótag elején ki kell írni a teljes rövid megfelelőjét. Pl. meny-nyi, ösz-sze, ágy-gyal, ész-szerű, megy-gyel.

6. Ha két magánhangzó között kettőnél több mássalhangzót jelölő betű van, csak az utolsó kerül a következő szótagba. Pl. cent-rum, ost-rom, nyolc-kor, bors-sal, vonz-za, kulcs-csal, metsz-szük, edz-dze, bridzs-dzsel.

7. A ch kétjegyű ugyan, de sokszor egyetlen hangot jelöl, az x pedig két hang jele, de egyetlen jegyű, s így mindkettő egy betűnek számít az elválasztáskor. Úgy tekintjük őket, mint a rövid mássalhangzókat jelölő magyar betűket. Pl. or-chi-dea, pszi-cho-ló-gus, pra-xis, tech-ni-ka (de: srác-hoz).

8. A ch és x végződésű szavak -val, -vel és -vá, -vé ragos alakjait a következő példák szerint kell elválasztani: pech-hel, bó-rax-szá, Bach-hal.

Az igekötős szavak és a leg-képzős melléknevek kivételével összetett szavakkal a programnak nem kell foglalkoznia. Ha a feldolgozandó szövegben összetett szó fordul elő, akkor az összetétel helyét egyenlőségjellel jelöljük meg. Az összetett szavakat alkotó tagok már a szótagolás szabályai szerint választandók el.

9. Ha a kötőjellel írt szóösszetételt a szóhatáron szakítjuk meg, a kötőjelet csak egyszer kell kitenni. Pl. égre-földre, dúl-fúl.

10. A felsőfokú melléknevek leg- képzője önálló szótag. Pl. leg-jobb, leg-job-ban, leg-in-kább (de: le-gen-da).

11. A mássalhangzóra végződő igekötő önálló szótagképző: agyon-, át-, benn-, el-, el-len-, fel-, föl-, fe-lül-, fö-lül-, fenn-, fönn-, ke-resz-tül-, kü-lön-, meg-, széj-jel-, szét-, to-vább-, túl-, vé-gig-. Pl. meg-int, fel-pró-bál.

12. Külön vizsgálandók a magánhangzóra végződő igekötők (ab-ba-, alá-, be-, be-le-, egy-be-, elő-, elő-re-, fél-be-, fél-re-, hát-ra-, ha-za-, hely-re-, hoz-zá-, ide-, ket-té-, ki-, köz-be-, köz-re-, le-, mellé-, ne-ki-, oda-, ösz-sze-, rá-, raj-ta-, szem-be-, vég-be-, visz-sza-), ha utánuk két vagy több mássalhangzó áll. Pl. ki-pró-bál, be-ska-tu-lyáz (de: ber-zen-ke-dik).

13. Külön vizsgálandók az alábbi képzőkre, ill. toldalékokra végződő szavak: -ság, -ség, -szerű, -szer, -szor, -ször. Pl. malac-ság, köz-ség, száz-szor (de ésszerű: ész-sze-rű).

Esztétikai kívánalmak:

14. Az összetett szavak összetételi határának kivételével két magánhangzó közé nem szabad elválasztójelet tenni. Pl. dió-nyi, kia-bál (de: ki-e-mel).

15. Szó elején, illetve végén álló magánhangzót nem szabad leválasztani. Pl. ele-mó-zsia, idill, kémia, ké-miai.

16. Szó elején álló magánhangzócsoportot nem szabad leválasztani. Pl. Euró-pa, aero-nau-ta.

Értékelési szempontok:

1. A programban megvalósított funkciók megbízható működése.

2. A megvalósított funkciók száma, sokrétűsége.

3. A program belső szerkezete: logikus tagolás, általános célú eljárások, bővíthetőség ...

4. A program olvashatósága: beszédes elnevezések, áttekinthető tördelés, magyarázatok, ...

Elérhető összpontszám: 105 pont

A verseny végeredménye:

I. kategória

1. Késmárki Mátyás	Katona József Gimnázium, Kecskemét
2. Salamon András	Zrínyi Miklós Gimnázium, Zalaegerszeg
3. Czunyi Nándor	Jókai Mór Gimnázium, Komárom
4. Lucz Géza	Táncsics Mihály Gimnázium, Kaposvár
5. Schermann Gábor	Zrínyi Miklós Gimnázium, Zalaegerszeg
6. Péter László	Lovassy László Gimnázium, Veszprém
7. Kolesár András	Piarista Gimnázium, Budapest
8. Borgulya Gábor	Nagy Lajos Gimnázium, Pécs
9. Oláh Ervin	Csány László Szakközépiskola, Zalaegerszeg
10. Ackermann Zoltán	Hámán Kató Szakközépiskola, Budapest

II. kategória

1. Günther Tamás	Révai Miklós Gimnázium, Győr
2. Czihó András	Bessenyei György Gimnázium, Kisvárda
Ürmössy Attila	Ságvári Endre Gimnázium, Kazincbarcika
4. Hornák Zoltán	Lovassy László Gimnázium, Veszprém
5. VicziánGergely	Móricz Zsigmond Gimnázium, Budapest
6. Turányi Zoltán	Berzsenyi Dániel Gimnázium, Budapest
7. Csilling Ákos	Fazekas Mihály Gimnázium, Budapest
8. Fejérvári Levente	Veres Pálné Gimnázium, Budapest
9. Gulyás László	Horváth Mihály Gimnázium, Szentes
10. Megyeri Gergely	Árpád Gimnázium, Budapest

1991. Első forduló

Első-második osztályosok

1. feladat: (5 pont)

Tekintsük az alábbi algoritmust, amelynek bemenete egy szó, kimenete pedig a "JÓ!" vagy a "ROSSZ!" felkiáltás! A VEREMBE és VEREMBŐL eljárások egy veremtár kezelésére szolgálnak. Az ÜRES eljárással megtudható, hogy a verem üres-e. A verem kezdetben üres.

Az OLVAS eljárás a bemenet következő betűjét olvassa be, a VANMÉG eljárás pedig megvizsgálja, hogy van-e a szónak további betűje.

```
Ha VANMÉG akkor OLVAS(betű): VEREMBE(betű)
Ciklus amíg (VANMÉG és nem ÜRES)
  OLVAS(betű)
  VEREMBŐL(felső)
  Ha (betű<>felső) akkor VEREMBE(felső): VEREMBE(betű)
Ciklus vége
Ha (VANMÉG vagy nem ÜRES) akkor "ROSSZ!" egyébként "JÓ!"
```

Mely szavakra ad "JÓ!" és melyekre "ROSSZ!" választ az algoritmus az alábbiak közül?

A. bab B. kiiktat C. odaado D. errearra E. kikerrearrakik

2. feladat: (8 pont)

Furcsa számítógépünk a következő formában várja tőlünk a programot. Először meg kell adni az alapismereteket. Másodszor föl kell sorolni azokat a következtetési szabályokat, amelyek alkalmazásával a problémát megoldhatónak gondoljuk.

Alapismeretek például:

```
1. anyja(Szilágyi Erzsébet, Mátyás).
```

Ez azt jelenti, hogy Szilágyi Erzsébet anyja Mátyásnak.

```
2. apja(Hunyadi János, Mátyás).
```

Ez azt jelenti, hogy Hunyadi János apja Mátyásnak.

Szabályok például:

```
1. szülője(x,y) HA anyja(x,y) VAGY apja(x,y).
2. nagyszülője(x,y) HA szülője(x,z) ÉS szülője(z,y).
```

Ez magyarul a következőt jelenti: akkor nagyszülője x y-nak, ha van olyan z személy, akinek x a szülője és ő (azaz z) szülője y-nak.

A logikai kifejezéseket balról jobbra haladva értékeli ki a gép. A kiértékelést abbahagyja, ha a részeredmény alapján már eldönthető a teljes kifejezés értéke.

A: Milyen rokonsági kapcsolatot határoznak meg a következő szabályok?

```
a. rokon1 (x,y) HA szülője(z,x) ÉS szülője(z,y) ÉS x<>y.
b. rokon2 (x,y) HA apja(x,z) ÉS szülője(z,y).
```

B: Írd meg a következő rokoni kapcsolatokat leíró szabályokat!

```
a. anyaiNagyszülője(x,y) HA ...   {azaz x anyai nagyszülője y-nak}
b. szülőpár(x,y) HA ... {azaz akiknek közös gyermekük van}
```

3. feladat: (8 pont)

Egy palacsintasütő és egy palacsintaevő ember számára készítettünk egy-egy algoritmust. Úgy tervezzük, hogy a két ember ezeket az algoritmusokat egyszerre – azaz egymással párhuzamosan – hajtja végre. Egy közös tárolót (tányért) használnak, amelyen egyszerre csak egy palacsinta fér el. Egymással nem beszélhetnek, csupán egy-egy, a vasútállomásokon alkalmazottakhoz hasonló szemaforral jelezhetnek egymásnak. Ehhez a következő utasításokat használhatják fel:

jelezz(SZ): szabadra állítja az SZ szemafort,

várj(SZ): várakozik, amíg az SZ szemafor nem szabadot jelez, majd ismét tilosra állítja, és abbahagyja a várakozást.

```
Sütő:                                   Evő:
  Ciklus                                  Jelezz(ÜRES A TÁNYÉR)
    Süss egy palacsintát!                 Ciklus
    Várj(ÜRES A TÁNYÉR)
    Tedd a palacsintát a tányérra!          *
    Jelezz(EHETSZ)
  Ciklus vége                             Ciklus vége
Eljárás vége.                           Eljárás vége.
```

A * helyébe négy utasítást teszünk, különféle sorrendben.

Add meg, hogy mely megoldások hibásak és miért! Ha több helyes megoldás is van, vizsgáld meg, hogy melyik mennyire hatékony!

```
A: Várj(EHETSZ)                 B: Vedd fel a palacsintát!
   Vedd fel a palacsintát!         Várj(EHETSZ)
   Jelezz(ÜRES A TÁNYÉR)           Jelezz(ÜRES A TÁNYÉR)
   Edd meg!                        Edd meg!

C: Várj(EHETSZ)                 D: Várj(EHETSZ)
   Jelezz(ÜRES A TÁNYÉR)           Vedd fel a palacsintát!
   Vedd fel a palacsintát!         Edd meg!
   Edd meg!                        Jelezz(ÜRES A TÁNYÉR)
```

4. feladat: (12 pont)

Pap Jancsi fogad Tasziló barátjával, hogy a padláson heverő lyukszalagok egyikén sem fordul elő a "LILLA" szó. De hogy biztos legyen a dolgában, programot ír a lyukszalagok átalakítására.

Az alábbi BASIC nyelvű programban az OLVAS(K$) parancs az 1-es számú lyukszalag következő karakterét olvassa be K$-ba, az ÍR(L$) parancs pedig a 2-es számú lyukszalagra írja ki az L$ tartalmát. A szalagok végét a "$" jel jelzi. (Mivel Pap Jancsi még nem nagyon tud programozni, programja hibajelzéssel fog leállni a szalag végén.)

Pap Jancsi minden szalagot átmásol a programmal, majd a már átalakított szalagokat is újra meg újra átfuttatja, amíg csak változást tapasztal. Végül minden szalag eredetije helyett annak utolsó változatát rakja vissza a padlásra.

```
10 OLVAS(K$)
15 IF K$ <> "L" THEN ÍR(K$): GOTO 10 ELSE OLVAS(K$)
20 IF K$ <> "I" THEN ÍR("L"): GOTO 15 ELSE OLVAS(K$)
30 IF K$ <> "L" THEN ÍR("LI"): GOTO 15 ELSE OLVAS(K$)
40 IF K$ <> "L" THEN ÍR("LIL"): GOTO 15 ELSE OLVAS(K$)
50 IF K$ <> "A" THEN ÍR("LILL"): GOTO 15 ELSE GOTO 10
60 END
```

Tasziló, aki megneszelte a turpisságot, már előre dörzsöli a kezét. Jogosan, mert a program valóban rossz.

A: Tasziló, hogy nyilvánvalóvá tegye diadalát, saját készítésű lyukszalagot dug el Pap Jancsi padlásán. Mi legyen ezen a szalagon, hogy a program biztosan felsüljön? (Azaz bennhagyjon legalább egy LILLÁt.)

B: Pap Jancsi közben maga is rájön a hibára, és kijavítja a programot. Két sorban végez apró módosítást, a többi soron nem változtat, és új sort sem szúr be. Add meg ezt a két sort kijavítva!

5. feladat: (11 pont)

A kompatibilis programozóhoz címzett mulatóban egy játékgép áll. A gép tetején egy piros, egy zöld és egy kék lámpa van, amelyek közül mindig csak egy ég; kezdetben a piros. A gépbe "1" és "0" feliratú zsetonokat lehet bedobálni. A gépen van még egy nyomógomb, amely csak akkor nyomható meg, amikor a piros lámpa ég; ilyenkor a gép annyi forintot fizet, amennyi az a 2-es számrendszerbeli szám, amelyet az addig bedobott zsetonokon lévő számjegyek a bedobás sorrendjében balról jobbra sorbaállítva kiadnak (pl. 1,1,0 bedobása 6 Ft-ot ér). A zsetonokat a kasszánál darabonként 10 forintért árulják. A gépbe azonban csak 9 zseton fér. Ha 9 zseton bedobása után a zöld vagy a kék lámpa ég, a pénzünk odaveszett. Só Sajó szerint az, hogy legközelebb melyik lámpa gyullad ki, csak attól függ, hogy éppen melyik lámpa ég, és hogy milyen zsetont dobunk be. Tapasztalatait a következő táblázatban összegezte:

Milyen zsetont dobunk be: \ Melyik lámpa ég:	P	Z	K	Melyik lámpa gyullad ki:
0	p	k	z	←
1	z	p	k	←

Serte Petra szerint a gép csak 9-cel osztható összegeket fizet ki. De Petra nem igazán ismeri a gépet, és lehet, hogy téved.

A. Kapunk-e a géptől pénzt a következő zsetonsorozatok bedobásával?

a. 1,1,0

b. 0,1,1,1

c. 1,0,1,0,0

d. 1,0,1,0,1

B. Igaza van-e Petrának? Ha nincs, akkor mi jellemzi a játékgép által kifizetett összegeket?

C. Maximum mennyit fizet a gép egyszerre, egy játékban?

D. Legalább hány zsetont kell bedobnod ahhoz, hogy a játék nyereséges legyen? (Hiszen a zsetonok is pénzbe kerülnek.)

6. feladat: (5 pont)

Itt van egy programrészlet:

```
Függvény REJTVÉNY(A,B)?
  Ha A>B akkor REJTVÉNY:=-REJTVÉNY(B,A)
  egyébként ha A=0 akkor REJTVÉNY:=B
  egyébként REJTVÉNY:=REJTVÉNY(A-1,B-1)
Függvény vége.
```

Mit adnak eredményül a következő függvényhívások?

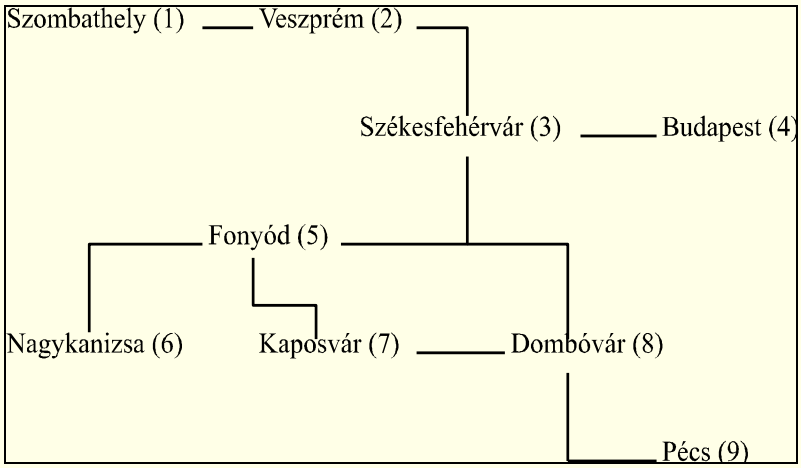
A. REJTVÉNY(3,12) ?

B. REJTVÉNY(4,6) + REJTVÉNY(6,4) ?

C. REJTVÉNY(X,Y), ahol X és Y tetszőleges nemnegatív számok?

7. feladat: (7 pont)

Egy vasúti menetrendhez a következő térkép tartozik:



Az alábbi algoritmus megvizsgálja, hogyan lehet eljutni Budapestről Nagykanizsára. Add meg, hogy milyen sorrendben írja ki az alábbi algoritmus az egyes városokat! A sor nevű adatszerkezetnek csak a végére lehet új elemet tenni (Sorba(I)), elvenni belőle pedig csak az elejéről lehet (I := Sorból).

A Voltunk(I) eljárással feljegyezzük, hogy az I-edik városban már voltunk a bejárás során.

```
Bejárás:
  Sorba(4): Voltunk(4)
  Ciklus amíg a sor nem üres és nem voltunk Nagykanizsán
    I:=Sorból
    Ciklus J=1-től 9-ig
      Ha a J. városban még nem voltunk és
         van közvetlen út az I.-ből a J.-be
      akkor Sorba(J): Voltunk(J); J-edik városnév kiírása
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

Elérhető összpontszám: 56 pont

Harmadik-ötödik osztályosok

1. feladat: (12 pont)

Az alábbi két függvényeljárás (nyissz és nyassz) rudakat darabol. A rudak hosszát a rúd tömbben tároljuk, amelynek a rudak számánál eggyel több eleme van. A tömb utolsó eleme az ún. strázsa, amelyben -1 van. A két függvényeljárás bemenő paramétereként azt kell megadni, hogy mekkora rúddarabot akarunk levágni.

```
VAR rúd: ARRAY [1..Maxrúd] OF INTEGER;
FUNCTION nyissz(h:INTEGER): INTEGER;
  VAR p: INTEGER;
BEGIN
  p:=1;
  WHILE (rúd[p+1]<>-1) AND (rúd[p]<h) DO p:=p+1;
  IF rúd[p]<h THEN nyissz:=0
  ELSE BEGIN nyissz:=p; rúd[p]:= rúd[p]-h END;
END {nyissz};

FUNCTION nyassz(h:INTEGER): INTEGER;
  VAR p,q,m: INTEGER;
BEGIN
  p:=1; q:=0; m:=MAXINT;
  WHILE (rúd[p]<>-1) DO
     BEGIN IF (rúd[p]-h>=0) AND (rúd[p]-h<m)
               THEN BEGIN m:=rúd[p]-h; q:=p END;
           p := p+1
     END;
  IF q<>0 THEN rúd[q]:=m;
  nyassz:=q
END {nyassz};
```

A. Hogyan alakul a tömb tartalma, és milyen értéket adnak vissza az egyes függvényeljárások, ha kezdetben a tömbben [3, 5, 2, 4, 7, -1] van, és a 3, 4, 2, 2 paraméterekkel hívjuk meg mind a nyissz, mind a nyassz eljárást, külön-külön, a megadott sorrendben?

B. A [2, 1, -1] tömbhöz adj meg olyan rúdigénylést, amelyre a nyissz függvényeljárás 0-t fog valamikor visszaadni, a nyassz pedig soha. (Ha nincs ilyen, válaszodat akkor is indokold meg!)

C. A [3, 2, -1] tömbhöz adj meg olyan rúdigénylést, amelyre a nyassz függvényeljárás fog valamikor 0-t visszaadni, a nyissz pedig soha. (Ha nincs ilyen, válaszodat akkor is indokold meg!)

2. feladat: (10 pont)

Furcsa számítógépünk a következő formában várja tőlünk a programot. Először meg kell adni az alapismereteket. Másodszor föl kell sorolni azokat a következtetési szabályokat, amelyek alkalmazásával a problémát megoldhatónak gondoljuk.

Alapismeretek például:

```
1. anyja(Szilágyi Erzsébet, Mátyás).
```

Ez azt jelenti, hogy Szilágyi Erzsébet anyja Mátyásnak.

```
2. apja(Hunyadi János, Mátyás).
```

Ez azt jelenti, hogy Hunyadi János apja Mátyásnak.

Szabályok például:

```
1. szülője(x,y) HA anyja(x,y) VAGY apja(x,y).
2. nagyszülője(x,y) HA szülője(x,z) ÉS szülője(z,y).
```

Ez magyarul a következőt jelenti: akkor nagyszülője x y-nak, ha van olyan z személy, akinek x a szülője és ő (azaz z) szülője y-nak.

A logikai kifejezéseket balról jobbra haladva értékeli ki a gép. A kiértékelést abbahagyja, ha a részeredmény alapján már eldönthető a teljes kifejezés értéke.

A: Milyen rokonsági kapcsolatot határoznak meg a következő szabályok?

```
a. rokon1 (x,y) HA szülője(z,x) ÉS szülője(z,y) ÉS x<>y.
```

```
b. rokon2 (x,y) HA apja(x,z) ÉS szülője(z,y).
c. rokon3 (x,y) HA szülője(x,y) VAGY (szülője(x,z) ÉS ro-
kon3_(z,y)).
```

B: Írd meg a következő rokoni kapcsolatokat leíró szabályokat!

```
a. anyaiNagyszülője(x,y) HA ...   {azaz x anyai nagyszülője y-nak}
b. szülőpár(x,y) HA ...           {azaz akiknek közös gyermekük van}
c. leszármazottja(x,y) HA ...     {azaz x leszármazottja y-nak}
```

3. feladat: (8 pont)

Egy palacsintasütő és egy palacsintaevő ember számára készítettünk egy-egy algoritmust. úgy tervezzük, hogy a két ember ezeket az algoritmusokat egyszerre – azaz egymással párhuzamosan – hajtja végre. Egy közös tárolót (tányért) használnak, amelyen egyszerre csak egy palacsinta fér el. Egymással nem beszélhetnek, csupán egy-egy, a vasútállomásokon alkalmazottakhoz hasonló szemaforral jelezhetnek egymásnak. Ehhez a következő utasításokat használhatják fel:

jelezz(SZ): szabadra állítja az SZ szemafort,

várj(SZ): várakozik, amíg az SZ szemafor nem szabadot jelez, majd ismét tilosra állítja, és abbahagyja a várakozást.

```
Sütő:                                Evő:
  Ciklus                               Jelezz(ÜRES A TÁNYÉR)
    Süss egy palacsintát!              Ciklus
    Várj(ÜRES A TÁNYÉR)
    Tedd a palacsintát a tányérra!         *
  Jelezz(EHETSZ)
  Ciklus vége                          Ciklus vége
Eljárás vége.                        Eljárás vége.
```

A * helyébe négy utasítást teszünk, különféle sorrendben.

Add meg, hogy mely megoldások hibásak és miért! Ha több helyes megoldás is van, vizsgáld meg, hogy melyik mennyire hatékony!

```
A: Várj(EHETSZ)                  B: Vedd fel a palacsintát!
   Vedd fel a palacsintát!          Várj(EHETSZ)
   Jelezz(ÜRES A TÁNYÉR)            Jelezz(ÜRES A TÁNYÉR)
   Edd meg!                         Edd meg!

C: Várj(EHETSZ)                  D: Várj(EHETSZ)
   Jelezz(ÜRES A TÁNYÉR)            Vedd fel a palacsintát!
   Vedd fel a palacsintát!          Edd meg!
   Edd meg!                         Jelezz(ÜRES A TÁNYÉR)
```

4. feladat: (15 pont)

A következő két rendező eljáráshoz írj olyan állításokat, amelyek a megjelölt helyeken érvényesek, és a rendezendő vektorra vonatkozó minden lényeges információt tartalmaznak!

```
Rendezés:
  Ciklus I=1-től N-1-ig
  { A. állítás }
     M:=I
     Ciklus J=I+1-től N-ig
     { B. állítás }
        Ha A(M)>A(J) akkor M:=J
     Ciklus vége
     Csere(A(I),A(M))
  Ciklus vége
Eljárás vége.
```

```
Rendezés:
  Ciklus I=2-től N-ig
  { C. állítás }
     J:=I-1
     Ciklus amíg J>0 és A(J)>A(J+1)
     { D. állítás }
        Csere(A(J),A(J+1)): J:=J-1
     Ciklus vége
  Ciklus vége
Eljárás vége.
```

5. feladat: (12 pont)

Adott az alábbi Pascal-nyelvű függvényeljárás:

```
FUNCTION hash(v: INTEGER): INTEGER;
  VAR h: INTEGER;
BEGIN
  h:=0;
  WHILE v<>0 DO BEGIN h:=h+v MOD 10; v:=v DIV 10 END;
  hash:=h MOD Max
END {hash};
```

A hash függvényt a következő eljárás használja:

```
PROCEDURE insert(v: INTEGER);
  VAR place,n: INTEGER;
BEGIN
  place:=hash(v); n:=0;
  WHILE (n<Max) AND (memo[place]>=0) DO
    BEGIN place:=(place+1) MOD Max; n:=n+1 END;
  IF memo[place]<0 THEN memo[place]:=v
END {insert};
```

A

```
VAR memo: ARRAY [0 .. Max-1] OF INTEGER
```

(globális) tömb minden elemének kezdetben -1 az értéke.

Mi lesz a memo tömb tartalma az alábbi insert–sorozatok adott sorrendű végrehajtása után, a Max állandó adott értéke mellett?

```
A._CONST_Max_=_7;
insert(2); insert(26); insert(117); insert(8); insert(18)
B._CONST_Max_=_11;
  insert(2); insert(26); insert(117); insert(8);
  insert(18); insert(14); insert(13)
```

6. feladat: (12 pont)

Egy adatátviteli rendszerben két számítógép között soros vonalon, négybites csomagokban továbbítják az információt. Mivel az átviteli csatorna nem tökéletes, kb. minden 50. átküldött bit megsérül. Annak azonban elenyésző a valószínűsége, hogy egymáshoz ennél közelebb lévő bitek sérüljenek meg. Ezért minden négybites csomagot 7 biten kódolva küldenek át. Így ha a 7 bitből csak egy sérül meg, akkor az eredetileg átküldött adat visszaállítható. Ha több bit sérül meg egyszerre, akkor az alábbi algoritmus nem működik helyesen. (A XOR bitenkénti KIZÁRÓ VAGY művelet.)

A következő programrészlet a vevőoldalon a dekódolás feladatát látja el. Bemenete a csat hételemű tömb, amely a csatornán érkezett és ugyanabba a csomagba tartozó biteket tartalmazza. Kimenetként az üzen négyelemű tömbben helyreállítja az eredeti üzenet 4 bitjét.

```
i:=0;
IF (csat[1] XOR csat[3] XOR csat[5] XOR csat[7])=1 THEN i:=i+1;
IF (csat[2] XOR csat[3] XOR csat[6] XOR csat[7])=1 THEN i:=i+2;
IF (csat[4] XOR csat[5] XOR csat[6] XOR csat[7])=1 THEN i:=i+4;
IF i > 0 THEN csat[i]:=NOT(csat[i]);
üzen[1]:=csat[3]; üzen[2]:=csat[5];
üzen[3]:=csat[6]; üzen[4]:=csat[7];
```

A következő programrészlet az adóoldalon a kódolás feladatát látja el. Értelemszerűen az előző eljárás fordítottját végzi. Néhány szám azonban hiányzik belőle. Ezeket *-gal helyettesítettük.

```
csat[3]:=üzen[1]; csat[5]:=üzen[2];
csat[6]:=üzen[3]; csat[7]:=üzen[4];
csat[1]:=csat[*] XOR csat[*] XOR csat[*];
csat[2]:=csat[*] XOR csat[*] XOR csat[*];
csat[4]:=csat[*] XOR csat[*] XOR csat[*];
```

Milyen értékeket vehet fel és mit fejez ki i hibátlan átvitel, ill. egyszeres hiba esetén? Add meg a *-gal helyettesített számokat a programban való előfordulásuk sorrendjében!

7. feladat: (12 pont)

Pap Jancsi fogad Tasziló barátjával, hogy a padláson heverő lyukszalagok egyikén sem fordul elő a "LILLA" szó. De hogy biztos legyen a dolgában, programot ír a lyukszalagok átalakítására.

Az alábbi BASIC nyelvű programban az OLVAS(K$) parancs az 1-es számú lyukszalag következő karakterét olvassa be K$-ba, az ÍR(L$) parancs pedig a 2-es számú lyukszalagra írja ki az L$ tartalmát. A szalagok végét a "$" jel jelzi. (Mivel Pap Jancsi még nem nagyon tud programozni, programja hibajelzéssel fog leállni a szalag végén.)

Pap Jancsi minden szalagot átmásol a programmal, majd a már átalakított szalagokat is újra meg újra átfuttatja, amíg csak változást tapasztal. Végül minden szalag eredetije helyett annak utolsó változatát rakja vissza a padlásra.

```
10 OLVAS(K$)
15 IF K$ <> "L" THEN ÍR(K$): GOTO 10 ELSE OLVAS(K$)
20 IF K$ <> "I" THEN ÍR("L"): GOTO 15 ELSE OLVAS(K$)
30 IF K$ <> "L" THEN ÍR("LI"): GOTO 15 ELSE OLVAS(K$)
40 IF K$ <> "L" THEN ÍR("LIL"): GOTO 15 ELSE OLVAS(K$)
50 IF K$ <> "A" THEN ÍR("LILL"): GOTO 15 ELSE GOTO 10
60 END
```

Tasziló, aki megneszelte a turpisságot, már előre dörzsöli a kezét. Jogosan, mert a program valóban rossz.

A: Tasziló, hogy nyilvánvalóvá tegye diadalát, saját készítésű lyukszalagot dug el Pap Jancsi padlásán. Mi legyen ezen a szalagon, hogy a program biztosan felsüljön? (Azaz bennhagyjon legalább egy LILLÁt.)

B: Pap Jancsi közben maga is rájön a hibára, és kijavítja a programot. Két sorban végez apró módosítást, a többi soron nem változtat, és új sort sem szúr be. Add meg ezt a két sort kijavítva!

8. feladat: (13 pont)

A Folttisztító Turbo-Pascal eljárás a grafikus képernyőn lévő és a háttértől elütő színű alakzat pontjait festi át háttérszínűre, bizonyos rendszer szerint. A felhasznált grafikus utasítások és függvények:

GetPixel(x,y): az (x,y) koordinátájú pont színét adja vissza,

GetBkColor: a háttér színét adja vissza,

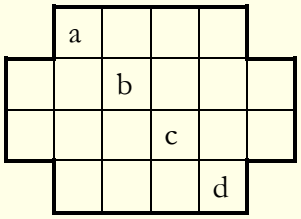
PutPixel(x,y,szín): az (x,y) pontot szín színűre festi.

(Megjegyzés: az origó a bal felső sarokban van, x vízszintes, y függőleges elmozdulást jelent.)

```
PROCEDURE Folttisztító(x,y: INTEGER);
  VAR x1,y1,d,n: INTEGER;
BEGIN
  x1:=x; y1:=y; d:=1; n:=0;
  REPEAT
     CASE d OF
        1: BEGIN x1:=x+1; y1:=y   END;
        2: BEGIN x1:=x;   y1:=y+1 END;
        3: BEGIN x1:=x-1; y1:=y   END;
        4: BEGIN x1:=x;   y1:=y-1 END
     END {CASE};
     IF (GetPixel(x1,y1)<>GetBkColor)
        THEN BEGIN PutPixel(x,y,GetBkColor); n:=0;
                   x:=x1; y:=y1; d:=(d+2) MOD 4+1
             END
        ELSE BEGIN d:=d MOD 4+1; n:=n+1 END
  UNTIL n=4
END {Folttisztító};
```

A. Mi a d változó szerepe, és mit jelentenek egyes értékei?

B. A képernyőn a következő, a háttértől elütő színű alakzat helyezkedik el. (Egy-egy mező egy-egy képernyőpontot jelent.)
Sorszámozd be a program által "kitisztított" pontokat a tisztítás sorrendjében, az **a**, **b**, **c** és **d** betűkkel megjelölt, négy különböző kiindulási mező esetén!
(Mind a négy esetre rajzolj egy-egy ábrát!)



Elérhető összpontszám: 84 pont

1991. Második forduló

Első-második osztályosok

1. Hetedhét: (40 pont)

A '*Hét szűk esztendő*' albizottság hétpecsétes titokként őrzi tervezetét, amelyet a bennfentesek '*Hetet egy csapásra*' törvénynek becéznek. Segítségre van szükségük, ezért fordulnak hozzád.

Csak akkor olvasd tovább, ha tudsz titkot tartani!

A tervezet egyik sarkalatos pontja szerint gazdasági gondjainkat az ősmagyar hetes számrendszer bevezetésével meg lehet oldani, ugyanis a hetes számnak köztudottan mágikus hatása van. A többség a helyi érték nélküli számábrázolás mellett tör lándzsát, mondván, hogy akkor a számok hosszabbak, és a fizetések többnek látszanak. (A hétpróbás ellenzék hetet-havat összehord, amikor azt állítja, hogy az árak is magasabbnak tűnnek majd.)

A javaslat szerint az új számrendszer számjegyeit, az ún. szűkszámjegyeket a hét honfoglaló törzsfőről nevezik el, és jelölésükre e nevek egyik jellemző betűjét használják. (Zárójelben megadjuk a számjegyeket jelölő betűket, és az elavult tízes számrendszerbeli megfelelőiket.)

Álmos (A=1) Előd (E=7) Ond (O=7^2)
Kont (K=7^3) Tas (T=7^4) Huba (H=7^5) Töhötöm (M=7^6)

(Jelenleg arról folyik még vita, hogy a nevek írásánál és ejtésénél a Gesta Hungarorumot avagy Anonymust tekintsék-e hiteles forrásnak.)

Az első ötven szűkszámot az alábbi táblázat mutatja.

	+1	+2	+3	+4	+5	+6	+7
0	A	AA	AAA	AAAE	AAE	AE	E
7	EA	EAA	EAAA	EAAAE	EAAE	EAE	EE
14	EEA	EEAA	EEAAA	EEAAAE	EEAAE	EEAE	EEE
21	EEEA	EEEAA	EEEAAA	EEEAAAE	EEEAAE	EEEAE	EEEO
28	EEEOA	EEEOAA	EEEOAAA	EEEOAAAE	EEEOAAE	EEEOAE	EEO
35	EEOA	EEOAA	EEOAAA	EEOAAAE	EEOAAE	EEOAE	EO
42	EOA	EOAA	EOAAA	AAAO	AAO	AO	O
49	OA						

A szűkszámok képzésének szabályai:

Szűkszámjegyet legfeljebb három nála kisebb, de egyforma szűkszámjegy előzhet meg. A kisebb számjegyeknek közvetlenül a nagyobb számjegy előtt kell állniuk, és ilyenkor a kisebb jegyek értékét le kell vonni a nagyobb számjegy értékéből. Ettől eltekintve a szűkszámok számjegyeit balról jobbra csökkenő sorrendben kell felírni, de úgy, hogy bármely szűkszámjegy után közvetlenül legfeljebb két vele azonos szűkszámjegy állhat. Ilyenkor a számjegyek értékét össze kell adni. (Ha e szabályok alkalmazásával többféle alak is lehetséges, akkor az a helyes, amely a lehető legkevesebb számjegyből áll. Pl. EOAE helyett AO a helyes alak.)

Egyetlen gondja van még a héttagú bizottságnak: abban a rövid időszakban, amíg az emberek megszokják a szűkszámokat, minden szűkszámot át kell alakítani tízes számrendszerbeli számmá. Ehhez kell a te segítséged!

Írj programot, amely tetszőleges szűkszámot átalakít tízes számrendszerbeli számmá!

A programnak csak szintaktikailag helyes számokat adunk, és a számok helyességét nem kell ellenőriznie. Az átalakítandó (garantáltan helyes) szűkszámokat a SZUK.BE szöveges állományból kell beolvasni; minden szűkszámot új sorba írunk, az első szűkszám az első sorban van. A szűkszámok

előtt és mögött tetszőleges számú szóköz lehet. A szűkszám tízes számrendszerbeli megfelelőjét a képernyőre és a SZUK.KI nevű állományba kell kiírni! Minden számot külön sorba kell rakni; az első szám az első sorban legyen!

Példaként megadunk egy minta SZUK.BE állományt. Ezt tetszés szerint módosíthatod, kiegészítheted. A programodat azonban más adatokkal is ki fogjuk próbálni!

ROSSZ

SZUK.BE	SZUK.KI
EAA	9
EEEO	28
MHT	136857
H	16807
HHAH	50420

Beadandó:

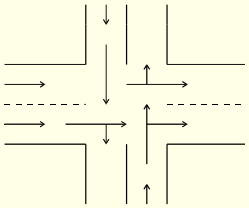
A program szövege lemezen és kinyomtatva; minden, a program futtatásához szükséges segédprogram, szubrutin lemezen; minden olyan dokumentáció, amely a megoldás megértését és értékelését segíti (programvázlat, adatok és adatszerkezetek specifikációja stb.).

2. Útkereszteződés: (100 pont)

Készíts programot, amely egy útkereszteződés forgalmát szimulálja! Az autók a két útra véletlenszerűen érkeznek, az úton a végéig haladnak (ott eltűnnek). Az ábrán függőlegesen rajzolt út kétirányú, a felfelé menő sávból lehet jobbra is kanyarodni. A vízszintes út kétsávos, a jobb oldali sávból jobbra, a bal oldaliból pedig balra is lehet kanyarodni. A kereszteződés forgalmát jelzőlámpák irányítják.

Megoldandó részfeladatok:

1. A kereszteződés, az éppen látható autók kirajzolása.
2. A jelzőlámpák pillanatnyi állapotának megjelenítése.
3. Az utakon mozgó, várakozó autók számának kijelzése.
4. Az autók a két útra véletlenszerűen érkeznek, (ez legyen paraméterezhető), az úton a végéig haladnak (ameddig a képernyőn ábrázoljuk az utat).
5. Az autók a kereszteződésben vagy egyenesen haladnak át, vagy a megfelelő irányba fordulnak.
6. Az autók a közlekedési lámpák miatt, illetve az előttük megállt más autók miatt az úton megállhatnak. Amikor lehetséges, tovább kell indulniuk.
7. A lámpaváltások idejét a felhasználó adja meg, amelyet a programnak a forgalomtól függően módosítania kell.
8. Az autók vagy állnak, vagy azonos sebességgel haladnak, egymást nem előzhetik.



Elérhető összpontszám: 140 pont

Harmadik-ötödik osztályosok

Szó ami szó: (140 pont)
Napjainkban egyre inkább előtérbe kerülnek a számítógép nem–műszaki jellegű felhasználásai. Ezek általában adatbázisokon, szövegeken való műveleteket jelentenek. E vonatkozásban fontos feladat az, amikor nem egyszerűen csak szavakat, karaktersorozatokat keresünk egy szövegben, hanem bizonyos szavak, szótöredékek, szócsoportok együttes előfordulását (szakszóval ezt nevezik információvisszakeresésnek). Alkothatunk például logikai kifejezést a keresni kívánt szavakból: "Keressük azt a szövegrészt, amelyben előfordul az egér szó, továbbá a kutya vagy a macska szavak közül legalább az egyik." Ennek formális leírására az egér & (kutya + macska) logikai kifejezés alkalmas, amelyben már a későbbiekben bevezetendő jelöléseket alkalmaztuk.

Írj programot, amely ilyen keresést valósít meg egy szövegben! A program tegyen eleget a következő kikötéseknek:

A szöveg
Az átvizsgálandó szöveg egy fájlban található, melynek a neve legyen SZOVEG.INP. (Legjobb, ha ezt a program magától tudja, és nem kérdez rá a névre.)

A szöveg bekezdésekből áll. Két bekezdést egy üres sor választ el egymástól, az utolsó bekezdés a fájl végéig tart. Egy bekezdés több, tetszőleges hosszúságú, nem üres sorból állhat, de teljes hossza nem haladhatja meg a 250 karaktert.

A bekezdés sorai szavakból állnak, amelyeket írásjelek (szóköz, '.' ',' ';' ':' '?' és '!') tetszőleges kombinációja választ el egymástól. Egy szó az ékezet nélküli ábécé kis- és nagybetűit, illetve számjegyeket tartalmazhat.

A program feltételezheti, hogy a bemeneti szöveg megfelel e szabályoknak, s ezt ellenőriznie sem kell.

A keresendő minta

A keresendő minta logikai kifejezés formájában adható meg. Ez mintaszavakból, az ÉS, VAGY, TAGADÁS műveletekből, valamint zárójelekből állhat. A műveletek precedenciája a szokásos (TAGADÁS > ÉS > VAGY); ezt azonban zárójelezéssel befolyásolni lehet.

A zárójelek a szokásos (és) karakterek. A műveleteket egy-egy karakter jelöli. Ezek a következők:

```
VAGY     +     ÉS &      TAGADÁS   !
```

A mintaszavak ékezet nélküli kis- és nagybetűket, számjegyeket, valamint ún. joker karaktereket tartalmazhatnak. Két mintaszó között a műveleti és zárójeleken kívül jelentés nélküli szóköz karakterek is állhatnak. (Ezek egyik szónak sem részei.)

A joker karaktert * jelöli. Ez egy szón belül bármilyen karaktersorozatra illeszkedik, beleértve az üres karaktersorozatot is. Joker a mintaszóban tetszőleges helyen előfordulhat, egy mintaszóban akár több helyen is; ilyenkor a különböző helyeken álló jokerek különböző karaktersorozatokat jelenthetnek. A mintaszó akkor illeszkedik egy szóra, ha a mintaszóban levő joker karakterek helyére a szabályoknak megfelelő karaktersorozatot illesztve a szó előállítható.

Az illeszkedés vizsgálatakor a kis- és nagybetűk között nem szabad különbséget tenni.

Példaként megadjuk, hogy a következő mintaszavak a hat, Alma, sok, HATALMAS, Heves, hasal szavak közül melyekre illeszkednek.

Minta	Illeszkedő szó
Hat	hat
hat*	hat, HATALMAS
alma	Alma, HATALMAS

```
*S*        sok, HATALMAS, Heves, hasal
H*s        HATALMAS, Heves
```

A program adjon lehetőséget arra, hogy a mintát akár a billentyűzetről, akár szöveges állományból beadhassuk. Ez utóbbi esetben a fájl neve MINTA.INP, és a minta egyetlen sorban helyezkedik el.

A keresés

A programnak azokat a bekezdéseket kell a bemeneti szövegből kiválasztania, amelyek megfelelnek a mintaként megadott logikai kifejezésnek! Egy bekezdés akkor felel meg a kifejezésnek, ha a kifejezésben levő mintaszavak logikai értékekkel való helyettesítése után a kifejezés logikai IGAZ értéket ad. A mintaszavak helyettesítési értéke IGAZ, ha a bekezdésben található a mintaszóra illeszkedő szó, és HAMIS, ha nem található ilyen. A különböző bekezdéseket a programnak egymástól függetlenül kell megvizsgálnia!

Eredmény

A program írja ki a szövegből a mintának megfelelő bekezdéseket a képernyőre és az EREDM.OUT nevű fájlba! A program a szöveg feldolgozásának végén üzenetben tudassa a felhasználóval a talált bekezdések számát!

Példaként találsz a lemezen egy egyszerű szöveget, amelyet tesztelési célra használhatsz, és tetszés szerint bővíthetsz, módosíthatsz. A programodat azonban bonyolultabb szövegekkel és különféle mintákkal fogjuk kipróbálni.

Ha nem tudtad befejezni a feladatot, akkor feltétlenül írd le, hogy szerinted mely részfeladatokat sikerült megoldanod!

Beadandó

A program szövege lemezen és kinyomtatva; minden, a program futtatásához szükséges segédprogram, szubrutin lemeze; minden olyan dokumentáció, amely a megoldás megértését és értékelését segíti (programvázlat, adatok és adatszerkezetek specifikációja stb.).

Elérhető összpontszám: 140 pont

A verseny végeredménye:

I. kategória

1. Szász Olivér	Berzsenyi Dániel Gimnázium, Budapest

II. kategória

1. Kiss Róbert	Révai Miklós Gimnázium, Győr

1992. Első forduló

Első-második osztályosok

1. feladat: (10 pont)

D.E. Knuth: A számítógép-programozás művészete című könyvében J. Stein alábbi algoritmusát közli:

"A program bemenő adatai az U és V természetes számok.

B1. Legyen k ← 0, majd ismételten k ← k+1, U ← U/2, V ← V/2, egyszer vagy többször, esetleg egyszer sem, mindaddig, amíg már U és V nem mindketten párosak.

B2. Ha U páratlan, legyen T ← -V, és menjünk B4-re. Egyébként pedig legyen T ← U!

B3. Legyen T ← T/2.

B4. Ha T páros, menjünk vissza B3-ra.

B5. Ha T>0, legyen U <- T, egyébként legyen V ← -T.

B6. Legyen T ← U-V! Ha T<>0, menjünk vissza B3-ra. Egyébként az algoritmus véget ér, az output U*2^K."

A. Mi lesz a program eredménye a (15,20), (24,28), (54,60) számpárokra?

B. Milyen feladatot old meg a program?

2. feladat: (15 pont)

Az alábbi programrészlet az A$ változó alapján állít elő egy szöveget a B$-ban. A$ csak nyomtatható karaktereket tartalmazhat! A felhasznált függvényeljárások jelentése:

RIGHT$(T$, i): a T$ szöveg jobb szélső i db karaktere,

LEFT$(T$, i): T$ szöveg bal szélső i db karaktere,

MID$(T$, k, i): a T$ szöveg középső i db karaktere a k.-tól kezdve,

LEN(T$): T$ szöveg karaktereinek száma,

CHR$(k): a k kódú karakter.

```
1000 S=1: B$=LEFT$(A$,1)
1010 IF B$=A$ THEN 1200
1020 FOR I=2 TO LEN(A$)
1030  IF RIGHT$(B$,1)=MID$(A$,I,1) THEN S=S+1: GOTO   1090
1040  IF S>2 THEN B$=B$+CHR$(0)+CHR$(S)
1050  IF S=2 THEN B$=B$+RIGHT$(B$,1)
1060  B$=B$+MID$(A$,I,1): S=1
1090 NEXT I
1200 ...
```

A. Mi lesz B$-ban az 1200-as sorban, ha A$ tartalma kezdetben

A1. "Ipafai fapipa"

A2. "Ezüsttel befuttatott."

A3. "13333333-szor ismételd meg."

A4. "Gyakorisága 888888"

B. Fogalmazd meg tömören, mit csinál a fenti program!

C. Milyen esetekben nem lehet előállítani B$-ból A$ értékét?

D. Egészítsd ki a programot két új sorral (1100 és 1110) úgy, hogy B$-ból A$ mindig előállítható legyen!

3. feladat: (13 pont)

A következő programrészlet bemenete az A(N) vektor, értékei 0 és M-1 közötti egész számok.

A. Mit tartalmaz a megoldásban a B vektor a (*) ponton?

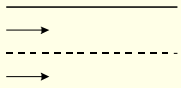
B. Mit tartalmaz a megoldásban a B vektor a (**) ponton?

C. Mit tartalmaz a megoldásban a B és a C vektor a (***) ponton?

```
Ciklus I=0-tól (M-1)-ig
  B(I):=0
Ciklus vége
Ciklus I=1-től N-ig
  B(A(I)):=B(A(I))+1
Ciklus vége
  (*)
Ciklus I=1-től (M-1)-ig
  B(I):=B(I)+B(I-1)
Ciklus vége
  (**)
Ciklus I=1-től N-ig
  C(B(A(I))):=A(I): B(A(I)):=B(A(I))-1
Ciklus vége
  (***)
```

4. feladat: (18 pont)

Egy kétsávos, egyirányú út forgalmát szimuláló programot készítettünk. A sávok forgalmát az A tömb írja le:



A(I,1) = -1 azt jelenti, hogy a felső sáv I. helyén nincs autó; A(I,1) ≥ 0 a felső sáv I. helyén található autó sebességét adja meg (azaz egy időegység alatt ennyit léphet előre). A(I,2) ugyanezt jelenti az alsó sávra.

Add meg, milyen forgalmi szituációkat szimulálnak az F1, ..., F5 eljárások közül a megírtak, ill. írd meg a szövegesen megadottakat! (A megoldásban S a maximális sebességet jelenti.)

```
Ciklus I=N-től 1-ig -1-esével
  Ciklus J=1-től 2-ig
     F1(I,A(I,J),J,K): F1(I,A(I,J),3-J,L)
     Ha K=A(I,J) és A(I,J)<S akkor F2(I,J)
     különben ha K=A(I,J) akkor F3(I,J)
     különben ha L<A(I,J) akkor F4(I,J,K)
     különben F5(I,J,L)
  Ciklus vége
Ciklus vége
```

```
F1(I,T,J,K):
```

Lehet-e a J. sávban az I. helyről T lépést előrelépni? K legyen az a lépésszám, amennyit lehet (maximum T, minimum 0); I+K = N+1, ha az út végére ért az autó.

```
Eljárás vége.
```

```
F2(I,J):
```

Előrelépés a sebességnek megfelelő távolságra, majd gyorsítás 1 egységgel.

```
Eljárás vége.
```

```
F3(I,J):
```

Előrelépés a sebességnek megfelelő távolságra.

```
Eljárás vége.
```

```
F4(I,J,K):
  A(I+K,J):=K-1: A(I,J):=-1
Eljárás vége.
```

```
F5(I,J,L):
  A(I+L,3-J):=A(I,J): A(I,J):=-1
Eljárás vége.
```

5. feladat: (8 pont)

Mit, milyen helyzetben és méretben rajzolnak a következő, LOGO nyelvű programok? Az egyes parancsok jelentése:

LEFT f, RIGHT f: balra-, illetve jobbrafordulás helyben f fokkal;

FORWARD h: előrelépés h egységgel, közben rajzol, ha kell

PENUP, PENDOWN: toll felemelése, illetve letevése a papírra

REPEAT db [utasítások]: utasítások megismétlése db-szer.

A.
```
LEFT 30
  REPEAT 3
     [REPEAT 2 [FORWARD x RIGHT 60 FORWARD x RIGHT 120]
      RIGHT 120]
```

B.
```
REPEAT 4
     [REPEAT 360 [FORWARD 1 RIGHT 1]
      PENUP RIGHT 90 FORWARD 10 LEFT 90 PENDOWN]
```

6. feladat: (13 pont)

Egy kurzormozgató program a kurzort a J, B, L, F billentyűkkel mozgatja. Mi a programban felhasznált többi billentyű hatása?

A 'Rajzol' eljárás egy szakasz pontjait váltja az ellenkezőjére a képen (sötétet világosra, illetve fordítva).

```
Kurzormozgatás
  M:=1: X:=100: Y:=100: RX:=X: RY:=Y: Rajzol(X,Y,X,Y)
  Ciklus
     Be: B
     Elágazás
       B='F' esetén XX:=X: YY:=Y-1: Változás
       B='L' esetén XX:=X: YY:=Y+1: Változás
       B='J' esetén XX:=X+1: YY:=Y: Változás
       B='B' esetén XX:=X-1: YY:=Y: Változás
       B='1' esetén M:=1: RX:=X: RY:=Y: Rajzol(X,Y,X,Y)
       B='2' esetén M:=2: RX:=X: RY:=Y: Rajzol(X,Y,X,Y)
     Elágazás vége
  Ciklus vége
Eljárás vége.
```

```
Változás:
  Ha M=1 akkor  Rajzol(RX,RY,X,Y): Rajzol(RX,RY,XX,YY)
      különben  Rajzol(RX,RY,RX,Y): Rajzol(RX,RY,RX,YY)
                Rajzol(RX,RY,X,RY): Rajzol(RX,RY,XX,RY)
                Rajzol(RX,Y,X,Y): Rajzol(RX,YY,XX,YY)
                Rajzol(X,RY,X,Y): Rajzol(XX,RY,XX,YY)
  Elágazás vége
  X:=XX: Y:=YY
Eljárás vége.
```

7. feladat: (10 pont)

Egy színes monitor a képen megjelenő színeket az RGB színmodell segítségével állítja elő (R (red) = piros, G (green) = zöld B (blue) = kék). A megjelenő szín fehér, ha mindhárom alapszín benne van; fekete, ha egyik sincs benne; alapszín, ha a három közül csak az egyik fordul elő benne; és keverékszín, ha kettő. A színeket tehát egy három elemű logikai vektorral ábrázoljuk, azoknál az alapszíneknél igaz értékkel, amelyek az adott színben előfordulnak. Milyen műveleteket végeznek a következő eljárások?

```
E1(S1,S2,S3):
  Ciklus S=red-től blue-ig
     S3(S):=S1(S) OR S2(S)
  Ciklus vége
Eljárás vége.
```

```
E2(S1,S2):
  Ciklus S=red-től blue-ig
     S2(S):=NOT S1(S)
  Ciklus vége
Eljárás vége.
```

```
E3(S1,SZ,I):
  I:=1
  Ciklus S=red-től blue-ig
     Ciklus J=1-től 3-ig
        SZ(J,S):=hamis
     Ciklus vége
     SZ(I,S):=S1(S)
     Ha S1(S) akkor I:=I+1
  Ciklus vége
Eljárás vége.
```

8. feladat: (13 pont)

Mit adnak eredményül a következő függvényeljárások (ahol L, L1, L2 karaktersorozatok, E pedig egy karakter):

A. Ismerjfel(L1,L2):
   Ha üres(L1) akkor eredmény:=L2
   különben eredmény:=egymásután(első(L1),Ismerjfel(elsőutániak(L1),L2))
   Függvény vége.

B. Találjki(E,L):
   Ha üres(L) akkor eredmény:=L
   különben Ha E=első(L) akkor eredmény:=elsőutániak(L)
   különben eredmény:=egymásután(első(L),Találjki(E,elsőutániak(L)))
   Függvény vége.

C. Mi az eredménye az

```
Ismerjfel("NEMES",egymásután("+","TIHAMÉR"))
```

   függvényeljáráshívásnak?

D. Mi az eredménye a

```
Találjki("E","NEMES")
```

   függvényeljáráshívásnak?

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

1. feladat: (18 pont)

A következő függvényeljárás az A tömb elemeiből számít ki egy értéket (a tömböt 1-től indexeljük):

```
F(P):
  Ha A(P)≥0 akkor F:=A(P)
   különben  I:=F(2*P): J:=F(2*P+1)
             Ha I=-1 vagy J=-1 akkor F:=-1
             különben Ha A(P)=-1 akkor F:=I+J
             különben Ha A(P)=-2 akkor F:=I-J
             különben Ha A(P)=-3 akkor F:=I*J
             különben Ha A(P)=-4 akkor Ha J=0 akkor F:=-1
                                       különben F:=I/J
             különben F:=-1
  Elágazások vége
Eljárás vége.
```

A. Mi az A tömbben lévő számok pontos szerepe, jelentése?

B. Mi lesz F(1) értéke, ha az A tömb tartalma [-2, 7, 5]?

C. Mi lesz F(1) értéke, ha az A tömb tartalma [-1, -3, -4, 2, 2, 6, 2]?

D. Milyen esetekben lesz az F(1) függvényeljárás-hívásnak -1 az eredménye?

2. feladat: (15 pont)

Az alábbi programrészlet az A$ változó alapján állít elő egy szöveget a B$-ban. A$ csak nyomtatható karaktereket tartalmazhat! A felhasznált függvényeljárások jelentése:

RIGHT$(T$, i): a T$ szöveg jobb szélső i db karaktere,

LEFT$(T$, i): a T$ szöveg bal szélső i db karaktere,

MID$(T$, k, i): a T$ szöveg középső i db karaktere a k.-tól kezdve,

LEN(T$): a T$ szöveg karaktereinek száma,

CHR$(k): a k kódú karakter.

```
1000 S=1: B$=LEFT$(A$,1)
1010 IF B$=A$ THEN 1200
1020 FOR I=2 TO LEN(A$)
1030  IF RIGHT$(B$,1)=MID$(A$,I,1) THEN S=S+1: GOTO 1090
1040  IF S>2 THEN B$=B$+CHR$(0)+CHR$(S)
1050  IF S=2 THEN B$=B$+RIGHT$(B$,1)
1060  B$=B$+MID$(A$,I,1): S=1
1090 NEXT I
1200 ...
```

A. Mi lesz B$-ban az 1200-as sorban, ha A$ tartalma kezdetben

A1. "Ipafai fapipa"

A2. "Ezüsttel befuttatott."

A3. "13333333-szor ismételd meg."

A4. "Gyakorisága 888888"

B. Fogalmazd meg tömören, mit csinál a fenti program!

C. Milyen esetekben nem lehet előállítani B$-ból A$ értékét?

D. Egészítsd ki a programot két új sorral (1100 és 1110) úgy, hogy B$-ból A$ mindig előállítható legyen!

3. feladat: (13 pont)

A következő programrészlet bemenete az A(N) vektor, értékei 0 és M-1 közötti egész számok.

A. Mit tartalmaz a megoldásban a B vektor a (*) ponton?

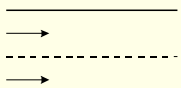
B. Mit tartalmaz a megoldásban a B vektor a (**) ponton?

C. Mit tartalmaz a megoldásban a B és a C vektor a (***) ponton?

```
Ciklus I=0-tól (M-1)-ig
  B(I):=0
Ciklus vége
Ciklus I=1-től N-ig
  B(A(I)):=B(A(I))+1
Ciklus vége
  (*)
Ciklus I=1-től (M-1)-ig
  B(I):=B(I)+B(I-1)
Ciklus vége
  (**)
Ciklus I=1-től N-ig
  C(B(A(I))):=A(I): B(A(I)):=B(A(I))-1
Ciklus vége
  (***)
```

4. feladat: (18 pont)

Egy kétsávos, egyirányú út forgalmát szimuláló programot készítettünk. A sávok forgalmát az A tömb írja le:



A(I,1)=-1 azt jelenti, hogy a felső sáv I. helyén nincs autó; A(I,1)≥0 a felső sáv I. helyén található autó sebességét adja meg (azaz egy időegység alatt ennyit léphet előre). A(I,2) ugyanezt jelenti az alsó sávra.

Add meg, milyen forgalmi szituációkat szimulálnak az F1, ..., F5 eljárások közül a megírtak, ill. írd meg a szövegesen megadottakat! (A megoldásban S a maximális sebességet jelenti.)

```
Ciklus I=N-től 1-ig -1-esével
  Ciklus J=1-től 2-ig
     F1(I,A(I,J),J,K): F1(I,A(I,J),3-J,L)
     Ha K=A(I,J) és A(I,J)<S akkor F2(I,J)
     különben ha K=A(I,J) akkor F3(I,J)
     különben ha L<A(I,J) akkor F4(I,J,K)
     különben F5(I,J,L)
  Ciklus vége
Ciklus vége
```

```
F1(I,T,J,K):
```

Lehet-e a J. sávban az I. helyről T lépést előrelépni? K legyen az a lépésszám, amennyit lehet (maximum T, minimum 0); I+K = N+1, ha az út végére ért az autó.

```
Eljárás vége.
```

```
F2(I,J):
```

Előrelépés a sebességnek megfelelő távolságra, majd gyorsítás 1 egységgel.

```
Eljárás vége.
```

```
F3(I,J):
```

Előrelépés a sebességnek megfelelő távolságra.

```
Eljárás vége.
```

```
F4(I,J,K):
  A(I+K,J):=K-1: A(I,J):=-1
Eljárás vége.
```

```
F5(I,J,L):
  A(I+L,3-J):=A(I,J): A(I,J):=-1
Eljárás vége.
```

5. feladat: (20 pont)

Egy tömbben tároljuk egy terület N*M pontjának tengerszint feletti magasságát. A pontok egymástól S távolságra vannak egy négyzetrácson. Az SZ magasságú szintvonalat a következő ciklussal rajzoltathatjuk meg:

```
Ciklus I=1-től (N-1)-ig
  Ciklus J=1-től (M-1)-ig
     Ha Szintvonal(I,J,I+1,J)
        akkor Szakaszrajz(I*S,J*S-S/2,I*S,J*S)
     Ha Szintvonal(I,J,I,J+1)
        akkor Szakaszrajz(I*S-S/2,J*S,I*S,J*S)
     Ha Szintvonal(I+1,J,I+1,J+1)
        akkor Szakaszrajz(I*S+S/2,J*S,I*S,J*S)
     Ha Szintvonal(I,J+1,I+1,J+1)
        akkor Szakaszrajz(I*S,J*S+S/2,I*S,J*S)
  Ciklus vége
Ciklus vége.
```

```
Szintvonal(I,J,K,L):
  Szintvonal:=(T(I,J)<SZ ÉS T(K,L)≥SZ) VAGY
              (T(I,J)≥SZ ÉS T(K,L)<SZ)
Eljárás vége.
```

Ez a megoldás pontnégyesekre bontja a síkot. Ha a pontnégyes egy pontpárja között átmegy szintvonal, akkor a pontpárt képzeletben összekötő szakasz felezőpontját összeköti a pontnégyes által meghatározott négyzet középpontjával. Pl. az alábbi ábrát kapjuk, ha feltesszük, hogy két szomszédos pontpár között halad át a láthatatlan szintvonal:

Alakítsd át az algoritmust úgy, hogy a vonal ne a felezőpontokon menjen át, hanem a megfelelő pontpárokban vett érték szerint arányosan határozd meg! A kiválasztott pontokat ne a négyzet középpontjával kösd össze, hanem egymással! Írd le azt is, hogy milyen esetek fordulhatnak elő (hány pontot lehet egy négyesen belül összekötni)! Pl.

6. feladat: (16 pont)

Egy agyafúrt bankrabló lefotózta a széf biztonsági programját. Balszerencséjére a programrész eleje és vége lemaradt!

A. Milyen értékeket vehet fel a B változó a 12310-es sorban?

B. Hogyan változtatja a ciklusmag az A és a B értékét?

C. Legfeljebb hány kódot kell kipróbálnia a betörőnek, hogy biztosan kinyithassa a széfet, és melyek lehetnek ezek?

```
...
12300 B=INT(ABS(X))
12310 B=B-INT(B/3)*3+1
12320 INPUT "Mi a titkos kód", A
12325 IF A<100 OR A>999 THEN GOTO 07
12330 A=INT(A-33)*2
12340 FOR I=1 TO 2
12345   C=INT(A/10)
12350   B=B+A-C*10
12360   A=C
12370 NEXT I
12380 IF B<>A OR B>3 THEN GOTO 12300
12390 PRINT "Máris nyitom!"
...
```

7. feladat: (13 pont)

Az alábbi rekurzív függvényeljárás bemenő paraméterei az X, M, A és B változók. X és M csak pozitív értékek lehetnek.

A. Milyen értéket számít ki a függvény?

B. Mi az M szerepe?

C. Mi az A és a B szerepe?

D. Tetszőleges pozitív X és M esetén adj meg olyan A és B kezdőértékpárt, amely biztosan jó!

```
Function Rekurzív(X,M,A,B: Real): Real;
  Var C: Real;
Begin
  C:=(A+B)/2;
  If C-A<M then Rekurzív:=C
  else if C*C>X Then Rekurzív:=Rekurzív(X,M,A,C)
  else Rekurzív:=Rekurzív(X,M,C,B)
End {Rekurzív};
```

8. feladat: (13 pont)

A lista mint adatszerkezet rekurzív definíciója a következő:

1. egyetlen elemet sem tartalmaz, azaz üres;
2. egy nemüres elemet és egy azt követő listát tartalmaz.

A lista-adattípust a fenti rekurzív definíciót hűen követve így definiálhatjuk egy képzeletbeli programozási nyelven (ahol NIL üres listát, | választást jelöl, Elem pedig bármilyen, már ismert adattípus lehet):

```
TYPE Lista  = NIL   |  (feje: Elem,farka: Lista)
```

Az alábbi függvényeljárás egy elemből és egy listából egy 1 elemmel hosszabb listát hoz létre:

```
FUNCTION hozzLétre(e: Elem,l: Lista): Lista:
  RETURN Lista: [e,l].
```

Ennyi bevezető után írd le saját szavaiddal az alábbi függvényeljárások hatását és működését:

A.
```
FUNCTION ismerjFel (l1,l2: Lista): Lista:
   RETURN IF l1=NIL THEN l2
          ELSE hozzLétre(l1.feje,ismerjFel(l1.farka,l2)
          ENDIF.
```

B.
```
FUNCTION találjKi(e: Elem,l: Lista): Lista:
     RETURN  IF l=NIL THEN l
             ELIF e=l.feje THEN l.farka
             ELSE hozzLétre(l.feje,találjKi(e,l.farka)
             ENDIF.
```

Tegyük föl, hogy már léteznek az alábbi (az egyszerűség kedvéért csupa karakterből álló) listák:

```
l1: (N(E(M(E(S))))) l2: (T(I(H(A(M(É(R))))))
```

Add meg az alábbi műveletek (függvényeljárás-hívások) eredményét:

C. ismerjFel(l1, hozzLétre('+', l2) = ?

D. találjKi('E', l1) = ?

Elérhető összpontszám: 126 pont

1992. Második forduló

Első-második osztályosok

Demográfia:

A feladat: nyúlpopuláció korcsoportjainak vizsgálata számítógépes szimulációval egy nyúltenyésztő számára.

A szimulációs program először is olvassa be a szükséges paramétereket (12 pont), majd a szimuláció során folyamatosan rajzoljon két grafikont: egyet a nyulak létszámáról az évek függvényében és egy másikat az egyes nyúlkorcsoportok létszámáról a korcsoportok függvényében! Például a 9. év elérésekor a következő oldalon láthatóhoz hasonló ábrát várunk.

Paraméterként adjuk meg a nyulak számát (max. 1000), maximális életkorukat (10-nél nem nagyobb természetes szám), az egyes nyulak életkorát (természetes szám), a születési, az elsődleges és másodlagos halálozási rátákat, továbbá a járvány valószínűségét (l. alább). A szimulációs program egy-egy lépésében elvégzi mindazt, ami a nyulakkal egy év alatt történhet.

A: A nyulak többsége öregszik és szaporodik, kisebb része a koruktól függő eséllyel (valószínűséggel) meghal. Ezeket az ún. elsődleges halálozási rátákat korcsoportonként 0 és 1 közötti valós számmal adjuk meg, az utolsó szükségképpen 1.

A nyulak (a nemeket nem különböztetjük meg) a koruktól függően szülnek utódokat. Ezeket az ún. születési rátákat korcsoportonként egy-egy egész számpárral adjuk meg; a számpár első tagja az átlagos utódszám, második tagja az átlagtól számított maximális eltérés. Az utódszámot egy-egy esetben az

[átlagos utódszám – eltérés, átlagos utódszám + eltérés]

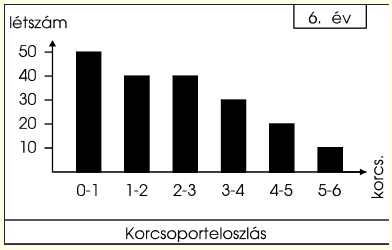
intervallumból vett véletlen egész szám adja meg.

B: Egyes években – véletlenszerűen – járvány tör ki a nyulak között; a járvány valószínűsége 0 és 1 közötti valós szám. Ilyenkor a halálozási ráták megnőhetnek. A szimulációs programnak a járvány évében ezeket a másodlagos halálozási rátákat kell alkalmaznia.

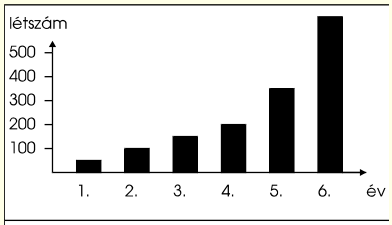
C: A nyúltenyésztő nem tarthat 1000-nél több nyulat. Ha számuk az 1000-et meghaladja, a korcsoportok létszámát arányosan csökkentve a felesleges nyuszikat megeszi.

A program szerkezetére, olvashatóságára, áttekinthetőségére max. 20 pont kapható.

A két grafikon ilyen lehet (Szlávi P.-Zsakó L.: Szimulációs modellek a populációgenetikában c. könyve alapján):



Korcsoporteloszlás



Populációlétszám

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

Petri gráf vizsgálata:

A feladat: Petri-gráf számítógépes ábrázolása és működésének szimulálása.

A Petri-gráf olyan hálózat, amely csomópontokból, átmenetekből és ezeket összekötő nyilakból áll. A csomópontokban (jelölésük: doboz) lehetnek ún. bigyók (jelölésük: o), tetszőleges számban. A csomópontokból nyilak indulnak ki, amelyek átmenetekhez (jelölésük: dupla vonal) vezetnek. Az átmenetekből újabb nyilak indulnak ki, amelyek csomópontokba vezetnek stb. A hálózat működése a következő:

Egy átmenet aktivizálódásának az a feltétele, hogy az őt közvetlenül megelőző csomópontokban legyen bigyó. Ha az átmenet aktivizálódik, minden befelé vezető nyílon elszív egy-egy bigyót a szomszédos csomópontokból, majd a belőle kivezető nyilakon (a számuk eltérhet a befelé mutató nyilak számától) elküld egy-egy bigyót az őt közvetlenül követő csomópontokba. Az átmenetekben a bigyók száma változhat: ha a bejövő nyilak vannak többen, a felesleges bigyók eltűnnek; ha a kifelé vezető nyilak száma a nagyobb, új bigyók keletkeznek. Ha egyidejűleg több átmenet is aktivizálódhat, azaz versenyhelyzet alakul ki, akkor véletlenszerűen egy átmenet aktivizálódik közülük.

Ha a hálózat kezdőállapotát (azaz a bigyók kezdeti eloszlását) ismerjük, meghatározhatjuk azokat az állapotsorozatokat, amelyek a hálózat működését jellemzik. (Több ilyen sorozat is lehet, hiszen a versenyhelyzetben lévő átmenetek véletlenszerűen aktivizálódnak.)

Egy adott állapotból egy másik állapot elérhető, ha van olyan átmenet-aktivizálási sorrend, amelynek eredményeképpen a másik állapot bekövetkezik.

Előfordulhat olyan állapot, amelyben egyetlen átmenet sem aktivizálható. Ekkor a hálózat holtpontba került.

A hálózat ciklusba esett, ha

- egy állapot megegyezik valamely korábbi állapottal;

- egy állapot csak abban tér el egy korábbi állapottól, hogy egy vagy több csomópontban több bigyó van, mint korábban volt (ui. ilyenkor végtelen számú bigyó halmozódik fel).

Feladatok: (semmi grafika, alfanumerikus be- és kivitel)

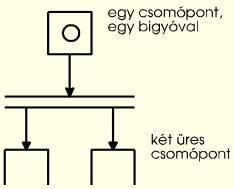
1. Készíts programot Petri-gráf tárolására! A program kezelője adja meg, hogy hány csomópontja és hány átmenete van a gráfnak, és hogy honnan hová vezetnek a nyilak. Csomópontot számjeggyel, átmenetet betűvel jelölünk. A program legyen képes a tárolt gráf szöveges kiírására, de grafikus megjelenítése nem szükséges. Korlátozások: a csomópontok és az átmenetek száma legfeljebb 8–8; nincsenek párhuzamos nyilak (azaz minden átmenet a működése során egy adott csomópontból legfeljebb egy bigyót szív el, ill. egy adott csomópontba legfeljebb egy bigyót küld); a kezdőállapotban egyetlen csomópontban sem lehet egynél több bigyó.

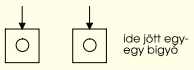
2. A kezdőállapot beolvasása után a program írjon ki egy max. 8 elemű állapotsorozatot (egyet a lehetségesek közül), amelyen a hálózat a működése során áthalad! A program vegye észre, ha a hálózat holtpontba kerül!

3. A kezdőállapot beolvasása és egy másik állapot megadása után a program döntse el, hogy a másik állapot elérhető-e legfeljebb 8 lépésben! Ezt úgy állapítsa meg, hogy az adott kezdőállapotból kiindulva állítsa elő és írja ki az összes elérhető állapotot, és vizsgálja meg, hogy ezek között szerepel-e a keresett állapot.

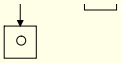
A program szerkezetére, olvashatóságára, áttekinthetőségére max. 20 pont kapható. ROSSZ ÁBRA

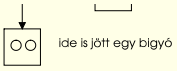
1. példa: Működés után:



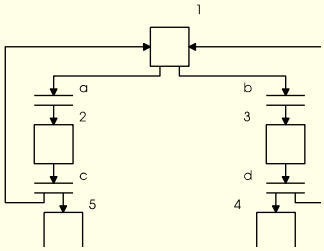


2. példa: Működés után:





3. példa: (kielégíti az 1. részfeladat feltételeit)



Feltéve, hogy kezdetben az "1" csomópontban van egy bigyó, az "a" vagy a "b" átmenet aktivizálódhat (versenyhelyzet). Ha az egyik aktivizálódik, akkor a másik már nem tud, tehát egy bigyó kerül a "2" vagy a "3" csomópontba. Így vagy a "c", vagy a "d" átmenet válhat aktívvá a következő lépésben. A "c" vagy a "d" aktivizálása visszaviszi a bigyót az "1" csomópontba, és minden kezdődhet elölről. A "4" és az "5" csomópontokban a bigyók csak gyűlnek-gyűlnek, és azt számlálják, hogy hányszor volt aktív az "a", ill. a "b" átmenet.

Elérhető összpontszám: 100 pont

A verseny végeredménye:

I. kategória

1. Birszki Bálint	Ipari Szakközépiskola, Vác
2. Szűcs Gábor	Radnóti Miklós Gimnázium, Szeged
3. Késmárki Attila	Teleki Blanka Gimnázium, Székesfehérvár
Szabó Balázs	Vetési Albert Gimnázium, Veszprém
5. Bencsáth Boldizsár	Budai Nagy Antal Gimnázium, Budapest
6. Varga Zoltán	Neumann János Szakközépiskola, Budapest
7. Molnár Dániel	Radnóti Miklós Gimnázium, Szeged
Blahut György Gábor	I. István Gimnázium, Budapest
9. Kárpáti Attila	Fazekas Mihály Gimnázium, Budapest
10. Horváth Péter	Fazekas Mihály Gimnázium, Budapest

II. kategória

1. Kiss Róbert	Révai Miklós Gimnázium, Győr
2. Szatmáry Zoltán	Berzsenyi Dániel Gimnázium, Budapest
Sziva Miklós	Révai Miklós Gimnázium, Győr
4. Urbán Péter	Fazekas Mihály Gimnázium, Budapest
5. Vizmathy Tamás	Egressy Gábor Szakközépiskola, Budapest
Szász Olivér	Berzsenyi Dániel Gimnázium, Budapest
7. Bajusz Péter	Fazekas Mihály Gimnázium, Budapest
8. Jüttner Alpár	Lovassy László Gimnázium, Veszprém
9. Salamon András	Zrínyi Miklós Gimnázium, Zalaegerszeg
10. Tóth Bálint	Berzsenyi Dániel Gimnázium, Budapest
Molnár Lajos	KLTE Gyakorló Gimnázium, Debrecen

1993. Első forduló

Első-második osztályosok

1. feladat: Szövegelés (10 pont)

Számítógépünk szövegekkel (TEXT) mindössze négyféle műveletet tud végezni:

1) Össze tudja hasonlítani szoveg1-et szoveg2-vel: szoveg1 = szoveg2.
2) Visszaadja szoveg első karakterét: elso(szoveg).
3) Visszaadja szoveg maradékát az első karaktere nélkül: maradek(szoveg).
4) szoveg2-t szoveg1-hez fűzi: fuz(szoveg1, szoveg2).

Pisti barátunk két új eljárást írt. Sajnos, sürgősen el kellett utaznia, nem maradt ideje az eljárások működésének ismertetésére. Rád vár a feladat, hogy megfejtsd, mi az eredménye Pisti eljárásainak, ha meghívják őket!

```
TEXT PROC elj1(TEXT CONST szoveg1,szoveg2):
  IF szoveg1=""
  THEN szoveg2
  ELSE
     elj1(maradek(szoveg1),fuz(elso(szoveg1),szoveg2))
  ENDIF
ENDPROC elj1;
```

```
TEXT PROC elj2(TEXT CONST szoveg):
  IF szoveg=""
  THEN ""
  ELSE fuz(elj2(maradek(szoveg)),elso(szoveg))
  ENDIF
ENDPROC elj2;
```

2. feladat: Portia ládikái (13 pont)

Shakespeare "A velencei kalmár" című vígjátékában Portia férjhez akar menni, de okos férjet szeretne. Van 3 ládikája: egy arany, egy ezüst és egy ólom. Az egyikbe beleteszi a saját képét, majd mind a három ládikára a kép helyére utaló igaz vagy hamis állításokat ír. A kérőknek az állítások és a "játékszabályok" alapján meg kell mondaniuk, hogy melyik ládikában van a kép. Az állítások pl. ilyenek lehetnek:

Az arany ládikán: A kép az arany ládikában van. A kép az ólom ládikában van.
Az ezüst ládikán: A kép az arany ládikában van. A kép az ólom ládikában van.
Az ólom ládikán: A kép az ezüst ládikában van. A kép az ólom ládikában van.

A fenti, ún. tényállítások PROLOG nyelven:

```
aranyládikán(aranyban). aranyládikán(ólomban).
ezüstládikán(aranyban). ezüstládikán(ólomban).
ólomládikán(ezüstben).  ólomládikán(ólomban).
```

Portia játékszabályként rendszerint azt mondja a kérőknek, hogy mind a három ládikára írt igaz állítást. Ehhez a játékszabályhoz a kép helyét – a kérők helyett – a következő PROLOG program találja ki:

```
háromigaz(X) ha aranyládikán(X) és ezüstládikán(X) és ólomlá-
dikán(X).
```

Most a háromigaz(X) kérdésre X=ólomban választ kapunk. X=aranyban választ akkor kapnánk, ha a fenti tényállítások között lenne aranyládikán(aranyban), ezüstládikán(aranyban) és ólomládikán(aranyban) is.

Néhanapján Portia megváltoztatja a játékszabályokat, pl. azt mondja a kérőknek, hogy a három ládika egyikén sincs igaz állítás, hogy legalább kettőn van igaz állítás stb.

a) Mondd meg, milyen játékszabálynak felel meg az alábbi PROLOG program!

```
nevezdel(X) ha ezüstládikán(X) és nem(ólomládikán(X)) és
nem(aranyládikán(X)) vagy aranyládikán(X) és nem(ólomládi-
kán(X)) és nem(ezüstládikán(X)) vagy
ólomládikán(X) és nem(ezüstládikán(X)) és nem(aranyládi-
kán(X)).
```

Itt pl. a nem(ólomládikán(X)) állítás X=aranyban esetén akkor teljesül, ha a tényállítások között nincsen ólomládikán(aranyban).

b) Írj egy-egy PROLOG programot az alábbi játékszabályokhoz!

- Legalább két ládikán van igaz állítás.
- Legalább egy ládikán van és legalább egy ládikán nincs igaz állítás.

3. feladat: Táblázatkezelő (11 pont)

Egy táblázatkezelő programról a következőket kell tudni:

- ún. cellákból áll, amelyek sorokba és oszlopokba vannak rendezve;
- a sorokat egész számok, az oszlopokat betűk azonosítják (1-től, illetve A-tól kezdve);
- minden cellába egyetlen dolog írható: vagy egy képlet, vagy egy szám;
- a képletekben hivatkozni lehet bármely cellára, a saját cellára is;
- cellára hivatkozni a cella oszlop- és sorazonosítójával lehet, pl. A1, Q55;
- a cellák értékét A1 A2 A3 ... B1 B2 B3 ... sorrendben számolja ki.

Adva van a következő program:

```
A1: ...
A2: Ha A1=0 akkor 0      B2: Ha A1=0 akkor 1
    különben A3              különben B3
A3: A2+1                 B3: A3*B2
```

a) Mit ad eredményül a program az A3 és B3 cellákban, ha az A1 helyre 0-t írunk és kiszámoltatjuk a képleteket, majd az A1 helyre 1-et írunk és további n-szer kiszámoltatjuk a képleteket? A3 és B3 mellett add meg A2 és B2 értékét is mind a két esetben!

b) Írj az A3 és B3 cellákban azonos eredményt adó programot, csak egyszerűbben (legfeljebb két képlettel)!

4. feladat: Bitvektorok (7 pont)

Legyenek A, B és C bitvektor-, p, q, r, s és t bit-, i pedig egész típusú változók, továbbá h – a bitvektor hossza – egész típusú állandó ($h \geq 1$). (Bitvektor: olyan egydimenziós tömb, amelynek bitek az elemei.) Legyenek továbbá A és B elemei egy-egy nemnegatív bináris egész szám bitjei. Tekintsük az alábbi programrészletet:

```
BEGIN
  t:=0;
  FOR i:=h DOWNTO 1 DO
  BEGIN
     p:=A[i];
     q:=B[i];
     s:=p XOR q;
     r:=s XOR t;
     t:=(p AND q) OR (s AND t);
     C[i]:=r
  END
END;
```

Bitműveletek:

		AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

a) Mit tartalmaz a végrehajtás után a C vektor?

b) Mi a t bitváltozó szerepe a ciklusban?

c) Mit fejez ki t értéke a programrészletből való kilépéskor?

5. feladat: LOGO (10 pont)

Mit rajzol az alábbi LOGO nyelvű program, ha az ELSO eljárást meghívjuk az N, M, X, Y paraméterekkel, és a teknőc észak felé indul el? Rajzold le, és írd be a rajzba a megfelelő helyre a jellemző adatokat (vonalhossz, ismétlődési szám stb.)!

Utasítások magyarázata:

PENUP :	ettől kezdve mozgás közben nem rajzol,
PENDOWN :	ettől kezdve rajzol,
FORWARD n, BACK n:	az aktuális helyről az aktuális irányban n egységnyit lép előre, illetve hátra,
LEFT f, RIGHT f :	az aktuális irányhoz képest f fokkal elfordul balra, illetve jobbra,
REPEAT n [ut] :	az ut utasítássorozatot n-szer megismétli.

```
ELSO N M X Y      ← eljárásnév és paraméterei
  PENUP
  REPEAT M [MASODIK N X Y LEFT 90 FORWARD Y RIGHT 90]
END               ← eljárás vége

MASODIK N X Y
  REPEAT N [HARMADIK X Y FORWARD X] BACK N*X
END
```

```
HARMADIK X Y
  PENDOWN
  REPEAT 2 [FORWARD X RIGHT 90 FORWARD Y/3 RIGHT 60
            FORWARD Y/3 LEFT 120 FORWARD Y/3 RIGHT 60
            FORWARD Y/3 RIGHT 90]
  PENUP
END
```

6. feladat: Lnko (12 pont)

Az alábbi négy algoritmus két szám legnagyobb közös osztójának meghatározására készült. (I|A jelentése: I osztója A-nak.)

a) Melyek helyesek, melyek hibásak közülük? (Logikai hibát keress!)

b) A hibásak mit csinálnak?

```
A)Lnko(A,B):
     Ciklus amíg A<>B
        Ha A>B akkor  A:=A-B
             különben B:=B-A
        Ciklus vége
     Lnko:=A
  Eljárás vége
```

```
B)Lnko(A,B):
     X:=1: H:=min(A/2,B/2)
     Ciklus I=2-től H-ig
        Ha I|A és I|B akkor X:=I
     Ciklus vége
     Lnko:=X
  Eljárás vége.
```

```
C)Lnko(A,B):
     Ha A<B akkor Csere(A,B)
     Ciklus amíg B<>0
        X:=A MOD B: A:=B: B:=X
     Ciklus vége
     Lnko:=A
  Eljárás vége.
```

```
D)Lnko(A,B):
     X:=A: Y:=B
     Ciklus amíg X<>Y
        Ha X<Y akkor  X:=X+A
              különben Y:=Y+B
     Ciklus vége
     Lnko:=X/(A*B)
Eljárás vége.
```

7. feladat: Bagdadi tolvaj (10 pont)

Tűz ütött ki a bagdadi felhőkarcolóban. A 150 emeletes toronyépületben minden emeleten arany-érmek vannak, zsákokban; az emeleteket 1-től 150-ig számozzák. Az alábbi programrészlet feladata az, hogy kiszámítsa, maximum hány aranyat 'lovasíthat meg' a tolvaj élete kockáztatása nélkül, a következő feltételek mellett:

1. A tolvaj felvonóval közlekedik az emeletek között. A felvonó 10 emeletet tesz meg percenként. A felvonó nem mehet át olyan emeleten, ahol tűz van.

2. A tűz kitörésekor a felvonó az 1. emeleten áll. A tűz lefelé terjed, percenként egy-egy emeletet tesz meg. A tolvaj élete veszélybe kerül, ha a tűz eléri az emeletet, amelyen éppen a zsákkal bíbelődik.

3. A tolvajnak emeletenként 10 másodpercre van szüksége a zsák arany elhozására. Mihelyst megkaparintotta a zsákot, azonnal elhagyja az emeletet.

A bemeneti adatok között az első annak az emeletnek a sorszáma, ahol kiütött a tűz; ezután számpárok következnek, a sorozatot 0 zárja. A számpárok első tagja egy-egy emelet sorszáma, a második pedig az ott található aranyak száma.

```
  VARtoronyhaz: ARRAY [1..150] OF INTEGER;
        tuzKezdete,felsoEmelet,aranyakSzama,i: INTEGER;
BEGIN
  FOR i:=1 TO 150 DO toronyhaz[i]:=0;
  read(tuzKezdete,i);
  WHILE i<>0 DO read(toronyhaz[i],i);
                      (* töltsd meg a házat arannyal *)
  felsoEmelet:=(30*tuzKezdete-4) DIV 33;
                      (* meddig lehet felmenni? *)
  aranyakSzama:=0;  (* számold össze az aranyakat *)
  FOR i:=1 TO felsoEmelet DO
        aranyakSzama:=aranyakSzama+toronyhaz[i];
  writeLn(aranyakSzama) (* írd ki az eredményt *)
END.
```

Kérdések:

a) Mi van akkor, ha a számpárokat nem az emeletek növekvő sorrendjében adjuk meg?

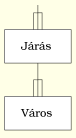
b) Mi van akkor, ha egy emelet sorszáma több számpár első tagjaként is előfordul?

c) Mi van akkor, ha egy emelet sorszáma egyetlen számpárban sem fordul elő?

d) A (30*tuzKezdete-4) DIV 33 kifejezésben az egyik állandó hibás. Melyik az, mi a helyes értéke, és miért?

e) Ha a tűz a 32. emeleten üt ki, melyik emeleten láthat "munkához" a tolvaj?

8. feladat: Karaván (7 pont)



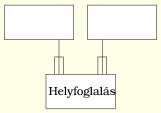
Egy számítógépes adatbázis létrehozásához az adott feladatban előforduló dolgokat (az ún. egyedeket) osztályozzuk, azaz halmazokba (ún. egyedtípusokba) soroljuk. Az egyedtípust téglalappal, az egyedek közötti kapcsolatokat pedig a téglalapokat összekötő vonalakkal ábrázoljuk. Háromfelé ágazó vonal, ún. "szarkaláb" jelöli, ha egy egyed több más egyeddel is kapcsolatban lehet.

Nézzük a következő példát!

A megyék halmazában megyék, a járásokéban járások, a városokéban városok vannak. A megyék halmazából kiválasztott tetszőleges megyében több járás is lehet, de minden egyes járás csak egy megyéhez tartozhat. Ehhez hasonlóan egy járásban több város is lehet stb. A megyék és a városok kapcsolatát nem jelöljük külön vonallal, mert egy megyében a városok halmaza ugyanaz, mint az adott megyéhez tartozó járásokban lévő városok halmaza.

A sivatagi karavántúrákat szervező KALEF Kft. a következő leírást adta számítógépes nyilvántartó rendszerének megtervezéséhez: "Minden karaván tapasztalt túravezető kísér. A karavánokra a turisták helyet foglalhatnak. A karaván különböző időpontokban indulnak, ezért egy vezető több

karaván is vezethet, és egy turista több karaványra is jelentkezhet." Másold le az alábbi ábrát, és egészítsd ki a hiányzó nevekkel és összekötő vonalakkal!



Elérhető összpontszám: 80 pont

Harmadik-ötödik osztályosok

1. feladat: Biokertészet (13 pont)

A következő PROLOG program a biokertészeket segíti: a növényekről beletáplált tudás alapján eldönti, hogy egyes vethető növényeket milyen sorrendben ültessenek egymás mellé, ha mindegyikből egy-egy sort szeretnének. Ha lehet, olyan növényeket kell egymás mellé ültetni, amelyek védik egymást. Ha ez nem megy, akkor olyanokat, amelyek szeretik egymást, és ha ez sem lehetséges, akkor olyanokat, amelyek legalább nem ellenségei egymásnak.

Gyomnövények:

```
gyom(pipacs).
gyom(csalán).
```

Vethető növények:

```
vethető(karalábé).
vethető(hagyma).
vethető(bab).
vethető(paradicsom).
vethető(retek).
```

A vethető növények közötti kapcsolatok:

```
védi(hagyma,paradicsom).
védi(paradicsom,hagyma).
védi(paradicsom,karalábé).
védi(karalábé,paradicsom).
```

```
szereti(paradicsom,bab).
szereti(bab,paradicsom).
szereti(karalábé,bab).
szereti(bab,karalábé).
szereti(retek,bab).
szereti(bab,retek).
```

```
ellenség(hagyma,karalábé).
ellenség(karalábé,hagyma).
ellenség(karalábé,retek).
ellenség(retek,karalábé,).
ellenség(bab,hagyma).
ellenség(hagyma,bab).
ellenség(hagyma,retek).
ellenség(retek,hagyma).
```

Egy lehetséges vetési sorrendet megadó program:

```
vetés(A,B,C,D,E) ha szomszéd(A,B) és szomszéd(B,C) és
  A<>C és szomszéd(C,D) és A<>D és B<>D és szomszéd(D,E)
  és A<>E és B<>E és C<>E.
```

A szomszéd(A,B) formula igaz értéket ad eredményül, ha talál olyan A és B, egymástól különböző, vethető növényeket, hogy A és B védik vagy szeretik egymást, vagy legalább nem ellenségei egymásnak. Ezután kerül sor a szomszéd(B,C) formulára, amely az előbb kapott B-hez egy új C-t keres s.í.t. Ha az adott B-hez nincsen olyan C, amelyre a szomszéd(B,C) formula igaz lenne, akkor a PROLOG rendszer az előző szomszéd(A,B) formulát újra kiértékelve előállít egy másik A,B párt, és a kiértékelést folytatja a szomszéd(B,C) formulával.

a) Írd fel a szomszéd(X,Y) formulát (felhasználható műveletek: és, vagy, nem)!

b) Add meg, hogy a fenti tudás alapján a felsorolt növények milyen sorrendben vethetők!

2. feladat: Sorozatok (16 pont)

Egy programnyelv sorozatokra a következő függvényeljárásokat definiálja (pl. a STRING típust tekinthetjük karakterek sorozatának):

első(S): az S sorozat első eleme,

elsőutániak(S): az S sorozat elsőt követő elemei,

elejére(E,S): az S sorozat az első eleme elé illesztett E elemmel együtt,

ürese(S): igaz, ha az S sorozat üres,

üres: az üres sorozat.

a) Mit csinál a következő két függvényeljárás?

```
példa1(S):   első(elsőutániak(S)).

példa2(E,S): Ha ürese(S) akkor HAMIS
             különben Ha E=első(S) akkor IGAZ
             különben példa2(E,elsőutániak(S)).
```

b) Az első, elsőutániak, elejére, ürese és üres függvényeljárások felhasználásával írd meg rekurzív függvényeljárásként az alábbiakat:

utolsó(S): az S sorozat utolsó eleme,

utolsóelőttiek(S): az S sorozat utolsót megelőző elemei,

végére(S,E): az S sorozat az utolsó eleme mögé illesztett E elemmel együtt.

3. feladat: Táblázatkezelő (9 pont)

Egy táblázatkezelő programról a következőket kell tudni:

- ún. cellákból áll, amelyek sorokba és oszlopokba vannak rendezve;
- a sorokat egész számok, az oszlopokat betűk azonosítják (1-től, illetve A-tól kezdve);
- minden cellába egyetlen dolog írható: vagy egy képlet, vagy egy szám;
- a képletekben hivatkozni lehet bármely cellára – a saját cellára is;
- cellára hivatkozni a cella oszlop- és sorazonosítójával lehet, pl. A1, Q55;
- a cellák értékét A1 A2 A3 ... B1 B2 B3 ... sorrendben számolja ki.

Adva van a következő program:

A1: 0	B1: Ha sgn(A6)=sgn(A4) akkor A3 különben A1	C1: ...
A2: 9	B2: Ha sgn(A6)=sgn(A5) akkor A3 különben A2	C2: ...
A3: (A1+A2)/2	B3: (B1+B2)/2	C3: ...
A4: +A1*A1-4	B4: +B1*B1-4	C4: ...
A5: +A2*A2-4	B5: +B2*B2-4	C5: ...
A6: +A3*A3-4	B6: +B3*B3-4	C6: ...

A további (C, D stb.) oszlopokba írt képletek értelemszerűen a B-beliek módosításával kaphatók meg; a C-beliek például úgy, hogy a B-k helyébe C-t, az A-k helyébe B-t írunk.

a) Mit számol ki ez a program?

b) Közelítőleg milyen értéket kapunk J3-ban?

4. feladat: Számvektorok (15 pont)

A következő algoritmus számvektorokban (egydimenziós tömbökben) tárolt egész számokat szoroz össze. A szorzandók az A() és a B(), 0-tól N-ig, ill. M-ig indexelt vektorokban vannak (N≥M), az eredményt a C() 0-tól K-ig indexelt vektorba kell tenni, a lehető legkevesebb művelet felhasználásával. Pl. az A() vektorban tárolt szám így értelmezendő:

$$a_0 + a_1*10 + a_2*10^2 + \ldots + a_n*10^n, \text{ ahol } a_n \neq 0.$$

Egészítsd ki az algoritmust a ?x?-lel megjelölt helyeken! (A közölt részletek elhagyása esetén nem jár pont.)

```
Szoroz(N,A,M,B,K,C):
  ÁTVITEL:=?1?
  Ciklus I=0-tól ?2?-ig
     C(I):=ÁTVITEL
     Ciklus J=max(0,I-M)-től ?3?-ig
        C(I):=C(I)+A(J)*B(I-J)
     Ciklus vége
     ÁTVITEL:=?4?
     C(I):=?5?
  Ciklus vége
  Ha ?6?  akkor  K:=N+M+1: C(K):=?7?
         különben K:=N+M
Eljárás vége.
```

5. feladat: Csak formálisan! (17 pont)

Egy nyelv formális leírására az alábbi szabályokat adjuk meg. A definiálandó fogalmakat <>-ek közé tettük, ::= vezeti be a definíciót, a | (vonás) vagy-kapcsolatot jelöl, a {}-ek a közéjük zártak ismétlődését jelentik legalább egyszer, a ... (3 pont) pedig a sorozat folytatódására utal.

```
<mondat> ::= { <elem> { <szóköz> } }.
```

Jelentése: a mondat elemek legalább egytagú sorozata, minden elem után legalább egy szóköz áll, és a mondatot pont zárja.

```
<elem> ::= <szó> | <szám> | <írásjel>
```

Jelentése: az elem szó vagy szám vagy írásjel.

```
<szó> ::= { <betű> }
```

Jelentése: a szó egy sorozat, amely legalább egy betűből áll.

```
<szám> ::= { <számjegy> }
<írásjel> ::= , | ;
<betű> ::= a | b | ... | z
<számjegy> ::= 0 | 1 | ... | 9
```

a) A fenti szabályok szerint hol és miért hibás ez a mondat:

ez egy „; 8szavas mondat : szava ez, de a -1 nem .

b) Módosítsd úgy a definíciót, hogy számnak lehessen előjele, és fixpontos valós alakban is fel lehessen írni (pl. +12.34)!

c) Egészítsd ki az alábbi programot a hiányzó Szóolvasás, Számolvasás, Írásjelolvasás eljárásokkal úgy, hogy a definíció szerint helyes mondatot elemekre bontson!

```
Elemző(P):
  i:=1
  Ciklus amíg i≤hossz(P)
     Elágazás
        P(i) ∈ Betűk      esetén Szóolvasás(P,i,SZO)
                                 Ki: SZO
        P(i) ∈ Számjegyek esetén Számolvasás(P,i,SZA)
                                 Ki: SZA
        P(i) ∈ Írásjelek  esetén Írásjelolvasás(P,i,IR)
                                 Ki: IR
        P(i)=Szóköz       esetén i:=i+1
     Elágazás vége
  Ciklus vége
Eljárás vége.
```

6. feladat: Turing-gép (12 pont)

A Turing-gép egy szalagból és egy író-olvasó fejből áll. A fej olvasás közben jobbra-balra mozoghat a szalagon, ha pedig nem mozog, akkor írhat rá. A szalagra írható, illetve onnan visszaolvasható jelek halmazát ábécének nevezzük.

A gép tetszőleges számú különböző állapotban lehet. A működését leíró program a gép minden lehetséges állapotához megadja, hogy a szalagról beolvasott jeltől függően melyik állapotba kerüljön, és a fej mit csináljon. A fejnek a következő parancsok adhatók: Jobbra lép (J), Balra lép (B), Jelet ír (jel). Ha valamelyik állapotban a beolvasott jelhez nincs parancs megadva, a gép megáll.

Pl. ábécé: {0,1,2}

q1:1 → q1:2 Ha a gép a q1 állapotban 1-et talál a szalagon, állapota nem változik meg, de az 1 helyére 2-t ír.

q1:2 → q1:J Ha a q1 állapotban 2-t talál a szalagon, jobbra lépteti a fejet, és q1 állapotban marad.

Feltéve, hogy kezdetben a q1 állapotban van, e program (szabályhalmaz) hatására a gép a fej kezdeti pozíciójától jobbra eső összes 1-et átírja 2-vé, amíg csak 0-t nem talál, és akkor megáll.

Tegyük föl, hogy a nemnegatív egész számokat a következőképpen ábrázoljuk:

```
  0: 1, 1: 11,  2: 111,  ...,  12: 1111111111111 stb.,
```

két ilyen számot pedig *-gal választunk el egymástól, pl. 11111*11.

Gépünk a q1 állapotban indul, a fej balról az első 1-en áll, és a szalagon két szám van, egymástól *-gal elválasztva:

```
  ...00011111*111111111000...
```

A szalag fenti tartalma mellett a gép a következő programot hajtja végre:

```
ábécé: {0,1,*}
     q1:1 → q1:J
     q1:* → q2:1
     q2:1 → q2:J
     q2:0 → q3:B
     q3:1 → q3:0
     q3:0 → q4:B
     q4:1 → q4:0
```

Válaszolj az alábbi kérdésekre:

a) Mi a fenti program végrehajtásának eredménye a megadott számokkal?

b) Hol áll a fej a program befejeződése után?

c) Mi a program eredménye két tetszőleges nemnegatív egész szám esetén?

d) Mi az eredmény akkor, ha a szalagon kezdetben nem két, hanem csak egy nemnegatív szám van, pl. ...0011111100...? Hol áll most a fej a program befejeződése után?

7. feladat: Assembly (12 pont)

Assembly programot írtunk morzejelek vételére. A morzejeleket ember adja, így az egyes jelek és a szünetek időtartama kis mértékben változhat. Ha a morzejel-bemeneten jelet érzékelünk, csupa 1-et tartalmazó byte-ot olvasunk be, ellenkező esetben csupa 0-t. Feltesszük, hogy a regiszterek nem csordulnak túl. Az alábbi programrészlet alapján válaszolj a kérdésekre!

```
     XOR BX,BX          ; BX := BX XOR BX (bitenkénti kizáró 'vagy')
C1:  INC BX             ; BX növelése 1-gyel
     IN AL,morzeport    ; 1 byte beolvasása a morzejel-bemenetről
     OR AL,AL           ; AL := AL OR AL
     JNZ C1             ; ugrás, ha az előző művelet eredménye nem 0
     MOV AX,P           ; az AX regiszterbe tölti a P változó értékét
     MOV CX,Q
     SUB AX,BX          ; AX := AX–BX
     NEG AX             ; AX := –AX
     SUB CX,BX
     JS C2              ; ugrás, ha az előző művelet eredménye negatív
     CMP AX,CX          ; összehasonlítás
     JG C2              ; ugrás, ha az előző műveletben AX > CX
     MOV P,BX
     MOV DL,0
     JMP C3             ; ugrás C3-ra
C2:  MOV Q,BX
     MOV DL,1
C3:
```

a) Hogyan különbözteti meg egymástól a vevő a rövid és a hosszú jeleket?

b) Mire szolgál a C1 címkénél kezdődő ciklus?

c) Mi a program eredménye, és hol keletkezik?

d) Milyen esemény bekövetkezése esetén kerül sor a JS C2, ill. a JG C2 ugrásokra?

e) Hogyan változnak a P és Q változók?

8. feladat: Monoton (6 pont)

a) Mi van h-ban az alábbi programrészlet végrehajtása után, ha a b[1..n] tömb elemei monoton növekvő sorozatot alkotnak?

```
  VAR i,j,h: INTEGER;
BEGIN
  i:=1; h:=1;
  WHILE i<n DO
     BEGIN
        j:=i+1;
        WHILE (j<=n) AND (b[i]=b[j]) DO j:=j+1;
        IF (h<j-i) THEN h:=j-i;
        i:=i+1
     END
END.
```

b) Írj a fentivel azonos eredményt adó programot, amely csak két változót használ (pl. i-t és h-t), és csak egyszer pásztáz végig a tömbön!

Elérhető összpontszám: 100 pont

1993. Második forduló

Első-második osztályosok

Gyűrűhálózat szimuláció:

Egy számítógéphálózat számítógépeit gyűrű alakban kapcsoltuk össze. A gyűrű N (≤4) szakaszból áll, s mindegyik szakasz végén van egy számítógép (sorszáma a szakasz sorszámával azonos).

Az egyes gépek üzeneteket küldhetnek egymásnak, amelyek a gyűrűben növekvő sorszámú szakaszok felé mozoghatnak, az utolsóból az elsőbe. Az üzenetek tetszőleges darabszámú csomagból állhatnak, ahol egy csomag szerkezete a következő:

- a címzett sorszáma,
- a feladó sorszáma,
- üzenet típus (NORMÁL, UTOLSÓ)

  (a legtöbb üzenetcsomag típusa NORMÁL, az utolsó csomagé UTOLSÓ)
- üzeneten belüli csomagsorszám,
- a csomagban levő szöveg (mindig 8 karakter),
- ellenőrző összeg (a szöveg karakterei kódjából kizáró vagy művelettel számolt érték).

Az üzenetek egy időegység alatt egy szakaszt tesznek meg, egy szakaszban legfeljebb egy üzenet lehet. Az egyes gépek véletlenszerűen indítanak üzeneteket. Minden gép csak a neki szóló üzeneteket veheti el a gyűrűből, a másoknak szólót tovább kell engednie.

Az csomagok szövege véletlenszerűen megváltozhat, s akkor az ellenőrző összeg alapján a vevő észlelheti a hibát, s a képernyőre kijelzi.

Készíts programot, amely beolvassa a hálózat konfigurációját (hány szakasz van), majd szimulálja a hálózat működését, azaz

A. Kezdőbeállítás: (13+12 pont)

A futás előtt véletlenszerűen generál 30 időegységre üzeneteket, vagy a felhasználó adhatja meg, hogy mikor, ki, kinek küldjön üzenetet (mindenki csak egyvalakinek küldhet, maximum 5 hosszúságút).

B. A forgalom követése: (25 pont)

- megjeleníti az egyes szakaszokon mozgó csomagokat,
- gyűjti a gépekhez érkezett csomagokat (az utolsó 5-öt őrzi meg),
- futás közben a megőrzött csomagok elolvashatók.

C. Hibagenerálás és -figyelés: (13 pont)

- az egyes csomagokhoz generálja az ellenőrző összeget,
- véletlenszerűen elrontja az csomagokat, (paraméter legyen egy hibázási valószínűség, pl. egy időegység alatt egy csomag 1% eséllyel romlik el)
- ellenőrzi az csomagok helyességét, s ha szükséges, akkor hibajelzést ad.

D. Futás végi statisztika: (12 pont)

- statisztikát ad az észlelt hibákról (hány volt, mely szakaszokon voltak),
- az egyes szakaszok foglaltságáról (t időegység alatt hány üzenet haladt át rajta).

E. A program minőségéért 15 pont adható.

Egy lehetséges képernyőterv (a gépek kapott, illetve küldendő csomagjai jobb elhelyezhetőség miatt lehetnek a képen tömörebben, pl. egy sorban kettő csomag):

Aktuális üzenet	Kapott, küldendő üzenetek: idő,(kitől vagy kinek,sorszám,szöveg)
idő=10	
1. szakasz: 0	kapott: nincs küld: 11,(2,2,"második ") 12,(2,3,"harmadik")
2. szakasz: 1,2,NORMÁL,1,"első ",ell.	kapott: nincs küld: 15,(1,1,"1 csomag")
3. szakasz: 0	kapott: nincs küld: nincs
4. szakasz: 3,4,UTOLSÓ,4,"sz4 ",ell.	kapott: (3,1,"sz1 ") (3,2,"sz2 ") (3,3,"sz3 ") küld: nincs

Elérhető összpontszám: 90 pont

Harmadik-ötödik osztályosok

Szövegformázó:

A feladat: szövegformázó program készítése, amely beolvas egy szöveget egy file-ból, s azt formázva kiírja a képernyőre, illetve egy másik file-ba!

A program a beolvasott szöveg minden egyes bekezdését egy feltételezett forma szerint kiírja, majd vár arra, hogy a felhasználó valamilyen funkcióbillentyű lenyomásával átformáztassa, vagy pedig jóváhagyja a kiírt formátumot. A bekezdés végét egy sorvég karakter jelzi. A formázási követelmények:

A. Tördelés: (20 pont)

- a szöveget úgy törje sorokra, hogy egy szót ne kelljen két sorba írni,
- a képernyőn két hasáb van (egyforma szélesek), amelyekben a program oszlopfolytonosan helyezi el a szöveget, (előbb a baloldalit egy lapon teljesen kitölti, s utána tér át a jobboldalira)
- egy lap mérete megegyezik a képernyőn látható sorok számával, a szöveg természetesen több lapos is lehet.

B. Bekezdések formázása: (30 pont)

- egy bekezdést formázhatunk úgy, hogy minden sorát a baloldali margóhoz igazítjuk,
- egy bekezdést formázhatunk úgy, hogy minden sorát a jobboldali margóhoz igazítjuk,
- egy bekezdést formázhatunk úgy, hogy minden sorát középre igazítjuk,
- egy bekezdést formázhatunk úgy, hogy minden sorát mindkét margóhoz igazítjuk szóközök egyenletes elhelyezésével a szavak között,
- egy bekezdést formázhatunk úgy, hogy minden sorát mindkét margóhoz igazítjuk szóközök elhelyezésével a szavak között úgy, hogy az egymás alatti sorokban minél kevesebb szóköz legyen egymás alatt,
- egy bekezdést formázhatunk úgy, mint az előző bekezdést.

C. Margók állítása: (8 pont)

- bekezdésenként lehessen átállítani a bal-, illetve a jobbmargót (csak szűkebbre vehető a hasáb méreténél),

D. Rejtett elválasztás: (12 pont)

- HA AZ EDDIGI FELADATOKKAL KÉSZEN VAGY: a szövegben levő ~ karakterek rejtett elválasztások, a képernyőn nem látszanak, de ha egy szó belsejébe tesszük, akkor a szó ott elválasztható.

E. Input-output: (15 pont)

- beolvasás file-ból,
- kiírás file-ba, illetve képernyőre,
- a képernyőre kiírt bekezdés újraformázása (akárhányszor) funkcióbillentyűkkel, a felhasználó jóváhagyásáig.

F. A program minőségéért 15 pont adható.

Feltételezett értékek: a hasábszélességek: 1..39, 41..79
sorokra kell tördelni, balmargóhoz igazítással

Ha egy hasáb egy sorában csak egyetlen szó fér el, vagy esetleg egy szó hosszabb, mind a hasáb szélessége, akkor az összes fenti szabály (értelmesen) felülbírálható.

Elérhető összpontszám: 100 pont

A verseny végeredménye:

I. kategória

1. Ali György	Neumann János Szakközépiskola, Eger
2. Kovács Gábor	Radnóti Miklós Gimnázium, Budapest
3. Blahut György Gábor	Szent István Gimnázium, Budapest
Nagy Péter	Neumann János Szakközépiskola, Budapest
5. Király Attila	Ipari Szakközépiskola, Vác
6. Gosztolya Gábor	Ságvári Endre Gimnázium, Szeged
7. Szabó Balázs	Vetési Albert Gimnázium, Veszprém
8. Papp László	Radnóti Miklós Gimnázium, Szeged
9. Fige Péter	Hermann Ottó Gimnázium, Miskolc
10. Bencsáth Boldizsár	Budai Nagy Antal Gimnázium, Budapest

II. kategória

1. Papp Zsombor	Zrínyi Miklós Gimnázium, Zalaegerszeg
Valkó László	Karinthy Frigyes Gimnázium, Budapest
3. Németh István	Radnóti Miklós Gimnázium, Szeged
4. Szász Olivér	Berzsenyi Dániel Gimnázium, Budapest
5. Molnár Lajos	KLTE Gyakorló Gimnázium, Debrecen
6. Zahorán Péter	Földes Ferenc Gimnázium, Miskolc
7. Jüttner Alpár	Lovassy László Gimnázium, Veszprém
8. Megyaszai Sándor	Fazekas Mihály Gimnázium, Debrecen
9. Vizmathy Tamás	Egressy Gábor Szakközépiskola, Budapest
Sinkovics Zoltán	Vajda János Gimnázium, Szarvas
Molnár György	PA Rt. Energetikai Szakképzési Intézet, Paks

1994. Első forduló

Első-második osztályosok

1. feladat: Az óra csak jár (14 pont)

Stopper kezelésére írtunk 3 eljárást. A gettime(or,p,mp,szmp) függvény a pontos időt adja vissza a paraméterként kapott változókban (óra, perc, másodperc, századmásodperc). Egy rögzített `I` paramétert figyelembe véve a függvények meghívásának sorrendje a következő:

```
Első(I); Második(I); Harmadik(I); Második(I); Harmadik(I); ...
stb.
```

a) Melyiknek mi a feladata?

b) Miért használunk tömböket és mire szolgálnak a K(), E(), V() tömbök?

c) Bizonyos esetekben a 2. eljárás hibás eredményt számol. Melyek ezek, s hogyan lehetne kijavítani a programot?

```
Első(I):
  E(I).or:=0; E(I).p:=0; E(I).mp:=0; E(I).szmp:=0
  gettime(or,p,mp,szmp)
  K(I).or:=or; K(I).p:=p; K(I).mp:=mp; K(I).szmp:=szmp
Eljárás vége.
```

```
Második(I):
  gettime(or,p,mp,szmp)
  E(I).szmp:=E(I).szmp+szmp-K(I).szmp
  E(I).mp:=E(I).mp+mp-K(I).mp
  E(I).p:=E(I).p+p-K(I).p
  E(I).or:=E(I).or+or-K(I).or
  V(I).or:=or; V(I).p:=p; V(I).mp:=mp; V(I).szmp:=szmp
Eljárás vége.
```

```
Harmadik(I):
  gettime(or,p,mp,szmp)
  K(I).or:=or; K(I).p:=p; K(I).mp:=mp; K(I).szmp:=szmp
Eljárás vége.
```

2. feladat: Táblatalálgatás (18 pont)

Egy táblázatkezelő programról a következőket kell tudni:

- ún. cellákból áll, amelyek sorokba és oszlopokba vannak rendezve; a sorokat egész számok (1-től), az oszlopokat betűk (A-tól) azonosítják;
- Minden cellába egyetlen szám vagy képlet és szám írható (utóbbi esetben a szám a cella kezdőértéke, a képlet pedig a további valahány kiszámításkor alkalmazandó formula;
- A képletekben bármely cellára hivatkozni lehet az adott cella oszlop- és sor-azonosítójával, pl. C5;
- A táblázatkezelő a cellák értékét minden lépésben balról jobbra haladva sor-oszlop sorrendben számolja ki, pl. A1, A2, ... B1, B2;
- If(f,A,B) feltételes kifejezés az f feltétel teljesülése esetén az A, egyébként pedig a B értéket jelenti.

a) Az A1, A2, A3, A4 cellák kezdőértéke a 0, 1, 1, X. (X tetszőleges pozitív szám.) A cellák következő értékeit kiszámító képletek:

```
A1: A1+1
A2: A2+2
A3: A2+A3
```

A képletek alkalmazását addig kell folytatni, amíg A3>A4 nem teljesül. Mi lesz A1, A2, A3 értéke?

b) Az A1, A2, A3 cellák kezdőértéke a 0, X, Y. (X, Y tetszőleges pozitív számok.) A cellák következő értékeit kiszámító képletek:

```
A1: If(páros(A3), A1, A1+A2)
A2: If(páros(A3), A2*2, A2)
A3: If(páros(A3), A3/2, A3-1)
```

A képletek alkalmazását addig kell folytatni, amíg A3=0 nem teljesül. Mi lesz ekkor A1, A2 értéke?

c) Az A1, A2, A3 cellák kezdőértéke az 1, X, Y. (X, Y tetszőleges pozitív számok.) A cellák következő értékeit kiszámító képletek:

```
A1: If(páros(A3), A1, A1*A2)
A2: If(páros(A3), A2*A2, A2)
A3: If(páros(A3), A3/2, A3-1)
```

A képletek alkalmazását addig kell folytatni, amíg A3=0 nem teljesül. Mi lesz ekkor A1, A2 értéke?

3. feladat: Flip-flop (18 pont)

Logikai áramköröket megadhatunk táblázattal vagy logikai formulával. A feladatban adattárolásra használható, tehát állapottal rendelkező alapáramköröket specifikálunk. Feltételezzük, hogy az áramkör n+1 időpontbeli állapotát a bemenőjelek és a kimenőjel n időpontbeli állapota határozza meg.

	T=h	T=i
yn=h	h	i
yn=i	i	h

Például a T tárolónak van egy bemenete (T) és egy kimenete (y). A táblázat a kimenet n+1 időpontbeli értékét tartalmazza. Logikai képletekkel ugyanez:

yn+1 = (yn és nem T) vagy (nem yn és T)

a) Adott az alábbi táblázat. Határozd meg a logikai képletet a lehető legkevesebb művelettel! Csak az *és*, *vagy*, *nem* műveleteket használhatod.

	R=h S=h	R=i S=h	R=h S=i	R=i S=i
yn=h	h	h	i	bármi
yn=i	i	h	i	bármi

b) Adott az alábbi táblázat. Határozd meg a logikai képletet a lehető legkevesebb művelettel! Csak az *és*, *vagy*, *nem* műveleteket használhatod.

	J=h K=h	J=i K=h	J=h K=i	J=i K=i
yn=h	h	i	h	i
yn=i	i	i	h	h

4. feladat: Fabejárás (10 pont)

Egy bináris fa rekurzív definíciója a következő:

```
fa = (balrészfa, gyökér, jobbrészfa)
```

Megengedjük azt is, hogy bármelyik részfa üres legyen (azaz ne tartalmazzon elemeket). Például egy 3 szintű teljes bináris fa a következő lehet:

```
      a
    /   \
   b     c
  / \   / \
 d   e f   g
```

Az ábrán egy 3 szintű, teljes bináris fa látható, a felső szinteken minden csúcsnak két részfája van, a legalsó szinten egy csúcshoz sem tartozik részfa (minden részfa üres).

Egy fa összes elemét írja ki a következő rekurzív program az ún. preorder bejárás szerint:

```
pre(fa):
  ha nem üres(fa) akkor írdki(gyökér),
                        pre(balrészfa),
                        pre(jobbrészfa).
```

A fenti fára a **pre** algoritmus a következő eredményt adja: `a b d e c f g`

Az inorder és a postorder bejárás algoritmusa a következő:

```
in(fa):
  ha nem üres(fa) akkor in(balrészfa),
                        írdki(gyökér),
                        in(jobbrészfa)

post(fa):
  ha nem üres(fa) akkor post(balrészfa),
                        post(jobbrészfa),
                        írdki(gyökér)
```

Egy négyszintű teljes bináris fa postorder bejárásával az alábbi betűsorozatot kaptuk (a fában ugyanaz a betű többször is előfordulhat):

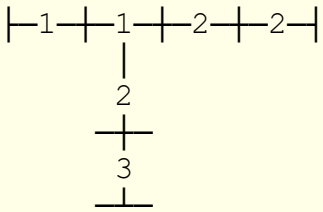
```
A B C D E F G H G F E D C B A
```

a) Rajzold fel a fát!

b) Milyen betűsorozatot kapunk ugyanennek a fának az inorder bejárásával?

5. feladat: Moszat (10 pont)

Egy moszat növekedését formális szabályokkal írjuk le. A moszat minden egységnyi szakaszának egy számjegyet feleltetünk meg (a szakasz hossza nem függ a számjegytől!). Ha leágazások keletkeznek, azokat zárójelbe tesszük. Például a 11(23)22 kódú moszat így néz ki:



A moszat növekedését úgy játsszuk le, hogy a kódjában szereplő jeleket a növekedést leíró szabályok szerinti jelsorozatra cseréljük. Folytasd 4 időegység növekedésének kódjával és ábrájával a megkezdett növekedést!

Szabályok:

0 -> 10	3 -> 3	6 -> 58	9 -> 39
1 -> 32	4 -> 56	7 -> 3(9)	(-> (
2 -> 3(4)	5 -> 37	8 -> 50	) ->)

A növekedés kiindulópontja és első 2 lépése:

```
4                 :   ├──┤
56                :   ├──┼──┤
3758              :   ├──┼──┼──┼──┤
33(9)3750         :
                  :
                  :
                  :
```

6. feladat: Rajzőrület (12 pont)

Az alábbi program egy térben mozgó teknőcöt vezérlő, LOGO-szerű nyelven készült.

a) Mit csinál a belső ciklus a Furcsa(A) eljárásban?

b) Mi a feladata a külső ciklusnak a Furcsa(A) eljárásban?

c) Mit rajzol ki az eljárás az A paraméter függvényében?

A programban használt utasítások magyarázata a következő:

`REPEAT db`	a következő, beljebb kezdett utasításokat db-szer megismétli
`forward hossz`	előrelép az aktuális irányban hossz egységgel,
`left szög`	balra fordul szög fokkal a teknőc síkjában,
`pitch szög`	a teknőc "felfelé" fordul (függőleges síkban) szög fokkal.

```
Furcsa(A):
  REPEAT 4
     forward A
     REPEAT 4
        pitch 90
        forward A
     left 90
```

7. feladat: Egyiket a másik után (18 pont)

Adottak a következő rekurzív függvények:

```
Nyissz(N,X):
  Ha N=1 akkor X
  különben Nyissz(N-1,elsőutániak(X))
Függvény vége.
```

```
Nyassz(A,B,X):
  Ha B=0 akkor Üres sorozat
  különben ha A=1 akkor elejére(első(X),
                          Nyassz(A,B-1,elsőutániak(X))
  különben Nyassz(A-1,B-1,elsőutániak(X))
Függvény vége.
```

A függvények értelmezése:

```
első([x1,x2,...,xN])=x1
```

```
elsőutániak([x1,x2,...,xN])=[x2,...,xN]
```

```
elejére(x0,[x1,x2,...,xN])=[x0,x1,x2,...,xN]
```

```
üres sorozat=[]
```

a) Mit csinál a Nyissz függvény, ha N>0 egész, X sorozat?

b) Mit csinál a Nyassz függvény, ha A>0 egész, B>=0 egész, X sorozat?

c) Készíts függvényt egy sorozat N. elemének meghatározásra! A függvény leírásában a fenti sorozatkezelő függvényeket lehet használni.

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

1. feladat: Nézd közelről (12 pont)

a) Mit csinál a Becserkész eljárás, mi az eredménye? Mit tartalmaznak az S és az N változók?

b) Mi a feladata a Valami eljárásnak és a Számol függvénynek?

```
Becserkész(X,S,N,E):
  S:=1: N:=1
  Ciklus amíg |X-S/N|≥E
     Ha X<S/N akkor N:=N+1 különben S:=S+1
  Ciklus vége
Eljárás vége.
```

```
Valami(A):
  Ha A=3 vagy A=6 vagy A=9 akkor Ki: "Igen"
  különben Ha A<10 akkor Ki: "Nem"
  különben B:=Számol(A): Valami(B)
Eljárás vége.
```

```
Számol(A):
  Ha A<10 akkor Számol:=A
       különben Számol:=(A mod 10)+Számol(A div 10)
Függvény vége.
```

2. feladat: Táblatalálgatás (14 pont)

Egy táblázatkezelő programról a következőket kell tudni:

- ún. cellákból áll, amelyek sorokba és oszlopokba vannak rendezve; a sorokat egész számok (1-től), az oszlopokat betűk (A-tól) azonosítják;
- Minden cellába egyetlen szám vagy képlet és szám írható (utóbbi esetben a szám a cella kezdőértéke, a képlet pedig a további valahány kiszámításkor alkalmazandó formula;
- A képletekben bármely cellára hivatkozni lehet az adott cella oszlop- és sorazonosítójával, pl. C5;
- A táblázatkezelő a cellák értékét minden lépésben balról jobbra haladva sor-oszlop sorrendben számolja ki, pl. A1, A2, ... B1, B2;
- If(f,A,B) feltételes kifejezés az f feltétel teljesülése esetén az A, egyébként pedig a B értéket jelenti.

a) Az A1, A2, A3, A4 cellák kezdőértéke a 0, 1, 1, X. (X tetszőleges pozitív szám.) A cellák következő értékeit kiszámító képletek:

```
A1: A1+1
A2: A2+2
A3: A2+A3
```

A képletek alkalmazását addig kell folytatni, amíg A3>A4 nem teljesül. Mi lesz A1, A2, A3 értéke?

b) Az A1, A2, A3 cellák kezdőértéke a 0, X, Y. (X, Y tetszőleges pozitív számok.) A cellák következő értékeit kiszámító képletek:

```
A1: If(páros(A3),A1,A1+A2)
A2: If(páros(A3),A2*2,A2)
A3: If(páros(A3),A3/2,A3-1)
```

A képletek alkalmazását addig kell folytatni, amíg A3=0 nem teljesül. Mi lesz ekkor A1, A2 értéke?

c) Az A1, A2, A3 cellák kezdőértéke az 1, X, Y. (X, Y tetszőleges pozitív számok.) A cellák következő értékeit kiszámító képletek:

```
A1: If(páros(A3),A1,A1*A2)
A2: If(páros(A3),A2*A2,A2)
A3: If(páros(A3),A3/2,A3-1)
```

A képletek alkalmazását addig kell folytatni, amíg A3=0 nem teljesül. Mi lesz ekkor A1, A2 értéke?

d) Készíts képleteket, melyek Fibonacci-számokat állítanak elő, s add meg a cellák kezdőértékeit is! A megoldásban maximum 3 cellát használhatsz fel! A Fibonacci-számok sorozatának kezdete 0, 1, 1, 2, 3, 5, ..., a soron következő szám mindig az előző kettő összege.

3. feladat: Flip-flop (14 pont)

Logikai áramköröket megadhatunk táblázattal vagy logikai formulával. A feladatban adattárolásra használható, tehát állapottal rendelkező alapáramköröket specifikálunk. Feltételezzük, hogy az áramkör n+1 időpontbeli állapotát a bemenőjelek és a kimenőjel n időpontbeli állapota határozza meg.

	T=h	T=i
yn=h	h	i
yn=i	i	h

Például a T tárolónak van egy bemenete (T) és egy kimenete (y). A táblázat a kimenet n+1 időpontbeli értékét tartalmazza. Logikai képletekkel ugyanez:

yn+1 = (yn és nem T) vagy (nem yn és T)

a) Adott az alábbi táblázat. Határozd meg a logikai képletet a lehető legkevesebb művelettel! Csak az *és*, *vagy*, *nem* műveleteket használhatod.

	R=h S=h	R=i S=h	R=h S=i	R=i S=i
yn=h	h	h	i	bármi
yn=i	i	h	i	bármi

b) Adott az alábbi táblázat. Határozd meg a logikai képletet a lehető legkevesebb művelettel! Csak az *és*, *vagy*, *nem* műveleteket használhatod.

	J=h K=h	J=i K=h	J=h K=i	J=i K=i
yn=h	h	i	h	i
yn=i	i	i	h	h

4. feladat: Vektorvarázslat (15 pont)

a) A egy pozitív egész számokat tartalmazó vektor. Mi kerül A[1]-be, illetve A[n]-be a Na_Micsoda(1,n,IGAZ,IGAZ) eljárás végrehajtása után. (A csere(i,j) eljárás felcseréli az A vektor i. és j. elemét.)

b) Adj közelítő értéket arra, hogy N függvényében hányszor hajtódik végre a > jellel jelölt művelet!

```
Na_Micsoda(kezdet,vég: Egész; alsó,felső: Logikai):
  Ha kezdet≠vég akkor
     k:=(kezdet+vég-1) DIV 2;
     Ha páratlan(k) akkor d:=k-kezdet+2;
                    különben d:=k-kezdet+1;
     Ciklus i=kezdet-től k-ig
        Ha A[i]>A[i+d]) akkor csere(i,i+d)
     Ciklus vége
     Ha páratlan(k) és A[k]>A[k+1]) akkor csere(k,k+1)
     Ha alsó akkor Na_Micsoda(kezdet,k,IGAZ,HAMIS)
     Ha felső akkor Na_Micsoda(k+1,vég,HAMIS,IGAZ)
  Elágazás vége
Eljárás vége.
```

5. feladat: Logikusan (12 pont)

Találd ki, hogy az alábbi kettő, PROLOG nyelvű program mit határoz meg (azaz milyen fajta bemenő adatra adnak igaz értéket véges lépés alatt az egyes "szabályok", mi lesz az eredményük, és mi a kiszámolás lényege?)

Jelmagyarázatként:

Div: az egészek osztásának műveleti jele (a hányados egészrészét adja)

Mod: az osztási maradék műveleti jele

a)
```
valami(X) ha X>1 és X1=X-1 és valami(X1) vagy X=1.
```

Mit csinál a valami(X)?

b)
```
csodaTudja(X) ha   X=2 vagy (X Mod 2)<>0 és
                  X2=(X Div 2) és nem naMi(X,X2).

naMi(X,Y) ha  Y>1 és (X Mod Y)=0
              vagy Y>1 és Y1=Y-1 és naMi(X,Y1).
```

Mit csinál a csodaTudja és a naMi?

6. feladat: Rajzőrület (15 pont)

Az alábbi programok LOGO-szerű nyelven íródtak. A Kiugró(A, B, C) eljárás meghívásakor az A paraméter csak 2-nél nagyobb egész értékeket vehet fel.

a) Mit csinál a belső ciklus az Éles(A) eljárásban?
b) Mi a feladata a külső ciklusnak az Éles(A) eljárásban?
c) Mit rajzol ki az Éles(A) eljárás az A paraméter függvényében?
d) Mit csinál a belső ciklus a Kiugró(A, B, C) eljárásban?
e) Mi a feladata a külső ciklusnak a Kiugró(A, B, C) eljárásban?
f) Mit rajzol ki a Kiugró(A, B, C) eljárás az A,B,C paraméterek függvényében?

A programban használt utasítások magyarázata a következő:

`REPEAT db`	a következő, beljebb kezdett utasításokat db-szer megismétli
`forward hossz`	előrelép az aktuális irányban hossz egységgel,
`left szög`	balra fordul szög fokkal a teknőc síkjában,
`pitch szög`	a teknőc "felfelé" fordul (függőleges síkban) szög fokkal.

```
Éles(A):
   REPEAT 4
      forward A
      REPEAT 4
         pitch 90
         forward A
      left 90
```

```
Kiugró(A,B,C):
   REPEAT A
      forward B
      REPEAT 2
         pitch 90
         forward C
         pitch 90
         forward B
      left 360/A
```

7. feladat: Vagány vágányok (18 pont)

Villamosvégállomáson a villamosvezetők az állomásfőnöktől egy szabad vágányt kérnek, oda beállnak, majd valamennyi ideig tartó várakozás után bejelentik az indulást, s elhagyják az állomást.

```
Villamos(I):
  Ciklus amíg az állomáshoz nem ér
     Előrehaladás, ha lehet
  Ciklus vége
  Vágányfoglalás(V)
  Beállás(V)
  ...
  Induláskérés(V)
  Indulás
Eljárás vége.
```

Az állomásfőnök eljárásai (egyszerre 1 vágányfoglalással és egy induláskéréssel tud foglalkozni, a végállomáson 2 vágány van) milyen stratégia szerint adja ki a vágányokat? (F kezdeti értéke 1 vagy 2, a működést a kezdeti értéktől függetlenül kell jellemezni).

a)

```
Vágányfoglalás(V):
  Ha FOGLALT(1) és FOGLALT(2) akkor Várj
  Ha FOGLALT(1) akkor V:=2 különben V:=1
  FOGLALT(V):=igaz
Eljárás vége.
```

```
Induláskérés(V):
  FOGLALT(V):=hamis
  Folytasd a vágányfoglalást, ha várakoztál benne
Eljárás vége.
```

b)

```
Vágányfoglalás(V):
  Ha FOGLALT(1) és FOGLALT(2) akkor Várj
  Ha FOGLALT(3-F) akkor V:=F különben V:=3-F
  FOGLALT(V):=igaz
Eljárás vége.
```

```
Induláskérés(V):
  FOGLALT(V):=hamis; F:=V
  Folytasd a vágányfoglalást, ha várakoztál benne
Eljárás vége.
```

c)

```
Vágányfoglalás(V):
  Ha FOGLALT(1) és FOGLALT(2) akkor Várj
  Ha FOGLALT(F) akkor V:=3-F különben V:=F
  FOGLALT(V):=igaz; F:=3-V
Eljárás vége.
```

```
Induláskérés(V):
  FOGLALT(V):=hamis
  Folytasd a vágányfoglalást, ha várakoztál benne
Eljárás vége.
```

Elérhető összpontszám: 100 pont

1994. Második forduló

Első-második osztályosok

1. feladat: (15 pont)

Készíts programot egy természetes szám prímtényezős felbontásának előállítására! A program olvassa be billentyűzetről N értékét, majd írja ki a prímtényezős felbontását a képernyőre!

Példa:

Bemenet	Kimenet
N=7	N=7
N=16	$N=2^4$ vagy N=2*2*2*2
N=24	$N=2^3*3$ vagy N=2*2*2*3
N=1	N=1

2. feladat: (15 pont)

Hamupipőke különböző színű lencséket válogat. Az egyező színűeket azonos tálkába kell tennie. Előre sajnos nem tudja, hogy hány darab tálat kell előkészítenie. Készíts programot, amely segít neki: megadja a lencsefajták számát, valamint leszámolja, hogy melyikből mennyi volt!

A program egy szekvenciális állományból olvassa a lencsesorozatot (a színeket kisbetűvel írjuk, az állományban soronként egy szín szerepel, csak a sorvég karakter választja el őket egymástól, szóköz az állományban sehol sincs), s a képernyőre írja az eredményt!

Példa:

File:
sárga
sárga
zöld
sárga
fehér
sárga
zöld
sárga

Kép:

sárga	5 db
zöld	2 db
fehér	1 db

3. feladat: (15 pont)

Egy bacilusfajta a következő jellegzetességekkel rendelkezik:

- keletkezése után egy órával szaporodóképessé válik;
- minden (szaporodóképes) óra végén osztódással szaporodik; a keletkező két új bacilus közül az egyik 'öreg' – ez megőrzi a korát, a másik 'fiatal' – ez csak egy óra múlva lesz szaporodóképes;
- a szaporodás elhanyagolható idő alatt megy végbe.

Ha adott pillanatban egy addig csíramentes környezetbe kerül egy újszülött bacilus, akkor hány bacilus lesz a tenyészetben **N** (**N** ≥ **1**) óra múlva? A program olvassa be a billentyűzetről N értékét, majd írja ki óránként a bacilusok számát!

Példa:

N=6, a bacilusok száma:

Óra	0	1	2	3	4	5	6
Bacilusok	1	1	2	3	5	8	13

4. feladat: (30 pont)

Egy szekvenciális szövegfile-ban tároljuk a magyar labdarúgó válogatott összes mérkőzésének eredményeit, dátum szerint növekvő sorrendben (dátum, adott, illetve kapott gólok, szóközökkel elválasztva – a file tartalmának helyességét nem kell ellenőrizni).

Határozd meg a válogatott leghosszabb nyeretlenségi, illetve leghosszabb pontnélküli sorozatának kezdő- és végdátumát (amely dátumok között, magukat a határokat is beleértve, a válogatott nem győzött, illetve nem szerzett pontot.)

A legelső előtti, illetve a legutolsó utáni napokról semmit nem tudunk, tehát akár e két időpont közvetlen szomszédjában is lehetett győztes mérkőzés.

Példa:

file:
1992.01.15. 3 0
1992.03.01. 1 1
1992.04.02. 2 6
1992.04.05. 0 2
1992.06.23. 2 1
1992.08.08. 0 1

kép:

Leghosszabb nyeretlenségi sorozat:
1992.01.16.-1992.06.22.

Leghosszabb pontnélküli sorozat:
1992.03.02.-1992.06.22.

Elérhető összpontszám: 75 pont

Harmadik-ötödik osztályosok

1. feladat: (15 pont)

Egy ékezetes betűket használó ábécé (ilyen például a magyar, cseh, spanyol stb.) ékezetes betűit két jellel kódoljuk: az ékezet nélküli betűvel, s a fölé írandó ékezettel. Például egyes betűk így nézhetnek ki:

á — a' ö — o: ń — n~ â — a^

é — e' ő — o" č — e` ...

A használható ékezetek jelei: **' " : ` ~ ^**

Készíts programot, amely a fenti jelekkel kódolt ékezetes betűket tartalmazó szöveget beolvassa egy szekvenciális file-ból, majd úgy írja ki a képernyőre, hogy a normál karakterek minden második sorban helyezkednek el, s a fölöttük levő sor megfelelő helyére kerülnek az ékezetek! A file-ban levő sorvégek hatására a képernyőn is új sort kell kezdeni.

Az ékezetek jeleit a szövegben más célra nem használjuk. A szöveg biztosan helyes, az ékezetek alkalmazhatóságát nem kell ellenőrizni.

Példa:

```
file:E'kezetes betu"ket          kép:'          "
          tartalmazo' szo:veg.       Ekezetes betuket
                                               '   :
                                      tartalmazo szoveg.
```

2. feladat: (15 pont)

Adott egy (elvileg végtelen) sakktábla. A tábla kockáit egész számpárokkal jelöljük. Írj egy olyan programot amely beolvassa két kocka koordinátáit és megadja az első kockáról a másodikra való eljutás módját abban az esetben, ha a mozgás mindig a sakkbeli lóugrás szabályai szerint történik! A képernyőre írt eredménynek tartalmaznia kell az összes érintett kocka koordinátáit.

Megjegyzések:

a) A megoldásban figyelembe kell venni, hogy a kiindulási, illetve végkocka **nagyon messze** is lehet egymástól.

b) Nem kötelező az optimális megoldást megkeresni, de törekedni kell olyan lépéssorozat megtalálására, amely közel áll az optimálishoz.

c) Az érkezési kocka bárhol lehet az indulási kockához képest.

Az alábbiakban megadjuk a feladat két lehetséges megoldását abban az esetben ha a *(3,7)* koordinátájú pontból indulunk és a *(10, 14)* koordinátájú pontba érkezünk:

(3,7) (5,8) (6,10) (7,12) (9,13) (11,12) (10,14)

(3,7) (4,9) (5,11) (6,13) (8,14) (9,12) (10,14).

3. feladat: (20 pont)

Adott egy N x M-es mátrix ($1 \leq N, M \leq 20$) amely egy terület domborzati térképét képezi le. (A mátrix pozitív valós számokat tartalmaz, amelyek a megfelelő pontok magasságát jelölik.) A mátrix egy adott pontjára ráhelyezünk egy labdát. A labda mindig lefelé gurul; a különböző elfogadható irányok között véletlenszerűen választ. Egy adott kockáról négy irányba lehet továbblépni: a fölötte, alatta, tőle jobbra, illetve balra levőre **(**pl. ha a (7,5) kockában vagyunk a (6,5), (8,5), (7,4) és (7,6) kockákra léphetünk**)**.

Írj egy olyan programot, amely beolvassa a domborzati térképet és a labda indulási pozícióját, és megállapítja, hogy a labda elérheti-e a térkép szélét (kigurul-e a térképről) vagy sem. Ha a labda kigurulhat, a program adja meg az (egyik) lehetséges útvonalat, ha nem, akkor jelezze, hogy a labda elakadt.

A táblázat elemeit számpárokkal jelöljük, a koordináták 1 és N, illetve M közöttiek. A bemeneti adatokat tartalmazó állomány struktúrája a következő:

sorok száma oszlopok száma (szóközökkel elválasztva)

az első sor adatai (szóközökkel elválasztva)

...

az utolsó sor adatai (szóközökkel elválasztva)

indulási pozíció sora indulási pozíció oszlopa (szóközökkel elválasztva)

A program a fentieken kívül legyen képes a labda sebességének kezelésére. Ha a labda N méter magasságból ereszkedik le, akkor a sebessége N m/s-mal nő, ha N métert emelkedik, akkor ugyanannyival csökken. (Ilyen körülmények között a labda kijöhet a völgyekből.)

4. feladat: (27 pont)

Egy szűkített LOGO programozási nyelv az alábbi utasításokból áll:

`forward hossz`	előrelépés hossz lépést az aktuális irányba
`left szög`	balrafordulás szög fokkal az aktuális helyen

`back hossz`	hátralépés hossz lépést az aktuális irányba
`right szög`	jobbrafordulás szög fokkal az aktuális helyen
`reset`	alaphelyzetbe-állás: üres képernyő, aktuális hely a képernyő közepe, aktuális irány a felfelé, toll a papíron
`penup`	toll felemelése
`pendown`	toll letevése
`repeat db [utasítások]`	utasítások db-szeri megismétlése

A hossz, a szög és a db tetszőleges egész számkonstansok lehetnek. Az utasításokat és paramétereiket legalább egy szóköz vagy sorvégjel választja el egymástól.

Készíts programot, amely egy szövegfile-ból beolvas egy, a fenti utasításokat tartalmazó LOGO programot, majd végrehajtja azt!

Példa:

file:

```
reset
left 90 forward 100
repeat 4 [forward 100 right 90]
```

kép:

Elérhető összpontszám: 77 pont

1994. Harmadik forduló

Első-második osztályosok

Tom és Jerry:

Tom, a macska és Jerry, az egér, egy négyzethálós labirintusban élnek. Tom szereti (megenni) Jerry-t, Jerry pedig szereti a sajtot. Így Tom kergeti Jerry-t, Jerry pedig keresi a sajtot. Ha Tomnak sikerül elkapnia Jerry-t, akkor megeszi szegényt, ha azonban Jerry a gyorsabb (szerencsésebb), akkor megtalálhatja és megeheti a világ legjobb sajtját. A labirintusban kétféle fal található: tömör, illetve lyukas. A lyukas falakon Jerry számára megfelelő méretű lyuk van, Tom azonban nem fér át ezen, azaz Jerry oda léphet, ahol út vagy lyuk van, Tom pedig csak oda, ahol út.

A programod a két fő tevékenységet engedje meg: labirintusgenerálást, valamint Tom és Jerry versenyfutását. A labirintusgenerálás a következőkből áll:

A. Generálj egy véletlenszerű, téglalap alapú, négyzethálós szerkezetű, kívülről zárt labirintust tömör (**f**) és lyukas (**o**) falakkal, valamint köztük levő utakkal (**u**)! A labirintusban az utak szélessége **1**, az utak kört alkothatnak, de nem biztos, hogy összefüggőek. **12 pont**

B. Válassz véletlenszerűen egy fix pozíciót a sajtnak és két kezdőpozíciót: egyet Tomnak, egyet Jerry-nek (természetesen az úton)! **4 pont**

A generálás eredménye szövegek sorozata legyen a következő formában: (lemezeden található példafile-ok: LAB1.INP, LAB2.INP, LAB3.INP)

LAB1.INP:

```
f f f f f f f f f f f f f f f
f u u u u u u u u u u u u S f
f o f f f f f f f f f f f u f
f u u u J o u u u u u u u u f
f u f f f f f u f f f f f f f
f u u u u u u u u u T u u u f
f f f f f f f f f f f f f f f
```

C. Írd ki az általad generált labirintust (sajttal, Tommal, Jerry-vel) egy szekvenciális szövegfile-ba!

Tom és Jerry *játékának* funkciói:

D. Olvasd be a labirintus kezdőállapotát egy szekvenciális szövegfile-ból, majd rajzold ki a képre karakterekkel vagy grafikusan (ez utóbbi értékesebb)! **7 pont**

E. Először Jerry, azután pedig Tom indul el, felváltva mozognak a következő szabályok szerint:

a. vagy csak vízszintes, vagy csak függőleges irányban léphetnek,

b. Jerry legfeljebb egyet léphet (az úton, illetve lyukas falban),

c. Tom legfeljebb kettőt léphet (csak az úton, irányváltás nélkül, kanyarban ugyanis lassítania kell),

d. ha Jerry látja a sajtot (a sajt sorában vagy oszlopában van és nincs köztük tömör fal), akkor egyenesen a sajt felé indul,

e. ha Jerry látja Tomot, akkor menekülni próbál (Tom felé biztosan nem megy) – ez az előző szabályt is felülbírálja,

f. ha Tom látja Jerry-t, akkor egyenesen felé indul (Tom a lyukakon sem lát át),

g. minden egyéb esetben Tom keresi Jerry-t, illetve Jerry keresi a sajtot (a képen láthatóan mozog keresés közben, a kereséshez egyébként bármilyen, akár az aktuális helyről nem látható információ is felhasználható). 10 pont

ha Tomot ketrecbe zárjuk, akkor Jerry megtalálja a sajtot (ha vezet hozzá út) 6 pont

ha nincs a labirintusban sajt és a falakon lyuk, akkor Tom megfogja Jerry-t 7 pont

F. Akár Jerry, akár Tom szerepét átvehesse a program használója, aki ilyenkor a négy kurzormozgató nyíl billentyűvel mozgathatja Jerry-t 1-1 lépést, valamint Tomot a megfelelő nyíl után az 1-es vagy a 2-es billentyű hatására 1, illetve 2 lépést. (A nyíl billentyűk kétbyte-os kódot adnak, az első byte mindig 0, a második: balra - 75, felfelé - 72, jobbra - 77, lefelé - 80) **24 pont**

G. A játék véget ér, ha Tom megeszi Jerry-t (amikor éppen rááll), **vagy** Jerry megeszi a sajtot (amikor éppen rááll), vagy a felhasználó leállítja a programot (a V billentyűvel). **3 pont**

Elérhető összpontszám: 75 pont + maximum 25 pont az 1-2. fordulóból

Harmadik-ötödik osztályosok

Táblázatkezelő:

Sok könnyen használható táblázatkezelő program létezik IBM PC-ken (pl. QuattroPro, Excel, ...). Készíts Te is egy egyszerű táblázatkezelő programot!

A táblázat oszlopait az angol ABC betűi, sorait 1 és 26 közötti egész konstansok azonosítják. Minden egyes mezőjébe vagy egy számot, vagy egy szöveget, vagy egy képletet lehet írni. A program a következő lehetőségekkel rendelkezzen:

A. a képernyőn induláskor jelenjen meg egy üres táblázat (a táblázat bal felső sarka látszik, minden mező 10 karakter hosszúságú legyen, aktuális mezőként az A1-et jelöljük meg); **3 pont**

B. a kurzormozgató billentyűkkel a táblázatban lehessen mozogni, az aktuális mezőt megjelenítésében valahogyan különböztesse meg a többitől; ha a képről kilépnénk, a képernyőt, mint ablakot mozgassa a táblázat fölött; **12 pont**

C. az aktuális mezőbe lehessen beírni maximum 10 karaktert (az ott találhatót felülírni, beíráskor szerkeszteni nem lehet, a beírást vagy az ENTER vagy pedig a valamelyik nyíl billentyű lenyomásával lehet befejezni – utóbbi esetben a mezőt a megfelelő irányban el is hagyjuk) **2 pont**

a. fixpontos valós számot a mezőben jobbra igazítva; 2 pont

b. szöveget a mezőben balra igazítva; (ez is tartalmazhat számjegyeket, de kötelez:ő aposztróffal kezdeni, de a szöveg végére nem kell még egyszer kitenni); 2 pont

c. képletet (amelynek kiszámított értékét írja ki a képernyőre a program, hibás képletek esetén a mez:őben a HIBA szöveg jelenjen meg), a képlet tartalmazhat: 19 pont

   1. számkonstansokat,
   2. mezőhivatkozásokat,
   3. +,-,*,/ műveleteket,
   4. zárójeleket;

Az ettől kezd:őd:ő funkciók valamely funkcióbillentyű lenyomására legyenek végrehajthatók!

D. mező értékét módosítani: (F1 billentyű) **7 pont**

a. a mező kiírt tartalmában a jobbra-, illetve a balra mutató nyilakkal mozogni;

b. az aktuális karaktert a DEL billentyűvel törölni;

c. az aktuális karakter elé egy karaktert beszúrni;

d. ENTER billentyű lenyomására a mező szerkesztését befejezni;

E. közvetlenül pozícionálni egy cellára (ekkor be kell írni a cella indexét); (F2 billentyű) **2 pont**

F. a táblázatot lemezre írni egy szekvenciális szövegfile-ba (az üres mezők nem kerülnek ki a fileba); (F3 billentyű) **5 pont**

Minden sorban egy mez:ő van, tartalma: **mezőazonosító:mezőtartalom** (a beírás formátumában)

Például: A1:15 A12:'szöveg B2:A1+A2*2

G. lemezről egy táblázatot beolvasni; (a lemezeden van: *.TAB – több próbafile) (F4 billentyű) **5 pont**

H. új sort beilleszteni az aktuális sor elé (ekkor a képletekben szereplő sorindexeket át kell számolni az új soroknak megfelelően), s a 26. sor korábbi tartalma elveszik; (F5 billentyű) **4 pont**

I. új oszlopot beilleszteni az aktuális oszlop elé (ekkor a képletekben szereplő oszlopindexeket át kell számolni az új oszlopoknak megfelelően), s a Z jelű oszlop korábbi tartalma elveszik; (F6 billentyű) **4 pont**

J. az aktuális sort törölni, minden mögötte levő sor eggyel előbbre kerül (ekkor a képletekben szereplő sorindexeket át kell számolni az új soroknak megfelelően), s a 26. sor üres lesz; (F7 billentyű) **4 pont**

K. az aktuális oszlopot törölni, a mögötte levők eggyel előbbre kerülnek (a képletekben szereplő oszlopindexeket át kell számolni az új oszlopoknak megfelelően), s a Z jelű oszlop üres lesz. (F8 billentyű) **4 pont**

Kódtáblázat: (a speciális billentyűkhöz kétbyte-os kód tartozik, az első byte mindig 0)

F1	F2	F3	F4	F5	F6	F7	F8	←	↑	→	↓
0, 59	0, 60	0, 61	0, 62	0, 63	0, 64	0, 65	0, 66	0, 75	0, 72	0, 77	0, 80

Elérhető összpontszám: 75 pont + maximum 25 pont az 1-2. fordulóból

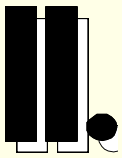
A verseny végeredménye:

I. kategória

1. Pósta Zoltán	Leővey Klára Gimnázium, Budapest
2. Benkő Tamás	Leővey Klára Gimnázium, Budapest
3. Smulovics Péter	Berzsenyi Dániel Gimnázium, Budapest
Koppány Csaba	Apáczai Csere János Gimnázium, Pécs
5. Ungár Péter	Fazekas Mihály Gimnázium, Budapest
6. Szalay Máté	Fazekas Mihály Gimnázium, Budapest
Kovács Gábor Zsolt	Lovassy László Gimnázium, Veszprém
Noll János	Fazekas Mihály Gimnázium, Budapest
Gyenes Gábor	Földes Ferenc Gimnázium, Miskolc
10. Kurucsai Tamás	Budai Nagy Antal Gimnázium, Budapest
Tóth László	Földes Ferenc Gimnázium, Miskolc

II. kategória

1. Blahut György Gábor	Szent István Gimnázium, Budapest
2. Kovács Gábor	Radnóti Miklós Gimnázium, Budapest
3. Marx Dániel	Szent István Gimnázium, Budapest
4. Fige Péter	Herman Ottó Gimnázium, Miskolc
5. Ipacs Zsolt	Széchenyi István Szakközépiskola, Nyíregyháza
6. Birszki Bálint	Boronkay György Szakközépiskola, Vác
6. Kovács Zoltán	Radnóti Miklós Gimnázium, Szeged
8. Gede Mátyás	Táncsics Mihály Gimnázium, Kaposvár
9. Tarján Dénes	Piarista Gimnázium, Budapest
10. Szabó Balázs	Vetési Albert Gimnázium, Veszprém



Megoldások, értékelések

II. Megoldások, értékelések

II.

1. Általános iskolások országos számítástechnikai versenye

1991. Első forduló

1. feladat: (11 pont)

A megoldás C-I*A B-vel osztásának maradékát vizsgálja.

```
Felbontás(A,B,C):
  N:=C div A: E:=hamis
  Ciklus i=0-tól N-ig
     Ha (C-I*A) mod B=0 akkor Ki: i,(C-I*A) div B: E:=igaz
  Ciklus vége
  Ha nem E akkor Ki: "Nincs megoldás"
Eljárás vége.
```

A. Ha a program C=i*A+j*B alakban keresi a megoldást, minden i,j-re vizsgálva 6 pont

B. Ha C-i*A B-vel való oszthatóságát vizsgálja 8 pont

C. Összes megoldás megadásáért 3 pont

2. feladat: (6 pont)

Először számoljuk ki, hogy az adott nap hányadik nap 1901 január elseje után, majd vegyük a 7-tel osztási maradékot. A `Hónap` tömb tartalmazza minden hónapra az előző hónapok összesített napszámát (a február 28 napos legyen)! A `Hét` tömb a hét egyes napjainak nevét tartalmazza, ahol `Hét`(0)="Hétfő".

Megjegyzés: Ebben és a következő feladatokban is kétféle lehetőség közül választhatunk. Feltölthetünk konstansokkal tömböket, s azokat megfelelően indexelve kapjuk a feladat megoldását, vagy pedig sokirányú elágazással választjuk szét az eseteket, s az egyes ágakra tesszük a konkrét kiírásokat.

```
Napmegadás(Év,Hó,Nap):
  DB:=(Év-1901)*365          [minden év]
  DB:=DB+(Év-1900) div 4       [4-gyel oszthatóak]
  DB:=DB-(Év-1900) div 100 [100-zal oszthatóak]
  DB:=DB+(Év-1600) div 400 [400-zal oszthatóak]
  DB:=DB+Hónap(Hó)+Nap-1       [hónap és nap]
  Ha (Év div 4=0 és Év div 100≠0 vagy Év div 100=0)
        és Hó>2 akkor DB:=DB+1
  Ki: Hét(DB mod 7)
Eljárás vége.
```

A. Nem szökőéveket 365 nappal számolja 2 pont

B. 4-gyel osztható éveket szökőévnek veszi 1 pont

C. 100-zal osztható, de 400-zal nem osztható évek nem szökőévek 1 pont

D. 7-tel osztás maradéka alapján jó a nap meghatározás 2 pont

3. feladat: (12 pont)

Külön kell választani a 0-ra végződőeket, majd a többi számot fel kell bontani egyes és tízes helyiértékre. Használjunk három tömböt:

- `Nulla`: a nullára végződő számok neve.
- `Egyes`: az egyes helyiértéken levő számok neve.
- `Tízes`: a tízes helyiértéken levő számok neve.

```
Szóvalírás(Szám):
  Ha Szám mod 10=0 akkor Ki: Nulla(Szám div 10)
  különben Ha Szám>10 akkor Ki: Tízes(Szám div 10)
           Ki: Egyes(Szám mod 10)
Eljárás vége.
```

A. 10-nél kisebb számok kiírása 3 pont

B. 10-nél nagyobb számok egyes helyiértéke 3 pont

C. 10-nél nagyobb számok tízes helyiértéke 3 pont

D. 0-ra végződő számok kiírása 3 pont

4. feladat: (8 pont)

A megoldás jó 2 pont

B megoldás jó 3 pont

C megoldás jó (hibás válasz, levonás) 4 pont

B magyarázata 3 pont

5. feladat: (18 pont)

Vegyünk fel egy annyi elemű tömböt, ahány csapat szerepel a bajnokságban! A tömb egyes elemei a feladatban szereplő mezőket tartalmazó rekordok legyenek (győz, dönt, ver, lőtt, kapott, pont)! Minden elem kezdőértéke 0. Az eredmények szintén rekordok (csap1, csap2, gól1, gól2). A csapatokat a sorszámukkal azonosítjuk.

```
Bajnokság(Ered,M,Tábl,N):
  Ciklus I=1-től M-ig
    Tábl(Ered(I).csap1) rekordnévvel:
      lőtt:=lőtt+Ered(I).gól1: kapott:=kapott+Ered(I).gól2
    Tábl(Ered(I).csap2) rekordnévvel:
      lőtt:=lőtt+Ered(I).gól2: kapott:=kapott+Ered(I).gól1
    Elágazás
      Ered(I).gól1>Ered(I).gól2 esetén
        Tábl(Ered(I).csap1) rekordnévvel: győz:=győz+1
                                          pont:=pont+2
        Tábl(Ered(I).csap2) rekordnévvel: ver:=ver+1
                          Ered(I).gól1=Ered(I).gól2 esetén
        Tábl(Ered(I).csap1) rekordnévvel: dönt:=dönt+1
                                          pont:=pont+1
        Tábl(Ered(I).csap2) rekordnévvel: dönt:=dönt+1
                                          pont:=pont+1
      Ered(I).gól1<Ered(I).gól2 esetén
        Tábl(Ered(I).csap1) rekordnévvel: ver:=ver+1
        Tábl(Ered(I).csap2) rekordnévvel: győz:=győz+1
                                          pont:=pont+2
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

A. eredmények beolvasása 2 pont

B. jó csapatnál számol 2 pont

C. győzelmek számolása 2 pont

D. döntetlenek számolása 2 pont

E. győzelmek számolása 2 pont

F. lőtt gólok számolása 2 pont

G. kapott gólok számolása 2 pont

H. összpontszám számolása 2 pont

I. eredmény táblázat kiírása 2 pont

Elérhető összpontszám: 55 pont

1991. Második forduló

1. feladat: (7 pont)

A megoldás például (más megoldás is elfogadható):

```
10 FOR I=1 TO 9
20   FOR J=0 TO 9
30     FOR K=0 TO 9
40       IF I*I*I+J*J*J+K*K*K=I*100+J*10+K THEN PRINT
I*100+J*10+K
50     NEXT K
60   NEXT J
70 NEXT I
```

A. Jó ciklusszervezés 3 pont

B. Jó feltételvizsgálat 3 pont

C. Jó kiírás 1 pont

2. feladat: (11 pont)

A. Egyenlő oldalú háromszög, oldalhossza 100 egység 3 pont

B. Négyzet, oldalhossza 100 egység 3 pont

C. Szabályos 360-szög, oldalhossza 1 egység, 3 pont
körnek látszik 2 pont

3. feladat: (13 pont)

```
Fénykereső:
  Ciklus
     Ha FÉNY=-1 akkor LÁNC(1,0): LÁNC(2,1)
     különben ha FÉNY=1 akkor LÁNC(1,1): LÁNC(2,0)
     különben LÁNC(1,1): LÁNC(2,1)
  Ciklus vége
Eljárás vége.
```

Bonyolultabb a megoldás, ha csak akkor szabad bekapcsolni a lánctalpat, ha nem mozog, illetve akkor szabad kikapcsolni, ha mozog.

```
Fénykereső:
  Egyik:=igaz: Másik:=igaz: LÁNC(1,1): LÁNC(2,1)
  Ciklus
    Ha FÉNY=-1 akkor Ha Egyik akkor LÁNC(1,0): Egyik:=hamis
                     Ha nem Másik akkor LÁNC(2,1): Másik:=igaz
    különben ha FÉNY=1 akkor Ha nem Egyik akkor LÁNC(1,1): Egyik:=igaz
                             Ha Másik akkor LÁNC(2,0): Másik:=hamis
    különben Ha nem Egyik akkor LÁNC(1,1): Egyik:=igaz
             Ha nem Másik akkor LÁNC(2,1): Másik:=igaz
  Ciklus vége
Eljárás vége.
```

A. Mozog az autó 1 pont

B. A baloldali lánctalp mozog, ha a fény jobbról jön 4 pont

C. A jobboldali lánctalp mozog, ha a fény balról jön 4 pont

D. Mindkettő mozog, ha a fény szemből jön 4 pont

4. feladat: (12 pont)

A. `130 PRINT U-V` 3 pont

B. `140 V=U` 3 pont

C. 150 H=H+D(I)-D(I-1) 3 pont

D. ezt a sort 115-be kell tenni 3 pont

5. feladat: (12 pont)

A. 120-as sorból az osztást ki lehet emelni a ciklus utánra 3 pont

B. 160-as sorból az osztást ki lehet emelni a ciklus utánra 3 pont

C. 160-as sorban négyzetreemelés helyett önmagával szorzás 3 pont

D. 160-as sorban szorzás esetén a különbséget nem kell kétszer kiszámítani 3 pont

6. feladat: (15 pont)

```
Bongó(Tipp,Nyeremény):
  Ciklus I=1-től 7-ig
     B(I):=Véletlen(0..9)     [0 és 9 közötti számjegy]
  Ciklus vége
  Ha Tipp(7)≠B(7) vagy Tipp(6)≠B(6) akkkor Nyeremény:=0
     különben Nyeremény:=70: I:=5
              Ciklus amíg I>0 és Tipp(I)=B(I)
                Nyeremény:=Nyeremény*10: I:=I-1
              Ciklus vége
  Elágazás vége
Eljárás vége.
```

A. 7 szám véletlenszerű sorsolása 3 pont

B. Jó a nyereségkiszámítás feltétele 9 pont

C. Jó a nyereség összege 3 pont

Elérhető összpontszám: 70 pont

1991. Harmadik forduló

Feladat: (50 pont)

A feladatot megoldó algoritmus lényege: Egy-egy tömböt feleltessünk meg a két útnak! Az egyes utak N egység hosszúak, a a közepüknél legyen a lámpa! Időegységenként minden autó egyet léphessen előre, ha előtte szabad hely van, illetve ha lámpánál áll, és a lámpa zöld. Az utak legelső pontjára jöhessenek be autók. A tömbökben 0-val jelöljük, ha valamilyen pozíción nincs autó, 1-gyel pedig ha van. A lámpa T(0) időegységig legyen piros, T(1)-ig piros-sárga, újra T(0)-ig zöld és T(1)-ig sárga színű! Kezdetben a vízszintesen rajzolt útnál zöld a lámpa.

```
Útkereszteződés:
  Idő:=0
  Ciklus
    Ciklus I=N-1-től 1-ig -1-esével [a képen vízszintes út]
      Ha (I≠N div 2 vagy Idő<T(0)) és V(I+1)=0
         akkor V(I+1):=V(I): V(I):=0
    Ciklus vége
    Ha véletlenszám<Belépés és V(1)=0 akkor V(1):=1
    Ciklus I=N-1-től 1-ig -1-esével [a képen függőleges út]
      Ha (I≠N div 2 vagy (Idő>T(0)+T(1) és Idő<2*T(0)+T(1)))
         és F(I+1)=0 akkor F(I+1):=F(I): F(I):=0
    Ciklus vége
    Ha véletlenszám<Belépés és F(1)=0 akkor F(1):=1
    Idő:=(Idő+1) mod (2*T(0)+2*T(1))
  Ciklus vége
Eljárás vége.
```

A. Az útkereszteződés kirajzolása	5 pont
B. Paraméterek beolvasása (autók belépése)	5 pont
C. Az autók vízszintesen mozognak	6 pont
D. Az autók függőlegesen mozognak	6 pont
E. Véletlenszerűen belépnek vízszintesen	3 pont
F. Véletlenszerűen belépnek függőlegesen	3 pont
G. A vízszintes út végén eltűnnek	1 pont
H. A függőleges út végén eltűnnek	1 pont
I. 4-féle lámpát kezel (piros, piros+sárga, zöld, sárga)	4 pont
J. A lámpaváltások időzítése (a sárga rövid)	4 pont
K. Jól kezeli az ellentétes irányú lámpát	2 pont
L. Az autók piros, sárga lámpánál megállnak	5 pont
M. Álló autó mögött a következők megállnak	5 pont

Elérhető összpontszám: 50 pont

1991. Negyedik forduló

1. feladat: (45 pont)

Négy eljárást kell készíteni a 4 részfeladat megoldására. A kettes és a tizenhatos számrendszerben a számokat N, illetve M elemű tömbökkel adjuk meg, a legkisebb indexű helyen a legnagyobb helyiértéket tárolva. A tizenhatos számrendszerbeli 9-nél nagyobb számjegyeket karakteresen ábrázoljuk, s a számoláshoz át kell alakítanunk őket.

```
Tízbőlkettő(T):
  I:=0
  Ciklus
     I:=I+1: K(I):=T mod 2: T:=T div 2
  amíg T>0
  Ciklus vége
  Ciklus amíg I>0
     Ki: K(I): I:=I-1
  Ciklus vége
Eljárás vége.
```

```
Kettőbőltíz(K,N):
  T:=0
  Ciklus I=N-től 1-ig -1-esével
     T:=T*2+K(I)
  Ciklus vége
  Ki: T
Eljárás vége.
```

A 0-tól 15-ig indexelt Számjegy tömb tartalmazza karakteresen a tizenhatos számrendszer számjegyeit.

```
Tizenhatbólkettő(H,M):
  J:=0
  Ciklus amíg Számjegy(J)≠H(1)
     J:=J+1
  Ciklus vége
  Tízbőlkettő(J)      [a bevezető 0-kat nem írja]
  Ciklus I=2-től M-ig
     J:=0
     Ciklus amíg Számjegy(J)≠H(1)
        J:=J+1
     Ciklus vége
     L:=8
     Ciklus amíg L>0        [a bevezetp 0-k is kellenek]
        Ki: J div L: J:=J mod L: L:=L div 2
     Ciklus vége
  Ciklus vége
Eljárás vége.
```

Egészítsük ki a kettes számrendszerbeli számot elölről annyi 0-val, hogy a számjegyek száma 4-gyel osztható legyen!

```
Kettőbőltizenhat(K,N):
  Nullával kiegészítés elölről
  Ciklus I=1-től N-ig 4-esével
     L:=0
     Ciklus J=I-től I+3-ig
        L:=L*2+K(J)
     Ciklus vége
     Ki: Számjegy(L)
  Ciklus vége
Eljárás vége.
```

A. Választás a lehetőségekből	2 pont
B. Beolvasás, kiírás	3 pont
C. 10 → 2 számrendszerbeli átalakítás	10 pont
D. 2 → 10 számrendszerbeli átalakítás	10 pont
E. 16 → 2 számrendszerbeli átalakítás	10 pont
F. 2 → 16 számrendszerbeli átalakítás	10 pont

2. feladat: (25 pont)

Általában a szóköz utáni betűket kell kiírni monogramként, kivéve akkor, ha az kettős betű, és a párja követi a névben.

```
Monogram(Név):
  Betűírás(1)
  Ciklus I=2-től hossz(Név)-ig
     Ha Név(i)=" " akkor Betűírás(I+1)
  Ciklus vége
Eljárás vége.

Betűírás(I):
  Ki: Név(I)
  Pár:=Név(I)+Név(I+1)
  Három:=Pár+Név(I+2)
  Ha Három="Dzs" akkor Ki: Három
  különben Ha Pár="Cs" vagy Pár="Dz" vagy Pár="Gy" vagy
              Pár="Ly" vagy Pár="Ny" vagy Pár="Sz" vagy
              Pár="Ty" vagy Pár="Zs" akkor Ki: Pár
  Ki: ". "
Eljárás vége.
```

A. Beolvasás, kiírás 3 pont

B. Felismer két kezdőbetűt 10 pont

C. Felismer tetszőleges számút +5 pont

D. Felismeri a kettős mássalhangzókat (cs,gy,ly,ny,sz,ty,zs) 7 pont

3. feladat: (25 pont)

Olyan képernyőt használunk, ahol a sorokat 0-tól SY-ig, az oszlopokat 0-tól SX-ig számozzuk.

```
Rajzolás:
  Sor:=SY div 2: Osz:=SX div 2: Pontrajzolás(Osz,Sor)
  Ciklus
     Be: Betű
     Ha Betű="J" akkor Osz:=Osz+1: Ha Osz>SX akkor Osz:=0
     Ha Betű="B" akkor Osz:=Osz-1: Ha Osz<0 akkor Osz:=SX
     Ha Betű="L" akkor Sor:=Sor+1: Ha Sor>SY akkor Sor:=0
     Ha Betű="F" akkor Sor:=Sor-1: Ha Sor<0 akkor Sor:=SY
     Pontrajzolás(Osz,Sor)
  Ciklus vége
Eljárás vége.
```

A. J betű kezelése 5 pont

B. B betű kezelése 5 pont

C. L betű kezelése 5 pont

D. F betű kezelése 5 pont

E. Üres kép, középről 5 pont

4. feladat: (45 pont)

Legyen `Első` az év első napjának héten belüli sorszáma (hétfő=1), `Hold` az első holdtölte napsorszáma (biztosan januárban van), `De` pedig igaz, ha az első holdtölte délelőtt van.

```
Húsvét(Év,Első,Hold,De):
  Első:=(8-Első) div 7     [Az első vasárnap]
  Napej:=31+28+21    [A tavaszi napejegyenloseg]
  Ha (Év mod 4=0 és Év mod 100≠0) vagy Év mod 400=0
     akkor Napej:=Napej+1
  Ciklus amíg Hold≤Napej   [Az első holdtölte utána]
     Hold:=Hold+29: Ha nem De akkor Hold:=Hold+1
     De:= nem De
  Ciklus vége
  Ciklus amíg (Hold mod 7)≠Első  [A következő vasárnap]
     Hold:=Hold+1
  Ciklus vége
  Ha Hold<Napej+12 akkor  Húsvét:="Március "
                          Nap:=Hold-Napej+20
                  különben Húsvét:="Április "
                          Nap:=Hold-Napej-11
  Ki: Húsvét,Nap,"."
  Ha Nap<31 akkor  Ki: Húsvét,Nap+1
           különben Ki: "Április 1."
Eljárás vége.
```

A. Holdtölték számolása	10 pont
B. Jó holdtölte megtalálása	5 pont
C. Ezutáni vasárnap megtalálása	20 pont
D. Szökőévek kezelése	5 pont
E. Beolvasás, kiírás	5 pont

Elérhető összpontszám: 180 pont

1992. Első forduló

1. feladat: (15 pont)

```
Összeadás(A,B,C,N):
  ÁT:=0
  Ciklus I=0-tól N-ig
     C(I):=A(I)+B(I)+ÁT
     Ha C(I)>9 akkor  ÁT:=1: C(I):=C(I)-10
              különben ÁT:=0
  Ciklus vége
  C(N+1):=ÁT
Eljárás vége.
```

A. Megfelelő számjegyek összeadása 6 pont

B. Ha 10-nél nagyobb, akkor átvitelt számol 3 pont

C. Az átvitelt hozzáadja a következő számjegyhez 3 pont

D. C(N+1)-et is kiszámolja (0 vagy 1 lesz) 3 pont

2. feladat: (28 pont)

A. Kurzormozgató billentyűk kezelése 2-2-2-2 pont

B. Kurzorválasztó billentyűk kezelése 2-2-2-2 pont

C. Kurzorok jó kirajzolása 3-3-3-3 pont

3. feladat: (22 pont)

```
Ütések(Bsor,Boszlop,Fsor,Foszlop,Hsor,Hoszlop):
  Bástya(Bsor,Boszlop,Fsor,Foszlop)
  Bástya(Bsor,Boszlop,Hsor,Hoszlop)
  Futó(Fsor,Foszlop,Bsor,Boszlop)
  Futó(Fsor,Foszlop,Hsor,Hoszlop)
  Huszár(Hsor,Hoszlop,Bsor,Boszlop)
  Huszár(Hsor,Hoszlop,Fsor,Foszlop)
Eljárás vége

Bástya(i,j,k,l):
  Ha i=k vagy j=l akkor Ki: "Bástya üt:",k,l
Eljárás vége.

Futó(i,j,k,l):
  Ha abs(i-k)=abs(j-l) akkor Ki: "Futó üt:",k,l
Eljárás vége.

Huszár(i,j,k,l):
  Ha abs(i-k)=2 és abs(j-l)=1 vagy abs(i-k)=1 és abs(j-l)=2
     akkor Ki: "Huszár üt:",k,l
Eljárás vége.
```

A. Bástya ütésének vizsgálata a bástya sorában 3 pont

B. Bástya ütésének vizsgálata a bástya oszlopában 3 pont

C. Futó ütésének vizsgálata az egyik átlóban 4 pont

D. Futó ütésének vizsgálata a másik átlóban 4 pont

E. Huszár ütésének vizsgálata a 8 lehetséges helyen 8 pont

4. feladat: (21 pont)

A. A betű	3 pont
20 egységnyi szárakkal	2 pont
középen 10 egységnyi összekötő vonal	2 pont
B. E betű	3 pont
vízszintes vonalai 10 egységnyiek	2 pont
függőleges vonala 20 egységnyi	2 pont
C. H betű	3 pont
vízszintes vonala 10 egységnyi	2 pont
függőleges vonalai 20 egységnyiek	2 pont

5. feladat: (14 pont)

A. 69 másodperc után növekszik a perc (59 helyett)	4 pont
(ugyanez a probléma az óra, perc pár hasonló állásánál)	1 pont
(1030-ban A(I)<=5-öt kellene vizsgálni)	
B. A perc változásakor a másodperc 10-es helyiértékű része nem lesz 0	4 pont
(ugyanez a probléma az óra, perc pár hasonló állásánál)	1 pont
(az 1040-es sorból hiányzik az A(I)=0)	
C. Az óra csak 23-ig mehetne, utána 0-ra kellene állítani	4 pont

Elérhető összpontszám: 100 pont

1992. Második forduló

1. feladat: (24 pont)

```
Szorzás(A,B,C,N):
  ÁT:=0
  Ciklus I=0-tól 2*N+1-ig
     C(I):=ÁT
     Ciklus J=max(0,N-I)-tól min(N,I)-ig
        C(I):=C(I)+A(J)*B(I-J)
     Ciklus vége
     ÁT:=C(I) div 10: C(I):=C(I) mod 10
  Ciklus vége
Eljárás vége.
```

A. Megfelelő számjegyek összeszorzása	8 pont
B. Az eredménybe jól számítja be a részletszorzatokat	10 pont
C. Ha 10-nél nagyobb, akkor átvitelt számol	3 pont
D. Az átvitelt hozzáadja a következő számjegyhez	3 pont

2. feladat: (18 pont)

A. Mitsubishi autóembléma (3 db	1 pont
x élhosszúságú	3 pont
rombusz	3 pont

120-120 fokkal elforgatva) 2 pont

B. Audi autóembléma (4 db 1 pont

10 egységnyire egymásbafonódó 3 pont

360 egység kerületű 2 pont

kör) 3 pont

3. feladat: (16 pont)

```
Hangrendek(Sor):
  Ciklus I=1-től hossz(Sor)-ig
     Ha Sor(i)∈{"a","á","o","ó","u","ú"} akkor Ki: "Mély"
     különben Ha Sor(i)∈{"e","é","i","í","ö","ő","ü","ű"}
                  akkor Ki: "Magas"
  Ciklus vége
Eljárás vége.
```

A. Magas magánhangzók felismerése 8 pont

B. Mély magánhangzók felismerése 6 pont

C. Mássalhangzóhoz nem rendel hangrendet 2 pont

4. feladat: (18 pont)

Jelenleg a ciklus belsejében 4 szorzás-osztás, valamint 1 hatványozás van, ezt kell csökkenteni.

A. Hatványozás megszüntetése 6 pont

B. Szorzások megszüntetése szorzásonként 5-5 pont

5. feladat: (10 pont)

A. Az autót egy zárt részbe helyeztük, ahonnan nem lehet kijutni 5 pont

B. Az autót egy olyan részbe helyeztük, ahol a "bal kéz a falon" szabállyal körbe lehet járni 5 pont

6. feladat: (14 pont)

A1. "Általános iskola" 2 pont

A2. "Fallal bekerített terület." 2 pont

A3. "13"+chr$(0)+chr$(7)+"-szor ismételd meg." 4 pont

B. szöveget tömörít 4 pont

a 2-nél többször ismétlődő karaktereket karakter + chr$(0) + chr$(darabszám) hármassal helyetesíti 2 pont

Elérhető összpontszám: 100 pont

1992. Harmadik forduló

1. feladat: (20 pont)

```
Háromszögfestő(a,b,c,d,e,f):
  h1:=négyzetgyök((a-c)^2+(b-d)^2)
  h2:=négyzetgyök((a-e)^2+(b-f)^2)
  h3:=négyzetgyök((e-c)^2+(f-d)^2)
  Ha h1≥h2+h3 vagy h2≥h1+h3 vagy h3≥h1+h2 akkor Hiba
     különben Szakasz(a,b,c,d): Szakasz(a,b,e,f)
              Szakasz(e,f,c,d)
              Ha d<b akkor Csere((a,b),(c,d))
              Ha f<b akkor Csere((a,b),(e,f))
              Ha f<d akkor Csere((e,f),(c,d))
              Ciklus I=b-től d-ig
                k:=((a,b),(c,d)) egyenes x-koordintája(i)
                v:=((a,b),(e,f)) egyenes x-koordintája(i)
                Szakasz(k,i,v,i)
              Ciklus vége
              Ciklus I=d+1-től f-ig
                k:=((e,f),(c,d)) egyenes x-koordintája(i)
                v:=((a,b),(e,f)) egyenes x-koordintája(i)
                Szakasz(k,i,v,i)
              Ciklus vége
  Elágazás vége
Eljárás vége.
```

A. A háromszög adatai beolvasása	3 pont
B. Ez tényleg egy háromszög	3 pont
C. Háromszög rajzolás	3 pont
D. Jó sorokban húz vonalat	3 pont
E. A sorban jó helyen kezdi	4 pont
F. A sorban jó helyen fejezi be	4 pont

2. feladat: (40 pont)

Az első eljárás sorokra tördel, a továbbiak pedig az egyes margóhoz igazításokat végzik már egy képernyő sorba kiférő szövegekre.

```
Tördel(Szöveg,bal,jobb,hogyan):
  Elsőszó(Szöveg,Szó): Sor:=Szó: db:=1
  Ciklus amíg Szöveg≠""
     Elsőszó(Szöveg,Szó)
     Ha hossz(Sor+Szó)+1≤jobb-bal+1 akkor Sor:=Sor+" "+Szó
                                          db:=db+1
        különben Igazítás(Sor,hogyan,bal,jobb,db): Sor:=Szó
                 db:=1
  Ciklus vége
  Igazítás(Sor,hogyan,bal,jobb,db)
Eljárás vége.
```

A fenti eljárás azt is meghatározza, hogy melyik sorba hány szó kerül. Ezt az információt csak a mindkét margóhoz igazítás használja fel.

```
Elsőszó(Szöveg,Szó):
  Szó:="": I:=1
  Ciklus amíg I≤hossz(Szöveg) és Szöveg(I)≠" "
     Szó:=Szó+Szöveg(I): I:=I+1
  Ciklus vége
  Szöveg:=Végéig(Szöveg,I+1)  [az adott pozíciótól a végéig]
Eljárás vége.
```

```
Balra(Sor,bal,jobb,db):
  Ciklus I=1-től bal-ig
     Sor:=" "+Sor
  Ciklus vége
  Ki: Sor
Eljárás vége.
```

```
Jobbra(Sor,bal,jobb,db):
  Ciklus amíg hossz(Sor)<jobb
     Sor:=" "+Sor
  Ciklus vége
  Ki: Sor
Eljárás vége.
```

```
Középre(Sor,bal,jobb,db):
  Ciklus amíg hossz(Sor)<jobb-1
     Sor:=" "+Sor+" "
  Ciklus vége
  Ki: Sor
Eljárás vége.
```

```
Margókhoz(Sor,bal,jobb,db):
  hány:=db div (jobb-bal+1): marad:=db mod (jobb-bal+1)
  Ujsor:=""
  Ciklus I=1-től hossz(Sor)-ig
     Ha Sor(I)≠" " akkor Ujsor:=Ujsor+Sor(I)
        különben k:=hány: Ha marad>0 akkor k:=k+1
                 Ciklus J=1-től K+1-ig
                   Ujsor:=Ujsor+" "
                 Ciklus vége
                 marad:=marad-1
     Elágazás vége
  Ciklus vége
  Ki: Ujsor
Eljárás vége.
```

A. Beolvasás ellenőrzéssel	7 pont
B. Jól tördel sorokra	10 pont
C. Balmargóhoz igazítás	3 pont
D. Jobbmargóhoz igazítás	4 pont
E. Középre igazítás	4 pont
F. Mindkét margóhoz igazítás	12 pont

3. feladat: (20 pont)

A `Hónap` tömb az adott hónap napjai számát adja meg, nem szökőévet feltételezve. A `Hét` tömb a hét napjai nevét tartalmazza, ahol Hét(0)="Hétfő".

```
Napmegadás(Év,Hó,Elsőnap):
  E:=0
  Ciklus amíg Elsőnap≠Hét(E)
     E:=E+1
  Ciklus vége
  DB:=E+(Év-1901)*365      [minden év]
  DB:=DB+(Év-1900) div 4      [4-gyel oszthatóak]
  DB:=DB-(Év-1900) div 100 [100-zal oszthatóak]
  DB:=DB+(Év-1600) div 400 [400-zal oszthatóak]
  Ciklus I=1-től Hó-1-ig
     DB:=DB+Hónap(I)       [hónap]
  Ciklus vége
  Ha (Év div 4=0 és Év div 100≠0 vagy Év div 100=0)
        és Hó>2 akkor DB:=DB+1
  Ciklus I=1-től Hónap(Hó)-ig
     DB:=DB+1: Ki: Hét(DB mod 7)
  Ciklus vége
Eljárás vége.
```

A. Beolvasás 3 pont

B. Hónapok napszámait jól kezeli 4 pont

C. A februárt szökőévben is jól kezeli 3 pont

D. A hónap első napját jó heti napra teszi 5 pont

E. Az év első napját jó heti napra teszi 5 pont

Elérhető összpontszám: 80 pont

1993. Első forduló

1. feladat: (14 pont)

A. Y=INT(100*X)/100 — 3 pont

B. Y=X*(1+K/100)^N (de lehet ciklussal is számolni) — 3 pont

C. Y=INT((B-A)*RND(0)+A) (a +A lehet a zárójelen kívül is) — 3 pont

D.
```
Y=0
DO WHILE X<>0
  H=INT(X/10): Y=Y+(X–10*H): X=H
LOOP
```
5 pont

2. feladat: (20 pont)

A. Téglalap alapú mozaik, (parkettaminta, padlóminta ...) — 3+3 pont

B. N sora, M oszlopa van, — 1+1 pont

C. az alaptéglalap oldalai X, illetve Y hosszúságúak, — 2+2 pont

D. az Y oldalhosszúságú oldalai egy-egy felét kivágja, — 2 pont

E. s befelé, illetve kifelé fordított — 2 pont

F. egyenlő oldalú háromszöggel helyettesíti. — 4 pont

3. feladat: (20 pont)

Összeadás(A,B,C):

```
X  = Lnko(An,Bn)                    3 pont
Csz= Asz*(Bn/X)+Bsz*(An/X)          2+2 pont
Cn = An*Bn
X  = Lnko(Csz,Cn)
Csz= Csz/X                          megmaradásuk 3 pont
Cn = Cn/X
```

Eljárás vége.

Szorzás(A,B,C):

```
X  = Lnko(An,Bsz)                   3 pont
Y  = Lnko(Asz,Bn)                   3 pont
Csz= (Asz/Y)*(Bsz/X)                1+1 pont
Cn = (An/X)*(Bn/Y)                  1+1 pont
```

Eljárás vége.

4. feladat: (20 pont)

A. 121 IF X+R<0 OR X+R>{kép oszlopai száma} THEN 120

131 IF Y+S<0 OR Y+S>{kép sorai száma} THEN 130

(Ez a két sor a fenti sorszám és a 150-es sor között bárhol lehet, a 130 helyett 120 is szerepelhet)

relációnként — 2-2 pont

jó helyre ugrásonként 2-2 pont

B. a hibás sorok kijavítva (a versenyzőnek csak jelölni kell a hibát, javítást nem várunk tőle)

110-es sor miatt már a kezdőpontra leáll a ciklus 2 pont

(ciklusba lépéskor ki kell kerülni, vagy a ciklus végén vizsgálni)

130 S=RND(3)–2 2 pont

140 OR helyett AND zárja ki a helybenmaradást 2 pont

160 X=X+R 2 pont

5. feladat: (26 pont)

Egy lehetséges, egyszerű megoldás:

```
100 FOR I=1 TO 4                                              2 pont
110 PRINT "AZ";I;". JÁTÉKOS LAPJAI:"
120 P=0: Z=0: M=0: T=0: A=0                                   1-1 pont
130 FOR J=1 TO 8                                              2 pont
140   IF LEFT$(A$(I,J),5)="PIROS" THEN P=P+1                  4 pont
150   IF LEFT$(A$(I,J),4)="ZÖLD" THEN Z=Z+1                   4 pont
160   IF LEFT$(A$(I,J),4)="MAKK" THEN M=M+1                   4 pont
170   IF LEFT$(A$(I,J),3)="TÖK" THEN T=T+1                    4 pont
180   IF RIGHT$(A$(I,J),3)="ÁSZ" THEN A=A+1                   4 pont
190 NEXT J
200 PRINT "PIROS:";P,"ZÖLD:";Z,"MAKK:";M,"TÖK:";T,"ÁSZ:";A
210 NEXT I
```

Elérhető összpontszám: 100 pont

1993. Második forduló

1. feladat: (20 pont)

```
100 FOR I=1 TO 4
110 PRINT "AZ";I;". JÁTÉKOS LAPJAI:"
120 P=0
130 FOR J=1 TO 8
140   IF LEFT$(A$(I,J),5)="PIROS" THEN A$(I,J):=MID$(A$(I,J),7)
150   IF LEFT$(A$(I,J),4)="ZÖLD" THEN A$(I,J):=MID$(A$(I,J),6)
160   IF LEFT$(A$(I,J),4)="MAKK" THEN A$(I,J):=MID$(A$(I,J),6)
170   IF LEFT$(A$(I,J),3)="TÖK" THEN A$(I,J):=MID$(A$(I,J),5)
180   IF A$(I,J)="ÁSZ" THEN P=P+11
181   IF A$(I,J)="KIRÁLY" THEN P=P+4
182   IF A$(I,J)="FELSŐ" THEN P=P+3
183   IF A$(I,J)="ALSÓ" THEN P=P+2
184   IF A$(I,J)="TIZES" THEN P=P+10
185   IF A$(I,J)="KILENCES" THEN P=P+9
186   IF A$(I,J)="NYOLCAS" THEN P=P+8
187   IF A$(I,J)="HETES" THEN P=P+7
190 NEXT J
200 PRINT "PONTÉRTÉKE:";P
210 NEXT I
```

A. A$(I,J)-ből a színek elhagyása. 4 pont

B. Az egyes figurákhoz jól rendel számokat. 8 pont

C. Jó összegzés. 4 pont

D. Mind a 4 versenyzőre megteszi. 4 pont

2. feladat: (30 pont)

A feladat egy lehetséges megoldó algoritmusa:

```
Jegylyukasztás:
  Ciklus I=1-től 7-ig
    Ciklus J=I+1-től 9-ig
      Ha nem (I=3 vagy I=6 vagy
              (I=2 vagy I=5) és(J MOD 3=0) )
         akkor Ki: I,J
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

A. Külső ciklus 1-től 8-ig (sőt 7-ig). 3 pont

B. Belső ciklus I+1-től 9-ig. 3 pont

C. (I,J) nem jó, ha I=3 vagy I=6 (I jobboldali oszlopban van). 6 pont

D. (I,J) nem jó, ha I középső, J pedig jobboldali oszlopban van. 18 pont

3. feladat: (25 pont)

Mitkeres:

Egyenesen megy előre, 3 pont
amíg falhoz nem ér, 3 pont
ezután balra fordul és 3 pont
végrehajtja a *mitcsinál* eljárást. 3 pont

Mitcsinál:

A megtalált falat körbejárja úgy, 5 pont
hogy a jobboldala mindig a falhoz ér. 4 pont
Egy helyben forog, ha nem fal mellett kezdték el használni. 4 pont

4. feladat: (25 pont)

A: A 3 magasságú oszlopok lebontása. 4 pont

A 2 magasságú oszlopok lebontása. 6 pont

A kockák jó összerakása. 5 pont

B. A kockák egymás mellett legyenek, s ne egymáson. 5 pont

Ekkor a megoldás vége: 5 pont

```
Fogd meg a zöld színű kockát!
Tedd a piros színű kockára!
Fogd meg a kék színű kockát!
Tedd a zöld színű kockára!
```

Elérhető összpontszám: 100 pont

1993. Harmadik forduló

1. feladat: (60 pont)

```
Biliárd:
  Be: X,Y,DX,DY,BX,BY
  Ciklus amíg nem (X=BX és Y=BY) és [bábut dönt]
    nem (Y=0 és (X=0 vagy X=maxX div 2 vagy XC=maxX)) és
    nem (Y=maxY és (X=0 vagy X=maxX div 2 vagy XC=maxX))
        Ha (X+DX<0) vagy (X+DX)>maxX akkor DX:=-DX
        Ha (Y+DY<0) vagy (Y+DY)>maxY akkor DY:=-DY
        X:=X+DX: Y:=Y+DY
        Pontrajzol(X,Y)
  Ciklus vége
Eljárás vége.
```

A. Beolvasás.	6 pont
B. Az asztal kirajzolása.	6 pont
C. A golyó mozog 8 irányba.	16 pont
D. A golyó a 4 falról visszapattan.	16 pont
E. A golyó a 6 lyukon leesik.	12 pont
F. A golyó a bábut feldönti.	4 pont

2. feladat: (40 pont)

A feladat lényege: hogyan lehet meghatározni egy számjegy tükörképét a buszjegyen:

```
Tükörkép(A):
  B:=3*((A-1) div 3)+3-((A-1) mod 3)
Eljárás vége.
```

A. Beolvasás.	5 pont
B. Keret kirajzolása.	6 pont
C. Számok kiírása.	9 pont
D. Jó helyen van a két lyuk.	10 pont
E. Jó a tükörkép meghatározása.	10 pont

Elérhető összpontszám: 100 pont

1994. Első forduló

1. feladat: (26 pont)

ELSOKEP: A oldalhosszúságú	2 pont
négyzetet rajzol	2 pont
MASODIKKEP: X,Y oldalhosszúságú	2+2 pont
téglalapot rajzol	2 pont
HARMADIKKEP: B oldalhosszúságú	2 pont
C, illetve 180-C fokos szöget bezáró oldalú	2+1 pont
rombuszt rajzol	2 pont
NEGYEDIKKEP: D,E oldalhosszúságú	2+2 pont
F, illetve 180-F fokos szöget bezáró oldalú	2+1 pont
paralelogrammát rajzol	2 pont

2. feladat: (25 pont)

Első: eldönti, hogy az X osztható-e 2-vel	5 pont

A megoldás módszere: párosak azok a számok, amelyek utolsó számjegye 0, 2, 4, 6, vagy 8

Második: eldönti, hogy az X osztható-e 3-mal	10 pont
(ha csak a 10-nél kisebb számokra ad jó választ, akkor csak	4 pont)

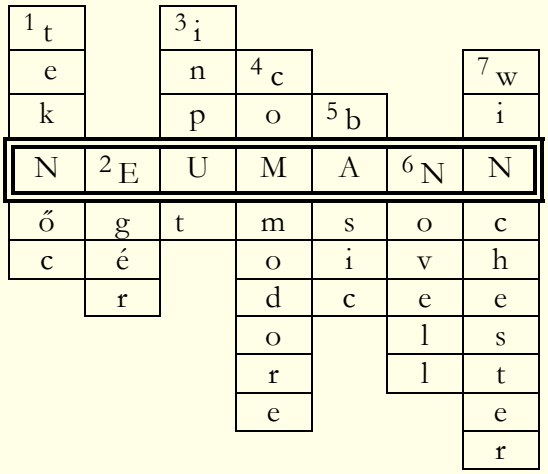
A megoldás módszere: azok a számok oszthatók hárommal, amelyek számjegyeinek összege 0, 3, 6, vagy 9.

Kiszámol: kiszámolja X számjegyeinek összegét,	3 pont
rekurzívan, amíg egyjegyű számot nem kap	3 pont
Harmadik: eldönti, hogy az X osztható-e 5-tel	4 pont

A megoldás módszere: azok a számok oszthatók öttel, amelyek utolsó számjegye 0 vagy 5.

3. feladat: (24 pont)

A. Helyes oszloponként	3-3 pont
B. Neumann János kitalálásáért	3 pont



4. feladat: (25 pont)

```
Összeadás(A,B):
  Ciklus amíg Hossz(A)≠Hossz(B)
    Ha Hossz(A)<Hossz(B) akkor A:="0"+A különben B:="0"+B
  Ciklus vége
  ÁT:=0: C:=""
  Ciklus I=hossz(A)-tól 1-ig -1-esével
    Ha A(I)="0" és B(I)="0" akkor Ha ÁT=0 akkor C:="0"+C
                                  különben C:="1"+C: ÁT:=0
    Ha A(I)≠B(I) akkor Ha ÁT=1 akkor C:="0"+C
                       különben C:="1"+C
    Ha A(I)="1" és B(I)="1" akkor Ha ÁT=1 akkor C:="1"+C
                                  különben C:="0"+C: ÁT:=1
  Ciklus vége
  Ha ÁT=1 akkor C:="1"+C
Eljárás vége.
```

A. A két szám beolvasása	4 pont
B. Helyiértékenkénti összeadás	10 pont
C. Átvitel jó kezelése	5 pont
D. Túlcsordulás jelzése	4 pont
E. Az eredmény kiírása	2 pont

Elérhető összpontszám: 100 pont

1994. Második forduló

1. feladat: (40 pont)

Csak a feldolgozó eljárást adjuk meg az alábbiakban:

```
Lencsék(LENCSE,N,LSZIN,DB,LDB):
  DB:=0
  Ciklus I=1-től N-ig
     J:=1
     Ciklus amíg J≤DB és LENCSE(I)≠LSZIN(J)
        J:=J+1
     Ciklus vége
     Ha J>DB akkor  DB:=DB+1: LSZIN(DB):=L(I): LDB(DB):=1
           különben LDB(J):=LDB(J)+1
  Ciklus vége
Eljárás vége.
```

A. 1-féle lencséből 1 darabot felismer	4 pont
B. 1-féle lencséből több darabot felismer	6 pont
C. sokféle lencséből 1 darabot felismer	6 pont
D. sokféle (szomszédos) lencséből sok darabot felismer	10 pont
E. sokféle (nem szomszédos) lencséből sok darabot felismer	10 pont
F. beolvasás	2 pont
G. darabszám kiírás	1 pont
I. H. lencsék kiírása	1 pont

2. feladat: (20 pont)

A három eljárást úgy kell megírni, hogy semmiféle számérték, még a szöveg hossza se szerepeljen benne.

```
Összead(A,B,C):
  C:=A
  Ciklus amíg B≠"|"
     C:=C+"|": B:=elsőutániak(B)
  Ciklus vége
Eljárás vége.
```

```
Kivon(A,B,C):
  Ha A≥B akkor C:=A
               Ciklus amíg B≠"|"
                 C:=elsőutániak(C): B:=elsőutániak(B)
               Ciklus vége
  Elágazás vége
Eljárás vége.
```

```
Szoroz(A,B,C):
  C:=""
  Ciklus amíg B≠"|"
     C:=C+A: B:=elsőutániak(B)
  Ciklus vége
Eljárás vége.
```

A. Számok beolvasása	2 pont
B. Összeadás	pont

C. Kivonás 5 pont

D. Szorzás 7 pont

E. Eredmény kiírása 1 pont

3. feladat: (40 pont)

Olyan eljárást adunk, amely egy N elemű tömb alapján hajtja végre az utasításokat.

```
Logo-értelmező(N,UT):
  X:=maxX div 2: Y:=maxY div 2: IR:=0: TOLL:=igaz
  Ciklus I=1-től N-ig
     Elágazás
       UT(I)="Előre" esetén Mozdulás(10)
       UT(I)="Hátra" esetén Mozdulás(-10)
       UT(I)="Balra" esetén IR:=IR-1: Ha IR<0 akkor IR:=3
       UT(I)="Jobbra" esetén IR:=IR+1: Ha IR>3 akkor IR:=0
       UT(I)="TollFel" esetén TOLL:=hamis
       UT(I)="TollLe" esetén TOLL:=igaz
     Elágazás vége
  Ciklus vége
Eljárás vége.
```

```
Mozdulás(T):
  Ha IR mod 2=0 akkor  UJY:=Y+(IR-1)*T: UJX:=X
                különben UJX:=X-(IR-2)*T: UJY:=Y
  Ha TOLL akkor Szakasz(X,Y,UJX,YJY)
  X:=UJX: Y:=UJY
Eljárás vége.
```

A. Előre 9 pont

B. Hátra 9 pont

C. Jobbra 4 pont

D. Balra 4 pont

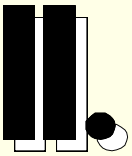
E. TollFel 2 pont

F. TollLe 2 pont

G. Kezdőállapot 3 pont

H. Beolvasás előre 7 pont

Elérhető összpontszám: 100 pont



2. Nemes Tihamér Nemzetközi Informatikai Tanulmányi Verseny

1985. Első forduló

Első-ötödik osztályosok

1. feladat: (4 pont)

A szubrutinnak átadott számokat az M1, M2 és M3 változókban balról jobbra (nem szigorúan) monoton csökkenő sorrendbe rendezi.

A. Ha szerepel, hogy nem szigorúan monoton csökkenő sorrendbe rendez: 3 pont

B. Ha csak annyi szerepel, hogy monoton csökkenő sorrendbe rendez: 2 pont

C. Ha az is szerepel, hogy balról jobbra rendez: +1 pont

2. feladat: (5 pont)

A program egy természetes szám (≥2) prímosztóit írja ki. A külső ciklus osztókat válogat ki, s a belső elosztja N-et az osztóval annyiszor, ahányszor csak lehet.

Ha a ≥2 feltétel elmarad, akkor legfeljebb 4 pont adható.

3. feladat: (6 pont)

A B() vektorban egyetlen olyan elem sem fordulhat elő kétszer, amely A()-ban is szerepel.

A. Ha B() minden eleméről kikötik, hogy csak egyszer fordulhat elő, akkor 4 pont

B. Ha A()-ban sem engedik meg, hogy egy elem kétszer szerepeljen, akkor 2 pont

C. Ha M=1 feltételt írnak, akkor 1 pont

D. Ha A()-ban nem engedik meg az ismétlődést, de B()-ben igen, akkor 0 pont

4. feladat: (6 pont)

A. Mindkét program 2**n-1 és -2**n értékeit nyomtatja ki (ahol a hatványozást ** jelöli). 1 pont

B. Mindkét program túlcsordulással (OV error) áll meg, mivel nem gondoskodtak arról, hogy L% ne váljék 32767-nél nagyobbá. 2 pont

C. Az 1. program már a K%=16383 és a -K%-1=-16384 értékek kiírása után megáll, mivel ezt követően (a következő szám kiírása előtt) bekövetkezik a túlcsordulás. A 2. program még ki tudja nyomtatni a K%= 32767 és a J%=-32768 értékeket (ez a legnagyobb, ill. a legkisebb ábrázolható egész szám a HT-1080Z gépen), mert túlcsordulás csak e számok kiírása után keletkezik. 3 pont

5. feladat: (6 pont)

A program a klasszikus *buborékos rendezés* algoritmus alapváltozata.

A. A THEN után az I index helyett J kell. 2 pont

B. A THEN után hibás a csere. 2 pont

C. I=N és J=I mellett az A(J+1) hivatkozás hibás (valamelyik ciklus határai hibásak). 2 pont

6. feladat: (8 pont)

A program a listás beillesztéses rendezés beillesztés szubrutinja, a mutatók tömbindexek, a lista végét a -1 mutató (tömbindex) jelzi.

A. N=0 2 pont

B. T(0)=-1 6 pont

7. feladat: (10 pont)

Egyenes szakaszt rajzol (összefüggő vonallal) az (X1,Y1) és az (X2, Y2) pontok közé. H1, H2 a végpontok X-, illetve Y-koordinátájának eltérése. Helyesen működik, ha a két pont a képernyőn van (HT-1080Z esetén 0≤X1≤127 és 0≤Y1≤47). A 60-as és 70-es sor a két pont koordinátánkénti távolságának maximumát határozza meg H-ban. Ha a távolság 1, akkor 2 pontot kell rajzolni: ha H, akkor H+1-et. S1, S2 az X-, illetve Y-irányú lépésköz. egyikük abszolút értéke biztosan +1, másikuk pedig ennél nem lehet nagyobb.

A. A működés kitalálása: 4 pont

B. A helyes működés feltétele: 2 pont

C. A 60-as és a 70-es sor szerepe: 2 pont

D. H+1 szerepe: 2 pont

8 feladat: (15 pont)

A. Ilyenkor B%-ba 0 kerül, ezért a gép az 1000-es és 1010-es sorokat ismételgeti. 1 pont

B. (Mind a számjegyek, mind a betűk ASCII-kódja követi azok természetes sorrendjét, ezért a kód a decimális értékből lineáris eltolással valóban előállítható. Az ASCII-kódtáblán azonban a számjegyek után nem a betűk következnek, hanem a kettőspont, a pontosvessző, a kisebb-jel stb. Ezért:)

Az 1040-es sorban a szubrutin a számjegyek kódját jól képezi, az A..F betűk kódja helyett azonban a kettőspont, pontosvessző, a kisebb-jel stb. kódját állítja elő. 2 pont

A válasz 1 pontot ér, ha csak azt tartalmazza, hogy az előállított kód körül van a hiba, de nem mondja meg, hogy a számjegyek kódja jó, a betűké a rossz.

C. Az 1040-es sor legyen pl. az alábbi: 2 pont

```
1040 IF A%<10  THEN A%=A%+ASC("0")
               ELSE A%=A%-10+ASC("A")
```

Nem szép, nem elegáns – de jó – megoldásért 1 pont jár.

D. (Az egyes billentyűk hatására 2 megfelelő hatványa, több billentyű egyidejű lenyomására pedig a hatványok összege kerül a B% változóba. Az 1030-as sor egész osztással az előforduló legnagyobb hatványkitevő értékét számítja ki. Tehát:)

Ha egyszerre több billentyűt ütünk le, akkor közülük mindig a legnagyobb értékűnek megfelelő számot állítja elő a szubrutin az A% változóban. Az "F" billentyű lenyomásakor B%-ba negatív szám kerül, ezért hatása eltér a többi billentyűétől. 2 pont

Kicsit pontatlan válaszért 1 pont adható.

E. Ha az "F" billentyűvel együtt egy vagy több másikat is leütünk, a szubrutin sohasem a neki megfelelő 15-öt írja A%-ba! Ugyanis az "F" hatványozás jele – a BASIC interpreter -32768-nak tekinti (a legnagyobb helyiértékű bit az előjelbit), s a többi billentyű lenyomásából adódó értéket ehhez adja hozzá.

Ezért a szubrutin a legkisebb értékű lenyomott billentyűnek megfelelő számot állítja elő A%-ban, ha az e fölött lévő összes billentyű le van nyomva, az ez alatt lévők pedig mind el vannak engedve.

Legyen ez a i-edik billentyű (0≤i≤14). Akkor B%-ban

```
-2**15+2**14+...+2**i=(-2**(15-i)+2**(14-i)+...+1)2**i=-2**i
```

lesz, A%-ba tehát i-t tesz a szubrutin.

Példa:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0  @: lenyomott billentyű
@ @ @ @ @ @ @ - - - - - - - - -  -: elengedett billentyű
```

A szubrutin most A%-ba 9-et tesz.

Ugyanakkor a szubrutin a legnagyobb értékű elengedett billentyűnek megfelelő számot adja vissza A%-ban, ha az e fölött lévő összes billentyű le van nyomva és az ez alatt lévők közül is legalább egy le van nyomva. Legyen ez j-edik billentyű (1≤j ≤14).

Ha a j-edik és alatta mind el lenne engedve, B% -2**(j+1) lenne.

(Ez volt az előző eset; most legalább egy be van nyomva a j alattiak közül.)

Ha a j-edik kivételével mindegyik le van nyomva, B%-ba

```
-2**(j+1)+2**(j-1)+2**(j-2)+...+2**0=
              =-2**j-(2**j-2**(j-1)-...-2**0)=-2**j-1
```

kerül. Vagyis 2**(j+1)>³B%³≥2**j+1, tehát a szubrutin valóban a j számot írja A%-ba.

Példa:

```
F E D C B A 9 8 7 6 5 4 3 2 1 0  @: lenyomott billentyű
@ @ @ @ @ @ @ - - - @ - - - -  -: elengedett billentyű
```

A szubrutin most A%-ba 8-et tesz.

Hibátlan válaszra 5 pontot lehet adni. Értékelhető részmegoldásokra arányosan kevesebb pont jár.

F. Az új 1020-as és 1030-as sorok a véges, előjeles 16 bites aritmetika egy tulajdonsága miatt működnek jól: 2**14 és 2**15-1 közé eső pozitív számok kétszerese negatív! 3 pont

Ha az "F" billentyűt is leütjük – akár egyedül, akár többedmagával –, B% negatív lesz, és ezért 1030 THEN ágára nem kerül sor, A%-ban 15 marad. Ha "F"-et nem nyomjuk le, akkor B%-ot addig szorozzuk 2-vel (azaz léptetjük a tartalmát balra) és egyúttal csökkentjük egyesével A%-ot, amíg B% negatívvá nem válik. Hogy ez mindenképpen bekövetkezzék, arról az 1010-es sor gondoskodik: addig nem is kerül 1020-ra a vezérlés, amíg B% valamelyik bitje nem billen be.

Értékelhető részmegoldásokért arányosan kevesebb pont adható.

Elérhető összpontszám: 60 pont

1985. Második forduló

Első-ötödik osztályosok

Jelölje **D** a darazsak számát, **Darázs**(**I**) pedig azt, hogy az **I**. darázs melyik szobában van! Az **X** tartalmazza az 1. szobában levő darazsak aktuális számát, **S**(**I**) pedig azt, hogy hányszor volt az 1. szobában éppen **I** darab darázs.

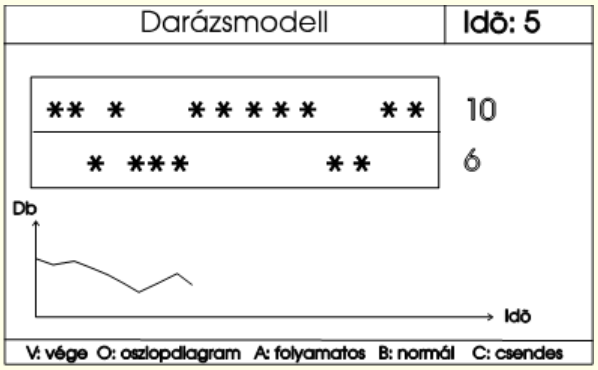
Az Üzemmód változó 4-es értéke jelzi, hogy befejeződött a szimuláció.

```
Darázsprogram:
  Tájékoztató írás
  Kezdeti értékek
  T:=0
  Kezdőábra
  Ciklus amíg Üzemmód<4
     Szimulációs lépés
     T:=T+1
     Eredmény írás
     Billentyűfigyelés
  Ciklus vége
  Összesítés
Program vége.
```

A darazsak számát és az első szobában levő darazsak számát beolvassuk, ezek alapján a darazsakat elhelyezzük a két szobában. Az eredmény változókat lenullázzuk. Az üzemmódot a normál megjelenítésre változtatjuk (2), s az időt is nullára állítjuk.

```
Kezdeti értékek:
  Be: D,X
  Ciklus I=1-től X-ig
     Darázs(I):=1
  Ciklus vége
  Ciklus I=X+1-től D-ig
     Darázs(I):=2
  Ciklus vége
  Ciklus I=0-tól N-ig
     S(I):=0
  Ciklus vége
  S(X):=1: Üzemmód=2
Eljárás vége.
```

A kezdőábra eljárás helyett egy mintát adunk a lehetséges képernyőképre:

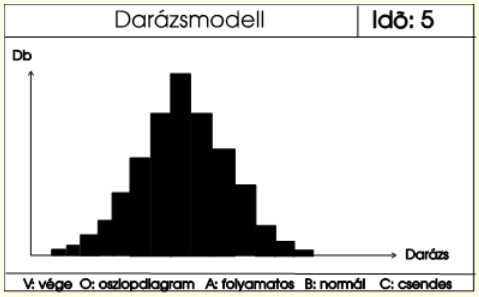



```
Szimulációs lépés:
  I:=Véletlen(N)      [1 és N közötti véletlen egész]
  Ha Darázs(I)=1  akkor  Darázs(I):=2: X:=X-1: S(X):=S(X)+1
                 különben Darázs(I):=1: X:=X+1: S(X):=S(X)+1
  Elágazás vége
Eljárás vége.
```

Az eredmények írását csak akkor kel elvégezni, ha nem a 3. üzemmódban (megjelenítés nélküli szimuláció) vagyunk. Ekkor a szimulációs lépésben elmozdított darazsat a régi helyéről le kell törölni, az új helyére ki kell rajzolni, újra ki kell írni mindkét szoba aktuális darázsszámát, majd az első szoba aktuális darázsszámát a grafikonra kell rajzolni.

```
Eredmény írás:
  Ki: T
  Ha Üzemmód<3 akkor Darázs törlés(I,3-Darázs(I))
                     Darázs rajzolás(I,Darázs(I))
                     Ki: X,D-X
                     Grafikon(X)
  Elágazás vége
Eljárás vége.

Billentyűfigyelés:
  Ha Üzemmód=1 akkor Billentyűlenyomásra vár
  Ha Van lenyomott billentyű
     akkor C:=Lenyomott karakter
            Ha C="A" akkor Üzemmód:=1
            Ha C="B" akkor Üzemmód:=2
            Ha C="C" akkor Üzemmód:=3
            Ha C="V" akkor Üzemmód:=4
            Ha C="O" akkor Oszlopdiagramrajzolás
                           Billentyűlenyomásra vár
                           Kezdőábra
Eljárás vége.
```

Az oszlopdiagram rajzolás eljárás megadása helyett szintén egy lehetséges ábrát mutatunk:



A. Szimulációs lépés	20 pont
B. A képernyőn látható az eltelt idő és az 1. szoba darázsszáma	2-2 pont
C. A képernyőn megjelenik a két szoba, és bennük a darazsak	5 pont
D. Megadható legyen a darazsak száma	2 pont
E. Megadható a kezdetben az első szobában levő darazsak száma	2 pont
F. A felhasználó menet közben bármikor beavatkozhat	2 pont
- befejezhet a program használatát	1 pont
- összegző oszlopdiagramot kérhet (hisztogram)	1 pont
- üzemmódot változtathat:	1 pont
a. minden röptetés után a program egy billentyű lenyomására vár	1 pont
b. folyamatos röpködés	1 pont
c. megjelenítés nélkül "gyorsan" végzi a szimulációs lépéseket	1 pont
G. Grafikon készítése a darazsak számáról az 1. szobában	20 pont
- a grafikonon mindig az utolsó 50 időegység állapota látszódik	7 pont
H. Összegző oszlopdiagram (hisztogram)	20 pont
- a grafikon szükséges kicsinyítése	10 pont
- a grafikon megnézése után mód van a szimuláció folytatására.	2 pont

Elérhető összpontszám: 100 pont

1986. Első forduló

Első-ötödik osztályosok

1. feladat: (11 pont)

1.1. B

1.2. A, B, D

1.3. C

1.4. A

1.5. A, E

1.6. B

1.7. A

1.8. B

1.9. D

1.10. C

Az 1.6. válasz 2 pontot ér, a többi 1 pontot. Csak a hibátlan megoldásokra jár pont!

2. feladat: (6 pont)

X tartalma:

7	6	5	4	3	2	1	0
5.	4.	3.	2.	1	1. vonat		

```
váltó állása:  sebessége:
 0 = egyenes   0...7 cm/s
 1 = kitérő
```

Y tartalma:

7	6	5	4	3	2	1	0
5.	4.	3.	2.	1	2. vonat		

```
jelző állása:  sebessége:
 0 = szabad    0...7 cm/s
 1 = tilos
```

A. A két vonat sebességét jelző bitek megadásáért: 2 pont

B. A váltók állását jelző bitek és jelentésük megadásáért: 2 pont

C. A jelzők állását jelző bitek és jelentésük megadásáért: 2 pont

D. Részmegoldásokért 1-1 pont adható.

3. feladat: (5 pont)

3.1. B

Tökéletes válaszért 3 pont, a C válaszért 2 pont, értékelhető magyarázatért 1 pont jár.

3.2. D

Tökéletes válaszért 2 pont, értékelhető magyarázatért 1 pont jár.

4. feladat: (4 pont)

4.1. tojáshab – BA, belekeverjük – A, tojás – BP, víz – KA, grízgaluska – KP, üvegesek nem lesznek – A.

4.2. Forró vízben addig főzzük, amíg a grízszemek üvegesek nem lesznek.

4.1. 2-2 jó válasz esetén 1-1 pont

4.2. Jó válasz esetén 1 pont

Más válasz is lehet jó!

5. feladat: (9 pont)

5.1. A 'hosszú várakozás' gondoskodik arról, hogy a botkormányt bármelyik irányba megnyomva, majd elengedve, a program csak egyetlen megnyomást érzékeljen.

Ha a botkormányt folyamatosan megnyomva tartjuk, a 'rövid várakozás' teszi lehetővé a gyors ismétlést.

5.2. Az 1010-es sor helyesen: A = A AND 15

5.3. Az 1020-1050-es sorokban a következő utasítások helyesek, tetszőleges sorrendben:

```
IF A = 14 THEN PRINT "É"; : RETURN
IF A = 11 THEN PRINT "D"; : RETURN
IF A = 13 THEN PRINT "K"; : RETURN
IF A =  7 THEN PRINT "N"; : RETURN
```

5.1. Ha csak egy válasz jó, 2 pont, ha mindkettő jó, 3 pont adható.

5.2. Jó válaszra 3 pont jár. 1 pont adható, ha rájön arra, hogy ez a sor hibás, de nem adja meg a helyes választ.

5.3. A három hibás utasítás kijavításáért darabonként 1-1 pont jár.

6. feladat: (8 pont)

A: $A*A \le N < (A+1)*(A+1)$, vagyis A egyenlő lesz N négyzetgyökének egészrészével 4 pont

B: $B = (A+1)*(A+1)$ 2 pont

C: C az (A+1). páratlan szám, azaz $2*A+1$ 2 pont

Részmegoldásonként arányosan kevesebb pont jár.

7. feladat: (11 pont)

7.1. A kép szürke pontjai világosabbak lesznek. A kontrasztok élesebbé válnak, mert a közepesen szürke képpontok jobban, a világosabb és a sötétebb képpontok viszont kevésbé világosodnak ki, a tiszta fehér és a tiszta fekete pedig nem változik meg. 5 pont

7.2. A képet felére kicsinyíti azáltal, hogy minden pontnégyest az átlagukkal helyettesít (az alábbi példában az azonos betűkkel jelölt pontokból lesz egy pont) : 3 pont

```
X X Y Y Z Z  ———>   X Y Z
X X Y Y Z Z
```

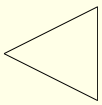
7.3. A kép bal felső sarkát 3-szorosra nagyítja azáltal, hogy az egyes pontokat vízszintes és függőleges irányban is megháromszorozza: 3 pont

```
X Y    ———>  X X X Y Y Y
A B           X X X Y Y Y
              X X X Y Y Y
              A A A B B B
              A A A B B B
              A A A B B B
```

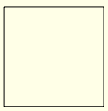
Részmegoldásokért arányosan kevesebb pont adható.

8. feladat: (6 pont) RAJZOLANDÓK!!!!!!

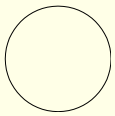
8.1. Egyenlő oldalú háromszög 100 egységnyi oldalhosszal. 1 pont



8.2. Négyzet 100 egység oldalhosszal. 1 pont

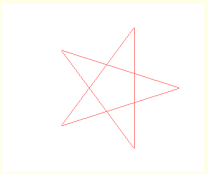


8.3. Szabályos 360-szög 1 egység oldalhosszal, azaz majdnem kör. 1 pont



8.4. Egyenlő oldalú háromszög 10 egység oldalhosszal. 1 pont

8.5. Ötágú csillag. 2 pont



Csak teljes értékű válaszokért jár a fenti pontszám!

9. feladat: (6 pont)

Leggyorsabb: ha az A() vektor eleve növekvően rendezett.

Leglassabb: ha az A() vektor eleve csökkenően rendezett.

A. Ha csak az egyik választ adja meg, 4 pont

B. ha mindkettőt. 6 pont

Elérhető összpontszám: 66 pont

1986. Második forduló

Első-ötödik osztályosok

Folyóiratküldő:

A feladat egy egyszerűsített adatbázis-kezelő program megírását tartalmazza, de igen erős korlátozásokkal, valamint a háttértár kezelést nem írja elő. Emiatt a memóriában egy egyszerű tömbben tárolhatók az alábbi adatok:

```
Újság=Rekord(név: string[25],város: 'A'..'P',kért: logikai)
Adatbázis: rekord(db: Egész,t: Tömb(1..30,Újság))
```

A feladat megoldására egy menürendszerű programot célszerű készíteni, amely főmenüje az alábbi lehet:

Tag és folyóirat nyilvántartás

A. Tagok adatainak nyilvántartása
B. Szállítási feladatok

Melyiket választod?

Mindkét feladathoz egy-egy újabb almenü tartozik:

Tag nyilvántartás

1. Üres adatbázis létrehozása
2. Új tag felvétele
3. Tag törlése
4. Tag adatainak módosítása
5. Tagok listája névsor szerint
6. Tagok listája városonként
0. Vissza az előző menühöz

Melyiket választod?

A 2., a 3 és a 4. pont megoldása elvileg kétféle lehet. Egyik esetben csupán egyetlen taggal kell foglalkozni. Ebben az esetben ezek egyszerűbbek, s a felhasználónak is könnyebb a tennivalója. Ha egyszerre több tagot is fel lehet vinnni, ..., akkor az egyes eljárásokban egy eszerinti ciklusnak kell szerepelnie, viszont minden egyes tag után rá kell kérdezni a folytatásra.

Mivel az adatbázis igen kis méretű, így a tárolás sorrendjének nincs semmilyen szerepe. Nézzük az egyes részfeladatokat megoldó algoritmusokat:

```
Változó A: Adatbázis

Üres adatbázis létrehozása:
  A.db:=0
Eljárás vége.
```

Az új tag felvételének, tag törlésének, adatai módosításának paramétere a tag megfelelő adata.

```
Új tag felvétele(név,város,kért):
  i:=1
  Ciklus amíg i≤A.db és név≠A.t(i).név
    i:=i+1
  Ciklus vége
  Ha i≤A.db akkor HIBAÜZENET('Volt már tag')
    különben A.db:=A.db+1; A.t(db).név:=név
             A.t(db).város:=város; A.t(db).kért:=kért
  Elágazás vége
Eljárás vége.

Tag törlése(név):
  i:=1
  Ciklus amíg i≤A.db és név≠A.t(i).név
    i:=i+1
  Ciklus vége
  Ha i>A.db akkor HIBAÜZENET('Nem tag')
    különben A.db:=A.db-1; A.t(i).név:=A.t(db).név
             A.t(i).város:= A.t(db).város
             A.t(i).kért:= A.t(db).kért
  Elágazás vége
Eljárás vége.

Tag adatainak módosítása(név,város,kért):
  i:=1
  Ciklus amíg i≤A.db és név≠A.t(i).név
    i:=i+1
  Ciklus vége
  Ha i>A.db akkor HIBAÜZENET('Nem tag')
    különben A.t(i).város:=város; A.t(i).kért:=kért
Eljárás vége.
```

Az adatok listázásához először egy rendező eljárásra van szükség:

```
Rendezés névsor szerint:
  Ciklus i=1-től N-1-ig
    Ciklus j=i+1-től N-ig
      Ha A.t(i).név>A.t(j).név akkor Csere(A.t(i),A.t(j))
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

A listázásnál figyelni kell arra, hogy egy lapra annyi sort írjunk ki, amennyi elfér; a lap végén várakozzunk billentyű lenyomásra; az utolsó lap végén akkor is várakozni kell, ha az nem telt be. Ha nincs egy adat sem, akkor nem szabad semmit kiírni, ezt a legcélszerűbb a menüből választásnál figyelni, s bele sem lépni a megfelelő almenüpontba, ha nincs mit listázni.

```
Listázás:
  Ciklus I=1-től A.db-ig
    Ha I mod SOROKSZÁMA=1 akkor Fejléc írás
    Kiírás(A.t(i))
    Ha I mod SOROKSZÁMA=0 akkor Várakozás
  Ciklus vége
  Ha A.db mod SOROKSZÁMA>0 akkor Várakozás
Eljárás vége.
```

A másik rendezési feladat az előbbihez hasonló, célszerű előbb újságfajta, majd pedig város szerint rendezni, s csak a listázásnál törődni a kétféle újsággal.

A második fő feladatcsoport nehezebb feladatokat tartalmaz, az alábbi menü szerint választhatunk:

Szállítási feladatok

1. Hálózat létrehozás
2. Újságonkénti lista egy induló helyről
3. Újságonkénti lista két induló helyről
0. Vissza az előző menühöz

Melyiket választod?

A városok hálózata egy gráf, amit legegyszerűbb a csúcsmátrixával megadni. A csúcsmátrix elemei azt tartalmazzák, hogy van-e közvetlen összeköttetés két város között, vagy pedig nincs. A mátrix kezdetben csupa hamis értéket tartalmaz.

```
G: Tömb('A'..'P','A'..'P',logikai):=(hamis,...,hamis)
```

A gráf megadása az élei megadásával történhet. A második részfeladathoz irányított gráf is elég, a harmadikhoz azonban irányítatlan kell. A harmadik részfeladat megoldása azonban megoldja a második részfeladatot is, így elég azzal foglalkozni.

```
Hálózat létrehozás(ÉLSZÁM):
  Ciklus i=1-től ÉLSZÁM-ig
    Be: X,Y; G(X,Y):=igaz; G(Y,X):=igaz
  Ciklus vége
Eljárás vége.
```

A második és a harmadik részfeladat megoldásában egy eljárást kell használni, amely egy adott újságra minden városhoz kiszámolja, hogy abból az adott városba hányat kell vinni, majd megadja, hogy a nyomdát tartalmazó városba inen milyen út vezet, s az út városaiban levő darabszámhoz hozzáadja a város újságszámát.

```
Számlálás(újság):
  DB:=0
  Ciklus i=1-től A.db-ig
    Ha A.t(i).kért=újság akkor DB(A.t(i).város):=DB(A.t(i).város)+1
  Ciklus vége
Eljárás vége.
```

A gráf szélességi bejárását a nyomtataás helyétől indítjuk:

```
SzélességiBejárás(hely):
  ÜresSor; Honnan():=0
  Sorba(hely); Honnan(hely):=hely
  Ciklus amíg a sor nem üres
    Sorból(pont)
    Ciklus i='A'-tól 'P'-ig
      Ha Honnan(i)=0 akkor Sorba(i); Honnan(i):=pont
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

Ennek hatására megtudtuk, hogy melyik városba közvetlenül honnan kell szállítani az újságot. Ha a sort is megtartjuk, akkor még azt is tudjuk, hogy itt a városokat mindig megelőzi az a város, ahonnan oda az újságot tovább kell vinni. Ebből következik, hogy a sor tömbjét felhasználhatjuk visszafelé bejárva az egyes városokba szállítandó mennyiség meghatározására (16 város van):

```
Mennyiségek:
  ÖSSZES:=DB
  Ciklus i=16-tól 1-ig -1-esével
    OSSZES(Honnan(SOR(i))):=OSSZES(Honnan(SOR(i)))+DB(i)
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A1. Először létrehozás; újra létrehozás; menürendszer	2 pont
A2. Mit tesz, ha egy személyt kétszer akarnak felvenni; ha sorrendben tárol: elejére, közepére, végére felvenni?; be kell-e írni az újság nevét, vagy "I/N"-re veszi fel? ; egyszerre egy vagy több beszúrás lehet? ; városjel ellenőrzés van-e? ; első ember felvétele	4 pont
A3. Mit tesz, ha nem létezőt töröl; mit tesz, ha mindenkit töröl; létezik-e egy vagy több törlés egyszerre? Mi azonosít egy embert: név vagy név-város? Lehet-e törlésből visszalépni "I/N" alapján?	4 pont
A4. Mit lehet módosítani? (város, újság, de nevet nem szabad) Módosításkor megadja-e, hogy mi volt a korábbi érték? Kell-e akkor is írni, ha az adatot nem akarja megváltoztatni? Egyszerre egy vagy több módosítás?	5 pont
A5. Rendezett-e a lista? Lapokra tördelés? Lehet-e a lapokat akármeddig nézni? Az utolsót is? Mi van, ha nincs még adat? Újságneveket ír, vagy kódolt értéket? Fejlécet ír-e laponként?	7 pont
A6. Lapozás, fejléc, városon belül a nevek névsorban vannak-e? A városok névsorban vannak-e?	8 pont
B1. Meg lehet-e adni a városok számát? Útmegadás milyen? Mi van, ha két város között két út van? Megadja-e az A–X útvonalat?	5 pont
B2. Újság-szám meghatározás jó-e? Eljut-e minden városba? Minden városhoz: mennyi újság ér oda, mennyit visznek tovább, mennyit raknak le?	15 pont
B3. Végtelen ciklusba kerül-e hurkok miatt? Két különböző városban levő nyomdát hogyan kezeli? Van-e utalás a legrövidebb útvonalra?	20 pont

Elérhető összpontszám: 70 pont

1987. Első forduló

Első-ötödik osztályosok

1. feladat: (12 pont)

A. N≥1 és egész — 2 pont

B. A(N) vektor tartalmazzon legalább egy negatív elemet! — 4 pont

C. F1 az A vektorbeli elemek közül az utolsó, F2 a nemnegatívak közül az utolsó sorszáma lesz. — 2 pont

(Ha az utolsót nem mondja, akkor 1 pont levonás.)

D. A B(N) vektor az A-beli elemek listába fűzésére szolgál a kívánt sorrendben. — 2 pont

E. A negatívak és a nemnegatívak sorrendje is megfordul – önmagukon belül - az eredetihez képest. — 2 pont

2. feladat: (5 pont)

Aritmetikai kifejezések kiértékelésének egyik nagyon egyszerű és gyors módszere az, amikor a kifejezéseket ún. lengyel formára hozzuk (egyik kitalálójuk, Lukasiewicz lengyel matematikus után elnevezve). Ebben a felírási formában a műveleti jeleket nem az operandusaik közé, hanem mögé kell írni. Ekkor ugyanis a kifejezés leírásához nincs szükség zárójelekre, a műveletek végrehajtási sorrendje pedig azonos a leírási sorrendjükkel.

1	3 1	4	2 4	5 2 4	4 5 2 4	20 2 4	8 20 2 4	12 2 4	6 12 2 4	2 2 4	4 4	0

Tévedésenként 1 pont levonás jár. 4-nél több hiba esetén 0 pont.

3. feladat: (12 pont)

Az itt leírt algoritmusok a LOGO programozási nyelv függvényszerű részében szinte szó szerint megvalósíthatók, csupán az alapszavakat kell LOGO-beli megfelelőjükkel helyettesíteni.

A. X magánhangzóinak száma — 4 pont

B. X minden betűje magánhangzó-e — 4 pont

C. X kódsorrendben (angol ABC sorrend) legnagyobb eleme — 4 pont

4. feladat: (9 pont)

A: négyzetgyök (N)-et kétszer számolja az osztók számába. — 2 pont

az 50-es sorban a feltétel az első és a második osztó megtalálása közötti összes számra teljesül (csak, ha az I osztó és a K=1, akkor lenne szabad az L=I utasítást végrehajtani). — 2 pont

a 80-as sorban, ha N prímszám volt, akkor nincs legkisebb valódi osztó, azaz a kiírást csak L>0 esetben szabad elvégezni. — 2 pont

B: minden olyan számra helyesen működik a program, amelynek a 2 és a 3 osztója, a szám négyzetgyöke nem osztó (nem egész), valamint maga a szám nem prímszám. — 3 pont

5. feladat: (12 pont)

A: f helyére -f — 3 pont

VAGY

minden LEFT helyére RIGHT és RIGHT helyére LEFT

B: x helyére -x, valamint f helyére -f — 3 pont

VAGY

minden LEFT helyére RIGHT és RIGHT helyére LEFT, valamint FORWARD helyére BACK és BACK helyére FORWARD

C: x helyére -x — 3 pont

VAGY

minden FORWARD helyére BACK és BACK helyére FORWARD

Ha az egyes feladatokra mindkét változatot megadja, akkor további 1-1 pont adható.

6. feladat: (5 pont)

A: B 2 hatvány és osztója A-nak — 2 pont

B: A és B előjele megegyezik — 3 pont

7. feladat: (10 pont)

Az IGEN címkén fejezi be a végrehajtást, ha B osztója A-nak,

egyébként pedig a NEM címkén. — 10 pont

A megoldás alapelve:

- ha A és B is páros, akkor az oszthatóság eldöntéséhez elég megvizsgálni A/2 és B/2 oszthatóságát;
- ha A páros, B páratlan, akkor elég A/2 és B vizsgálata;
- ha A páratlan és B páros, akkor B biztosan nem osztó;
- ha mindkettő páratlan, akkor B pontosan akkor osztja A-t, ha A–B-t is osztja.

Részmegoldásonként 1-1 pont adható a következő válaszokért:

- A=B esetén az IGEN címkén fejezi be,
- A=0 esetén az IGEN címkén fejezi be,
- B=1 esetén az IGEN címkén fejezi be,
- A<B esetén a NEM címkén fejezi be.

8. feladat: (8 pont)

Az első módszer intervallumfelezéssel keresi meg a függvény gyökhelyét, a második pedig húrmódszerrel.

A: $F(A)\leq 0$ és $F(B)\geq 0$ — 3 pont

B: $F(A)*F(B)\leq 0$ — 4 pont

$F(A)=F(B)=0$ egyszerre nem lehet — 1 pont

9. feladat: (7 pont)

Néhány lehetséges megoldás:

```
100 INPUT M,H0
110 DIM X(H0),Y(H0)
120 G=9.81
121 MG=M*G
122 E0=MG*H0
130 FOR H=H0 TO 0 STEP -1
140   X(H)=MG*H
150   Y(H)=E0-X(H)
180 NEXT H
```

```
100 INPUT M,H0
110 DIM X(H0),Y(H0)
120 G=9.81
121 MG=M*G
122 EH=MG*H0
123 EM=0
130 FOR H=H0 TO 0 STEP -1
140   X(H)=EH
150   Y(H)=EM
160   EH=EH-MG
170   EM=EM+MG
180 NEXT H
```

Pontszám= 6 mínusz a ciklus belsejében levő szorzások száma, illetve plusz 1 pont, ha a ciklusmagban az értékadások, összeadások, kivonások, tömbindexelések együttes száma ≤ 8.

10. feladat: (15 pont)

A: A beolvasott számok bináris alakban:

240 = 1111 00 00	↑		Az utolsó két bit változik, a többi változatkan.
242 = 1111 00 10	↑		
243 = 1111 00 11	jobbra	balra	
241 = 1111 00 01	↓	↓	

243 = 1111 00 11	↑		A következő két bit változik, a többi változatlan
251 = 1111 10 11	↑		
255 = 1111 11 11	előre	hátra	
247 = 1111 01 11	↓	↓	

A beolvasott byte felső 4 bitje állandóan 1. 1 pont

Az alsó két bit az X irányú, a következő két bit az Y irányú elmozdulástól függő kódszót tartalmaz. 1 pont

Az elmozdulás irányát a kódok sorrendje adja meg. 1 pont

B: (X,Y)= az egér pozíciójának koordinátái. 1 pont

(X1,Y1)= az egér pillanatnyi állapotának 2-2 bites kódja. 1 pont

(X0,Y0)= az egér előző állapotának 2-2 bites kódja. 1 pont

(X9,Y9)= az előző és a pillanatnyi állapotból számított decimális érték. 1 pont

C: A program követi az egér mozgását és pontsorozatként felrajzolja a bejárt utat a képernyőre. 1 pont

A képernyőn a (10,10) pontból indul és nem ellenőrzi, hogy a kirajzolandó pont a képernyőre esik-e. 1 pont

D: feltétel1: X9=2 OR X9=23 OR X9=31 OR X9=10 (jobbra)

feltétel2: X9=20 OR X9=32 OR X9=13 OR X9=1 (balra)

feltétel3: X9=2 OR X9=23 OR X9=31 OR X9=10 (előre)

feltétel4: X9=20 OR X9=32 OR X9=13 OR X9=1 (hátra)

Ha egy feltételt hibátlanul ad meg:

3 pont

Ha az ellenkező feltételt is hibátlanul adja meg: +2 pont

A további két feltétel megadásáért feltételenként: +1 pont

Elérhető összpontszám: 95 pont

1987. Második forduló

Első-ötödik osztályosok

Üzenetközvetítő:

Első lépésként meg kell gondolni, hogy a feladat megoldásához milyen adatszerkezet tartozik. Ez a tárolandó adatok mennyisége miatt lehet egészen egyszerű szerkezetű is.

Először definiáljuk a levéltípusokat. Ez a küldő-fogadó típusa szerint ötféle lehet, a törlési szabály szerint pedig négyféle. Mindkét szempont szerint egy-egy sorszám azonosíthatja a leveleket. Figyelni kell azonban majd a megoldás során arra, hogy a több embernek is szóló üzeneteket az első olvasás után senki (sem a rendszer, sem az olvasó) ne törölhesse. Ezen kívül szükség lesz még arra is, hogy a levelet pontosan ki küldte, ez pedig egy index lehet a dolgozói adatbázisban. Egyik típushoz még egy dátum megadására is szükség van, két másiknál pedig a címzett azonosítására is (osztály, illetve személy).

```
Levél=Rekord(típus,törlés,küldő: Egész, címzett: Egész,
             mikor: Dátum, tartalom: Szöveg)
```

A leveleket a feladat szerint nem kell háttértárra kitenni, így egy tömbben, ömlesztve tárolhatjuk őket.

Mivel a feladat szerint a háttértáron tárolást nem kellett megoldani, ezért a dolgozók adatbázisa egy, a memóriában tárolt konstans tömb lehet, aminek egy eleme az alábbi szerkezetű:

```
Dolgozó=Rekord(név: Szöveg,beosztás: egész, osztály: egész)
```

A beosztás és az osztály itt egy-egy egész szám, ami egy újabb táblázatra mutat, ahol szövegesen van le írva a beosztás (vezérigazgató, igazgató, titkár, osztályvezető, dolgozó,programozó), illetve az osztály neve. Aki nem osztályhoz tartozik, annál a beosztásához tartozó jelző (pl. gazdasági igazgató, szakszervezeti titkár stb.)

A program, mivel nincs háttértárolási funkciója, folyamatosan fut, kilépni nem lehet belőle. A felhasználókat azonosítania kell, ezért létre kell hozni benne bejelentkezés és kijelentkezés funkciókat. A feladat szerint nincs két azonos nevű ember, tehát a név egyértelműen azonosít mindenkit, valamint titkosítással sem kell foglalkozni, azaz nem kell a felhasználóknak jelszót adnunk.

A főprogram ezek szerint egy menürendszer kezeléséből áll:

ZAFFO

1. Bejelentkezés
2. Levél küldése
3. Levelek kiírása
4. Levél törlése
5. Kijelentkezés

Melyiket választod?

A bejelentkezés funkcióban, ha Kiss Attila jelentkezett be, akkor meg kell tőle kérdezni a napi dátumot (ezt csak ő írhatja be). A dátumot és a bejelentkező nevét és táblázatbeli sorszámát egy globális változóban kell tárolni. Bejelentkezéskor ellenőrizni kell, hogy valóban dolgozó jelentkezett-e be.

A dátum beírásakor kell kitörölni a leveleket tartalmazó tömbből az adott dátumig törlendő levelek közül azokat, amiknek lejárt a határideje.

A kijelentkezés funkció után csak újabb bejelentkezést szabad elfogadni.

A levél küldése részben egy újabb menű jelenik meg, ez azonban a bejelentkező beosztásától függően különböző kell legyen. A címzett megadása a kiválasztott levéltípustól függ. Minden levélhez meg lehet adni törlési szabályokat (egyikhez dátum is szükséges), de itt figyelni kell arra, hogy melyik típushoz milyen törlési szabály alkalmazható.

A levelek kiírása részben a bejelentkezett beosztásától függően kell nézni, hogy ő milyen típusú leveleket olvashat. Kiss Attila minden levelet elolvashat, a vezérigazgató csak az 1. és a személyesen neki szóló 5. típusút, az igazgatókés a titkárok ezen kívül még olvashatják a 2. típusút, az osztályvezetők pedig a 3. típusút. A beosztott dolgozók az 1. típusú leveleket olvashatják, valamint az ő osztályukra érkező 4. típusúakat és a személyesen nekik küldött 5. típusúakat. Ebben a részben kell törölni az egy olvasás után törlendő leveleket

A levél törlése funkcióban mindenkinek csak azokat a leveleket kellene megmutatni, amiket töröl-het, s rákérdezni, hogy töröljük-e.

Értékelési szempontok:

I. Vezérlő rész:		11 pont
Bejelentkezés		3 pont
- nem létező vezető beengedése:	-1	
- nem létező osztály beengedése:	-1	
- nem fogad minden személyt:	-1	
Menü vagy kérdések sorozata		5 pont
- minden funkció használható:	3	
- ugyanaz-e Kiss Attilának, NEM:	2	
- beenged illegális funkcióba:	-1	
- ha lehet bejelentkezés nélkül bármit tenni:	-1	
Dátum beírás		3 pont
- létezik:	1	

- ha a beírása előtt nem lehet semmit csinálni, akkor:	2	
II. Levél küldése		17 pont
Csak olyannak küldhet, akinek szabad	4 pont	
Ellenőriz törlési kapcsolatokat	2 pont	
(követelmény: körlevelet ne törölhesse a kapó és ne törlődjön 1 olvasás után, más kapcsolatot nem kell tiltani, de nem baj, ha esetleg tiltják)		
Létezik mindegyik fajta levél	5 pont	
Létezik mindegyik törlési mód	4 pont	
Küldhet egymás után több levelet	1 pont	
Ki lehet lépni küldés nélkül	1 pont	
III. Összes levél kiírása		4 pont
Levelenkénti lapokra tagolás	2 pont	
- ha nincs levél, akkor rosszul működik:	-1	
Látni kell: címzett, küldő, tartalom, törlési fajta	2 pont	
IV. A dolgozónak szóló levelek kiírása		12 pont
Levelenkénti lapokra tagolás	2 pont	
Megkapja a személyesen neki szólókat	2 pont	
Megkapja az osztályának szólókat	2 pont	
Megkapja a vele azonos beosztásúaknak szólókat	2 pont	
Megkapja a mindenkinek szólókat	2 pont	
Megtudja, hogy kitől kapta a levelet	2 pont	
- ha nem konkrétan:	-1	
Ha másnak szóló levelet is kap:	-4	
V. Levél törlése		16 pont
Egy olvasás után törlődik-e	2 pont	
Adott dátumkor törlődik-e	3 pont	
- dátumegyenlőség:	-1	
A küldő visszavonhatja	3 pont	
- ha más is visszavonhatja:	-1	
- ha a sikertelen törlésről nincs visszajelzés:	-1	
- ha más fajtát is lehet:	-1	
A kapó törölheti	2 pont	
- ha a sikertelen tőrlésről nincs visszajelzés:	-1	
Kiss Attila törölhet-e bármit	2 pont	
- csak amit kap:	-1	
A törlő megtudhatja-e, hogy miből vonhat vissza	2 pont	
Ki lehet lépni törlés nélkül	1 pont	
A törlésről van visszajelzés	1 pont	
VI. Adatszerkezet		14 pont

Levél: CIMZETT (kódolva is), KULDO (kódolva is), TARTALOM,		
TORLESI MOD, DATUM, LEVEL FAJTA (ha a címzettből nem derül ki)		6 pont
Dolgozó: NEV, KULDESI JOG, BEOSZTAS	3 pont	
Osztály: NEV, KOD	1 pont	
Levelek tárolása (lista, mutató, gyors művelet)	2 pont	
Menü, mint adat	2 pont	
VII. Programszerkezet, stílus		10 pont
Eljárásokra tagolás, modularitás	3 pont	
Közös funkciók kiemelése 1 eljárásba	2 pont	
Megjegyzések, a programszöveg tagolása	2 pont	
Egyszerűség, világosság	2 pont	
Értelmes változónevek	1 pont	
VIII. Felhasználói tulajdonságok		10 pont
Kérdések szövegezése	1 pont	
Hibajelzések szövege, láthatósága	1 pont	
Csak olyan funkciót lát a programból, amit használhat is	2 pont	
Törlést csak törölhetőre kérdez	1 pont	
Felesleges dolgot nem kérdez (pl. osztályt)	1 pont	
Képernyőkezelés esztétikussága	1 pont	
A lehetőségekből választás egyszerűsége (menü)	3 pont	
főprogram, levélfajta, törlési mód		
- ha teljes szavakat kell beírni:	-1	
- következetes beolvasás hiánya:	-1	
- nem szólít fel billentyűlenyomásra:	-1	
IX. Programterv		5 pont
Adatleírás		5 pont
- tömbök:	1	
- a kódokat magyarázza	3	
- fontos változók	1	

Elérhető összpontszám: 99 pont

1988. Első forduló

Első-ötödik osztályosok

1. feladat: (6 pont)

A. A b**2–4*a*c diszkriminánsban i növekedésével b**2 értéke sokkal gyorsabban nő 4*a*c értékénél, és a számábrázolás pontatlansága miatt a b**2 mellett a 4*a*c előbb-utóbb elhanyagolható értékűvé válik. 1 pont

B. Ekkor x2 = –b+SQRT(b**2) = 0.0 lesz az eredmény a várt 1.0 helyett! 1 pont

C. A 24 bites mantisszával max. 2**23 = 8 388 608-at, azaz 10**6 nagyságrendű számokat lehet pontosan ábrázolni. 1 pont

D. A kivonáshoz a kitevőknek egyenlőknek kell lenniük, ezért i>6 esetén már fellép az említett jelenség. 1 pont

E. Az adott számábrázolás mellett max. 2**(2**7) = 2**128 ≈ 10**38 nagyságrendű számok ábrázolhatók. 1 pont

F. Ezért túlcsordulás b>10**19, vagyis i>19 esetén lép fel. 1 pont

2. feladat: (15 pont)

A. [1] meghatározza azt a hossz-t, amely mellett az [1,hossz–10] intervallumba eső számok összege még kisebb, az [1,hossz] intervallumba eső számok összege pedig már nagyobb m-nél. 1 pont

[1] eredményéből kiindulva [2] azt a hossz-t határozza meg, amely mellett az [1,hossz] intervallumba eső számok összege még éppen kisebb, az [1,hossz+1] intervallumba eső számok összege pedig már éppen nagyobb m-nél. 1 pont

[2] eredményéből kiindulva [3] azt vizsgálja, hogy [az első hossz szám összege,m] intervallum felbontható-e hossz számú, egyenlő hosszúságú részintervallumra. Ha igen, megvan a megoldás, hiszen csak e részintervallumok hosszával kell megnövelni az [1,hossz] intervallum minden elemét. Egyébként hossz-t 1-gyel csökkentjük, és folytatjuk. 2 pont

B. 4095 esetén egyszer sem, ugyanis 4095 éppen az első 90 természetes szám összege, és ezért [3] ismétlési feltétele már kezdetben hamis. 1 pont

4096 esetén 89-szer, ugyanis 4096 2 hatványa, amelynél a keresett intervallum hossza 1 (ld. a 2.4. választ). Mivel a [2] ismétlés végén hossz=90, a hossz=1 érték megtalálásáig éppen 89-szer kell megismételni a törzset. 1 pont

C. Itt három ismétlés van, amelyekben – elvileg – elakadhatna a program. De [1] biztosan befejeződik, mert hossz-t 10-esével növelve az első hossz szám összege előbb-utóbb biztosan nagyobb lesz a véges m-nél (vagy túlcsordulás lép fel). 1 pont

De [2] is befejeződik, mert hossz-t 1-esével csökkentve legfeljebb 9 lépésben m-nél megint kisebb lesz az első hossz szám összege. 1 pont

Végül [3] is biztosan véget ér, hiszen a legrosszabb esetben is hossz=1-gyel minden szám maradék nélkül osztható. 1 pont

D. Ha m felbontható az [i,j] intervallumba eső számok összegeként, akkor m = hossz*(i+j)/2 = hossz*(2*i+hossz-1)/2, hiszen j = i+ hossz–1. A vizsgálandó esetben m = 2**k alakú (ahol ** a hatványozás jele), vagyis tetszőleges k-ra teljesülnie kell a következő egyenlőségnek: 2**(k+1)= hossz*(2*i+hossz–1). De ha hossz páros, akkor 2*i+hossz –1 páratlan, és fordítva, tehát mindkét tényező egyszerre csak hossz=1 mellett lehet 2 hatványa. 3 pont

E. A program gyorsabbá tehető, ha a szorzások és az osztások helyett összeadást és kivonást végzünk. Erre van lehetőség, ugyanis az első hossz szám összegénél az első hossz+1 szám összege éppen (hossz +1)-gyel nagyobb. Ha bevezetünk egy változót az összeg tárolására, akkor ennek értékét az [1], [2] és [3] ismétlésekben minden lépésben meghatározott értékkel kell növelni, ill. csökkenteni. 2 pont

Ha az adott programozási nyelven ábrázolható legnagyobb egész számot ismerjük, akkor tudunk felső korlátot adni a hossz-ra, és így az [1] és [2] ismétlés helyett a gyorsabb bináris (logaritmikus) keresést használhatjuk. 1 pont

3. feladat: (8 pont)

3.1. A fekete szemek száma 1-esével nő, ha két fehéret húzunk, egyébként, 1-esével csökken. 1 pont

A fehér szemek száma 2-esével csökken, ha két fehéret húzunk, egyébként nem változik. 1 pont

3.2. A fehér szemek számának párossága (ez ún. invariáns tulajdonság)! Ha kezdetben páratlan volt a fehér szemek száma, akkor végig páratlan is marad, ha páros volt, akkor páros is marad. 3 pont

3.3. Minden lépésben 1-gyel csökken a kávészemek száma a dobozban. 1 pont

A fehérek kezdeti számától függ, hogy milyen színű lesz az utolsó: ha páratlan, akkor fehér; ha páros, akkor fekete. 2 pont

4. feladat: (9 pont)

4.1. A példákból elölről hátrafelé kiolvashatók a következő kódok: 0=0000, 1=1000, 2=0100, 3=1100, 5=1010, 6=1101, 7=1111, 9=1001. 2 pont

Visszafelé olvasva néhány kód más:
1=0001, 2=0010, 3=0011, 5=0101, 6=1011. 1 pont

Hiányzik a 4-es és a 8-as számjegy, amelyekhez a még hiányzó 0110, 0111 és 1110 kódszavak rendelhetők. 1 pont

A hozzárendelés a 4.2 választól függ.

4.2. A számjegyek kódjából ki lehet deríteni az olvasási irányt, és így helyre lehet állítani az eredeti sorrendet. 1 pont

Ehhez az szükséges, hogy a négy közül legalább egy számjegynek ne legyen szimmetrikus a kódja: ezért írjuk elő, hogy a legnagyobb helyiértékű jegy csak 1,2,3,4,5 és 6 lehet. 2 pont

Ebből következik, hogy a 4 számjegynek két kódja van: 0111 és 1110, a 8-nak pedig egy: 0110. 2 pont

5. feladat: (6 pont)

A. A beolvasott decimális számot normalizálja. 1 pont

B. A számot b darab számjeggyel írja ki. 1 pont

C. A többi számjegyet kerekíti. 1 pont

D. Ha a$ 0-val kezdődik akkor hibásan normalizál. 1 pont

E. Ha az utolsónak kiírandó számjegy 9-es lenne, és ezt a kerekítés miatt meg kell növelni, akkor nem számjegyet ír ki, hanem a 9-es után következő kódú karaktert. 2 pont

6. feladat: (7 pont)

6.1. A és B másképpen működik, ha a számsorozat 1-esekkel kezdődik vagy végződik. 1 pont

Ekkor csak B veszi figyelembe az első sorozat hosszát. 2 pont

6.2. Mindkettő hibás, ha a számsorozat 1-esekkel végződik. 1 pont

Ekkor B kiszámolja, de nem írja ki a megoldást. 2 pont

Ekkor A-ban a belső ciklus feltételében indexhatár-túllépés lesz. 1 pont

7. feladat: (7 pont)

A. A 90-es sorbeli RAJZOLÁS a koordináták által kijelölt négyszöget rajzol a képernyőre. 2 pont

B. A 110-es sorbeli RAJZOLÁS hasonló, de ALFA szöggel elforgatott négyszöget rajzol a képernyőre. 4 pont

C. Az utóbbi a (0,0) pont mint középpont körül van elforgatva. 1 pont

8. feladat: (9 pont)

A. A program befejeződik, ha olyan pontot talál, amely minden szomszédjánál magasabb, de mégsem a legmagasabb. 2 pont

B. Ha fennsíkra ér, végtelen ciklusba kerül. (A fennsík egynél több, azonos magasságú pontból áll, amelyeket alacsonyabb pontok vesznek körül.) 3 pont

C. Síkon haladva mindig "délkeleti" irányba törekszik. 1 pont

D. Akkor is végtelen ciklusba kerül, ha egy síkon haladva alacsonyabb pontokba ütközik, 2 pont

E. vagy eléri a térkép szélét. 1 pont

9. feladat: (8 pont)

9.1. Törtvonalban vagy egyenes vonalban halad a fényforrás felé. 2 pont

De ha a fényforrás az autó mögött van, távolodik tőle. 1 pont

9.2. Az e a fényerősség-különbségre vonatkozó érzékenységi küszöb. 1 pont

9.3. Ha e olyan kicsi, hogy az autó a lehető legkisebb elfordulás után is azonnal az ellenkező irányba akar fordulni (azaz nem tud elég kicsit fordulni). 2 pont

Ha e olyan nagy, az autó állandóan elkerüli a fényforrást. 1 pont

Ha a fényforrás olyan gyorsan mozog, hogy a játékautó nem tud utánafordulni. 1 pont

Elérhető összpontszám: 75 pont

1988. Második forduló

Első-ötödik osztályosok

Mini-LOGO:

A feladat szövege nem szól arról, hogy a program egyetlen sorból áll-e vagy sem. Emiatt is célszerű egy szóolvasó eljárást írni, ami a bemenetről beolvas egy szót. A szót szóköz karakter vagy pedig a sorvég karakter határolja. A főprogram ezek alapján egy ciklus lesz, ami beolvassa a programot, s amikor a V utasításhoz ért, akkor a tárolt programot végrehajtja. Az egységes tárolás úgy valósítható meg, ha egy táblázatot veszünk fel, melynek elemei utasításkódból és annak paraméterőből álló párok. Ebből a szerkezetből kilóg az eljárás és a ciklus, ahol az egyik paraméter nagyon hosszú is lehet. Ezt úgy oldhatjuk meg, hogy ilyenkor egy másik táblázatra hivatkozunk, ahol a leírás szerepel.

```
Programbeolvasás:
  Szóolvasás(SZÓ); Tábla.DB:=0;
  Ciklus amíg SZÓ≠'V'
    Tábla.DB:=Tábla.DB+1
    Elágazás
      SZÓ∈{'M','F'} esetén Tábla.elem(Tábla.db).kód:=SZÓ
                          Szóolvasás(SZÓ)
                          Tábla.elem(Tábla.db).kód:=SZÓ
      SZÓ∈{'N','R','T','H'} esetén Tábla.elem(Tábla.db).kód:=SZÓ
      SZÓ='I' esetén Szóolvasás(SZÓ)
                     Tábla.elem(Tábla.db).kód:=SZÓ
                     Ciklusfeldolgozás
      SZÓ='E' esetén Szóolvasás(SZÓ)
                     Tábla.elem(Tábla.db).kód:=SZÓ
                     Eljárásfeldolgozás(SZÓ)
      SZÓ='D' esetén Szóolvasás(SZÓ); Eljárástörlés(SZÓ)
    Elágazás vége
  Ciklus vége
  Tábla.elem(Tábla.db).kód:=SZÓ
Eljárás vége.
```

A ciklusmagok és az eljárások egy külön táblába kerülnek, amelyek pontosan olyan szerkezetűek, mint a főprogram, ezért majd végrehajtáskor a programvégrehajtás eljárás rekurzívan meghívhatja magát, ha ilyen programstruktúrát kell végrehajtania.

A ciklusokat a táblázatukban előfordulási sorszámukkal azonosítjuk, az eljárásokat pedig a nevükkel.

A ciklusok, illetve az eljárások feldolgozása hasonló a fő feldolgozáshoz (bár eljárás belsejében nem lehet másik eljárás, illetve annak sincs sok értelme, hogy ciklus belsejében definiáljunk eljárást). Az az alapvető különbség, hogy itt nem a V utasításra kell befejezni a feldolgozást, hanem a csukó zárójelre, a programrészek végére azonban a későbbi egységes feldolgozás miatt be kell írni a V utasítást.

Különbség lehet még az eljárások és a ciklusok között, hogy a ciklustáblát programonként újra kezdjük, az eljárástáblát azonban nem szabad, azaz azokat a következő program felhasználhatja (ezért van külön eljárás törlés funkció).

```
Programvégrehajtás(Tábla):
  i:=1; X:=0; Y:=0; IR:=0; R:=igaz
  Képernyőtörlés; Kurzormozgatás(XX,YY)
  Ciklus amíg Tábla.elem(i).kód≠'V'
    Elágazás
      Tábla.elem(i).kód='M' esetén
                  XX:=XX+Tábla.elem(i).érték*cos(IR)
                  YY:=YY+Tábla.elem(i).érték*sin(IR)
                  Ha IR akkor Szakaszrajzolás(X,Y,XX,YY)
                     különben Kurzormozgatás(XX,YY)
                  X:=XX; Y:=YY
      Tábla.elem(i).kód ='F' esetén IR:=IR+Tábla.elem(i).érték
      Tábla.elem(i).kód ='N' esetén R:=hamis
      Tábla.elem(i).kód ='R' esetén R:=igaz
      Tábla.elem(i).kód ='T' esetén Képernyőtörlés
      Tábla.elem(i).kód ='H' esetén X:=0; Y:=0; IR:=0; R:=igaz
                                    Kurzormozgatás(XX,YY)
      Tábla.elem(i).kód(1) számjegy esetén
                  Ciklus j=1-től egész(Tábla.elem(i).kód)-ig
                    Programvégrehajtás(c(Tábla.elem(i).érték))
                  Ciklus vége
      egyéb esetén Eljáráskeresés(Tábla.elem(i).kód,Altábla)
                  Programvégrehajtás(Altábla)
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

Alapfunkciók: 28 pont

M	-	5
F	-	2
N,R	-	1
T	-	1
H	-	2
I	-	10 (I-ben eljáráshívás: 3, a többiek: 7 pont)
V	-	2 (alapállapotba állítás)
szakasz	-	5 ("nem folytonos" esetben csak 2 pont)

Eljárásfunkció: 19 pont

E	-	7 (időleges eljárásdefiníció: csak 4 pont)
D	-	2
hívás	-	10 (eljárásban I: 3, a többiek: 7 pont)

Vezérlőrész: 7 pont

beolvasás	-	2 (nem teljes programbeolvasás esetén: 0)
végrehajtás vezérlés	-	5

Szerkezet: 17 pont

adatszerkezet	-	5
strukturáltság	-	5 (eljárások, algoritmikus szerkezetek használata)
tagoltság	-	3 (eljárások, algoritmikus szerkezetek formája)
magyarázatok	-	2
fejleszthetőség	-	2

Bővítések: 13 pont

hibakezelés	8 pont
utasításnév	- 1
paraméter	- 1
ismétlés	- 1 (zárójelhiba)
eljárásdefiníció	- 2 (zárójelhiba, már létező újradefiniálása)
fenntartott név	- 1
kimegy a képről	- 1
hibás törlés	- 1 (Dobd el utasításban)
értelmes hibajelzés	3 pont
teljes utasításnév	2 pont

Elérhető összpontszám: 84 pont

1989. Első forduló

Első-ötödik osztályosok

1. feladat: (3 pont)

Az eredmény 64, mivel a 20-as és az 50-es sort egyszer, a 60-ast kétszer, a 70-est négyszer, ... hajtja végre a program.

A. Jó eredmény: 1 pont

B. Magyarázat : 2 pont

2. feladat: (8 pont)

1030 DO WHILE J>=1 AND IS$<T$(J) 1 pont

1040 T$(J+1)=T$(J): J=J–1 1 pont

1060 T$(J+1)=IS$ 1 pont

1.——> az I–1. elemig a T$() vektor rendezett, 1 pont

és IS$ a beszúrandó 1 pont

2.——> a J+1..I intervallumban a T$() vektor rendezett, 1 pont

IS$ helye az 1..J indexű elemek valamelyike 1 pont

3.——> az 1..I intervallumban rendezve van a T$() vektor 1 pont

3. feladat: (6 pont)

A. az x tizedestörtet kettedestörtté (azaz kettes számrendszerbeli törtté alakítja) 1 pont

B. befejeződik, ha már előállítottunk 30 számjegyet, vagy ha végtelen szakaszos kettedestört első ismétlődő szakaszának vége van, 2 pont

C. vagy ha nulla maradékot kapunk valamelyik lépésben, azaz az utolsó értékes jegynél. 1 pont

D. x=0.3-ra az eredmény 0.01001 1 pont

4. feladat: (6 pont)

Kulcstranszformációs függvénnyel az értékből címet számít, s a kiszámított címre, vagy az azt (ciklikusan) követő első szabad helyre teszi az elemet.

A. A K1$ kulcs szerinti helyre igyekszik tenni a (K1$,K2$) sort a táblázatban. 2 pont

B. Ha az így kiszámított hely szabad, akkor oda teszi. 1 pont

C. Ha nem szabad, akkor megkeresi a mögötte levő legközelebbi üres helyet, és odateszi. 2 pont

D. Ha nincs üres hely, akkor kiírja, hogy "baj!!!". 1 pont

5. feladat: (6 pont)

A D() vektor tartalmazza az egyes érmék darabszámát.

```
Pénzvisszaadás(ÖSSZEG):
  Ciklus I=1-től 5-ig
    Ha ÖSSZEG>=FT(I) akkor DB:=INT(ÖSSZEG/FT(I))
                           Ha D(I)<DB akkor DB:=D(I)
                           D(I):=D(I)-DB
                           Ki: DB;" db. ";FT(I);" Ft-os"
                           ÖSSZEG:=ÖSSZEG-DB*FT(I)
    Elágazás vége
  Ciklus vége
  Ha ÖSSZEG>0 akkor Ki: "Nem tudok visszaadni"
Eljárás vége.
```

A. DB korlátozása 2 pont

B. D(I) csökkentése 2 pont

C. Visszaadás lehetetlenségének kezelése 2 pont

6. feladat: (7 pont)

A. A program egy egyszerű hóembert rajzol, három kört egymás tetején. (egymás melletti körök – fekvő hóember – csak 2 pontot ér) 3 pont

B. A felső két kör sugara mindig kétharmada az alatta levő kör sugarának. 1 pont

C. A legalsó kör sugara 30 egység. 1 pont

D. A GVONAL eljárás egy félkörívet közelít szabályos sokszöggel. (Kihasználja, hogy a félkörív hossza sugár*pi/180 egység.) 1 pont

E. Az :Y paraméter a körív sugarát adja meg. 1 pont

7. feladat: (8 pont)

A. A G1 0-ba ragadásának felismeréséhez A=0, B=1, C=1 szükséges, így a 100-as sor első száma : 6 1 pont

B. Ha a kapu jól működik, akkor Y=0, egyébként Y=1, tehát a második szám: 0 1 pont

A másik két sor hasonlóan:

C. 110-es sor: 1,0 vagy 3,0 vagy 5,0 a jó számpár 2 pont

D. 120-as sor: 7,1 a jó számpár 2 pont

Indoklás:

E. A tesztelt kapu kimenetének hibája akkor deríthető ki, ha a kapu vezérlése olyan, hogy a kimeneten 1-nek kellene lennie 1 pont

F. és ez az 1-es ki tud jutni az áramkör Y kimenetére 1 pont

(Például a G1 kimenete akkor lesz 1, ha A=0 és B,C mindegyike 1, mivel ekkor G1 második bemenete 0 lesz. A jel akkor jut ki G3 kimenetére – ami az áramkár kimenete – ha G3 másik bemenete 0, ez itt teljesül. A hiba tehát kideríthető a fenti vezérléssel, mert G1 hibája esetén 0 helyett 1 jelenik meg az áramkör kimenetén. Itt természetesen kihasználtuk azt, hogy az áramkörben egyszerre csak egy hibás kapu lehet.)

8. feladat: (8 pont, problémánként 1-1 pont)

Problémánként 1 pont (csak algoritmikus problémákra adható), néhányat felsorolunk:

- a ki-, illetve beszállást 10 másodpercen belül be kell fejezni,

- a lift, miközben egy szinten áthalad, egyetlen hívást és egyetlen parancsot képes fogadni (hiába nyomnak meg egyszerre több gombot),
- nem veszi észre, hogy az adott szintre már hívták (ugyanazt többször is megnyomhatják),
- nem veszi észre, hogy a többszörösen hívott szintekre nem kell még egyszer elmennie,
- nem áll meg közbülső szinteken,
- 32-nél több hívás esetén indexhatár-túllépéssel leáll,
- ha az utasok utolsó, liftben maradt csoportja nem száll ki az általuk megjelölt szinten, akkor az ajtó becsukódása után mindaddig a liftben kénytelenek maradni, míg kívülről valaki nem hívja a liftet,
- beszállás után, amennyiben a liftnek nincs több tárolt úticélja, akkor az utasoknak a következő lifthívást meg kell várniuk, mert a lift nem figyeli a parancs gombot.

Elérhető összpontszám: 52 pont

1989. Második forduló

Első-ötödik osztályosok

Digitális áramkör szimulációja:

Az áramkört egy speciális gráfként foghatjuk fel., mivel azonban nincs szükség sehol sem a gráf csomópontjainak valamilyen stratégia szerinti bejárására, ezért igazából a gráfot nem kell felépítenünk.

Jó megoldás lehet, ha a bemeneteknek és a kimeneteknek is fenntartunk egy-egy tömböt, s egy harmadikat a kapuknak. Ebben az esetben a feladat lépésenként az összes kapun előállítjuk a bemenetből a kimenetet, majd a kimeneti tömböt átmásoljuk a bemeneti tömbbe.

Az áramkör megadásánál szükségünk lesz a csomópontok számára, valamint a közöttük levő kapuk leírására (milyen kapu, mely csomópontok a bemenetei és melyek a kimenetei). Ezen kívül be kell olvasni az áramkör bemeneteit, illetve kimeneteit.

A szükséges adatok és a fő eljárás tehát:

```
BE,KI: Tömb(1..maxcsp,Egész)

KAPU: Tömb(1..maxkapu,Rekord(név:Szöveg,csp:Tömb(1..3,Egész)))
KAPUDB: Egész

Szimuláció:
  Ciklus amíg nem STABIL
    Ciklus i=1-től KapuDB-ig
      Elágazás
        KAPU(i).típus=NOT esetén NOTKAPU(i)
        KAPU(i).típus=OR esetén ORKAPU(i)
        KAPU(i).típus=AND esetén ANDKAPU(i)
        KAPU(i).típus=EXOR esetén EXORKAPU(i)
        KAPU(i).típus=NOR esetén NORKAPU(i)
        KAPU(i).típus=NAND esetén NANDKAPU(i)
      Elágazás vége
    Ciklus vége
    STABIL:=(BE=KI); BE:=KI
  Ciklus vége
Eljárás vége.
```

Az egyes kapuk működését hasonlóan lehet megoldani, közülük például az AND-kapu így néz ki:

```
ANDKAPU(i):
  KI(KAPU(i).csp(3)):=BE(KAPU(i).csp(1))*BE(KAPU(i).csp(2))
Eljárás vége.
```

A szimuláció elkezdéséhez a bemeneteket be kell másolni a BE tömb megfelelő elemeibe. Az áramkör bemeneteit a következőképpen írhatjuk le:

```
ABE: Tömb(1..maxcsp,Rekord(csp,érték: Egész))
ABEDB: Egész

Bemenetfeltöltés:
  Ciklus i=1-től ABEDB-ig
    BE(ABE(i).csp):=ABE(i).érték
  Ciklus vége
Eljárás vége.
```

A szimuláció végén a kimenet feltöltését a fentihez hasonlóan lehet megoldani.

Az egyik részfeladatban az összes lehetséges bemeneti jelkombinációra kell végrehajtani a szimulációt. Ez azt jelenti, hogy ABEDB helyen kell előfordulni az összes 0-1-ekből álló jelkombinációnak. Ebből tehár összesen 2^{ABEDB} darab van. A legegyszerűbben úgy állíthatók elő, hogy egy számlálóval számolunk 0-tól 2^{ABEDB}-1-ig, s a számlákó értékét kettes számrendszerbeli számmá konvertáljuk. Az így kapott számjegyek lesznek a bemenetek.

Az utolsó részfeladatban ellenőrzéseket kell végezni. Ezek a következőképpen tehetők meg:

- Kapu kimenetét sehova sem kötötték: egy kimeneti csomópont nem szerepel sem bemenetként, sem áramkör kimenetként.
- Kapu bemenetét sehova sem kötötték: egy bemeneti csomópont nem szerepel sem kimenetként, sem áramkör bemenetként.
- Két kapu kimenetét összekötötték: ugyanaz a kimeneti csomópont legalább 2 kapunál jelenik meg.
- Az áramkör kimenete nem valamely kapu kimenete.
- Az áramkör bemenete nem valamely kapu bemenete.
- Van olyan kapu a hálózatban, amelyet a bemenetről nem lehet elérni: ez az egyetlen részfeladat, amelynél gráfbejáró algoritmusra van szükség. Meghatározásuk a következőképpen történhet: Vegyük a bemeneti csomópontokat, s az elért kapu közé vegyük fel azokat, amelyeknek ezek bemenetei. Az így elért kapuk kimeneteire alkalmazzuk ugyanezt a módszert, mindadig, amíg új kapuhoz jutunk. Az alábbi eljárás után a VOLT tömbben igaz értékek szerepelnek azon kapuknál,amelyek elérhetők a bemenetről. Ha maradt hamis érték, akkor a hálózat hibás.

```
Elérhetőség:
  VOLT:=hamis
  Ciklus i=1-től ABEDB-ig
    Szomszédok(ABE(i).CSP)
  Ciklus vége
  VÁLTOZÁS:=hamis
  Ciklus
    Ciklus i=1-től KAPUDB-ig
      Ha VOLT(i) akkor Szomszédok(KAPU(i).csp(3))
    Ciklus vége
  amíg VÁLTOZÁS
  Ciklus vége
Eljárás vége.
```

```
Szomszédok(i):
  Ciklus j=1-től KAPUDB-ig
    Ha KAPU(j).csp(1)=i vagy KAPU(j).csp(2)=i
              akkor VOLT(j):=igaz; VÁLTOZÁS:=igaz
  Ciklus vége
Eljárás vége.
```

- Ha az áramkör olyan, hogy stabil állapot nem biztos, hogy kialakul, akkor a szimulációs ciklust lépésszám korláthoz is köthetjük, s ha a koráton belül nem alakul ki stabil állapot, akkor hibajelzést adunk.

Értékelési szempontok:

1. Adatszerkezet		12 pont
1.1. Kapufajták táblázata	3 pont	
1.2. Kapuk táblázata	2 pont	
1.3. Kapubemeneti jelek táblázata	1 pont	
1.4. Kapukimeneti jelek táblázata	1 pont	
1.5. Csomópontok táblázata	2 pont	
1.6. Eseménylista (a változások követése)	3 pont	
2. Hálózat beolvasása		8 pont
2.1. A kapuneveket és csak azokat lehet beolvasni	4 pont	
2.2. Megfelelő be- és kimenetszámmal lehet a kapukat beolvasni	4 pont	
3. Működés		30 pont
3.1. Az egyes kapuk helyesen működnek	12 pont	
3.2. Jelek terjednek a következő kapu bemenetére	4 pont	
3.3. A kimenetre kijutnak a jelek	2 pont	
3.4. Leáll, ha már nincs a kapuk kimenetein változás	6 pont	
3.5. Visszacsatolásos hálózatot is tud kezelni	4 pont	
3.6. Bemeneti kombinációk beolvasása	2 pont	
4. Az összes bemenet előállítása		10 pont
4.1. Jól előállítja	8 pont	
4.2. A képernyőről nem futnak ki a szimuláció eredményei – lapozás	2 pont	
5. Ellenőrzések		20 pont
5.1. Kapu kimenetét sehova nem kötötték, s nem is áramkörkimenet	2 pont	
5.2. Kapu bemenetét sehova nem kötötték, s nem is áramkörbemenet	2 pont	
5.3. Két vagy több kapu kimenetét összekötötték	2 pont	
5.4. Van olyan kapu a hálózatban, amit a bemenetről nem lehet elérni	8 pont	
5.5. Az áramkör kimenete nem valamelyik kapu kimenete	1 pont	
5.6. Az áramkör bemenetét nem kötötték egyik kapu bemenetére sem	1 pont	
5.7. A korlátnak állított idő letelése után sem alakul ki stabil kimenet	2 pont	
6. Programminőség		20 pont
6.1. Képernyőtörlés, címkiírás a megfelelő helyeken	2 pont	
6.2. Barátságos beolvasás, kiírás	4 pont	

6.3. Magyarázatok a programszövegben	2 pont
6.4. Bekezdéses leírás	2 pont
6.5. Sorok közötti tagolás (pl. üres sorokkal)	2 pont
6.6. Világos szerkezetű eljárásokra bontás	4 pont
6.7. Rövid programsorok	1 pont
6.8. Értelmes változónevek	1 pont
6.9. Menürendszer	2 pont

Elérhető összpontszám: 100 pont

1990. Első forduló

Első-második osztályosok

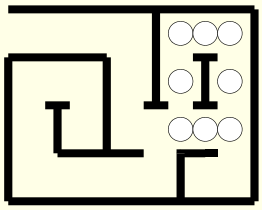
1. feladat: (4 pont)

A. HIBA1 – egy csukózárójel előbb volt, mint a neki megfelelő nyitózárójel; több a csukózárójel, mint a nyitó — 2 pont

B. HIBA2 – nincs minden nyitózárójelnek csukó párja — 2 pont

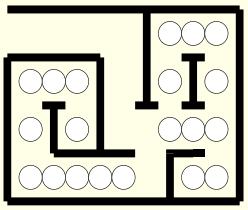
2. feladat: (15 pont)

A. Megkezdett két O-önként +1, ha jó, -1, ha rossz a válasz, — összesen max. 4 pont



B. Biztosan kijutunk, ha nincs benne olyan kör, amelyet az óramutató járásával ellentétes irányban kellene bejárnunk. — 4 pont

C. Megkezdett öt O-önként +1, ha jó, két O-önként -1, ha rossz a válasz, — összesen max. 4 pont.



D. Biztosan kijutunk, ha a kijárat fent vagy balra van, és ha a vizsgált pontból fölfelé, s ha nem megy, balra törekedve csökkenő távolságú út vezet hozzá. — 3 pont

3. feladat: (10 pont)

a +1 pont indoklás esetén jár:

a) hibás, mert nem betűre végződik: — 1+1 pont

b) helyes: — 1 pont

c) hibás, mert nagybetűket tartalmaz: — 1+1 pont

d) hibás, mert két dollárjel van benne: — 1+1 pont

e) hibás, mert két speciális jellel kezdődik: — 1+1 pont

f) helyes: — 1 pont

4. feladat: (13 pont)

A: A (*2*) és a (*4*) változat kever szabályosan. — 6 pont

(6 pont a hibátlan válaszra. Ebből 2-2 pont levonás minden tévedésre; 0 pont, ha kettőnél többet téved.)

B: Az (*1*) és a (*3*) változat egyes lapokat megtöbbszörözhet, másokat kihagyhat. — 2 pont

Az (*1*) változat nem jegyzi meg, hogy mely lapokat osztotta már ki. — 2 pont

A (*3*) ezt ugyan megteszi, de a már egyszer kiosztott lapokból is újra oszt. 2 pont

C: A (*2*) változat a VÉLETLEN() függvényt a pakli leosztása közben várhatóan egyre gyakrabban hívja meg egy-egy lap kiosztásához, ugyanis egyre valószínűbb, hogy a függvény egy már kiosztott lap számát állítja elő. 1 pont

(Háttér-információ a javító tanárnak: Az n-edik lap kiosztásához szükséges hívások száma geometriai eloszlást követ, amelynek várható értéke a kívánt esemény valószínűségének reciproka, esetünkben 32/(33-n). Ezért a teljes program lefutásához várhatóan 32/32 + 32/31 + 32/30 + ... + 32/1 hívás szükséges. Megsejtése plusz pont!)

5. feladat: (7 pont)

Növekvő sorrendbe rendez 1 és 300 közötti egész számokat.

A. Rendez: 3 pont

B. növekvően: 2 pont

C. 1 és 300 közötti egészeket : 2 pont

6. feladat: (8 pont)

A: SZÓ = HIDEG; 3 pont

B: KULCS = K; 1 pont

C1: A művelet KÓD(I-1) operandusából és KÓD(I) eredményéből kiszámítható SZÓ(I) operandusa, és így a szó hátulról visszafelé haladva helyreállítható. 2 pont

Vagy

C2: Az itt alkalmazott XOR művelet asszociatív és kommutatív; ha két operandusa azonos, akkor 0 az eredménye; ha egyik operandusa 0, akkor az eredménye azonos a másik operandusával. Ezért SZÓ(I) = KÓD(I) művelet KÓD(I-1), ami az első betű kivételével mindig igaz. Az első betűre pedig KULCS = KÓD(0) = SZÓ(1) művelet KÓD(1) igaz. 4 pont

Elérhető összpontszám: 57 pont

Harmadik-ötödik osztályosok

1. feladat: (15 pont)

```
A:    3  4  5                                  5 pont
      2  1  6
      9  8  7

B:    6  3  8                                  5 pont
      2  1  4
      7  5  9

C:    9  3  8                                  5 pont
      2  1  4
      6  5  7
```

2. feladat: (6 pont)

Az egységkör területét. (2 pont, ha negyedkör a válasz.)

3. feladat: (11 pont)

A: helyes 1 pont

```
(1), (5), (9), (6), (12);                      2 pont
```

B: hibás; 1 pont

C: helyes 1 pont

```
(1), (3), (8), (10), (6), (13);
```
2 pont

D: helyes 1 pont

```
(2), (6), (13), (3), (8), (10);
```
2 pont

E: hibás. 1 pont

4. feladat: (10 pont)

HIBA1	-	)-jel a (-párja nélkül	2 pont
HIBA2	-	)-jel le nem zárt [-jellel	2 pont
HIBA3	-	]-jel a [-párja nélkül	2 pont
HIBA4	-	]-jel le nem zárt (-jellel	2 pont
HIBA5	-	nincs minden (-jelnek)-párja	1 pont
HIBA6	-	nincs minden [-jelnek]-párja	1 pont

5. feladat: (8 pont)

A: SZÓ = KABALA; 3 pont

B: KULCS = C; 1 pont

C1: A művelet KÓD(I-1) operandusából és KÓD(I) eredményéből kiszámítható SZÓ(I) operandusa, és így a szó hátulról visszafelé haladva helyreállítható. 2 pont

vagy

C2: Az itt alkalmazott XOR művelet asszociatív és kommutatív; ha két operandusa azonos, akkor 0 az eredménye; ha egyik operandusa 0, akkor az eredménye azonos a másik operandusával. Ezért SZÓ(I) = KÓD(I) művelet KÓD(I-1), ami az első betű kivételével mindig igaz. Az első betűre pedig KULCS = KÓD(0) = SZÓ(1) művelet KÓD(1) igaz. 4 pont

6. feladat: (16 pont)

A: Csak a (*2*) változat kever szabályosan. max. 6 pont

(6 pont a hibátlan válaszra. Ebből 4 pont a levonás, ha a (*4*) változatot is szabályosnak vette; és tévedésenként 2 pont a levonás, ha az (*1*), (*2*) és (*3*) változatokra vonatkozó valamelyik válasz hibás.)

B: Az (*1*) és a (*3*) változat egyes lapokat megtöbbszörözhet, másokat kihagyhat.

Az (*1*) változat nem jegyzi meg, hogy mely lapokat osztotta már ki. 1 pont

A (*3*) változat ezt ugyan megteszi, de a már egyszer kiosztott lapokból is újra oszt. 2 pont

A (*4*) változat minden lapot csak egyszer oszt ki, de nem szabályosan, mivel nem gondoskodik a lap helybenhagyásáról. Ezért például az első lap nem maradhat az első helyen. 4 pont

C: A (*2*) változat a VÉLETLEN() függvényt a pakli leosztása közben várhatóan egyre gyakrabban hívja meg egy-egy lap kiosztásához, 2 pont

ugyanis egyre valószínűbb, hogy a függvény egy már kiosztott lap számát állítja elő. 1 pont

(Háttér-információ a javító tanárnak: Az n-edik lap kiosztásához szükséges hívások száma geometriai eloszlást követ, amelynek várható értéke a kívánt esemény valószínűségének reciproka, esetünkben 32/(33-n). Ezért a teljes program lefutásához várhatóan 32/32 + 32/31 + 32/30 + ... + 32/1 hívás szükséges. Megsejtése plusz pont!)

7. feladat: (7 pont)

Növekvő sorrendbe rendez 1 és 300 közötti egész számokat.

A. Rendez:	3 pont
B. Növekvően:	2 pont
C. 1 és 300 közötti egészeket:	2 pont

8. feladat: (14 pont)

A1: SELECT név, fizetés	2 pont
FROM alkalmazottak	1 pont
WHERE munkakör = "osztvez" AND fizu > 10000	2 pont
A2: SELECT név, osztnév	2 pont
FROM alkalmazottak, osztályok	2 pont
WHERE oszt = kód AND hely = "Pécs"	2 pont
B: Kiírt nevek: Éber, Élő, Buzgó	3 pont

Minden hibáért egy pont levonás jár a háromból!

Elérhető összpontszám: 87 pont

1990. Második forduló

Első-második osztályosok

Metró:

A feladatot 4, egyre bővülő lépésben kellett megoldani. Az első változatban csupán a metróállomás peronja, valmint a metrószerelvények szerepelnek. Az első metrószerelvény érkezési idejét az egyik irányból az EGYIK, a másikból pedig a MÁSIK változó tartalmazza. A LE változóba tesszük a leszállók számát, az EVAR és az MVAR pedig az egyik, illetve a másik irányba haladó metróra várakozók száma lesz.

```
Szimuláció:
  IDŐ:=0
  Ciklus
    IDŐ:=IDŐ+1
    DB:=véletlen érkező szám egyik irányba; EVAR:=EVAR+DB
    DB:=véletlen érkező szám másik irányba; MVAR:=MVAR+DB
    Ha (IDŐ mod TE)=EGYIK akkor LE:=véletlen leszálló szám
    Ha (IDŐ mod TE)=MÁSIK akkor LE:=véletlen leszálló szám
    LE:=0
    Ha ((IDŐ-TA) mod TE)=EGYIK akkor EVAR:=0
    Ha ((IDŐ-TA) mod TE)=MASIK akkor MVAR:=0
  Ciklus vége
Eljárás vége.
```

A második változatban két mozgólépcsőt is kell kezelni. Ezek tárolására a LELEP(1..N) és a FELLEP(1..N) tömböket használjuk. Új változó jelenik meg a lefele és a felfele menő mozgólépcsőre várakozók számára: LEVAR, FELVAR.

```
Szimuláció:
  IDŐ:=0
  Ciklus
    IDŐ:=IDŐ+1
    DB:=véletlen érkező szám; LEVAR:=LEVAR+DB
    Ha LELEP(N)>0 akkor Ha véletlenszám<.5 akkor EVAR:=EVAR+1
                                          különben MVAR:=MVAR+1
    Ha LELEP(N)>1 akkor Ha véletlenszám<.5 akkor EVAR:=EVAR+1
                                          különben MVAR:=MVAR+1
    Ciklus i=N-től 2-ig
      LELEP(i):=LELEP(i-1)
    Ciklus vége
    Ha LEVAR≤2 akkor DB:=LEVAR különben DB:=2
    LELEP(1):=DB; LEVAR:=LEVAR-DB

    Ha (IDŐ mod TE)=EGYIK akkor LE:=véletlen leszálló szám
    Ha (IDŐ mod TE)=MÁSIK akkor LE:=véletlen leszálló szám
    FELVAR:=FELVAR+LE; LE:=0
    Ciklus i=2-től N-ig
      FELLEP(i-1):=FELLEP(i)
    Ciklus vége
    Ha FELVAR≤2 akkor DB:=FELVAR különben DB:=2
    FELVAR:=FELVAR-DB; FELLEP(N):=DB
    Ha ((IDŐ-TA) mod TE)=EGYIK akkor EVAR:=0
    Ha ((IDŐ-TA) mod TE)=MASIK akkor MVAR:=0
  Ciklus vége
Eljárás vége.
```

A harmadik változat lényege az egyes helyiségek méretének korlátozása. Emiatt a fentről érkezés a következővé alakul:

```
    DB:=véletlen érkező szám; LEVAR:=LEVAR+DB
    Ha LEVAR≥VDB akkor LEVAR:=VDB
```

A metróról le, illetve felszállás is átalakul (csak az egyik irányra fogalmazzuk meg, a másik hasonló lehet). A metrón levők száma csökken a leszállókkal, nő a felszállókkal, amíg meg nem telik. A leszállást és a felszállást időegységekre kell bontani,egyszerre legfeljebb SZ ilyen ember lehet. A leszállást az is befolyásolja, hogy van-e hely a peronon, a felszállást pedig az, hogy van-e hely a metrószerelvényen.

```
    Ha (IDŐ mod TE)=EGYIK akkor LE:=véletlen leszálló szám
                                EMETRO:=igaz; ESZAM:=ESZAM-LE
    Ha EMETRO akkor
       Ha LE>0 akkor Ha LE≤SZ akkor DB:=LE különben DB:=SZ
                     Ha EVAR+MVAR+FELVAR+DB≥PDB
                        akkor DB:=PDB-EVAR-MVAR-FELVAR
                     LE:=LE-DB; FELVAR:=FELVAR+DB
            különben Ha EVAR≤SZ akkor DB:=EVAR különben DB:=SZ
                     Ha DB>MDB-ESZAM akkor DB:=MDB-ESZAM
                     EVAR:=EVAR-DB; ESZAM:=ESZAM+DB
    Ha ((IDŐ-TA) mod TE)=EGYIK akkor EMETRO:=hamis
```

Ha a fentről jövők miatt a peron betelt, akkor hibajelzést kell adni.

```
Ha LELEP(N)>0 akkor Ha véletlenszám<.5 akkor EVAR:=EVAR+1
                                       különben MVAR:=MVAR+1
Ha LELEP(N)>1 akkor Ha véletlenszám<.5 akkor EVAR:=EVAR+1
                                       különben MVAR:=MVAR+1
Ha EVAR+MVAR+FELVAR≥PDB akkor HIBA
```

A negyedik részfeladat újdonsága a harmadik mozgólépcső. Meg kell oldani az indítását, leállítását, illetve a kezelését. HLE igaz, ha a harmadik lépcső lefelé megy, hamis ha nem, HFEL igaz, ha felfelé megy, hamis ha nem Az automatikus indítás a következő lehet:

```
Ha LEVAR=VDB és nem HFEL akkor HLE:=igaz
```

```
Ha EVAR+MVAR+FELVAR=PDB és nem HLE akkor HFEL:=igaz
```

A két változó hamisra állítása, azaz a harmadik lépcső megállítása a felhasználó feladata. A harmadik lépcsőt a HLEP(1..N) tömbbel ábrázoljuk.

```
Ha HLE akkor Lefelé menő lépcső kezelés
    különben Felfelé menő lépcső kezelés
```

A kétféle lépcső kezelés jellegében azonos az eddigi LELEP, illetve FELLEP lépcsők kezelésével.

Mivel az első lefelé menő lépcső leállítására is lehetőség van, ezért az ELE változó legyen igaz, ha az első lépcső lefelé megy, illetve hamis, ha áll. Ehhez a második változat algoritmusában szereplő dőltbetűs rész végrehajtását kell feltételhez kötni:

```
Ha ELE akkor {ide jön a dőltbetűs rész}
```

Értékelési szempontok:

A:		15 pont
1.	érkezés	1 pont
	irányválasztás	1 pont
2.	sorba beállás	2 pont
3.	metrók érkezése TE időközönként	2 pont
	metrók várakozása TA időegységig	1 pont
4.	utasok kiszállása	1 pont
5.	várakozók felszállása	1 pont
6.	leszállók kilépése az állomásról	1 pont
	Eredmény: két várakozó sor,	2 pont
	metró érkezés, indulás	2 pont
	idő kijelzés	1 pont
B:		15 pont
7.	sorbaállás	1 pont
8.	lépcsőrelépés	2 pont
9.	lefelé menő lépcső kezelése	2 pont
	felfelé menő lépcső kezelése	2 pont
10.	felfelé várakozók sora	1 pont
11.	(azonos a 6.-kal)	------
	Eredmény: két újabb sor,	2 pont

lépcsők állapota	4 pont	
lépcsők iránya	1 pont	
C:		15 pont
12. érkező metró utasszáma	1 pont	
13. váróterem betelik	1 pont	
14. metrókocsi betelik	1 pont	
15. szimuláció leáll, ha kell	1 pont	
16. egyszerre max. SZ számú utas	3 pont	
felszállók a leszállók után	2 pont	
17. nem elférők nem szállnak ki	1 pont	
Eredmény: metró telítettség,	2 pont	
betelések	3 pont	
D:		20 pont
18. felhasználó lépcsőt indít	2 pont	
felhasználó lépcsőt leállít	2 pont	
leállítás jókor	4 pont	
a harmadik lépcső mozgatása	2 pont	
a harmadik lépcsőre fellépés	2 pont	
19. automatikus indítás lefelé	2 pont	
automatikus indítás felfelé	2 pont	
20. peron megtelés (15. benne van)	1 pont	
Eredmény: harmadik lépcső kijelzése	2 pont	
harmadik lépcső iránya	1 pont	
Programminőség (struktúra):		25 pont
Tagolás soronként	3 pont	
Bekezdések	4 pont	
Magyarázatok	5 pont	
Szabályos szerkezetek	5 pont	
Eljárások alkalmazása	5 pont	
Beszédes azonosítók	3 pont	
Program minőség (megjelenítés):		10 pont
Paraméterezés (mit és hogyan)	3 pont	
Paraméterezés ellenőrzése	2 pont	
Képernyő esztétikussága	5 pont	

Megjegyzések:

- az egyes részfeladatokra pluszpontok adhatók,
- paraméterezés helyett használhatók alkalmasan megválasztott konstansok.

Elérhető összpontszám 100 pont

Harmadik-ötödik osztályosok

Szóelválasztás:

A feladat megoldásának első lépése egy szóolvasó eljárás megírása, mert utána már csak szavakkal kell foglalkozni.

```
Szóolvasás(f,SZÓ):
  Olvas(f,C); SZÓ:=''
  Ciklus amíg C≠' ' és C≠sorvég
    SZÓ:=SZÓ+C; Olvas(f,C)
  Ciklus vége
Eljárás vége.
```

Lehetnek a szavakban több jellel jelölt, de egységként kezelendő betűk (ékezetes magánhangzók, többjegyű mássalhangzók). Ezeket célszerű egyetlen jellel kódolni, kiegészítve ezzel a hagyományos karakterkészletet. Figyelni kell arra is, hogy a többjegyű hosszú mássalhangzókat duplázni kell (pl. ccs → cs,cs).

Ugyancsak érdemes már a szóolvasásnál megszámolni, hogy a szóban hány magánhangzó van. Az 1. feltétel szerint ugyanis nem választjuk el azokat a szavakat, amelyek legfeljebb 3 hangból állnak, illetve amelyek egyetlen magánhangzót tartalmaznak.

Következő lépésként el kell készíteni egy eljárást, amely egy szóból levágja az első szótagot. Ezt sok változatban készítjük el, a feladat szövegében leírtaknak megfelelően.

```
Szótagolvasás(SZÓ,SZÓTAG):
  SZÓTAG:=''
  Ciklus amíg SZÓ(1) nem magánhagzó
    SZÓTAG:=SZÓTAG+SZÓ(1); SZÓ:=elsőutániak(SZÓ)
  Ciklus vége
  Ciklus
    SZÓTAG:=SZÓTAG+SZÓ(1); SZÓ:=elsőutániak(SZÓ)
  amíg SZÓ(1) nem magánhangzó és SZÓ(2) nem magánhangzó
  Ciklus vége
Eljárás vége.
```

Ezzel az eljárással az 1-8.szabályok mindegyikét teljesítettük, hála a jó hangkódolásnak.

A 9. szabályhoz figyelni kell a szótagolvasás második ciklusában a kötőjelet:

```
  Ciklus
    SZÓTAG:=SZÓTAG+SZÓ(1); SZÓ:=elsőutániak(SZÓ)
  amíg SZÓ(1) nem magánhangzó és SZÓ(2) nem magánhangzó
       és SZÓ(1)≠'-'
  Ciklus vége
  Ha SZÓ(1)='-' akkor SZÓ:=elsőutániak(SZÓ)
```

A 10. szabály az első olyan, amely egy szóra az eddigi szabályoktól eltérő szótagolást okozhat (leg-in-kább, le-gen-da). Mivel a programnak nem kell tudnia a magyar nyelvet, ezért az előző két példára az egyébként nyelvtanilag hibás (le-gin-kább, leg-en-da) változatokat is ki kell írnia. Ez azt jelenti, hogy az eddigi eljárás mellett meg kell hívni egy másikat is, ami az új szabály szerinti szótagolást végzi:

```
Leg-es Szófeldolgozás(SZÓ):
  Ha SZÓ(1..3)='leg' akkor Ki: 'leg-'
                           Szófeldolgozás(SZÓ(4..hossz(SZÓ))
Eljárás vége.
```

11. A mássalhangzóra végződő igekötő önálló szótagképző: agyon-, át-, benn-, el-, el-len-, fel-, föl-, fe-lül-, fö-lül-, fenn-, fönn-, ke-resz-tül-, kü-lön-, meg-, széj-jel-, szét-, to-vább-, túl-, vé-gig-. Pl. meg-int, fel-pró-bál.

12. Külön vizsgálandók a magánhangzóra végződő igekötők (ab-ba-, alá-, be-, be-le-, egy-be-, elő-, elő-re-, fél-be-, fél-re-, hát-ra-, ha-za-, hely-re-, hoz-zá-, ide-, ket-té-, ki-, köz-be-, köz-re-, le-, mel-lé-, ne-ki-, oda-, ösz-sze-, rá-, raj-ta-, szem-be-, vég-be-, visz-sza-), ha utánuk két vagy több mássalhangzó áll. Pl. ki-pró-bál, be-ska-tu-lyáz (de: ber-zen-ke-dik).

A 11. és a 12. szabály az igekötőkre ír elő újabb szabályokat. Mivel ezek is olyanok, hogy formailag többféleképpen is értelmezhetők, itt is kétféle feldolgozást kell elvégezni, sőt formailag ez a két szabály összevonható a 10. szabállyal. Ehhez fel kell venni egy konstans tömböt a lehetséges igekötőkkel és elválasztásukkal:

```
Igekötők: Tömb(1..MaxIgekötőszám,Rekord(mit,mire:Szöveg))=
          (('agyon','agyon-'),…('bele','be-le-'),…)
```

Az igekötő feldolgozása ezek után egy táblázatban kereséssé alakul:

```
Igekötő feldolgozása(SZÓ):
  i:=1
  Ciklus amíg i≤MaxIgekötőszám és
              Igekötők(i).mit≠SZÓ(1..hossz(Igekötők(i)))
    i:=i+1
  Ciklus vége
  Ha i≤MaxIgekötőszám
     akkor Ki: Igekötők(i).mire
              Szófeldolgozás(SZÓ(hossz(Igekötők(i))..hossz(SZÓ))
  Elágazás vége
Eljárás vége.
```

A 13. szabály ragokról és képzőkről szól. Ezeket a szavak végéről ugyanúgy kell levágni, mint az igekötőket a szavak elejéről. Ehhez ugyanolyan táblázatot kell kitölteni, mint az igekötők táblázata az előző két szabálynál.

Az esztétikai szabályok (14-16) a szótagolási szabályt, azaz a szótagolvasást módosítják.

Értékelési szempontok:

Tesztelésre csak értelmes szavakat érdemes használni, mert a programoknak a magyar nyelv szabályait nem kell tesztelnie, és hibára sem kell felkészülnie):

A) Általában:			20 pont
Egyértelmű esetet sohasem tekint kétesnek.		2 pont	
Billentyűzetről és állományból is működik.		2-3 pont	
Állomány végét felismeri.		1 pont	
Szóközzel és sorvéggel tagolható a szöveg.		1-1 pont	
Egyenlőségjelet felismeri.		1 pont	
Kétes eseteket egyenként végigkérdezi, és végül a teljes szöveget egyben írja ki.		5 pont	
Ékezetes betűket felismeri.		2 pont	
Nagy- és kisbetűket jól kezeli.		2 pont	
B) Egyszerűbb elválasztások: (1.-8. szabályok)			20 pont
Egy szótagút nem választ el.	1 pont	bal	
Többszótagút elválaszt, ha a két magánhangzó között:			
- egyjegyű rövid mássalhangzó van.	1 pont	ka-lap	
- több egyjegyű rövid mássalhangzó van.	1 pont	tor-ta	

- lehet egyjegyű hosszú mássalhangzó is.	1 pont	dur-ran elv-vel csekk-hez
- többjegyű rövid mássalhangzó egyedül áll.	2 pont	bo-dza
- többjegyű rövid mássalhangzó nem egyedül áll.	2 pont	mor-zsa fin-dzsa racs-ni
- többjegyű hosszú mássalhangzó egyedül áll.	2 pont 1 pont	meny-nyi bridzs-dzsel
- többjegyű hosszú mássalhangzó nem egyedül áll.	1 pont 1 pont	menny-be kulcs-csal
X-et jól elválasztja.	0 pont	pra-xis
Ch-t többjegyűnek veszi.	0 pont	or-chi-dea
X-val,-vel,-vá,-vé ragos alakját jól választja el.	2 pont	bórax-szá
Ch-val,-vel,-vá,-vé ragos alakját jól választja el.	2 pont	pech-hel
Kétes esetben többet is kijelez.	3 pont	srác-hoz srá-choz
C) Bonyolultabb elválasztások: (9.-16. szabályok)		30 pont
Kötőjelesen (- jellel!) beadott összetett szót jól választ el (nem tesz ki két kötőjelet).	1 pont	tip-top
Egyenlőségjelet (? !) többjegyű betű esetén és két magánhangzó között jól értelmez.	1 pont	ház-szem-le cse-re-a-lap
Jól felismeri és leválasztja a leg-et.	1 pont	leg-a-lább
Jól felismeri és leválasztja az igekötőket (2 pont ha csak néhányat).	4 pont	fel-e-sel, ki-pró-bál, meg-int
Jól felismeri és leválasztja (csak szó végéről) a -ság, -ség -szerű,... képzőket (2 pont ha csak néhányat).	4 pont	ma-lac-ság száz-szor ész-sze-rű
Leg-, igekötő, -ság, -ség, -szerű stb. esetén kétes esetben több változatot kijelez (ui. ha az értelmét nem ismerjük, nem tudhatjuk, melyik a helyes)	3 pont	le-ga-lább leg-a-lább fel-e-sel, fe-le-sel meg-int, me-gint ma-lac-ság,ma-la-cság száz-szor, százs-zor
Igekötőknél és -ság,-ször... -nél ékezetes és ékezet nélküli betű között különbséget tesz.	1 pont	
Két magánhangzó közé nem tesz elválasztójelet.	2 pont	tea-fű
Mégis tesz két mgh. közé elválasztójelet, ha ott mgh-ra végződő igekötő van (1 pont, ha csak néhánynál tudja).	2 pont	le-áll ki-e-mel
Ilyenkor mindkét esetet felkínálja (nem tudhatja, melyik jó).	2 pont	le-áll,leáll ki-e-mel kie-mel
Igekötő, ill. leg- után, ha nincs teljes szótag, akkor ott nem választ el.	3 pont	kü-lönb be-lem fe-le
Szó elején álló mgh-t nem választ le.	2 pont	idill

Szó végén álló mgh-t nem választ le.	2 pont	ké-mia	
Szó elején álló mgh-csoportot nem választ le.	2 pont	Euró-pa	
Programminőség: struktúra.			25 pont
Tagolás soronként.		3 pont	
Bekezdések.		4 pont	
Magyarázatok.		5 pont	
Szabályos (strukturált) szerkezetek.		5 pont	
Eljárások alkalmazása.		5 pont	
Beszédes azonosítók.		3 pont	
Programminőség: megjelenítés.			10 pont
Kényelmes adatbevitel.		5 pont	
Képernyő esztétikussága.		5 pont	

Megjegyzés: az egyes részfeladatokra – indoklással – pluszpontok adhatók.

Elérhető összpontszám: 105 pont

1991. Első forduló

Első-második osztályosok

1. feladat: (5 pont)

Minden olyan szóra "JÓ!" választ ad, amelyek vagy teljes egészükben tükörképei önmaguknak, vagy pedig egy tükörkép rész zár közre bennük újabb tükörképszavakat.

A, B, D: "ROSSZ!"; C, E: "JÓ!".

Minden helyes válaszra 1-1 pont

2. feladat: (8 pont)

A feladatok, illetve a megoldások a PROLOG programozási nyelv egy *magyarított* változatán készültek.

A. a. testvér(x,y). Jobb: testvér(x,y) VAGY féltestvér(x,y). 2 pont

b. nagyapja(x,y). 2 pont

B. a. anyaiNagyszülője(x,y) HA szülője(x,z) ÉS anyja(z,y). 2 pont

b. szülőpár(x,y) HA szülője(x,z) ÉS szülője(y,z) ÉS x<>y. 2 pont

3. feladat: (8 pont)

A. Helyes. 1 pont

B. Hibás, mert Evő megpróbálhat palacsintát felvenni a tányérról, mielőtt Sütő szólna, hogy EHETSZ. 2 pont

C. Hibás, mert ha Sütő elég gyors, akkor az ÜRES A TÁNYÉR jelzésre hamarabb ráteheti a következő palacsintát a tányérra, minthogy Evő elvenné az ott levőt. 2 pont

D. Helyes. 1 pont

D lassúbb A-nál, mert Sütő csak azután folytathatja a sütést, miután Evő megette a palacsintát, azaz amíg Evő eszik, Sütő nem sütheti a következőt. 2 pont

4. feladat: (12 pont)

A: Bármilyen szöveg, amelyben LILILLA vagy LILLILLA előfordul. Semmilyen más megoldás nem jó. 2-2 pont

B: 40 IF K$ <> "L" THEN ÍR("LI"): GOTO 20 ELSE OLVAS(K$) 4-4 pont

50 IF K$ <> "A" THEN ÍR("LIL"): GOTO 20 ELSE GOTO 10

Más megoldások is elfogadhatók, de a fentieknél bonyolultabbakra csak 2-2 pont adható.

5. feladat: (11 pont)

A: a, d: igen; b, c: nem.

Két vagy három helyes válasz esetén: 1 pont

illetve ha mind a négy válasz helyes: 2 pont

B: Nincs igaza. A gép 3-mal osztható összegeket hajlandó kifizetni. (Ui. a piros lámpa akkor gyullad fel, ha az addig bedobott összeg osztható 3-mal. A zöld lámpa ég, ha 1 a maradék, és a kék ég, ha 2 a maradék.) 4 pont

C: 510 Ft-ot. (Ui. 29 – 1 = 511, de mivel ez nem osztható 3-mal, 510 a helyes megoldás.) 2 pont

D: Legalább 6 zsetont. (Ui. 6 zsetonért 60 Ft-ot kell fizetnem, és 63 Ft-ot nyerhetek vele.) 3 pont

6. feladat: (5 pont)

A. 9 2 pont

B. 2 – 2 = 0 1 pont

C. Y – X 2 pont

7. feladat: (7 pont)

Az algoritmus a vasúti menetrendhez tartozó gráf ún. *szélességi bejárásán* alapszik.

A. Székesfehérvár 1 pont

B. Veszprém, Fonyód, Dombóvár 3 pont

C. Szombathely 1 pont

D. Nagykanizsa, Kaposvár 2 pont

de ha Kaposvárt nem adja meg 0 pont

ha Pécs is szerepel a listán -2 pont

Azaz városonként 1-1 pont, amíg a sorrend megegyezik az itt megadottal.

Elérhető összpontszám: 56 pont

Harmadik-ötödik osztályosok

1. feladat: (12 pont)

A nyissz eljárás az ún. *first fit* algoritmust alkalmazza: az első olyan rúdból vág le h hosszúságú darabot, amelynek hossza legalább h.

A nyassz eljárás az ún. *best fit* algoritmusra épül: a h-nál nem rövidebb rudak közül abból vág le h hosszúságú darabot, amelyiknél a legkisebb lesz a maradék.

A. A visszatérési értékek, ill. a tömb tartalma a két függvényeljárás esetén: nyissz: 1, 2, 3, 4 és [0, 1, 0, 2, 7, -1] 3 pont

nyassz: 1, 4, 3, 2 és [0, 3, 0, 0, 7, -1] 3 pont

B. 1, 2 esetén nyissz a 2-re ad 0-t eredményül, nyassz jó. 2 pont

C. 1, 2, 2 esetén nyassz az utolsó 2-esre 0-t ad eredményül, nyissz jó. (A B. és C. kérdésekre nincs más jó megoldás!) 4 pont

2. feladat: (10 pont)

A feladatok, illetve a megoldások a PROLOG programozási nyelv egy *magyarított* változatán készültek.

A. a. testvér(x,y). Jobb: testvér(x,y) VAGY féltestvér(x,y). 2 pont

b. nagyapja(x,y). 2 pont

c. őse(x,y). 2 pont

B. a. anyaiNagyszülője(x,y) HA szülője(x,z) ÉS anyja(z,y). 2 pont

b. szülőpár(x,y) HA szülője(x,z) ÉS szülője(y,z) ÉS x<>y. 2 pont

c. leszármazottja(x,y) HA szülője(y,x) VAGY (szülője(y,z) ÉS leszármazottja(x,z)). 3 pont

3. feladat: (8 pont)

A. Helyes. 1 pont

B. Hibás, mert Evő megpróbálhat palacsintát felvenni a tányérról, mielőtt Sütő szólna, hogy EHETSZ. 2 pont

C. Hibás, mert ha Sütő elég gyors, akkor az ÜRES A TÁNYÉR jelzésre hamarabb ráteheti a következő palacsintát a tányérra, minthogy Evő elvenné az ott levőt. 2 pont

D. Helyes. 1 pont

D lassúbb A-nál, mert Sütő csak azután folytathatja a sütést, miután Evő megette a palacsintát, azaz amíg Evő eszik, Sütő nem sütheti a következőt. 2 pont

4. feladat: (15 pont)

A bal oldali kiválasztó, a jobb oldali pedig buborékrendezés. Mindkét algoritmus nagyság szerint (nem szigorúan monoton) növekvő sorba rendez.

A. állítás: Minden h-ra 1 és I–2 között: $A(h) \le A(h+1)$, és 2 pont
minden k-ra I és N között: $A(I-1) \le A(k)$. 2 pont

Szóban: Az A(1..I–1) résztömb elemei nagyság szerint (nem szigorúan monoton) növekvő sorrendben vannak, és közülük a legnagyobb – A(I–1) – sem nagyobb az A(I..N) résztömb bármely eleménél.

B. állítás: Teljesül az A. állítás, és 2 pont
minden k-ra I és J–1 között: $A(M) \le A(k)$. 2 pont

Szóban: Teljesül az A. állítás, és az A(M) elem biztosan nem nagyobb az A(I..J–1) résztömb bármely eleménél.

C. állítás: Minden h-ra 1 és I–2 között: $A(h) \le A(h+1)$. 2 pont

Szóban: Az A(1..I–1) résztömb elemei nagyság szerint (nem szigorúan monoton) növekvő sorrendben vannak.

D. állítás: Minden h-ra 1 és J–1 között: $A(h) \le A(h+1)$, és 2 pont
$A(J) > A(J+1)$, és 1 pont
minden k-ra J+1 és I–1 között: $A(k) \le A(k+1)$. 2 pont

Szóban: Az A(1..J) résztömb elemei nagyság szerint (nem szigorúan monoton) növekvő sorrendben vannak, és A(J) nagyobb A(J+1)-nél, és az A(J+1..I) résztömb elemei is nagyság szerint (nem szigorúan monoton) növekvő sorrendben vannak.

5. feladat: (12 pont)

A hash – kulcstranszformációs – függvény megadja paramétere számjegyei összegének Max-szal való osztási maradékát. Az insert eljárást e függvényt felhasználva helyez el elemeket egy tömbben, a függvényérték által megadott helyen, illetve ha az foglalt, akkor – ciklikusan – a következő szabad (-1 értékű) helyen.

A. -1, 26, 2, 117, 8, 18, -1 5 pont

B. 18, -1, 2, -1, 13, 14, -1, -1, 26, 117, 8 7 pont

Azaz minden jó helyre írt jó érték után 1-1 pont jár.

6. feladat: (12 pont)

A. Hibátlan átvitel esetén i értéke 0. 1 pont

B. Egyszeres hiba esetén i 1 és 7 közötti érték, a meghibásodott bit sorszáma. 2 pont

3, 5, 7,
3, 6, 7,
5, 6, 7.

C. Minden helyes szám: 1 pont

7. feladat: (12 pont)

A: Bármilyen szöveg, amelyben LILILLA vagy LILLILLA előfordul. Semmilyen más megoldás nem jó. 2-2 pont

B: 40 IF K$ <> "L" THEN ÍR("LI"): GOTO 20 ELSE OLVAS(K$) 4-4 pont

50 IF K$ <> "A" THEN ÍR("LIL"): GOTO 20 ELSE GOTO 10

Más megoldások is elfogadhatók, de a fentieknél bonyolultabbakra csak 2-2 pont adható.

8. feladat: (13 pont)

A. A d változóban az aktuális haladási irányt tárolja a program: 1: jobbra, 2: le, 3: balra, 4: fel. 1 pont

(Ha csak egy válasz is rossz, nem jár érte a pont!)

B. A jó megoldásokra egyenként 3-3 pont adható. Kis hiba esetén (pl. egy-két számot hibásan írt be, de egyébként következetes a megoldás) 1-1 pontot le kell vonni. Ha a **-gal megjelölt mezőkbe beírja a következő sorszámot, akkor ugyancsak le kell vonni 1-1 pontot, ugyanis az eljárás az utolsó mezőt sohasem törli. 4*3 pont

a)

	1	2	3	4	
15	16	17	18	5	6
14	13	**	19	8	7
	12	11	10	9	

b)

5	4	3	**	
6	1	2	11	
7	8	9	10	

c)

		5	4	3	
		6	1	2	
		7	8	**	

d)

	9	10	11	12	
7	8	19	**	13	14
6	5	18	17	16	15
	4	3	2	1	

Elérhető összpontszám: 84 pont

1991. Második forduló

Első-második osztályosok

1. Hetedhét: (40 pont)

A szűkszámok képzésére a következő szigorú szabály érvényes: Szűkszámjegyet legfeljebb három nála kisebb, de egyforma szűkszámjegy előzhet meg. A kisebb számjegyeknek közvetlenül a nagyobb számjegy előtt kell állniuk, és ilyenkor a kisebb jegyek értékét le kell vonni a nagyobb számjegy értékéből. Ettől eltekintve a szűkszámok számjegyeit balról jobbra csökkenő sorrendben kell

felírni, de úgy, hogy bármely szűkszámjegy után közvetlenül legfeljebb két vele azonos szűkszámjegy állhat. Ilyenkor a számjegyek értékét össze kell adni.

A használt szűkszámjegyek: A = 1, E = 7, O = 7^2, K = 7^3, T = 7^4, H = 7^5, M = 7^6.

Fel kell vennünk egy konstans tömböt, amely a fenti betűket és számokat tartalmazza:

```
S: Tömb(1..7,Rekord(Szűk: Karakter,Szám: HosszúEgész))
```

Ezután a szabály szerint venni kell a számjegyeket, s vagy pozitív, vagy negatív előjellel hozzáadni az eredményhez.

```
Átalakítás(SZÓ):
  i:=1; SZÁM:=0
  Ciklus amíg i≤hossz(SZÓ)
    j:=i
    Ciklus amíg j≤hossz(SZÓ) és SZÓ(i)=SZÓ(j)
      j:=j+1
    Ciklus vége
    Kiválasztás(S,SZÓ(i),iHely)
    Ha j≤hossz(SZÓ) akkor Kiválasztás(S,SZÓ(j),jHely)
    Ha j≤hossz(SZÓ) vagy iHely>jHely
       akkor SZÁM:=SZÁM+(j-i)*S(iHely).Szám); i:=j
    különben SZÁM:=SZÁM+S(jHely).szám-(j-i)*S(iHely).Szám)
             i:=j+1
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Tesztesetek:

A	1	jól kezeli a számjegyeket
E	7	
O	49	
K	343	
T	2401	
H	16807	
AAA	3	jól kezel egymás mellett 3 számjegyet
EO	42	jól kezel előtagot
KKKT	1372	jól kezel 3 előtagot
KO	392	jól kezel utótagot
TKKK	3430	jól kezel 3 utótagot
EOA	43	jól kezel elő- és utótagot egyszerre
AOAE	54	előtag lehet több helyen
AAT	2399	szóközt szűr
M	117649	jól kezel nem ábrázolható egész számot
MHTKOEA	137257	nagyon nagy szám

Értékelési szempontok:

A. Beolvasás		3 pont
B. Kiírás		3 pont
Csak képernyőre	1 pont	
Csak állományba	2 pont	

C. Összes szűkszámjegy felismerése		7 pont
(PC-n: Hosszú egészek kezelése is.)		
Számjegyenként	1-1 pont	
D. Előtagok kivonása		7 pont
Előtagok felismerése	3 pont	
Mind a 3 előtag felismerése	2 pont	
Jó érték levonása	2 pont	
E. Utótagok hozzáadása		7 pont
Utótagok felismerése	3 pont	
Mind a 3 utótag felismerése	2 pont	
Jó érték hozzáadása	2 pont	
F. Szóközök szűrése		3 pont
G. Programszerkezet, stílus		10 pont
Szabályos programszerkezetek	2 pont	
Megfelelő sorokra tagolás	1 pont	
Bekezdéses leírás	2 pont	
Sorok közötti tagolás	1 pont	
Eljárások megfelelő alkalmazása	2 pont	
Magyarázatok a programszövegben	1 pont	
Beszédes azonosítók alkalmazása	1 pont	

2. Útkereszteződés: (100 pont)

Mivel az útketesztеződésben 2 út mindkét sávját kell vizsgálni, célszerű a megoldásban a 4 sávot 1-1 tömbben elhelyezni. A közlekedési lámpák beállíthatók úgy, hogy vagy az ábrán vízszintesen haladó sávoknak van zöld jelzése, vagy a függőlegesen haladóknak. A lámpák kezeléséhez egy konstans tömböt hozunk létre, amely az egyes lámpaállások idejét tartalmazza a kezdőidőhöz képest `zöld, sárga,piros,piros-sárga` sorrendben. Az időt a megtett szimulációs lépések számával adjuk meg:

```
Lámpa: Tömb(0..3,Egész)
```

Legyen a `Hossz` változó a teljes lámpaváltás ciklusa, azaz a fenti négy időtartam összege. Ennek megfelelően a főprogram 4 eljárást tartalmaz:

```
Szimuláció:
  idő:=1; Állás:=0
  Ciklus
    Balrólfelül(T1); Balrólalul(T2); Felülről(T3); Alulról(T4)
    idő:=idő+1
    Ha idő MOD Hossz=Lámpa(Állás) akkor Állás:=Állás+1
  Ciklus vége
Eljárás vége.
```

A négy eljárás lényegében hasonló, így elég közülük illusztrációként egyet megírnunk. Legyen ez a `Balrólfelül` eljárás! Az útszakaszt akkora szakaszonként ábrázoljuk egy-egy tömbelemmel, amekkorát egy autó egy időegység alatt meg tud tenni: 1 lesz ott, ahol éppen van autó, 0 pedig ott, ahol nincs. Így első változatként az N hosszúságú úton az autók mozgását egyszerű léptetéssel lehet megoldani:

```
Balrólfelül(T):
  Ciklus i=N-1-től 1-ig -1-esével
    T(i+1):=T(i)
  Ciklus vége
  Ha véletlenszám<BE akkor T(1):=1 különben T(1):=0
Eljárás vége.
```

A második változatban vegyük hozzá, hogy az autók az útkereszteződésben balra is fordulhatnak. Az útkereszteződés legyen az út K. és K+1. pozíciójában! A fordulás esetünben annyit jelent, hogy a K+1. pozícióba lépő autók átlépnek a megfelelő keresztező útra (most a T4 K+1. pozíciójára):

```
Balrólfelül(T):
  Ciklus i=N-1-től 1-ig -1-esével
    Ha i≠K vagy T(i)=0 akkor T(i+1):=T(i)
    különben ha véletlenszám<FORD akkor T4(K+1):=1
    különben T(K+1):=1
  Ciklus vége
  Ha véletlenszám<BE akkor T(1):=1 különben T(1):=0
Eljárás vége.
```

Az utolsó változatban törődni kell a lámpákkal is.A vízszintes úton akkor lehet haladni, ha az `Állás` változó értéke 0, azaz eben az irányban zöld jelzést kapunk. Ekkor a függőleges utakon piros jelzés érvényes, ott autók éppen nem haladhatnak. Ha a sárga ideje elég nagy (legalább 2), akkor a kereszteződésben levő autók is kilépnek onnan, mire valamelyik irányból zöldre vált a lámpa. A lámpa előtti autók mozgását másképpen kell megoldani, mert ők csak akkor mozoghatnak, ha az előttük levő pozíción nincs autó.

```
Balrólfelül(T):
  Ciklus i=N-1-től K+1-ig -1-esével
    T(i+1):=T(i)
  Ciklus vége
  Ha T(K)=0 akkor T(K+1):=T(K)
  különben ha véletlenszám<FORD akkor T4(K+1):=1
  különben T(K+1):=1
  Ha ÁLLÁS=0 akkor T(K):=T(K-1); T(K-1):=0
  Ciklus i=K-2-től 1-ig -1-esével
    Ha T(i+1)=0 akkor T(i+1):=T(i); T(i):=0
  Ciklus vége
  Ha véletlenszám<BE akkor T(1):=1 különben T(1):=0
Eljárás vége.
```

Meg kell még oldani a forgalomtól függő lámpaváltást. Ehhez először megkell számolni azt, hogy a lámpánál hány autó áll. Ez megoldható az előbbi eljárás utolsó ciklusának módosításával, a `BFÁll` változó bevezetésével (a másik három út megfelelő változója: `BAÁll`, `FÁll`,`AÁll`):

```
  BFÁll:=0
  Ciklus i=K-2-től 1-ig -1-esével
    Ha T(i+1)=0 akkor T(i+1):=T(i); T(i):=0
              különben Ha T(i)=1 akkor BFÁll:=BFÁll+1
  Ciklus vége
```

Ezek után már csak azt kell eldönteni, hogy a 4 sávon éppen várakozó autók számától hogyan függjön a zöld lámpa ideje. Egy lehetséges megoldás (a zöld és a sárga 1 időegységgel később/korábban vált, a piros időtartama így a zöldével ellenkezőleg változik):

```
  Ha BFÁll+BAÁll>FÁll+AÁll akkor Lámpa(0):=Lámpa(0)+1
                                 Lámpa(1):=Lámpa(1)+1
  különben ha BFÁll+BAÁll<FÁll+AÁll akkor Lámpa(0):=Lámpa(0)-1
                                          Lámpa(1):=Lámpa(1)-1
```

Értékelési szempontok:

A. Ábrázolás		14 pont
Az útkereszteződés kirajzolása	6 pont	
Paraméterek beolvasása (sávonként 1)	4 pont	
Darabszámok kijelzése (sávonként 1)	4 pont	
B. Az autók mozgása		26 pont
Az autók vízszintesen mozognak	3 pont	
A kereszteződésnél tudnak fordulni balra	3 pont	
A kereszteződésnél tudnak fordulni jobbra	3 pont	
Az autók függőlegesen mozognak lefelé	3 pont	
Az autók függőlegesen mozognak felfelé	3 pont	
A kereszteződésnél tudnak fordulni is	3 pont	
Véletlenszerűen belépnek vízszintesen	3 pont	
Véletlenszerűen belépnek függőlegesen	3 pont	
A vízszintes út végén eltűnnek	1 pont	
A függőleges út végén eltűnnek	1 pont	
C. Lámpakezelés		20 pont
4 féle lámpát kezel (piros, piros+sárga, zöld, sárga)	4 pont	
A lámpaváltások időzítése (a sárga rövid)	4 pont	
Jól kezeli az ellentétes irányú lámpát	2 pont	
A forgalomtól függő lámpaállítás		
(van min-max idő a zöld lámpára,	2 pont	
az úton lévők akt. számától függ,	4 pont	
darabszámmal arányos)	4 pont	
D. Autók megállása, elindulása		15 pont
Az autók piros, sárga lámpánál megállnak	5 pont	
Álló autó mögött a következők megállnak	5 pont	
Álló autó elindul, ha lehetséges	5 pont	
E. Programszerkezet		25 pont
Szabályos programszerkezetek	5 pont	
Megfelelő sorokra tagolás	3 pont	
Bekezdéses leírás	4 pont	
Sorok közötti tagolás	2 pont	
Eljárások alkalmazása	5 pont	
Magyarázatok	3 pont	
Beszédes azonosítók	3 pont	
Minden egyébre pluszpont adható.		

Elérhető összpontszám: 140 pont

Harmadik-ötödik osztályosok

Szó ami szó: (140 pont)

Ez a feladat több részfeladatból áll. Egyrészt meg kell oldani a kifejezések kiértékelését (lengyel formára hozás, lengyel forma kiértékelése), másrészt pedig az elemi feltételek megvizsgálását: a szavak keresését.

Foglalkozzunk először a kifejezések lengyel formára hozásával. Egy szöveg típusú változóban adott kifejezést bontunk lexikális elemekre, amelyek egy szöveg típusú elemeket tartalmazó vektorba kerülnek.

```
Lexikális analízis(kifejezés,db,elemek):
  db:=0; i:=1
     Ciklus amíg i≤hossz(kifejezés)
        x:=kifejezés(i); i:=i+1
        Elágazás
          x∈Műveletijelek esetén db:=db+1; elem(db):=x
          x="(" vagy x=")" esetén db:=db+1; elem(db):=x
          x=' ' esetén {semmit nem kell tenni}
          egyéb esetben Szóolvasás(x,i); db:=db+1; elem(db):=x
        Elágazás vége
   Ciklus vége
Eljárás vége.
```

A `Szóolvasás(x,i)` eljárás az x változóba szedi ki a kifejezés i-edik pozícióján kezdődő szót, miközben i-t növeli

```
Szóolvasás(x,i):
  Ciklus amíg i≤hossz(kifejezés) és
                (kifejezés(i)='*' vagy kifejezés(i)∈Betűk)
     x:=x+kifejezés(i); i:=i+1
  Ciklus vége
Eljárás vége.
```

Másodjára olvassuk a lexikális elemeket és elkészítjük a lengyel formát. Ha olvasás közben operandust találunk, azt az eredménykifejezésbe automatikusan leírhatjuk, ha operátor jön, akkor azt egyelőre megjegyezzük és majd csak a következő operátor (-ok) ismerete után döntünk sorsáról. Fontos dolog, hogy a megjegyzett operátorok tárolási sorrendjét tudjuk, hisz az utoljára betettet kell majd először elővenni. (Ennek megoldására használunk un. verem adatszerkezetet, hisz az éppen ilyen tulajdonságokkal bír). Ha az aktuális operátor prioritása kisebb, mint az előzőé, akkor az előzőt kivesszük a veremből és a készülő kifejezésbe tesszük. Ha a verem tetején megint egy nála nagyobb prioritású operátor van, akkor ezt az eljárást folytatjuk.

A zárójelek használata annyiban bonyolítja a helyzetet, hogy a zárójelben levő kifejezésrészletek, mint önálló egységek kerülnek feldolgozásra. Ez úgy történik, hogy a nyitózárójelet betesszük a verembe és a csukó érkeztével az esetleg a nyitón levő operátorok mindegyikét a kifejezéshez másoljuk.

```
Lengyel formára hozás(db,elemek,ldb,lengyel):
  ldb:=0: Verembe(Végjel)
  Ciklus i=1-től db-ig
     A:=elemek(i)
     Elágazás
        A='(' esetén Verembe(A)
        A=')' esetén Zárójelfeldolgozás
        A∈{'+','&','!'} esetén Műveletfeldolgozás(A)
        egyéb esetben ldb:=ldb+1; lengyel(ldb):=A
     Elágazás vége
  Ciklus vége
  Veremürítés
Eljárás vége.

Zárójelfeldolgozás:
  Ciklus amíg Veremtető≠"("
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
  Veremből(B)
Eljárás vége.

Műveletfeldolgozás(A):
  Ciklus amíg prioritás(A)≤prioritás(Veremtető)
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
  Verembe(A)
Eljárás vége.

Veremürítés:
  Ciklus amíg Veremtető≠Végjel
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
Eljárás vége.
```

Harmadik fázis a kiértékelés. Ezt meg kell hívni a bemenő állomány összes bekezdésére.

```
Kiértékelés(X,Y,Z,ldb,lengyel):
  Ciklus i=1-től ldb-ig
     A:=lengyel(i)
     Elágazás
        A∈{'+','&'} esetén Veremből(B); Veremből(C)
                           Művelet(A,B,C,D); Verembe(D)
        A='!' esetén Veremből(B); D:=nem B; Verembe(D)
        egyéb esetben Keresés(A,B); Verembe(B)
     Elágazás vége
  Ciklus vége
  Veremből(Z)
Eljárás vége.

Művelet(A,B,C,D):
  Elágazás
     A='+' esetén D:=B vagy C
     A='&' esetén D:=B és C
  Elágazás vége
Eljárás vége.
```

A második fő feladat a keresés. Ez egyszerű lenne, ha nem használhatnánk joker (*) karaktereket. Ekkor ugyanis csak a bekezdésből való szóolvasást kellene megírnunk, majd a szavakat egyesével kellene hasonlítani a keresett A szóval.

A szóolvasásra ebben az esetben is szükségünk lesz:

```
Szóolvasás(SOR,SZÓ):
  i:=1; SZÓ:=''
  Ciklus amíg i≤hossz(SOR) és SOR(i)≠' '
    SZÓ:=SZÓ+SOR(i); i:=i+1
  Ciklus vége
   SOR:=SOR(hossz(SZÓ)+1..hossz(SOR))
Eljárás vége.
```

A beolvasott szavakat össze kell hasonlítani a keresett szóval. Az első részben megvizsgáljuk, hogy a keresett minta azonos-e a szóval, vagy a minta *-ra végződik és eleje azonos a szó elejével.

A második vizsgálat azzal az esettel foglalkozik, amikor a minta *-gal kezdődik. Ekkor a minta következő *-ig levő részét (vagy az egészet) kell keresni a szóban. Ha megtaláltuk, akkor a mintában és a szóban is meg kell még vizsgálnunk a maradék részeket.

A harmadik vizsgálatnál a minta biztos nem *-gal kezdődik. Ekkor a minta *-ig tartó részének meg kell egyezni a szó megfelelő hosszúságú részével, s ezután mindkettő maradék részét kell vizsgálni.

```
Hasonlítás(A,SZÓ,VAN):
  Ha A=SZÓ vagy A(1..hossz(A)-1)=SZÓ(1..hossz(A)-1) és   {1}
     A(hossz(A))='*' akkor VAN:=igaz
  különben ha A(1)='*' akkor                  {2}
    A:=A(2..hossz(A)); Keresés(A,'*',V,i)
    Ha nem V akkor VAN:=(SZÓ=A)
    különben Keresés(SZÓ,A(1..i-1),V,j)
             Ha V akkor A:=A(i..hossz(A))
                        SZÓ:=SZÓ(j+i-1..hossz(SZÓ))
                        Hasonlítás(A,SZÓ,VAN)
               különben VAN:=hamis
             Elágazás vége
    Elágazás vége
  különben Keresés(A,'*',V,i)                    {3}
           Ha nem V vagy A(1..i-1)≠SZÓ(1..i-1) akkor VAN:=hamis
           különben A:=A(I..hossz(A))
                    SZÓ:=SZÓ(i..hossz(SZÓ))
                    Hasonlítás(A,SZÓ,VAN)
           Elágazás vége
  Elágazás vége
Eljárás vége.
```

Tesztesetek:

Minta	**a bekezdés elé írt sorszám**
autok	3,4,8
*lampak	2
ut*	3,4
mo*o	3
lam*va*ok	7
ss	8
mozgo&autok&varakozo	3
harom+ket+egy	4
lathato&autok+allnak&haladnak	8

!autok	2,7
jelzolampak&!autok	2
!autok&jelzolampak	2
jelzolampak+(mozgo+autok)&adja	2,7
semmi+!(autok+akarmi)	2,7

Értékelési szempontok:

A. Szövegbeolvasás		5 pont
B. Mintabeolvasás		5 pont
Billentyűzetről beolvasás	1 pont	
Állományból beolvasás	3 pont	
A kettő között választani lehet	1 pont	
C. Kiírás		8 pont
Képernyőre kiírás	3 pont	
Állományba kiírás	5 pont	
D. Bekezdéskezelés		7 pont
E. Szóillesztés		25 pont
Szódarabolás, határoló karakterek felismerése	2 pont	
Illesztés joker nélkül	5 pont	
Illesztés egy jokerrel	10 pont	
Illesztés több jokerrel	8 pont	
F. Kifejezéskiértékelés		60 pont
ÉS, VAGY, TAGADÁS művelet helyes	10-10-10 pont	
Helyes precedencia	5 pont	
Helyez zárójeles kifejezéskiértékelés	25 pont	
G. Programszerkezet, stílus		30 pont
Szabályos programszerkezetek	6 pont	
Megfelelő sorokra, illetve sorok közötti tagolás	4-3 pont	
Bekezdéses leírás	5 pont	
Eljárások alkalmazása	6 pont	
Magyarázatok	3 pont	
Beszédes azonosítók	3 pont	

Elérhető összpontszám: 140 pont

1992. Első forduló

Első-második osztályosok

1. feladat: (10 pont)

A. 5 — 1 pont
4 — 1 pont
6 — 2 pont

B. A két szám legnagyobb közös osztója. — 6 pont

2. feladat: (15 pont)

A1. "Ipafai fapipa" — 1 pont

A2. "Ezüsttel befuttatott." — 1 pont

A3. "13"+CHR$(0)+CHR$(7)+"-szor ismételd meg." — 1 pont

A4. "Gyakorisága 8" — 1 pont

B. Szöveget tömörít, — 2 pont
2-nél többször ismétlődő karaktereket — 1 pont
a karakter + CHR$(0) + CHR$(darabszám) hármassal helyettesíti. — 1 pont

C. Ha A$ két vagy több azonos karakterrel ér véget, — 3 pont
ha 255-nél több egyforma karakter van A$-ban. — 2 pont

D. Az 1100-as és 1110-es sorba át kell másolni az 1040-es, ill. az 1050-es sorok tartalmát. (Bármilyen, ezzel azonos hatású megoldás is jó.) — 1-1 pont

3. feladat: (13 pont)

A. B(j) az A vektor j értékű elemeinek száma: — 3 pont

B. B(j) az A vektor j értékű vagy j-nél kisebb elemeinek száma: — 3 pont

C. B(j) az A vektor j-nél kisebb elemeinek száma: — 3 pont

D. C-ben az A vektor elemei vannak nagyság szerint növekvő sorrendben: — 4 pont

4. feladat: (18 pont)

F1. Egy lehetséges megoldás: — 6 pont

```
F1(I,T,J,K):
  K:=1
  Ciklus amíg K≤T és I+K≤N és A(I+K,J)=-1
     K:=K+1
  Ciklus vége
  Ha I+K≤N akkor K:=K-1
Eljárás vége.
```

Részpontszámok: K 0 és T közötti érték: — 2 pont
autót nem lép át: — 2 pont
I + K = N + 1 eset jó: — 2 pont

F2. A(I+K,J) := A(I,J) + 1 : A(I,J) : =-1 — 3 pont

F3. A(I+K,J) := A(I,J) : A(I,J) := -1 — 3 pont

F4. Lassítás 3 pont

F5. Sávváltás 3 pont

5. feladat: (8 pont)

A. Mitsubishi autóembléma: azaz 3 db 1 pont
x élhosszúságú 1 pont
rombusz. 1 pont
120-120 fokkal elforgatva 1 pont

B. Audi autóembléma, azaz 4 db 1 pont
10 egységnyire egymásbafonódó 1 pont
360/(2*pi) sugarú 1 pont
kör. 1 pont

6. feladat: (13 pont)

A. '1' - ettől kezdve egy szakasz végpontja mozog a képen, 2 pont
(J, B, L, F hatására) kezdőpontja az ekkori kurzorhelyzet. 2 pont

B. '2' ¤ ettől kezdve egy téglalap egyik sarka mozog a képen, 3 pont
szemben levő sarka az ekkori kurzorhelyzet. 3 pont

C. Az egyes funkciók (1 és 2) által rajzolt alakzatokat egy tetszőleges újabb funkció (1, 2) kiválasztása zárja le. 3 pont

7. feladat: (10 pont)

E1. Két szín keverékszíne (S3-ban): 3 pont

E2. Egy szín kiegészítő színe (S2-ben azok az alapszínek vannak, amelyek S1-ben nincsenek): 3 pont

E3. Egy szín (S1) felbontása alapszínekre (SZ): 3 pont
I-ben az előforduló alapszínek száma: 1 pont

8. feladat: (13 pont)

A. Az L2 karaktersorozatot az L1-hez fűzi: 4 pont

B. Az L karaktersorozatból kitörli az E elemet: 3 pont
de csak E első előfordulását vizsgálja: 2 pont

C. "NEMES+TIHAMÉR": 2 pont

D. "NMES": 2 pont

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

1. feladat: (18 pont)

A. -1,-2,-3,-4: az összeadás, kivonás, szorzás, osztás azonosítói, 2 pont
pozitív számok: aritmetikai műveletek operandusai. 2 pont

B. 7 - 5 = 2 2 pont

C. 2 * 2 + 6 / 2 = 7 (jó műveletenként 2-2 pont) 6 pont

D. 0-val osztáskor, 2 pont

ismeretlen művelet végrehajtásakor (azaz ha nem -1, -2, -3 vagy -4 a művelet kódja), 2 pont

ha éppen -1 a végeredmény. 2 pont

2. feladat: (15 pont)

A1. "Ipafai fapipa" 1 pont

A2. "Ezüsttel befuttatott." 1 pont

A3. "13"+CHR$(0)+CHR$(7)+"-szor ismételd meg." 1 pont

A4. "Gyakorisága 8" 1 pont

B. Szöveget tömörít, 2 pont

a 2-nél többször ismétlődő karaktereket 1 pont

a karakter + CHR$(0) + CHR$(darabszám) hármassal helyettesíti. 1 pont

C. Ha A$ két vagy több azonos karakterrel ér véget, 3 pont

ha 255-nél több egyforma karakter van A$-ban. 2 pont

D. Az 1100-as és 1110-es sorba át kell másolni az 1040-es, ill. az 1050-es sorok tartalmát. (Bármilyen, ezzel azonos hatású megoldás is jó.) 1-1 pont

3. feladat: (13 pont)

A. B(j) az A vektor j értékű elemeinek száma: 3 pont

B. B(j) az A vektor j értékű vagy j-nél kisebb elemeinek száma: 3 pont

C. B(j) az A vektor j-nél kisebb elemeinek száma: 3 pont

D. C-ben az A vektor elemei vannak nagyság szerint növekvő sorrendben: 4 pont

4. feladat: (18 pont)

F1. Egy lehetséges megoldás: 6 pont

```
F1(I,T,J,K):
  K:=1
  Ciklus amíg K≤T és I+K≤N és A(I+K,J)=-1
     K:=K+1
  Ciklus vége
  Ha I+K≤N akkor K:=K-1
Eljárás vége.
```

Részpontszámok:

K 0 és T közötti érték: 2 pont

autót nem lép át: 2 pont

I + K = N + 1 eset jó: 2 pont

F2. A(I+K,J) := A(I,J) + 1 : A(I,J) : =-1 3 pont

F3. A(I+K,J) := A(I,J) : A(I,J) := -1 3 pont

F4. Lassítás 3 pont

F5. Sávváltás 3 pont

5. feladat: (20 pont)

A. Az egyes négyzetek oldalain csak páros számú (0, 2, 4) szintvonal haladhat át. (Ha 1 vagy 3 szintvonalat is megenged, egyszer vagy kétszer -2 pont, de összesen: ≥ 0 pont.) 2-2-2 pont

B. (I,J) és (K,L) közötti szakaszvég kiszámolása:

Pl. (I,J) és (I,J+1) között:

```
S1:=abs(T(I,J)-T(I,J+1))
S2:=abs(T(I,J)-SZ)
TAV:=S*S2/S1
végpont:=(I*S-S/2,J*S-S/2+TAV)
```

Részpontszámok:
aránypár [S2:S1 = TAV:S]: 5 pont
hely: 3 pont

C. Végpontok összekötése

2 végpont esetén jól köt össze: 2 pont

4 végpont esetén jól köt össze: 4 pont

6. feladat: (16 pont)

A. B értéke 1, 2 vagy 3 lehet. 1-1-1 pont

B. B-t a 12330-as sorban módosított A változó értékének utolsó számjegyével növeli meg. 2 pont

A ennek az értéknek a tizedrésze (az utolsó számjegyet hagyja el B-ből). 2 pont

C. Legfeljebb 3 kódot kell kipróbálnia. 1 pont

Ki kell próbálnia az alábbi kódokat:

B = 3 esetére a 183-at, és 1 pont

B = 2 esetére a 133-at vagy a 188-at, és 1-1 pont

B = 1 esetére a 138-at vagy a 184-et vagy a 193-at 1-1-1 pont

ha B=2 és B=1 esetén utal a választhatóságra 1-1 pont

7. feladat: (13 pont)

A. A függvény sqrt(X) közelítő értékét adja meg. 5 pont

B. M a pontosságot írja elő. 2 pont

C. A ≤ sqrt(X) ≤ B, az [A,B] intervallum felezéssel egyre csökken. 3 pont

D. A = 0, 1 pont

B = X + 1 2 pont

Magyarázat: A ≤ sqrt(X) ≤ B, ezért

– ha X > 1, akkor 0 < sqrt(X) < X < X+1

– ha 0 ≤ X ≤ 1, akkor 0 ≤ sqrt(X) ≤ 1 ≤ X+1

8. feladat: (13 pont)

A. Hatása: ismerjFel egy új listát ad eredményül azzal, hogy l2-t l1-hez fűzi – ezért a neve pl. fűzzHozzá lehetne. 2 pont

Működése: ha l1 nem üres, először is leválasztja róla az elemet (a fejét), majd feltételezve, hogy már létezik az l1 farkából és az l2-ből (saját maga rekurzív meghívásával) összefűzött lista, hozzLétre segítségével létrehozza a kívánt listát. 3 pont

B. Hatása: találjKi egy új listát ad eredményül azzal, hogy az első e elemet kitörli az l listából; ha nincs benne, akkor magát l-et adja vissza – ezért a neve pl. töröljKi lehetne. 2 pont

Működése: ha l nem üres, megnézi, hogy a feje e-vel egyenlő-e; ha igen, e-t elhagyva l farka az eredmény; ellenkező esetben feltételezi, hogy már létezik az l farkából az e elem kitörlésével kapott lista (saját maga rekurzív meghívásával), és így az l fejéből és az e-t már biztosan nem tartalmazó listából létrehozza az eredményt. 3 pont

C. N (E (M (E (S (+ (T (I (H (A (M (É (R)))))))))))) 1 pont

D. (N (M (E (S)))) 2 pont

Elérhető összpontszám: 126 pont

1992. Második forduló

Első-második osztályosok

Demográfia:

Legyen N a populáció egyedszáma, K pedig korcsoportjainak száma! Egyszerű esetben -és mi ezt vizsgáljuk- a korcsoportok egyenlő időtartamot ölelnek fel, méghozzá 1 évet. T(N)-ben tároljuk a modell objektumainak, az egyes egyedeknek a jellemző tulajdonságát: a kort. H(I) jelenti azt, hogy valaki I éves korában milyen valószínűséggel hal meg. Az I éves nyulaknak M(I)±E(I) utódja születik.

A modell algoritmusának első változata:

```
Szimulációs lépés:
  J:=N                    {a jelenlegi utolsó egyed}
  Ciklus i=1-től N-ig
    Ha RND<H(T(i)) akkor T(i):=0              {halál}
    különben Darab:=Véletlen(M(T(i))-E(T(i))..M(T(i))+E(T(i)))
             Ciklus ii=1-től Darab-ig
               J:=J+1; T(J):=1                {születés}
             Ciklus vége
             T(i):=T(i)+1                  {öregedés}
    Elágazás vége
  Ciklus vége
  T-táblázat tömörítése
Eljárás vége.
```

Most nézzük meg, hogyan hatnak a populációra a járványok! Legyen JV annak a valószínűsége, hogy egy évben van járvány! Járvány esetén az egyes korcsoportokban mások a halálozási ráták. Ebben az esetben JH(K) tárolja a halálozási rátákat, s a járvány algoritmusa, amit a fő szimulációs eljárásban a táblázat tömörítése előtt kell meghívni:

```
Járvány:
  Ha RND<JV akkor Ciklus i=1-től N-ig
                    Ha véletlenszám<JH(T(i)) akkor T(i):=0
                  Ciklus vége
  Elágazás vége
Eljárás vége.
```

A táblázat tömörítés kihagyja a táblázatból a 0 értékeket, s közben számolhatjuk, hogy melyik korcsoportból hány nyulunk van (DB vektor) és összesen hány nyulunk van (ODB):

```
T-táblázat tömörítése:
  ii:=0; DB():=0
  Ciklus i=1-től J-ig
    Ha T(i)>0 akkor ii:=ii+1; T(ii):=T(i); DB(T(ii)):=DB(T(ii)+1
  Ciklus vége
  N:=ii; ODB:=szumma(DB); Ha ODB>1000 akkor Csökkentés
Eljárás vége.
```

Ezután megoldjuk, hogy az összlétszám ne nőhessen 1000 fölé:

```
Csökkentés:
  Szorzó:=(ODB-1000)
  Ciklus i=1-től K-ig
    CS(i):=[DB(i)/ODB*Szorzó]
  Ciklus vége
  Ciklus i=1-től N-ig
     Ha CS(T(i))>0 akkor CS(T(i)):=CS(T(i))-1; T(i):=0
  Ciklus vége
  T-táblázat tömörítése
Eljárás vége.
```

Értékelési szempontok:

Adatbeolvasás		12 pont
Kezdőállapot beolvasása	3 pont	
Halálozási paraméterek beolvasása	3 pont	
Születési paraméterek beolvasása	3 pont	
Járvány paraméterek beolvasása	3 pont	
Beolvasások 3 pontjának megoszlása:		
van:	1 pont	
kérdés:	1 pont	
ellenőrzés:	1 pont	
A. (alapszimuláció)		21 pont
Halálozás jó megoldása	7 pont	
Születés jó megoldása	7 pont	
Öregedés jó megoldása	7 pont	
B.(járvány)		12 pont
Járvány véletlenszerűen tör ki	5 pont	
Járvány esetén mások a halálozási ráták	7 pont	
C. (létszámtúllépés)		10 pont
Túllépés felismerése	3 pont	
Túllépés megszüntetése	2 pont	
Arányos megszüntetés	5 pont	
Eredmény		25 pont
Korcsoporteloszlás grafikon van	6 pont	
képernyőre/ablakra normált grafikon	2 pont	

skálabeosztás, feliratok	2 pont	
Populációlétszám grafikon van	6 pont	
képernyőre/ablakra normált grafikon	2 pont	
időbeli betelés problémájának megoldása	5 pont	
skálabeosztás, feliratok	2 pont	
Programminőség:		20 pont
Értelmesen sorokra tördelt programszöveg:	2 pont	
Bekezdéses leírás a struktúrák elejének, végének jelzésével:	4 pont	
Programegységek tagolása (elválasztó sorok):	2 pont	
Eljárások alkalmazása:	4 pont	
Megjegyzések, magyarázatok:	4 pont	
Beszédes azonosítók:	2 pont	
Állandók (szinonimák) alkalmazása:	2 pont	

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

Petri-gráf vizsgálata: (100 pont)

Az első részfeladat a Petri gráf bolvasása, ellenőrzésekkel. Ehhez először dönteni kell az ábrázolásról. Szükségünk lesz arra, hogy egy csomóponthoz van-e kötve egy átmenet, illetve fordítva, valamint a csomópontok és átmenetek maximális (MaxCs,MaxÁt) és aktuális (CsDB,ÁtDB) számára.

```
CS: Tömb(1..MaxCs,1..MaxÁt,Logikai)
ÁT: Tömb(1..MaxÁt, 1..MaxCs,Logikai)
```

```
Gráf beolvasás:
  Be: CsDB,ÁtDB
  Ciklus amíg van adat
    Be: ADAT; X=ADAT csomópont része,Y=ADAT átmenet része
    L:=előbb volt-e a csomópont rész
    Ha L akkor Ha CS(X,Y) akkor HIBA különben CS(X,Y):=igaz
      különben Ha ÁT(Y,X) akkor HIBA különben ÁT(Y,X):=igaz
  Ciklus vége
  Független csomópont ellenőrzés(HIBA)
  Bemenet nélküli átmenet ellenőrzés(HIBA)
  Kimenet nélküli átmenet ellenőrzés(HIBA)
Eljárás vége.
```

Szükségünk lesz az egyes csomópontokban levő bigyók számára, amelyet kezdetben beolvasással töltünk fel.

```
CSOMÓ: Tömb(1..MaxCs,Egész)
```

A működés a feladat szerint legfeljebb 8 időegységig tart, illetve hamarabb befejeződhet, ha holtpont helyzetbe kerülünk. Egy lépésben ki kell válogatni az aktiválható átmeneteket, majd közülük egyet kell véletlenszerűen választani:

```
Működés:
  i:=1; HOLTPONT:=hamis
  Ciklus amíg i≤8 és nem HOLTPONT
    Aktiválható átmenetek kiválogatása(AktDB,AktÁt)
    Ha AktDB=0 akkor HOLTPONT:=igaz
            különben X:=AktÁt(véletlen(AktDB)); i:=i+1
                     Változtatás(X)
  Ciklus vége
Eljárás vége.
```

Egy átmenet aktivizálódásának az a feltétele, hogy az őt közvetlenül megelőző csomópontokban legyen bigyó.

```
Aktiválható átmenetek kiválogatása(AktDB,AktÁt):
  AktDB:=0
  Ciklus x='A'-tól ÁtDB-ig
    Ha Aktiválható(x) akkor AktDB:=AktDB+1; AktÁt(AktDB):=x
  Ciklus vége
Eljárás vége.
```

```
Aktiválható(x):
  i:=1
  Ciklus amíg i≤CsDB és (nem ÁT(x,i) vagy CSOMÓ(i)≠0)
    i:=i+1
  Ciklus vége
  Aktiválható:=(i>CsDB)
Függvény vége.
```

Egy átmenet működtetése azt jelenti, hogy bigyókat kell elvenni csomópontokból, illetve hozzáadni csomópontokhoz:

```
Változtatás(X):
  Ciklus i=1-től CsDB-ig
    Ha CS(i,X) akkor CSOMÓ(i):=CSOMÓ(i)-1
  Ciklus vége
  Ciklus i=1-től CsDB-ig
    Ha ÁT(X,i) akkor CSOMÓ(i):=CSOMÓ(i)+1
  Ciklus vége
Eljárás vége.
```

Nehezebb feladat annak megállapítása, hogy egy állapot elérhető-e egy másik állapotból. Ehhez gráfbejáró algoritmust kell használnunk. A szélességi bejárást választottuk, amihez egy sorra lesz szükségünk.

```
Elérhetőség(CSOMÓ,CÉL):
  Sorba(CSOMÓ,0); H:=0
  Ciklus amíg a Sor nem üres és CÉL∉SOR és H<8
    Sorból(CS,H)
    Aktiválhatók átmenetek kiválogatása(AktDB,AktÁt)
    Ha AktDB>0 akkor Aktiválhatók feldolgozása(AktDb,AktÁt,CS,H)
  Ciklus vége
Eljárás vége.
```

Az aktiválhatók feldolgozásában figyelni kell a ciklusba vezető átmenetekre, azokat nem kell többször is bejárni.

```
Aktiválhatók feldolgozása(AktDb,AktÁt,CS,H):
  Ciklus i=1-től AktDB-ig
    Változtatás(AktÁt(i))
    Ha CS∉SOR akkor Sorba(CS,H+1) különben CIKLUS
  Ciklus vége
Eljárás vége.
```

Tesztesetek:

Háló-1: 3 csomópont, 1 átmenet (holtpont egy lépésben)

"a" átmenetbe bemegy a nyíl: "1" csomópontból, ahol van bigyó
"a" átmenetbe bemegy a nyíl: "2" csomópontból, ahol nincs bigyó
"a" átmenetből kimegy a nyíl: "3" csomópontba

Háló-2: 3 csomópont, 1 átmenet (holtpont két lépésben)

"a" be: 1, 2, ki: 2, 3

Háló-3: 1 csomópont, 1 átmenet (ciklus)

"a" be: 1, ki: 1

Háló-4: 2 csomópont, 1 átmenet (ciklus lehetősége)

a" be: 1, ki: 1, 2

Háló-5: 5 csomópont, 4 átmenet (ez a 3. példa hálózata)

"a" be: 1, ki: 2
"b" be: 1, ki: 3
"c" be: 2, ki: 1, 5
"d" be: 3, ki: 1, 4

Háló-6: 5 csomópont, 8 átmenet

"a" be: 1, ki: 2
"b" be: 2, ki: 1
"c" be: 2, ki: 3
"d" be: 3, ki: 1
"e" be: 3, ki: 4
"f" be: 4, ki: 1
"g" be: 4, ki: 5
"h" be: 5, ki: 1

Elérhető-e az 10000 állapotból a 00010 állapot ? (igen)
Elérhető-e a 00001 állapotból az 10010 állapot ? (nem)
Elérhető-e a 01110 állapotból a 00300 állapot ? (igen)

Háló-7: 4 csomópont, 4 átmenet

"a" be: 1, 2, ki: 3
"b" be: 3, ki: 1, 2
"c" be: 3, ki: 4
"d" be: 4, ki: 3

Elérhető-e az 1100 állapotból a 0001 állapot ? (igen)
Elérhető-e az 1100 állapotból a 0020 állapot ? (nem)
Elérhető-e az 1111 állapotból a 3300 állapot ? (igen)

Háló-8: 4 csomópont, 4 állapot

"a" be: 1, ki: 2, 3
"b" be: 1, 3, ki: 4
"c" be: 4, ki: 3
"d" be: 2, ki: 1

Elérhető-e az 1000 állapotból a 0010 állapot ? (igen)
Elérhető-e a 0010 állapotból az 1000 állapot ? (igen)
Elérhető-e az 1111 állapotból a 0002 állapot ? (igen)

Háló-2 – Háló-8-ban feltesszük, hogy kezdetben minden bemenő csomópontban van bigyó.

Értékelési szempontok:

1. feladat:	23 pont
1.1. A hálózat szerkezete megadható, a csomópontokat számjegyek (1, 2, ...), az átmeneteket betűk (a, b, c, ...) jelölik:	4+1+1 pont
1.2. A tárolt gráf kiíratható:	4 pont
1.3. Megadható a kezdőállapot (melyik csomópontban hány bigyó van):	2 pont
1.4. Az előző kérdésekre nem lehet értelmetlen adatot megadni (pl. negatív csomópontszám, negatív bigyószám, nem létező helyre irányuló nyíl, független csomópont, átmenet bemenő nyíl nélkül, átmenet kimenő nyíl nélkül):	1+1 pont
1.5. Nem lehet a feltételnek ellentmondó hálózatot megadni (azaz nem lehetnek párhuzamos nyilak és 1-nél több bigyó a kezdeti állapotban):	3+2 pont
2. feladat:	27 pont
2.1. A szimuláció működik, az egyes állapotokat kijelzi:	10+5 pont
2.2. A versenyhelyzeteket kezeli, azaz az egyidejűleg aktivizálható átmenetek közül véletlenszerűen választ:	5 pont
2.3. Végtelen működési sorozat esetén a 8. állapot után a szimuláció leáll:	2 pont
2.4. A holtpontot – a 8. lépés megtétele előtt már nincs aktivizálható átmenet – észreveszi és kijelzi:	5 pont
3. feladat:	30 pont
3.1. A versenyző az adott hálózathoz és kezdőállapothoz előállítja és kiírja a 8 lépésben elérhető állapotok halmazát:	15 pont
3.2. Észreveszi, ha egy már korábban bejárt állapotba jut: (ciklus–1)	5 pont
3.3. Észreveszi, ha olyan állapotba jut, amely egy előzőleg már figyelembevett állapottól csak abban különbözik, hogy egyik csomópontban sincs kevesebb bigyó: (ciklus–2)	5 pont
3.4. Egy adott állapot elérhetőségét jól állapítja meg:	5 pont
Programminőség:	20 pont
Értelmesen sorokra tördelt programszöveg:	2 pont
Bekezdéses leírás a struktúrák elejének, végének jelzésével:	4 pont
Programegységek tagolása (elválasztó sorok):	2 pont
Eljárások alkalmazása:	4 pont
Megjegyzések, magyarázatok:	4 pont
Beszédes azonosítók; állandók (szinonimák) alkalmazása:	2+2 pont

Elérhető összpontszám: 100 pont

1993. Első forduló

Első-második osztályosok

1. feladat: Szövegelés (10 pont)

A. elj1 szoveg1 megfordítottjához fűzi szoveg2-t. 5 pont

B. elj2 megfordítja az átadott szöveget. 5 pont

2. feladat: Portia ládikái (13 pont)

A. Pontosan egy ládikán van igaz állítás. 4 pont

B. legalább_két_igaz(X) ha ezüstládikán(X) és ólomládikán(X) vagy
aranyládikán(X) és ólomládikán(X) vagy
aranyládikán(X) és ezüstládikán(X).

páronként 1-1-1 pont

a három részállítás közötti 'vagy' kapcsolatra 1 pont

legalább_egy_egy(X) ha (aranyládikán(X) vagy ezüstládikán(X) vagy
ólomládikán(X)) és nem(aranyládikán(X) és
ezüstládikán(X) és ólomládikán(X)).

hármasonként 2-2 pont

a két részállítás közötti 'és' kapcsolatra 1 pont

3. feladat: Táblázatkezelő (11 pont)

A. A2 = 0, B2 = 1, A3 = 1, B3 = 1 1 pont

A2 = n esetén B2 = n! 3 pont

A3 = n+1 1 pont

B3 = (n+1)! 1 pont

B. A1: ...

A3: Ha A1 = 0 akkor 1 különben A3+1 2 pont

B3: Ha A1 = 0 akkor 1 különben A3*B3 3 pont

4. feladat: Bitvektorok (7 pont)

A. Az A és a B bitvektorokban lévő számok összegét. 3 pont

B. Átvitel a következő, eggyel magasabb helyértékre. 2 pont

C. Az esetleges túlcsordulást jelzi: 0 — nem volt, 1 — volt. 2 pont

5. feladat: LOGO (10 pont)

A. Téglalap alapú mozaikot, 2 pont

B. amelynek N sora és M oszlopa van, 1+1 pont

C. az alaptéglalap oldalai X, illetve Y hosszúak; 1+1 pont

D. az Y hosszú oldalak középső harmadát kivágja, 2 pont

E. és két ugyanilyen hosszú, befelé fordított szakasszal helyettesíti; 2 pont

a végeredmény: téglalapmozaikban rombusz alakú kivágások.

6. feladat: Lnko (12 pont)

A. Helyesek: A), C); hibásak: B), D). 3+3+1+1 pont

B. B) nem adja meg a legnagyobb közös osztót, ha az éppen az egyik szám (A vagy B) lenne, 1 pont

C. egyébként jól működik. 1 pont

D. a legnagyobb közös osztó reciprokát adja. 2 pont

7. feladat: Bagdadi tolvaj (10 pont)

A. Semmi, a program közömbös a számpárok sorrendjére. 1 pont

B. A program sorrendben az utolsó értékpárt veszi figyelembe. 2 pont

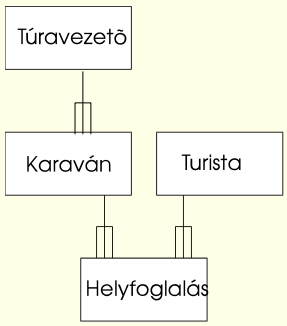
C. A program 0-nak veszi az aranyak számát a hiányzó emeleteken. 1 pont

D: A 4 helyett 2 kell; 1+1 pont

rosszul írtuk át programmá a 6*(felsoEmelet–1)+10≤60*(tuzKezdete – felsoEmelet) kifejezést. 2 pont

E. A (30*32–2) DIV 33 = 29-en. 2 pont

8. feladat: Karaván (7 pont)



A. "Túravezető—Karaván" kapcsolat berajzolása 3 pont

B. "Karaván" megnevezés beírása 2 pont

C. "Turista" megnevezés beírása 2 pont

Elérhető összpontszám: 80 pont

Harmadik-ötödik osztályosok

1. feladat: Biokertészet (13 pont)

A. Egy lehetséges megoldás:

szomszéd(X,Y) ha védi(X,Y) vagy szereti(X,Y) vagy (vethető(X) és vethető(Y) és X<>Y és nem(ellenség(X,Y))).

Egy másik lehetséges megoldás:

szomszéd(X,Y) ha (védi(X,Y) vagy szereti(X,Y) vagy nem(ellenség(X,Y))) és vethető(X) és vethető(Y) és X<>Y.

Részpontszámok:

ha szerepel benne a védi(X,Y) 1 pont

ha szerepel benne a szereti(X,Y) 1 pont

ha szerepel benne a nem(ellenség(X,Y)) 1 pont
ha a 'vagy' műveletet alkalmazza közöttük 1 pont
ha szerepel benne a vethető(X) és vethető(Y), 'és' kapcsolatban 2 pont
ha szerepel benne az X<>Y, 'és' kapcsolatban 2 pont

B. hagyma van a szélen, mert csak egy szomszédja lehet 1 pont
hagyma, paradicsom, karalábé, bab, retek 2 pont
hagyma, paradicsom, retek, bab, karalábé 2 pont
fordított sorrendért nem jár külön pont; ha hibás sorrendet is megad, darabonként 1-1 pont levonás

2. feladat: Sorozatok (16 pont)

A. példa1 = második (a sorozat második elemét adja eredményül) 1 pont
példa2 = eleme (eldönti, hogy az E elem benne van-e az S sorozatban) 2 pont

B. utolsó(S): Ha ürese(elsőutániak(S)) 1 pont
akkor első(S) 1 pont
különben utolsó(elsőutániak(S)) 2 pont
utolsóelőttiek(S): Ha ürese(elsőutániak(S)) 1 pont
akkor üres 1 pont
különben elejére(első(S), 1 pont
utolsóelőttiek(elsőutániak(S))). 2 pont
végére(S,E): Ha ürese(S) akkor elejére(E,S) 1 pont
különben elejére(első(S), 1 pont
végére(elsőutániak(S),E))). 2 pont

3. feladat: Táblázatkezelő (9 pont)

A. Az X*X–4 függvény 2 pont
egy zérushelyét keresi meg 2 pont
intervallumfelező módszerrel 2 pont
a [0,9] intervallumban 1 pont

B. 2 (kettő) 2 pont

4. feladat: Számvektorok (15 pont)

A. ?1?:0 (nulla) 1 pont
B. ?2?: N+M 2 pont
C. ?3?: min(I,N) 4 pont
D. ?4?: C(I) DIV 10 2 pont
E. ?5?: C(I) MOD 10 2 pont
F. ?6?: ÁTVITEL ≠ 0 2 pont
G. ?7?: ÁTVITEL 2 pont

5. feladat: Csak formálisan! (17 pont)

A. „; két jel között nincsen szóköz 1 pont

8szavas egybeírva nem szám 1 pont

: ismeretlen jel 1 pont

ez, a szót nem szóköz zárja 1 pont

- ismeretlen jel 1 pont

B. <szám> ::=<előjel><egész szám> 2 pont

|<előjel><egész szám>.<egész szám> 2 pont

<előjel> ::=+|– 1 pont

<egész szám> ::= {<számjegy>} 1 pont

```
Szóolvasás(P,i,SZO):
  SZO:=P(i): i:=i+1
  Ciklus amíg P(i)≠" "
     SZO:=SZO+P(i): i:=i+1
  Ciklus vége
Eljárás vége.
```

3 pont

A Számolvasás ugyanez. 2 pont

```
Írásjelolvasás(P,i,IR):
  IR:=P(i): i:=i+1
Eljárás vége.
```

1 pont

6. feladat: Turing-gép (12 pont)

A. ...00011111111111100000... = 12 3 pont

B. Az 1-esek utáni első 0-n. 2 pont

C. A két nemnegatív egész szám összege. 4 pont

Ha csak annyit mond, hogy összefűzi a két 1-esekből álló sorozatot, akkor 4 pont helyett csak 2 pont adható.

D. Az eredeti szám. 2 pont

A fej most is az 1-esek utáni első 0-n áll. 1 pont

7. feladat: Assembly (12 pont)

A. Egy jel rövid, ha időtartama közelebb esik az utolsó rövid jel idejéhez, és hosszú, ha az utolsó hosszú jel idejéhez esik közelebb. 4 pont

B. A C1-es ciklus addig vár, amíg a program jelet érzékel; leállásakor BX-ben a jel hosszával arányos érték van. 2 pont

C. rövid jel esetén DL = 0 1 pont

hosszú jel esetén DL = 1 1 pont

D. a JS C2-t hajtja végre, ha a jel hosszabb volt a legutóbbi hosszú jelnél 1 pont

a JG C2-t hajtja végre, ha a jel rövidebb volt a legutóbbi hosszú jelnél, de közelebb esik hozzá, mint a legutóbbi rövid jelhez 1 pont

E. rövid jel esetén P = BX 1 pont

hosszú jel esetén Q = BX 1 pont

8. feladat: Monoton (6 pont)

A. h-ban a b-beli leghosszabb fennsík hossza van, ahol fennsíknak az egymással azonos szomszédos értékek egy sorozatát nevezzük 2 pont

B. Egy lehetséges megoldás 4 pont

```
  VAR i,h: INTEGER;
BEGIN
  i:=2; h:=1;
  WHILE i<=n DO
     BEGIN
        IF (b[i-h]=b[i]) THEN h:=h+1;
        i:=i+1
     END
END.
```

Elérhető összpontszám: 100 pont

1993. Második forduló

Első-második osztályosok

Gyűrűhálózat szimuláció:

A feladat megoldását a szükséges adatszerkezetekkel kezdjük. Kell az üzenetek típusa, a generált üzenetek (küldési idő szerint növekvő sorrendben), a hálózatban körbemenő üzenetek, valamint az egyes gépek által tárolt legutolsó 5 üzenet:

```
Típus Üzenet=Rekord(címzett,feladó: Egész,
                    fajta: (normál,utolsó), sorszám: Egész
                    tartalom: Szöveg(8), ell: karakter)

Generált: Tömb(1..MaxGen,Rekord(mikor: Egész, üz: Üzenet))

Háló: Tömb(1..MaxN,Üzenet)

Tárolt: Tömb(1..MaxN,1..5,Üzenet)
```

Legyen az üres üzenet olyan, hogy a címzettje a 0 sorszámú gép. A tömböket kezdetben töltsük fel üres üzenetekkel!

A szimuláció időegységenkét lép egyet, két fő részből áll, a hálózatban levő üzenetek körbeléptetéséből, valamint az új üzenetek beléptetéséből. A körbeléptetésnél kell figyelni arra, hogy az üzenet annak szól-e, akihez éppen került.

```
Körbeléptetés:
  T:=Háló(N)
  Ciklus i=N-1-től 1-ig -1-esével
    Ha Háló(i).címzett=i+1 akkor Elvesz(i+1)
                           különben Háló(i+1):=Háló(i)
  Ciklus vége
Eljárás vége.
```

Az üzenet elvételénél a tárolt legutóbbi 5 üzenetet léptetni kell:

```
Elvesz(j):
  Ciklus k=1-től 4-ig
    Tárolt(j,k):=Tárolt(j,k+1)
  Ciklus vége
  Tárolt(5):=Háló(j-1); Háló(j-1):=üres üzenet
Eljárás vége.
```

Minden generált üzenetet végig kell nézni (1-től GenDB-ig), s amelyiket az adott időpontban kell indítani, azt betesszük a hálózatba, ha a küldőnél nincs éppen üzenet. Ha van, akkor a küldési időpontot eggyel növeljük (azaz késleltetjük az új üzenet elküldését):

```
Új üzenet belépés(idő):
  Ciklus i=1-től GenDb-ig
    Ha Generált(i).mikor=idő
       akkor U:=Generált(i).üz
             Ha Háló(U.feladó)=üres üzenet
                akkor Háló(U.feladó):=U
             különben Generált(i).mikor:=Generált(i).mikor+1
  Ciklus vége
Eljárás vége
```

A feladat többi részfeladata elemi tevékenységeket tartalmaz (ellenőrző összeg generálása, ellenőrzése, statisztikák készítése, forgalom megjelenítése, …) emiatt ezek megvalósításával itt nem foglalkozunk.

Értékelési szempontok:

A. Kezdőbeállítás:		13 pont
- választ a véletlenszerű és a felhasználói között	1 pont	
- van a felhasználói megadás	1 pont	
- van a véletlenszerű megadás	1 pont	
- maximum 30 időegységre lehet	1 pont	
- mindenki küldhet	1 pont	
- mindenki egyvalakinek küldhet	1 pont	
- generál címzettet (nem önmaga)	2 pont	
- generál csomagsorszámot (növekvő)	2 pont	
- ez maximum 5 lehet	1 pont	
- lehet több csomagos üzenet(vannak típusok)	2 pont	
B. A forgalom követése:		25 pont
- megjeleníti az egyes szakaszokon levő üzeneteket	5 pont	
- gyűjti a gépekhez érkezett üzeneteket	5 pont	
- az utolsó 5-öt őrzi meg	2 pont	
- az üzenetek mozognak	4 pont	
- a küldő jó helyre teszi	1 pont	
- nem teszi oda, ha foglalt	1 pont	
- a küldőnél a küldés után eltűnik	2 pont	
- a vevő elveszi	2 pont	
- a vevő beteszi a kapottjai közé	2 pont	
- más nem veszi el	1 pont	
C. Hibagenerálás és -figyelés:		13 pont
- generál ellenőrző összeget	1 pont	
- jó az ellenőrző összeg	4 pont	
- véletlenszerűen elrontja az üzeneteket	3 pont	

- ellenőrzi az üzenetek helyességét	5 pont	
D. Futás végi statisztika:		12 pont
- statisztikát ad az észlelt hibákról	4 pont	
- szakaszonként adja	3 pont	
- az egyes szakaszok foglaltsága	5 pont	
E. Felhasználói kapcsolat minősége:		12 pont
- barátságos választás	3 pont	
- időpont beolvasás (barátságos, biztonságos)	3 pont	
- üzenet beolvasás		
- hány csomag	1 pont	
- ki	1 pont	
- kinek	1 pont	
- szöveg	1 pont	
- a hibázás valószínűség beolvasás	1 pont	
- idő kijelzés	1 pont	
F. Programminőség		15 pont
- értelmes eljárásokra tagolás	3 pont	
- bekezdéses struktúra	3 pont	
- sorokra tagolás	3 pont	
- sorok közötti tagolás	3 pont	
- beszédes azonosítók	3 pont	

Elérhető összpontszám: 90 pont

Harmadik-ötödik osztályosok

Szövegformázó:

A fő feladat a szöveg lapokra, azon belül hasábokra, azon belül pedig sorokra tördelése. Mivel szavakat nem törhetünk két sorra, ezért célszerű megírni a szóolvasást. Így a program felső szintje a következő lehet:

```
Tördelés:
  Nyit(f); Lapnyitás(g); Vége:=igaz; Igazít:=balra
  Ciklus amíg nem vége(f)
    Ha vége akkor Beavatkozás
    Szóolvasás(f,SZÓ,Vége); Lapraírás(g,SZÓ)
  Ciklus vége
  Zár(f); Lapzárás(g)
Eljárás vége.

Szóolvasás(f,SZÓ,Vége):
  Olvas(f,C); SZÓ:=''
  Ciklus amíg C≠' ' és C≠sorvég
    SZÓ:=SZÓ+C; Olvas(f,C)
  Ciklus vége
  Vége:=(C=sorvég)
Eljárás vége.
```

A lapon legyen két hasáb, s mindegyikben LapDB darab sor, a sorok maximális hossza legyen MaxHossz. A lap definíciója az alábbi lehet:

```
Lap: Rekord(AktHasáb,AktSor: Egész, BMargó,JMargó: Egész,
            Sor: Tömb(1..2,1..LapDB,Szöveg(MaxHossz)
```

A Lapnyitás eljárás beállítja a kezdő változókat:

```
Lapnyitás(g):
  Lap.AktHasáb:=1; Lap.AktSor:=1
  Lap.BMargó:=1; Lap.JMargó:=MaxHossz
  Nyit(g)
Eljárás vége.
```

A Lapraírás eljárás elhelyezi a következő lapon az aktuális szót.

```
Lapraírás(g,SZÓ):
  Ha hossz(SZÓ)+hossz(Lap.Sor(AktHasáb,AktSor))+1≤
     Lap.JMargó-Lap.BMargó+1
  akkor Lap.Sor(AktHasáb,AktSor):=Lap.Sor(AktHasáb,AktSor)+' '+SZÓ
  különben Ha AktSor<LapDB akkor AktSor:=AktSor+1
           különben AktSor:=1
                    Ha AktHasáb=1 akkor AktHasáb:=2
                    különben Lapkiírás(g,Lap); AktHasáb:=1
           Lap.Sor(AktHasáb,AktSor):=Szóközök(BMargó-1)+SZÓ
Eljárás vége.
```

A Lapzárás eljárás kiírja az utolsó lapot, majd lezárja az output file-t.

```
Lapzárás(g):
  Lapkiírás(g,Lap); Zár(g)
Eljárás vége.
```

A fentiek jól működnek, ha csak kimenő állományba kell írni az eredményt, a képernyőt azonban nem célszerűen kezelik, mert csak akkor írnak rá, ha megtelt a lap. Úgy kell módosítani a Tördelés eljárást, hogy minden bekezdés beolvasása után frissítse a képernyőt! Ehhez a ciklusa vége előtt az alábbi utasításra van szükség:

```
Ha Vége akkor Képernyőreírás
```

A következő feladat a formázás. Ehhez tárolni kell, hogy az aktuális bekezdés mettől meddig tart a képernyőn. Ezekre a sorokra kell meghívni a megfelelő margóhoz igazítás eljárást.

```
Balraigazít(h,s):
  Ciklus amíg hossz(Lap.Sor(h,s))<Lap.JMargó-Lap.Bmargó+1
    Lap.Sor(h,s):=Lap.Sor(h,s)+' '
  Ciklus vége
Eljárás vége.

Jobbraigazít(h,s):
  Ciklus amíg hossz(Lap.Sor(h,s))<Lap.JMargó-Lap.BMargó+1
    Lap.Sor(h,s):=' '+Lap.Sor(h,s)
  Ciklus vége
Eljárás vége.

Középreigazít(h,s):
  Ciklus amíg hossz(Lap.Sor(h,s))<Lap.JMargó-Lap.BMargó+1
    Lap.Sor(h,s):=' '+Lap.Sor(h,s)+' '
  Ciklus vége
Eljárás vége.
```

A mindkét margóhoz igazítás kicsit nehezebb feladat. Itt le kell számolni, hogy a sorban hány szó van, majd közöttük egyenletesen kell elhelyezni a megfelelő számú szóközt. A szavakra tördeléshez

az ElsőSzó nevű eljárást használjuk, ami lényegében azonos a Szóolvasás eljárással, csupán file helyett a Lap.Sor(h,s) tömbből olvas

```
MargóhozIgazít(h,s):
  kell:=Lap.JMargó-Lap.BMargó+1-Hossz(Lap.Sor(h,s))
  hely:=Szószámlálás(Lap.Sor(h,s))-1
  kellplusz:=kell MOD hely; kell:=kell DIV hely
  ElsőSzó(Lap.Sor(h,s),szó); t:=Szóközök(Lap.BMargó-1)+szó+' '
  Ciklus i=1-től hely-ig
    Ha i≤kellplusz akkor t:=t+Szóközök(kell+1)
                    különben t:=t+Szóközök(kell)
    ElsőSzó(Lap.Sor(h,s),szó); t:=t+szó+' '
  Ciklus vége
  Lap.Sor(h,s):=t
Eljárás vége.
```

Utolsó részfeladatként szerepelt a rejtett elválasztások kezelése. Ezt úgy célszerű megoldani, hogy a rejtett elválasztás legyen szóhatár. Utána a sorban nem szabad szóközt tennünk. Ha a következő szó még kifér az aktuális sorba, akkor a rejtett elválasztást ki kell hagyni a sorból.

Értékelési szempontok:

A. Tördelés:		20 pont
- szavak helyes sorokra tördelése	8 pont	
- hasábokba írás	5 pont	
- oszlopfolytonos hasábkitöltés	4 pont	
- lapozás a képen előre(betelés után várakozás)	3 pont	
B. Bekezdések formázása:		30 pont
- baloldali margóhoz igazítás	1 pont	
- ez a feltételezett érték	1 pont	
- jobboldali margóhoz igazítás	3 pont	
- középre igazítás	3 pont	
- mindkét margóhoz igazítás bárhogyan	3 pont	
- szóközök egyenletes elhelyezése	8 pont	
- egymás alatti sorokban egymás alatti szóközök	8 pont	
- az előző formázás alapján formázás	3 pont	
C. Margók állítása:		8 pont
- balmargó az 1. hasábra	2 pont	
- balmargó a 2. hasábra	2 pont	
- jobbmargó az 1. hasábra	2 pont	
- jobbmargó a 2. hasábra	2 pont	
D. Rejtett elválasztás:		12 pont
- rejtett elválasztás nem látszik, ha nem kell	6 pont	
- rejtett elválasztásnál elválaszt	6 pont	
E. Input-output:		15 pont
- beolvasás file-ból,	2 pont	
- kiírás file-ba,	2 pont	

- kiírás képernyőre,	2 pont	
- a képernyőre kiírt újraformázása	3 pont	
- többször is tudja	1 pont	
- filenevek beolvasása(minőség)	2 pont	
- az input file létezésének ellenőrzése	2 pont	
- újraformázás funkcióbillentyűi látszanak	1 pont	
F. Programminőség		15 pont
- értelmes eljárásokra tagolás	3 pont	
- bekezdéses struktúra	3 pont	
- sorokra tagolás	3 pont	
- sorok közötti tagolás	3 pont	
- beszédes azonosítók	3 pont	

Elérhető összpontszám: 100 pont

1994. Első forduló

Első-második osztályosok

1. feladat: Az óra csak jár (14 pont)

A. Első: elindítja a stoppert — 1 pont
   Második: leállítja a stoppert — 1 pont
   Harmadik: továbbindítja a stoppert — 1 pont

B. Azért használunk tömböket, hogy a programunk egyszerre több stoppert is szimulálhasson. — 1 pont
   K(): az utolsó indítás időpontja — 1 pont
   E(): a kezdéstől mért idő (ennyi időn keresztül járt a stopper, tehát nem számít bele a közbülső leállítás és újraindítás között eltelt idő) — 1 pont
   V(): az utolsó leállítás időpontja — 1 pont

C. Ha a leállításban bármelyik különbség negatív lenne, hibás eredményt kapunk. — 4 pont
   Javítása: kivonáskor átvitelszámítás. — 3 pont

2. feladat: Táblatalálgatás (18 pont)

A. A1: Négyzetgyök(X) egészrésze. — 3 pont
   A2: 2*A1 + 1 — 1 pont
   A3: (A1+1) négyzete — 2 pont

B. A1: X*Y — 4 pont
   A2: X*2^k, ahol 2^k az Y-nál nem nagyobb, legnagyobb kettőhatvány — 2 pont

C. A1: X^Y — 4 pont
   A2: X^(2^k), ahol 2^k=Y-nál nem nagyobb, legnagyobb kettőhatvány — 2 pont

3. feladat: Flip-flop (18 pont)

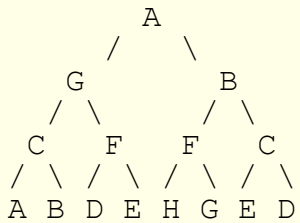
Ha több művelettel ír fel helyes táblázatot eredményező képletet, felesleges műveletenként 2-2 pont levonás, de egyik részfeladatra sem adható negatív pontszám.

A. y_{n+1}=(y_n és nem R) vagy S — 8 pont

B. y_{n+1} = (y_n és nem K) vagy (J és nem y_n) — 10 pont

4. feladat: Fabejárás (10 pont)

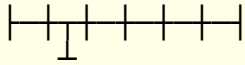
A.

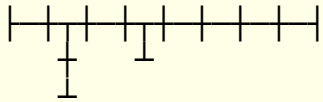


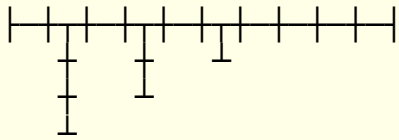
5 pont

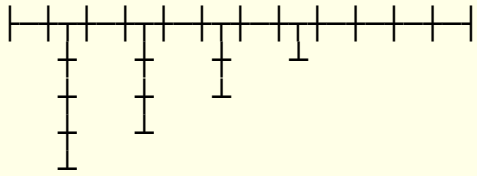
B. A C B G D F E A H F G B E C D — 5 pont

5. feladat: Moszat (10 pont)

`33(9)3750:`  1 pont

`33(39)33(9)3710:`  2 pont

`33(339)33(39)33(9)3210:`  3 pont

`33(3339)33(339)33(39)33(4)3210:`  4 pont

6. feladat: Rajzőrület (12 pont)

A. Álló négyzetet rajzol és visszatér a kiindulási helyzetébe. 3 pont

B. Fekvő négyzetet rajzol (Vagy: a kocka alaplapját). 2 pont

És minden éle fölé egy álló négyzetet. 2 pont

C. Kockát rajzol, 4 pont

melynek az oldalhossza A. 1 pont

7. feladat: Egyiket a másik után (18 pont)

A. Meghatározza egy sorozat N. és utána következő elemeit. 4 pont

B. Meghatározza egy sorozat A-adik és B-edik elemek közé eső elemeit. 4 pont

Az A-adik és a B-edik elem is benne lesz. 3 pont

A > B esetén ez üres sorozat lesz. 3 pont

```
C.Nedik(N,X):
Ha N=1  akkor első(X)
     különben Nedik(N-1,elsőutániak(X))
Függvény vége.
```

4 pont

Egy másik megoldás:

```
Nedik(N,X):
  első(Nyissz(N,X))
Függvény vége.
```

Elérhető összpontszám: 100 pont

Harmadik-ötödik osztályosok

1. feladat: Nézd közelről (12 pont)

A. Keres egy olyan racionális számot (törtet), 2 pont

amely az X pozitív valós számot E-nél jobban közelíti meg, 1 pont

és a racionális szám tört alakjában a számláló és a nevező a legkisebb. 2 pont

Az S a számláló, az N a nevező. 1 pont

B. Valami kiírja, hogy a szám osztható-e 3-mal. 3 pont

A számol függvény kiszámolja a számjegyek összegét. 3 pont

2. feladat: Táblatalálgatás (14 pont)

A. A1: Négyzetgyök(X) egészrésze. 3 pont

A2: 2*A1 + 1 1 pont

A3: (A1+1) négyzete 2 pont

B. A1: X*Y 4 pont

A2: X*2^k, ahol 2^k az Y-nál nem nagyobb, legnagyobb kettőhatvány 2 pont

C. A1: X^Y 4 pont

A2: X^(2^k), ahol 2^k=Y-nál nem nagyobb, legnagyobb kettőhatvány 2 pont

D. A1: A2 1 pont

A2: A3

A3: A2 + A3

A cellák kezdőértéke: (A1=0, A2=0, A3=1) vagy (A1=0, A2=1, A3=1) 1 pont

3. feladat: Flip-flop (14 pont)

Ha több művelettel ír fel helyes táblázatot eredményező képletet, felesleges műveletenként 2-2 pont levonás, de egyik részfeladatra sem adható negatív pontszám.

A. y_{n+1}=(y_n és nem R) vagy S 8 pont

B. y_{n+1}=(y_n és nem K) vagy (J és nem y_n) 10 pont

4. feladat: Vektorvarázslat (15 pont)

A. Meghatározza az A vektorban az 1 és N közötti indexű elemek maximumát 4 pont

és minimumát. 4 pont

A maximum az A vektor utolsó, 1 pont

míg a minimum az első elemébe kerül. 1 pont

B. Közelítőleg (N/2)+(N/2)+(N/4)+(N/8)+(N/16)+... 4 pont

Ha ezen kívül megad egy közelítést, amely N ≤...≤ 2*N intervallumba esik (pl. (3/2)*N) 1 pont

Megjegyzés: a matematikailag bizonyítható minimális lépésszám (3/2)*N–2.

5. feladat: Logikusan (12 pont)

A. Eldönti, hogy a számparaméter természetes szám-e. 4 pont

B. csodaTudja eldönti, hogy a számparaméter prímszám-e. 4 pont

Részpontszám adható, ha megmondja, hogy csodaTudja igaz értéket ad X=2 esetén (1 pont), vagy olyan páratlan szám esetén, amire NaMi hamis (1 pont).

NaMi igaz, ha Y (valódi) osztója X-nek, 1 pont

vagy ha van Y-nál kisebb (valódi) osztója X-nek. 3 pont

6. feladat: Rajzőrület (15 pont)

A. Álló négyzetet rajzol és visszatér a kiindulási helyzetbe. 1 pont

B. Fekvő négyzetet rajzol (vagy: a kocka alaplapját),	1 pont
és minden éle fölé egy álló négyzetet.	1 pont
C. Kockát rajzol,	2 pont
melynek az oldalhossza A.	1 pont
D. Álló téglalapot rajzol.	1 pont
E. Fekvő szabályos sokszöget rajzol (vagy: a hasáb alaplapját),	1 pont
s föléjük álló téglalapokat.	1 pont
F. Szabályos sokszög alapú	1 pont
hasábot rajzol,	2 pont
melynek oldalszáma A,	1 pont
oldalhossza B,	1 pont
magassága C.	1 pont

Megjegyzés: a hasáb határesetben henger lesz.

7. feladat: Vagány vágányok (18 pont)

A. Az első vágányt választja,	3 pont
ha az szabad.	1 pont
Különben a másikat választja.	1 pont
Ha mindkettő foglalt, akkor vár, míg egyik szabaddá nem válik.	1 pont
B. Azt a vágányt választja, amelyikről régebben indultak,	3 pont
ha az szabad.	1 pont
Különben a másikat választja.	1 pont
Ha mindkettő foglalt, akkor vár, míg egyik szabaddá nem válik.	1 pont
C. A két vágányt felváltva adja,	3 pont
ha a sorra kerülő szabad.	1 pont
Különben a másikat választja (ha a sorra kerülő éppen foglalt).	1 pont
Ha mindkettő foglalt, akkor vár, míg egyik szabaddá nem válik.	1 pont

Elérhető összpontszám: 100 pont

1994. Második forduló

Első-második osztályosok

1. feladat: Prímtényezős felbontás (15 pont)

A megoldás ötlete, hogy a számot osszuk az első lehetséges prímszámmal, a 2-vel addig, amíg csak lehet, majd keressük meg a következő osztót (ami biztosan prím lesz):

```
Felbontás(N):
  i:=2; db:=0
  Ciklus amíg N>1
    Ciklus amíg (N mod i)>0
      i:=i+1
    Ciklus vége
    db:=db+1; Alap(db):=i; Kitevő(i):=0
    Ciklus amíg (N mod i)>0
      Kitevő(i):=Kitevő(i)+1; N:=N div i
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

1. Prímszámokra jó	2 pont
2. Prímhatványokra jó	3 pont
3. Különböző prímek szorzatára jó	2 pont
4. Tetszőleges egyéb számra jó	5 pont
5. 1-re jó	1 pont
6. Esztétikus kép	2 pont

(cím, kérdésszöveg, válasz)

2. feladat: Hamupipőke (15 pont)

A feladat megoldásában az N elem közül ki kell válogatnunk a különböző szavakat (lencseszíneket), majd meg kell számolnunk, hogy melyik hányszor fordul elő.

```
Lencsék:
  db:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤db és elem(i)≠lencse(j)
      j:=j+1
    Ciklus vége
    Ha j≤db akkor darab(j):=darab(j)+1
    különben db:=db+1; lencse(db):=elem(i); darab(db):=1
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A. 1 színt felismer	2 pont
B. 1 színt számol	2 pont
C. minden új színt felismer	3 pont
D. szomszédos színeket felismeri	3 pont

E. újra megjelenő színt felismeri 4 pont

F. Üres file-ra jó 1 pont

G. File-név beolvasása 1 pont

H. Esztétikus kép 2 pont

3. feladat: Bacilusok (15 pont)

A feladat a Fibonacci-féle számok előállítását jelenti, melyek definíció szerint az őket megelőző két Fibonacci-szám összegeként állnak elő.

```
Fibonacci(N):
  F(0):=1; F(1):=1
  Ciklus i=2-től N-ig
    F(i):=F(i-1)+F(i-2)
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A. A program 20-nál kisebb pozitív egészek beadására kiír egy eredménysorozatot. Ha csak az N. eredményt írja, és az előzőket nem, nem jár pont. (Ez a pont jár akkor is, ha a sorozat hibás értékeket tartalmaz.) 4 pont

B. A kiírt eredmény hibátlan. A kiírás minőségét itt nem értékeljük. Az eredmény ellenőrzéséhez nézzük meg a 20. órában a bacilusok számát, ez 10946 lesz. Ennél nagyobb N-ekkel nem kell próbálkozni, mert már túlcsordulás lehet, és nem volt követelmény az egész számábrázolás korlátait meghaladni. 6 pont

C. Esztétikus kép 2 pont

4. feladat: Labdarúgó válogatott (30 pont)

Ebben a feladatban leghosszabb intervallumokat kell megadni úgy, hogy a sorozat két szélén nekünk kell megadni az intervallumhatárt. Emiatt a sorozatot ki kell egészíteni egy-egy elemmel a két végén, hogy egységesen kezelhessük.

A feladat nehézsége a dátum típus kezelése (reláció, illetve eggyel növelés és csökkentés).

A megoldásban az eredmények leírására szolgáló típust használunk:

```
Típus Eredmény=Rekord(mikor: Dátum, lőtt,kapott: Egész)
```

A két részfeladatot szinte ugyanúgy kell megoldani, csupán az egyiknél az intervallumhatárokat a győztes, a másiknál pedig a győztes és döntetlen mérkőzések jelentik.

Itt az egyiket nézzük meg:

```
Leghosszabb nyeretlen(N,Mérk):
  Mérk(0).mikor:=csökkent(Mérk(1).mikor)
  Mérk(N+1).mikor:=növel(Mérk(1).mikor)
  Kezd:=Mérk(0).mikor; Maxkezd:=Kezd; Maxvég:=Kezd
  Ciklus i=1-től N+1-ig
    Ha Mérk(i).lőtt>Mérk(i).kapott
       akkor Vég:=csökkent(Mérk(i).mikor)
             Ha Vég-Kezd>Maxvég-Maxkezd
                akkor Maxkezd:=Kezd; Maxvég:=Vég
             Kezd:=növel(Mérk(i).mikor)
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A. felismer nyeretlen sorozatot	3 pont
B. felismer pontnélküli sorozatot	3 pont
C. felismeri a leghosszabb nyeretlen sorozatot	3 pont
D. felismeri a leghosszabb pontnélküli sorozatot	3 pont
E. felismer nyeretlen sorozatot az elején	1 pont
F. felismer pontnélküli sorozatot a végén	1 pont
G. napváltás előre	1 pont
H. napváltás visszafelé	1 pont
I. hónapváltás előre	1 pont
J. hónapváltás visszafelé	1 pont
K. február – március váltás előre, nem szökőévben	1 pont
L. március – február váltás visszafelé, nem szökőévben	1 pont
M. február – március váltás előre, szökőévben	1 pont
N. március – február váltás visszafelé, szökőévben	1 pont
O. évváltás előre	1 pont
P. évváltás visszafelé	1 pont
Q. elején váltás nélküli nap	1 pont
R. végén váltás nélküli nap	1 pont
S. nincs nyeretlen sorozat	1 pont
T. nincs pontnélküli sorozat	1 pont
U. File-név beolvasás, esztétikus kép	1+1 pont

Elérhető összpontszám: 75 pont + 25 pont az 1. fordulóból

Harmadik-ötödik osztályosok

1. feladat: Ékezetes betűk (15 pont)

A megoldás arra épül, hogy a képernyőn tudunk egyszerre két sorba írni. Párhuzamosan írunk mindkettőbe, a felsőbe majdnem mindig szóközt, kivéve, ha az alsóba írt karaktert a bemeneten ékezet jele követi. A megoldásrészlet a SOR sort írja a képernyő K-adik és K+1-edik sorába (feltehető volt, hogy minden sor kifér a képernyő egy sorába):

```
Ékezetesít(SOR,K):
  Ciklus i=1-től hossz(SOR)-ig
    Pozícionál(K,i)
    Ha SOR(i+1)∈Ékezetek akkor Ki: SOR(i+1)
                       különben Ki: ' '
    Pozícionál(K+1,i); Ki: SOR(i)
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A. 6-féle ékezet felismerése	6 pont
B. ékezetek jó helyre helyezése	7 pont
C. sorvég karakter hatására új sor kezdése	1 pont

D. File-név beolvasása 1 pont

2. feladat: (15 pont)

Mivel a sakktábla nagyon nagy is lehet, ezért nem ábrázolhatjuk közvetlenül a sakktáblát. Olyan lépéssorozatot kell tehát generálnunk, amelyre garantált, hogy ugyanoda kétszer nem léphetünk.

A megoldás első fázisának ötlete az, hogy közelítsünk a célhely felé, mindig abban az irányban lépve a lóval kettőt, amely koordinátában nagyobb az aktuális és a célhely eltérése.

A második fázisba akkor érünk, ha a célhelyre vagy annak közvetlen szomszédságába értünk. Az utóbbi esetben minden pozícióra konstansként megadhatjuk, hogy mely lépésekkel jutunk el a célba.

```
Lóugrás(KX,KY,CX,CY):
  Ciklus amíg |KX-CX|>1 vagy |KY-CY|>1
    Ha |KX-CX|≥|KY-CY| akkor Ha KX>CX akkor KX:=KX-2
                                  különben KX:=KX+2
                             Ha KY>CY akkor KY:=KY-1
                                  különben KY:=KY+1
                  különben Ha KY>CY akkor KY:=KY-2
                                  különben KY:=KY+2
                             Ha KX>CX akkor KX:=KX-1
                                  különben KX:=KX+2
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A) Ha a program bekéri a pozíciókat, és útvonalat ír ki, még ha helytelent is, ez a pont megadható. (A teszteléskor csak pozitív egész számokat adunk be; a programnak nem kell működnie nem egész számokra, negatív számokra pedig nem teszteljük a működést.) 2 pont

B) Rövid utak. Ellenőrizni kell az indulási, illetve a végpozíció egyezését, és menet közben a lóugrás betartását. A feladat nem tiltja, hogy többször is ugyanarra a mezőre lépjünk.

C) A feladat kritikus része: nagyon messze levő pontok között is talál-e utat. Ha visszalépéses (backtrack) algoritmust alkalmaz, a program "nagyon lassú" lesz, ekkor nem jár a pont. Ha más kézenfekvő heurisztikus megoldást használ, a futás nagyon gyors lesz. A pont akkor jár, ha a futás gyors, a két végpont a megadottal egyező és a sorozatban a lóugrás szabályai érvényesülnek. (Mivel a sorozat hosszú, itt csak az elejét és a végét kell ellenőrizni.)

Ha futás közben 5 percnél hosszabb ideig nem ad magáról életjelt a program (nem listáz új eredményt), vagy hibajelzést kapunk (pl. elfogy a memória), akkor nem adható pont. A helyes megoldás időveszteség nélkül (folyamatosan) adja az eredmény pozíciókat. Nem az időkorlát a lényeges, tetszőlegesen nagy (idő vagy memória) korlátot is megszabhatnánk, hiszen a keresést alkalmazó megoldás biztosan meghaladja ezt is, ha a távolságot növeljük: pl. a maximális integernyi távolságon. Márpedig kikötés volt, hogy a két kocka "nagyon messze" lehet egymástól.

Kezd	**Cél**	
(0,0)	(1000,1000)	1 pont
(0,1000)	(1000,1000)	1 pont
(1500,2000)	(1000,1000)	1 pont
(0,1000)	(1000,0)	1 pont

3. feladat: (20 pont)

A feladat megoldása egy gráfbejárás, melynek során azt kell vizsgálni, hogy elérjük-e a mátrix szélét. Használjuk például a szélességi bejárást:

```
Szélességi bejárás(x,y,Kiért):
  px:=x; py:=y; Sor:=üres; Sorba(px,py); Szín(px,py):=szürke
  Kiért:=hamis
  Ciklus amíg nem üres a Sor és nem Kiért
    Sorból(px,py); Szín(px,py):=fekete
    Ha Mag(px,py)>Mag(px-1,py) és Szín(px-1,py)=fehér
       akkor Sorba(px-1,py); Szín(px-1,py):=szürke
             Ha px=2 akkor Kiért=igaz
    Ha Mag(px,py)>Mag(px+1,py) és Szín(px+1,py)=fehér
       akkor Sorba(px+1,py); Szín(px+1,py):=szürke
             Ha px=N-1 akkor Kiért=igaz
    Ha Mag(px,py)>Mag(px,py+1) és Szín(px,py+1)=fehér
       akkor Sorba(px,py+1); Szín(px,py+1):=szürke
             Ha py=M-1 akkor Kiért=igaz

    Ha Mag(px,py)>Mag(px,py-1) és Szín(px,py-1)=fehér
         akkor Sorba(px,py-1); Szín(px,py-1):=szürke
               Ha py=2 akkor Kiért=igaz
  Ciklus vége
Eljárás vége.
```

Értékelési szempontok:

A) Az algoritmustól független funkciók értékelése. Ez a pontszám akkor jár, ha a program beolvasta a bemeneti adatokat, és valamilyen választ kiírt (pl. ha a próbafile-ok bármelyikével futási eredményt produkál, még ha algoritmikusan helytelent is). Nulla pontot kell adni viszont, ha bármi más, az algoritmussal nem kapcsolatos hiba van: pl. a program nem kéri be a file nevét, hibásan írja ki a pozíciókat stb. — 2 pont

B) A program első változata (sebesség nélküli változat) egyszerű esetekre. Akkor adható pont, ha mindhárom próbafile-ra hibátlan eredményt kapunk:

- Ha a széléről indul, akkor azonnal leáll. — 1 pont
- Csak lefelé lépésekkel kijut. — 1 pont
- Egyszerű folyosón kijut-e, csapda kikerülésével. — 1 pont
- Több kijutási hely közül legalább egyet megtalál-e. — 1 pont
- Nincs megoldása. Erre rájön-e. Ha megengedi, hogy a golyó vízszintesen haladjon, vagy átlósan lépjen, akkor itt hibás megoldást fog jelezni. — 1 pont

C) A program első változata (sebesség nélküli eset) nagyobb terjedelmű keresésre. Akkor adható pont, ha mindkét futás hibátlan. — 4 pont

D) A program megkérdezi, hogy melyik változat szerint szeretnénk futtatni. — 1 pont

E) A második változat (sebességgel rendelkezik a golyó) futásának tesztelése.

- Egyszerű bukkanón vagy vízszintesen átjut-e. — 2 pont
- Nincs megoldás. — 2 pont

F) A második változat bonyolultabb futásának tesztelése. Itt a program végtelen ciklusba mehet vagy nagyon lelassulhat, ha nem tiltja meg a golyó "kóválygását", vagyis azt, hogy körbe járjon, például úgy, hogy megtiltja a már bejárt út ismétlését. (Az itt megadott megoldás kóválygással tűzdelt megoldása elfogadható, ha a program nem kerül végtelen ciklusba vagy nem lassul le látványosan, mert ezt egyébként a feladatban leírt törvények megengedik.) 4 pont

Megjegyzés: a második résznél a feladat nem definiálja egyértelműen, hogy a golyó átlendülhet-e egy csúcson úgy, hogy a sebessége eközben épp 0-ra csökken. Ezért az értékeléskor ezt az esetet nem teszteljük.

4. feladat: (27 pont)

Egy értelmező programot kell írnunk, amely olvassa a program utasításait, majd egyesével végrehajtja őket. Itt is, mint az előző években sokszor, egy szóolvasó eljárást kell írnunk.

```
Értelmezés(f):
  x:=0; y:=0; ir:=0; toll:=igaz
  Ciklus amíg nem vége(f)
    Szóolvasás(f,SZÓ)
    Elágazás
      SZÓ='forward' esetén Szóolvasás(s); hossz:=szám(s)
                           xx:=x+hossz*cos(ir)
                           yy:=y+hossz*sin(ir)
                           Ha toll akkor szakasz(x,y,xx,yy)
                           x:=xx; y:=yy
      SZÓ='back' esetén Szóolvasás(s); hossz:=szám(s)
                       xx:=x-hossz*cos(ir)
                       yy:=y-hossz*sin(ir)
                       Ha toll akkor szakasz(x,y,xx,yy)
                       x:=xx; y:=yy
      SZÓ='left' esetén Szóolvasás(s); szög:=szám(s)
                       ir:=ir-szög
      SZÓ='right' esetén Szóolvasás(s); szög:=szám(s)
                        ir:=ir+szög
      SZÓ='penup' esetén toll:=hamis
      SZÓ='pendown' esetén toll:=igaz
      SZÓ='reset' esetén x:=0; y:=0; ir:=0; toll:=igaz; törlés
      SZÓ='repeat' esetén Szóolvasás(s); db:=szám(s)
                          Ciklusmag beolvasás(szöveg)
                          Ciklus i=1-től db-ig
                            Ciklusmag feldolgozás(szöveg)
                          Ciklus vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

A `Ciklusmag feldolgozás` eljárás tulajdonképpen ugyanolyan, mint a fenti eljárás ciklusa, csupán nem file-ból olvassa a szavakat, hanem a `szöveg` változóból.

A `Ciklusmag beolvasás` pedig beolvassa a [és a] jelek közötti szavakat.

Értékelési szempontok:

A. forward végrehajtása 2 pont

B. left végrehajtása 2 pont

C. back végrehajtása 2 pont

D. right végrehajtása 2 pont

E. reset végrehajtása	1 pont
F. penup végrehajtása	1 pont
G. pendown végrehajtása	1 pont
H. repeat végrehajtása	9 pont
I. többjegyű számok felismerése	3 pont
J. szóközelválasztó kezelése	2 pont
K. sorvégelválasztó kezelése	1 pont
L. File-név beolvasása	1 pont

Elérhető összpontszám: 77 pont + 23 pont az 1. fordulóból

1994. Harmadik forduló

Első-második osztályosok

Tom és Jerry:

A játék alapja egy gráfbejárás, a problémából kiindulva a mélységi bejárás. Ennek első változata így nézhet ki, Jerry mozgatására:

```
Mélységi bejárás(Jx,Jy):
  px:=Jx; py:=Jy; i:=1; Verem:=Üres
  Verembe(0,0); Szín(px,py):=szürke
  Ciklus amíg nem üres a Verem és Lab(px,py)≠'S'
    Ha Szín(px-1,py)=fehér és (Lab(px-1,py)='u' vagy
                               Lab(px-1,py)='o')
       akkor Verembe(px-1,py); Szín(px-1,py):=szürke; px:=px-1
    különben ha Szín(px,py-1)=fehér és (Lab(px,py-1)='u' vagy
                                        Lab(px,py-1)='o')
       akkor Verembe(px,py-1); Szín(px,py-1):=szürke; py:=py-1
    különben ha Szín(px+1,py)=fehér és (Lab(px+1,py)='u' vagy
                                        Lab(px+1,py)='o')
       akkor Verembe(px+1,py); Szín(px+1,py):=szürke; px:=px+1
    különben ha Szín(px,py+1)=fehér és (Lab(px,py+1)='u' vagy
                                        Lab(px,py+1)='o')
       akkor Verembe(px,py+1); Szín(px,py+1):=szürke; py:=py+1
    különben Veremből(px,py)
  Ciklus vége
Eljárás vége.
```

A feladat szövege kicsit módosít az alap bejárási stratégián, amikor azt mondja, hogy Jerry a sajt felé indul, ha látja. Ez azt jelenti, hogy a ciklusban a 4 elágazás-ág vizsgálata előtt meg kell nézni, hogy valamelyik irányban látja-e Jerry a sajtot:

```
  Ha balrasajt(px,py) akkor Verembe(px-1,py)
                            Szín(px-1,py):=szürke; px:=px-1
  Ha felfelesajt(px,py) akkor Verembe(px,py-1)
                              Szín(px,py-1):=szürke; py:=py-1
  Ha jobbrasajt(px,py) akkor Verembe(px+1,py)
                             Szín(px+1,py):=szürke; px:=px+1
  Ha lefelesajt(px,py) akkor Verembe(px,py+1)
                             Szín(px,py+1):=szürke; py:=py+1
```

A valamelyik irányban sajkeresés egy egyszerű keresés programozási tétel alkalmazását jelenti:

```
Balrasajt(x,y):
  Ciklus amíg Lab(x-1,y)≠'S' és Lab(x-1,y)≠'f'
    x:=x-1
  Ciklus vége
  Balrasajt:=(Lab(x-1,y)≠'S')
Függvény vége.
```

A sajt felé haladás szabályát újabb szabály írhatja felül: ha Jerry Tomot látja valamerre, akkor abba az irányba biztosan nem megy. Ennek egyik iránybeli vizsgálata így nézhet ki:

```
  Ha balraTom(px,py) akkor btilos:=igaz különben btilos:=hamis
  ...
```

Ezután a balra lépés két lehetőségét az alábbiakra kell módosítani:

```
  Ha balrasajt(px,py) és nem btilos
     akkor Verembe(px-1,py); Szín(px-1,py):=szürke; px:=px-1
  ...
  Ha Szín(px-1,py)=fehér és nem btilos és
    (Lab(px-1,py)='u' vagy Lab(px-1,py)='o')
       akkor Verembe(px-1,py); Szín(px-1,py):=szürke; px:=px-1
  ...
```

A gráfbejárást még egy szempont miatt kell módosítani: a bejárás véget szokott érni, ha nincs több lehetőség a továbbhaladásra. Most azonban ez nem tehető meg. Így, ha a verem kiürül, akkor nem szabad befejezni a fő ciklust, hanem el kell kezdeni újra.

Tom mozgását Jerry mozgásához hasonlóan lehet megvalósítani, de mivel Tom kettőt is léphet egy irányba, így több elágazás-ágat kell megvalósítani.

Értékelési szempontok:

A.	Labirintusgenerálás szövegfile-ba	2+1 pont
	Van út	2 pont
	Csak 1 szélességű utak vannak benne	6 pont
	Van lyukas fal is	1 pont
B.	Sajtelhelyezés (nem falban)	1+1 pont
	Tom elhelyezése (nem falban)	1+1 pont
	Jerry elhelyezése (nem falban)	1+1 pont
C.	Szekvenciális szövegfile-ba írás	l. A részfeladat

Tom és Jerry *játékának* funkciói:

D.	Labirintusbeolvasás	1 pont
	Labirintusrajzolás (grafikával)	2+4 pont
E.	Először Jerry, azután pedig Tom indul el, felváltva mozognak a következő szabályok szerint:	
	a. vagy csak vízszintes, vagy csak függőleges irányban léphetnek	1 pont
	b. Jerry legfeljebb egyet léphet (a falak között, illetve lyukas falban),	2 pont
	c. Tom legfeljebb kettőt léphet (a falak között, irányváltás nélkül),	2 pont
	d. ha Jerry látja a sajtot (a sajt sorában vagy oszlopában van és nincs köztük fal), akkor egyenesen a sajt felé indul,	1 pont

e. ha Jerry látja Tomot, akkor menekülni próbál (Tom felé biztosan nem megy) – ez az előző szabályt is felülbírálja, 1 pont

f. ha Tom látja Jerry-t, akkor egyenesen felé indul, 1 pont

g. egyéb esetben Tom keresi Jerry-t, illetve Jerry keresi a sajtot. 1+1 pont

ha Tomot ketrecbe zárjuk, akkor Jerry megtalálja a sajtot 6 pont

ha nincs a sajt és a falakon lyuk, akkor Tom megfogja Jerry-t 7 pont

F. Jerry helyett a felhasználó (falba nem megy) 4*2 pont

Tom helyett a felhasználó (falba, lyukba nem megy, 2-t is léphet egy irányba) 4*4 pont

G. A játék véget ér, ha Tom megeszi Jerry-t, 1 pont

vagy Jerry megeszi a sajtot, 1 pont

vagy a felhasználó leállítja a programot. 1 pont

Elérhető összpontszám: 75 pont + maximum 25 pont az 1-2. fordulóból

Harmadik-ötödik osztályosok

Táblázatkezelő:

A feladat szerint egy 26*26-os, 10 karakteres szövegeket tartalmazó táblázatot kell tárolnunk:

```
Tábla: Tömb('A'..'Z',1..26,Szöveg(10))
```

A főprogram egy ciklus, amelyben a felhasználó dönt a továbblépésről.

```
Táblázatkezelő:
  sor:='A'; oszlop:=1;
  Ciklus
    Be: X
    Elágazás
      X=lefele esetén sor:=következő(sor)
      X=felfele esetén sor:=előző(sor)
      X=balra esetén oszlop:=oszlop-1
      X=jobbra esetén oszlop:=oszlop+1
      X=F1 esetén Adatmódosítás(sor,oszlop)
      X=F2 esetén Pozícionálás(sor,oszlop)
      X=F3 esetén Adatbeolvasás
      X=F4 esetén Adatkiírás
      X=F5 esetén Sorbeszúrás(sor)
      X=F6 esetén Oszlopbeszúrás(oszlop)
      X=F7 esetén Sortörlés(sor)
      X=F8 esetén Oszloptörlés(oszlop)
      egyéb esetben Tábla(sor,oszlop):=X
  Ciklus vége
Eljárás vége.
```

Az egyszerűség kedvéért a fenti eljárásokat fordított sorrendben írjuk meg. Törlés esetén a táblázat sorait/oszlopait eggyel léptetni kell, s üresre töltjük fel az új sort/oszlopot.

A két eljárás nagyon hasonló, most az oszloptörlést írjuk meg.

```
Oszloptörlés(osz):
  Ciklus s='A'-tól 'Z'-ig
    Ciklus o=osz-tól 25-ig
      Tábla(s,o):=Tábla(s,o+1)
    Ciklus vége
    Tábla(s,26):=''
  Ciklus vége
  Oszlophivatkozáselőre(osz)
Eljárás vége.
```

A beszúrás a törléshez hasonló művelet, itt a beszúrás helyét kell üressé tenni. Most a sor beszúrását valósítjuk meg:

```
Sorbeszúrás(sor):
  Ciklus o=1-től 26-ig
    Ciklus s='z'-től következő(sor)-ig
      Tábla(s,o):=Tábla(előző(s),o)
    Ciklus vége
    Tábla(sor,o):=''
  Ciklus vége
  Sorhivatkozáshátra(sor)
Eljárás vége.
```

Az oszlopok, illetve a sorok törlése és beszúrása esetén módosítani kell a képletekben szereplő hivatkozásokat is.

Az adatok beolvasása és kiírása egy-egy egyszeű ciklusparral megvalósítható, a `Tábla` mátrixot minden változtatás nélkül kell kiírni, illetve beolvasni. Az egyszerűség kedvéért minden mezőt külön sorba írunk, de csak azokat, amelyek nem üresek.

A `Pozícionálás(sor,oszlop)` is nagyon egyszerű, be kell olvasni a `sor` és az `oszlop` változó új értékét.

Az `Adatmódosítás(sor,oszlop)` eljárás egy szöveges szerkesztő tevékenység:

```
Adatmódosítás(sor,oszlop):
  poz:=1; vége:=hamis; sz:=Tábla(sor,oszlop)
  Ciklus amíg nem vége
    Be:X
    Elágazás
      X=balranyíl esetén Ha poz>1 akkor poz:=poz-1
      X=jobbranyíl esetén Ha poz<hossz(sz)) akkor poz:=poz+1
      X=DEL esetén sz:=sz(1..poz-1)+sz(poz+1..hossz(sz))
      egyéb esetben sz:=sz(1..poz-1)+X+sz(poz..hossz(sz))
                    Ha hossz(sz)>10 akkor sz:=sz(1..10)
    Elágazás vége
  Ciklus vége
  Tábla(sor,oszlop):=sz
Eljárás vége.
```

Ezután kell foglalkoznunk a megjelenítéssel. az aktuális cellát más színnel írjuk, így mielőtt elhagynánk, normál színűre kell váltanunk. A képernyőn biztosan nem fér el 26 sor és 26 oszlop. Emiatt ablakmozgatást kell elvégeznünk minden pozícionálás esetén.

A felső szintet úgy állítjuk be, hogy a bal felső sarok legyen a képernyő bal felső sarkában:

```
Táblázatkezelő:
  sor:='A'; oszlop:=1; Ks:='A'; Ko:=1
  Ciklus
    Be: X; Cellaírás(sor,oszlop)
    ...
    Megjelenítés(Ks,Ko)
  Ciklus vége
Eljárás vége.
```

A képernyőablak mozgatása egyszerű függőleges irányban, hiszen 25 sor fér ki a képernyőre, s 26 van összesen. Ez azt jelenti, hogy csak akkor kell mozgatni, ha a két szélső sor valamelyikére lépünk. A vízszintes mozgatás gyakoribb, egy oszlop 10 karakter szélességű, így egyszerre 8 oszlop látható a képen.

```
Megjelenítés(Ks,Ko):
  Ha sor<Ks akkor Ks:=sor; frissít:=igaz
  különben Ha sor='Z' és Ks='A' akkor Ks:='B'; frissít:=igaz
  Ha oszlop<Ko akkor Ko:=oszlop; frissít:=igaz
  különben Ha oszlop>Ko+7 akkor Ko:=oszlop-7; Frissít:=igaz
  Ha frissít akkor Teljesképírás(Ks,Ko,sor,oszlop)
Eljárás vége.
```

A teljes képernyő kiírása egy részmátrix kiírását jelenti, azzal a feltétellel, hogy az aktuális cellát (sor,oszlop) más színnel írjuk ki.

A cellák írása egy képletfeldolgozást jelent. Ehhez meg kell oldani a kifejezés lengyel formára hozását, valamint a lengyel forma kiértékelését.

```
Cellaírás(sor,oszlop):
  Lengyel(sor,oszlop,B); Pozícionálás(sor,oszlop); Kiírás(B)
Eljárás vége.
```

Foglalkozzunk először a kifejezések lengyel formára hozásával. Egy szöveg típusú változóban adott kifejezést bontunk lexikális elemekre, amelyek egy szöveg típusú elemeket tartalmazó vektorba kerülnek.

```
Lexikális analízis(kifejezés,db,elemek):
  db:=0; i:=1
     Ciklus amíg i≤hossz(kifejezés)
        x:=kifejezés(i); i:=i+1
        Elágazás
          x∈Műveletijelek esetén db:=db+1; elem(db):=x
          x="(" vagy x=")" esetén db:=db+1; elem(db):=x
          egyéb esetben Szóolvasás(x,i); db:=db+1; elem(db):=x
        Elágazás vége
   Ciklus vége
Eljárás vége.
```

A `Szóolvasás(x,i)` eljárás az x változóba szedi ki a kifejezés i-edik pozícióján kezdődő szót, miközben i-t növeli

```
Szóolvasás(x,i):
  Ciklus amíg i≤hossz(kifejezés) és
                (kifejezés(i)∈Számjegyek ∪ Betűk)
     x:=x+kifejezés(i); i:=i+1
  Ciklus vége
Eljárás vége.
```

Másodjára olvassuk a lexikális elemeket és elkészítjük a lengyel formát. Ha olvasás közben operandust találunk, azt az eredménykifejezésbe automatikusan leírhatjuk, ha operátor jön, akkor azt egyelőre megjegyezzük és majd csak a következő operátor (-ok) ismerete után döntünk sorsáról. Fontos dolog, hogy a megjegyzett operátorok tárolási sorrendjét tudjuk, hisz az utoljára betettet kell majd először elővenni. (Ennek megoldására használunk un. verem adatszerkezetet, hisz az éppen ilyen tulajdonságokkal bír). Ha az aktuális operátor prioritása kisebb, mint az előzőé, akkor az előzőt kivesszük a veremből és a készülő kifejezésbe tesszük. Ha a verem tetején megint egy nála nagyobb prioritású operátor van, akkor ezt az eljárást folytatjuk.

A zárójelek használata annyiban bonyolítja a helyzetet, hogy a zárójelben levő kifejezésrészletek, mint önálló egységek kerülnek feldolgozásra. Ez úgy történik, hogy a nyitózárójelet betesszük a verembe és a csukó érkeztével az esetleg a nyitón levő operátorok mindegyikét a kifejezéshez másoljuk.

```
Lengyel formára hozás(db,elemek,ldb,lengyel):
  ldb:=0: Verembe(Végjel)
  Ciklus i=1-től db-ig
     A:=elemek(i)
     Elágazás
        A='(' esetén Verembe(A)
        A=')' esetén Zárójelfeldolgozás
        A∈Műveletijelek esetén Műveletfeldolgozás(A)
        egyéb esetben ldb:=ldb+1; lengyel(ldb):=A
     Elágazás vége
  Ciklus vége
  Veremürítés
Eljárás vége.

Zárójelfeldolgozás:
  Ciklus amíg Veremtető≠"("
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
  Veremből(B)
Eljárás vége.

Műveletfeldolgozás(A):
  Ciklus amíg prioritás(A)≤prioritás(Veremtető)
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
  Verembe(A)
Eljárás vége.

Veremürítés:
  Ciklus amíg Veremtető≠Végjel
     Veremből(B); ldb:=ldb+1; lengyel(ldb):=B
  Ciklus vége
Eljárás vége.
```

Harmadik fázis a kiértékelés. Ezt rekurzívan kell meghívni akkor, ha a kifejezésben más cellára hivatkozunk.

```
Kiértékelés(X,Y,Z,ldb,lengyel):
  Ciklus i=1-től ldb-ig
     A:=lengyel(i)
     Elágazás
        A∈Műveletijelek esetén Veremből(B); Veremből(C)
                               Művelet(A,B,C,D); Verembe(D)
        A(1)∈Betűk esetén Lengyel(A(1),szám(A(2..hossz(A)),B)
                         Verembe(B)
        egyéb esetben Verembe(B)
     Elágazás vége
  Ciklus vége
  Veremből(Z)
Eljárás vége.

Művelet(A,B,C,D):
  Elágazás
     A='+' esetén D:=B+C
     A='-' esetén D:=B-C
     A='*' esetén D:=B*C
     A='/' esetén D:=B/C
  Elágazás vége
Eljárás vége.
```

Értékelési szempontok:

A.	Kezdőállapot	3 pont
B.	Mozgás 4 irányban	1+1+1+1 pont
	Mozgatás 4 irányban	1+1+1+1 pont
	A táblázat szélein nem lép ki	1+1+1+1 pont
C.	az aktuális mezőbe lehessen beírni	2 pont
	a. fixpontos valós számot a mezőben jobbra igazítva	2 pont
	b. szöveget a mezőben balra igazítva	2 pont
	c. képletet, képlet tartalmazhat:	
	1. számkonstansokat,	2 pont
	2. mezőhivatkozásokat,	2 pont
	3. +,-,*,/ műveleteket,	1+1+1+1 pont
	4. zárójeleket;	4 pont
	szintaktikus ellenőrzés	7 pont
D.	mező értékét módosítani:	
	a. nyilakkal mozgás	2+2 pont
	b. az aktuális karaktert a DEL billentyűvel törölni;	1 pont
	c. az aktuális karakter elé egy karaktert beszúrni;	1 pont
	d. ENTER billentyű lenyomására a szerkesztést befejezni;	1 pont
E.	közvetlenül pozícionálni egy cellára	2 pont
F.	a táblázatot lemezre írni	5 pont
G.	lemezről egy táblázatot beolvasni	5 pont

H.	új sort beilleszteni az aktuális sor elé	1 pont
	a képletekben szereplő sorindexek átszámolása	1 pont
	sorok eltolása, a Z. sor elveszik	2 pont
I.	új oszlopot beilleszteni az aktuális oszlop elé	1 pont
	a képletekben szereplő oszlopindexek átszámolása	1 pont
	oszlopok eltolása, a 26. oszlop elveszik	2 pont
J.	az aktuális sort törölni	1 pont
	a képletekben szereplő sorindexek átszámolása	1 pont
	sorok eltolása, a Z. sor üres lesz	2 pont
K.	az aktuális oszlopot törölni	1 pont
	a képletekben szereplő oszlopindexek átszámolása	1 pont
	oszlopok eltolása, a 26. oszlop üres lesz	2 pont

Elérhető összpontszám: 75 pont + maximum 25 pont az 1-2. fordulóból