Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Rekurzió

TÁMOP-4.2.3.-12/1/KONV

A projektek az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósulnak meg.

1. A faktoriális függvény

A rekurzió, mint eszköz felbukkan specifikációs, algoritmikus, implementációs (nyelvi) eszközként. Kezdjük a *specifikáció*nál, amellyel a matematikusok a problémák megoldását kezdik! A függvény egy igen hatékony absztrakciós eszköz, ugyanis jól kidolgozott formalizmussal rendelkezik és sokrétű, „bejáratott” operációval (függvényművelettel) lehet építkezni. Mivel sok mindent (érts ezalatt bármilyen tevékenységsort) úgy lehet tekinteni, mint valamiféle függvényt, ami a kezdetben meglévő adatokhoz hozzárendeli a kívánt valamit, ezért nem reménytelen vállalkozás a függvény definiálást választani „általános” leíró eszközként. Ilyen ok miatt nincs mit csodálkozni azon, hogy példáink java része rekurzív függvény lesz, és elindulásként is az egyik legismertebbiket választottuk ki: a *faktoriális*t.

Definíciója:

$$n! = \begin{cases} n*(n-1)! & ha \;\; n>0 \\ 1 & ha \;\; n=0 \end{cases}$$

A faktoriális definíciója két részre bomlik. Az egyik épít a már meglévő, és működő definícióra, és azt eggyel csökkentett értékkel „újra meghívja”. A másik – bízva abban, hogy valamikor „eljő az ő ideje” – kijáratot biztosít a(z előbbi) végtelenségig való önhívogatásból. Vagyis valahogy így:

```
Faktoriális(n):
  Ha n=0 akkor [a definíció nem rekurzív része]
               f:=1
      különben [a definíció rekurzív része]
               f:=n*Faktoriális(n-1)
  Elágazás vége
  Faktoriális:=f
Függvény vége.
```

Nézzük meg, hogy pontosan mi is történik pl. a Faktoriális(3) hívásakor!

Faktoriális(3) => 3 * Faktoriális(2) 3 * 2 * 1 * 1

Faktoriális(2) => 2 * Faktoriális(1) 2 * 1 * 1

Faktoriális(1)=> 1 * Faktoriális(0) 1 * 1

Faktoriális(0) => 1 1

A faktoriális függvény értéke nagyon gyorsan nő, azaz nem tudjuk nagyobb N értékekre kiszámolni:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	6	24	120	720	5 040	40 320	362 880	3 628 800	39 916 800	479 001 600	6 227 020 800	87 178 291 200

Érdekességként az utolsó K számjegyét viszont számolhatjuk megfelelő számtípus esetén:

```
Faktoriális(n,k):
  Ha n=0 akkor f:=1
      különben f:=(n*Faktoriális(n-1)) mod 10^k
  Elágazás vége
  Faktoriális:=f
Függvény vége.
```

K=6 esetén: néhány következő tag:

10	11	12	13	14	15	16	17	18	19	20
628 800	916 800	1 600	20 800	291 200	368 000	888 000	96 000	728 000	832 000	640 000

Könnyű belátni, hogy K=6 esetén N=25-tól kezdődően csupa 0 lesz az így kiszámolt sorozatban.

2. A Fibonacci-számok

A rekurzív függvények matematikai elméletében éppúgy ismert, mint a biológiában a Fibonacci olasz matematikusról elnevezett számsorozat. E híres matematikus (aki egyébként a matematikának sok – mai szóhasználattal élve – ágában jeleskedett) Európában talán elsőként nyúlt egzakt eszközökhöz mindennapos – mondhatnánk „háztáji" – probléma megoldásához. Ugyanis azt vizsgálta, hogy egy nyúlpár „alapította" nyúlnemzetség adott idő alatt mekkora létszámúra növekszik, figyelembe véve, hogy a leszármazottak is alaposan „besegítenek" a létszámnövelésbe.

Ha a szaporodás eléggé „szabályosan" történik, akkor az új generáció létszámát az előzőek ismeretében könnyen kiszámíthatjuk. Szerinte az új generáció növekedését az előző 2 generáció gyerekei teszik ki. E mögött az a feltételezés húzódik meg, hogy minden nyúlpár egyszerre éppen 2 utóddal járul a népességhez, és e „szokásukat" születésüket követő 2 egymásutáni időpontban „gyakorolják" (mert – mondjuk – mielőtt a harmadik szaporodásra sor kerülhetne, fazékba kerülnek).

Definíciója:

$$Fib(n) = \begin{cases} 0 & ha \;\; n = 0 \\ 1 & ha \;\; n = 1 \\ Fib(n-1) + Fib(n-2) & ha \;\; n > 1 \end{cases}$$

Példa: (a Fibonacci-szám sorozat első néhány tagja)

```
0,1,1,2,3,5,8,13,21,34,55,...
```

```
Fib(N):
  Elágazás
    N=0 esetén Fib:=0
    N=1 esetén Fib:=1
    egyéb esetben Fib:=Fib(N-1)+Fib(N-2)
  Elágazás vége
Függvény vége.
```

Nézzük meg Fib(4) kiszámítását!

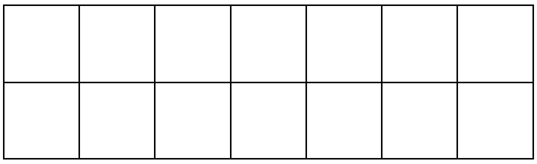
```
Fib(4) => Fib(3) + Fib(2)
          Fib(3) => Fib(2) + Fib(1)
                    Fib(2) => Fib(1) + Fib(0)

                    Fib(2) => Fib(1) + Fib(0)
```

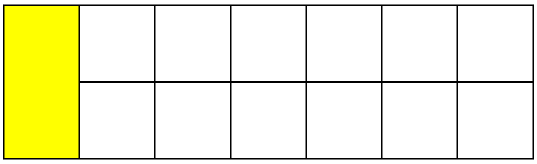
Többször is kiszámításra kerülnek ugyanazok a tagok!

Járdakövezés

Számítsuk ki, hogy hányféleképpen lehet egy 2xn egység méretű járdát kikövezni 1x2 méretű lapokkal!



Az első lapot lerakhatjuk függőlegesen:



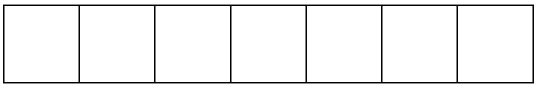
vagy kettőt vízszintesen:

Az első esetben (n-1)*2 cellát kell még lefednünk, a másodikban pedig (n-2)*2 cellát. Azaz az n*2 cella lefedéseinek Lefed(n) száma Lefed(n-1)+Lefed(n-2).

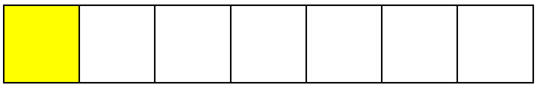
```
Lefed(N):
  Elágazás
    N=0 esetén Lefed:=0
    N=1 esetén Lefed:=1
    egyéb esetben Lefed:=Lefed(N-1)+Lefed(N-2)
  Elágazás vége
Függvény vége.
```

Járdakövezés újra

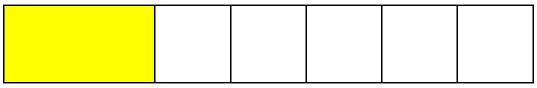
Számítsuk ki, hogy hányféleképpen lehet egy n egység méretű járdát kikövezni 1x1, 1x2 és 1x3 méretű lapokkal!



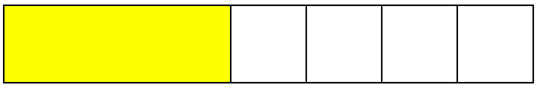
Az első helyre tehetünk 1x1-es lapot:



Az első helyre tehetünk 1x2-es lapot:



Az első helyre tehetünk 1x3-as lapot:

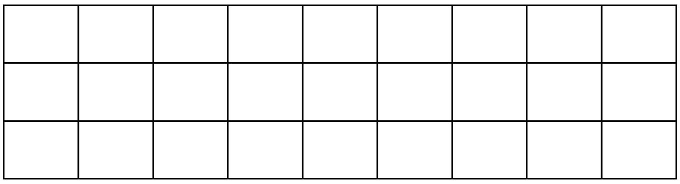


Az első esetben n-1, a másodikban n-2-t, a harmadikban pedig n-3 cellát kell még lefednünk. Azaz az n cella lefedéseinek Lefed(n) száma Lefed(n-1)+Lefed(n-2)+Lefed(n-3).

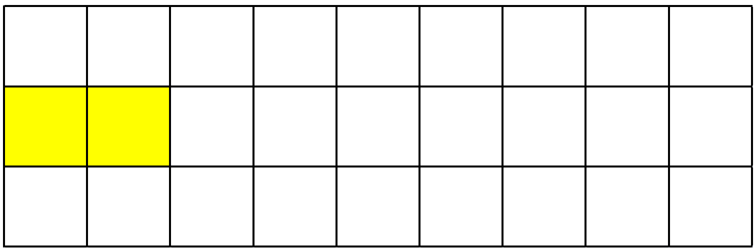
```
Lefed(N):
  Elágazás
    N=0 esetén Lefed:=0
    N=1 esetén Lefed:=1
    N=2 esetén Lefed:=2
    egyéb esetben Lefed:=Lefed(N-1)+Lefed(N-2)+Lefed(N-3)
  Elágazás vége
Függvény vége.
```

Gondolkodtató

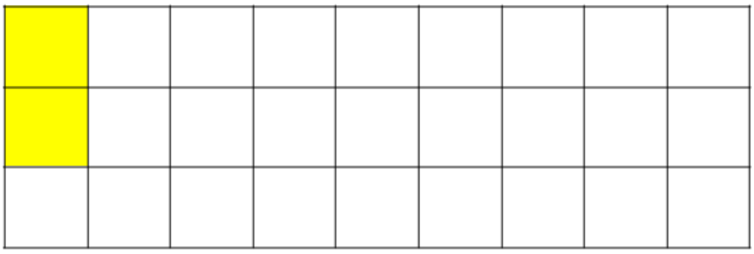
Hányféleképpen lehet egy 3xn egység méretű járdát kikövezni 1x2 méretű lapokkal!



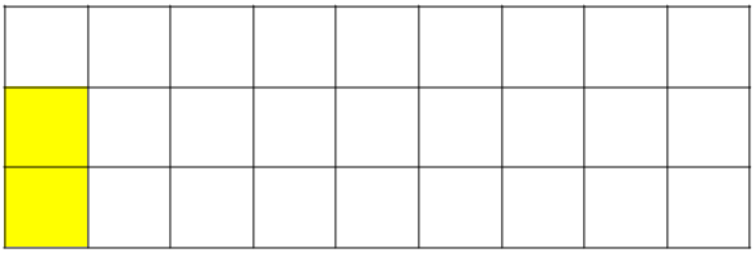
Az első oszlop középső négyzete háromféleképpen fedhető le. A második és harmadik eset ráadásul egymás tükörképe.



1. eset

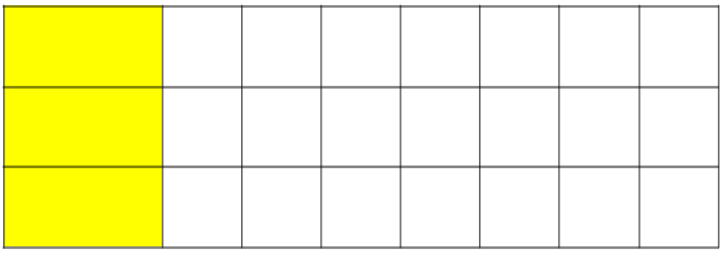


2. eset

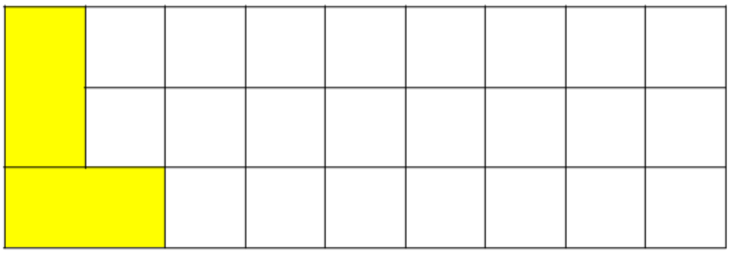


3. eset

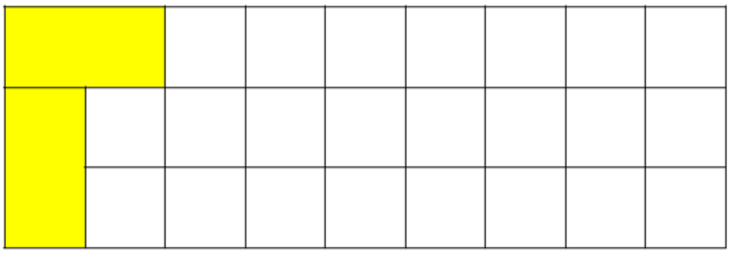
Az első oszlop betöltéséhez mindegyik csak egyféleképpen folytatható:



1. eset



2. eset



3. eset

Lépcső

Az iskola bejáratánál N lépcsőfok van. Egyszerre maximum K fokot tudunk lépni, ugrani fölfele. Minden nap egyszer megyünk be az iskolába. Készíts programot, amely megadja, hogy hány napig tudunk más és más módon feljutni a lépcsőkön!

Másképp megfogalmazva a feladatot, arra vagyunk kíváncsiak, hogy az N. lépcsőfokra hányféleképpen lehet feljutni.

Nézzük először a K=2 esetet! Az N. lépcsőfokra két helyről tudunk lépni, az N-1-edikről és az N-2-edikről. Azaz annyiféle feljutási lehetőség van az N. lépcsőfokra, amennyi az N-1-edikre plusz amennyi az N-2-dikre, tehát Lépcső(N)=Lépcső(N-1)+Lépcső(N-2). Itt is a Fibonacci számokat kapjuk.

Ezt kell általánosítanunk K-ra: Lépcső(N)=Lépcső(N-1)+…+Lépcső(N-K).

```
Lépcső(N):
  Ha N=0 akkor L:=1
  különben L:=0
           Ciklus i=1-től K-ig
             Ha N-i≥0 akkor L:=L+Lépcső(N-i)
           Ciklus vége
  Lépcső:=L
Függvény vége.
```

3. M-nél kisebb vagy egyenlő kettő-hatványok előre

Adjuk meg az M-nél nem nagyobb kettő-hatványokat növekvő sorrendben! A feladatot átfogalmazhatjuk úgy, hogy adjuk meg a K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványokat növekvő sorrendben (ha K kettő-hatvány)! Már itt is a rekurzió: ezek a számok a K, majd pedig a 2*K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványok növekvő sorrendben:

```
Hatványok(K,M):
  Ha K≤M akkor Ki: K; Hatványok(2*K,M)
Eljárás vége.
```

Példa: K=1, M=1000

```
  1 2 4 8 16 32 64 128 256 512
```

4. M-nél kisebb vagy egyenlő kettő-hatványok visszafelé

Adjuk meg az M-nél nem nagyobb kettő-hatványokat csökkenő sorrendben! A feladatot átfogalmazhatjuk úgy, hogy adjuk meg a K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványokat csökkenő sorrendben (ha K kettő-hatvány)! Már itt is a rekurzió: ezek a számok a 2*K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványok csökkenő sorrendben, majd pedig a K:

```
Hatványok(K,M):
  Ha K≤M akkor Hatványok(2*K,M); Ki: K
Eljárás vége.
```

Példa: K=1, M=1000

```
  512 256 128 64 32 16 8 4 2 1
```

5. M-nél kisebb vagy egyenlő kettő-hatványok oda és vissza

Adjuk meg az M-nél nem nagyobb kettő-hatványokat növekvő, majd visszafelé csökkenő sorrendben! A feladatot átfogalmazhatjuk úgy, hogy adjuk meg a K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványokat ilyen sorrendben (ha K kettő-hatvány)! Már itt is a rekurzió: ezek a számok a K, majd a 2*K-nál nagyobb vagy egyenlő, M-nél kisebb vagy egyenlő kettő-hatványok csökkenő sorrendben, végül pedig újra a K:

```
Hatványok(K,M):
  Ha K≤M akkor Ki: K; Hatványok(2*K,M); Ki: K
Eljárás vége.
```

Példa: K=1, M=1000

```
  1 2 4 8 16 32 64 128 256 512 512 256 128 64 32 16 8 4 2 1
```

6. Binomiális együtthatók

Egy véges halmaz, melynek N darabszámú elemeiből K elemszámú halmazokat (kombinatorika nevén osztályokat) akarunk mindenféle módon képezni (és minden elem csak egyszer fordul elő). Ezt úgy hívjuk, hogy n elem k-ad osztályú ismétlés nélküli kombinációja. Ezen kombinációk száma megegyezik a matematikából máshonnan is ismert binomiális együtthatókkal.

A binomiális együtthatókat a következő képlettel definiálhatjuk:

$$\binom{n}{k} = \frac{n!}{k! * (n-k)!}$$

Ennek kiszámítása hosszú programot igényelne, próbálkozzunk inkább egy másik kiszámítási módszerrel!

Ehhez nézzük meg e számok elrendezését, a Pascal háromszöget:

```
                              1
                          1       1
                      1       2       1
                  1       3       3       1
              1       4       6       4       1
          1       5      10      10       5       1
      1       6      15      20      15       6       1
    1     7      21      35      35      21       7       1
  1    8     28      56      70      56      28       8       1
 1   9    36     84     126     126     84      36       9       1
1  10   45   120   210   252   210   120    45     10    1
1  11   55  165   330   462   462   330  165   55   11  1
1   12   66   220   495   792   924   792   495   220   66   12  1
```

Felfedezhetjük, hogy a fenti táblázatban minden szám a fölötte levő két szám összege, azaz N elemből K elem választása leírható az alábbi módon:

- az első elemet választjuk, majd még N-1 elemből választunk K-1 elemet. vagy
- az első elemet nem választjuk és a maradék N-1 elemből választunk K elemet.

$$B(n,k) = \begin{cases} 1 & ha\ \ k = 0 \\ B(n-1,k) + B(n-1,k-1) & ha\ \ 0 < k < n \\ 1 & ha\ \ k = n \end{cases}$$

```
Bin(n,k):
  Ha k=0 vagy k=n akkor Bin:=1
  különben Bin:=Bin(n-1,k)+Bin(n-1,k-1)
Eljárás vége.
```

Egy gyorsabb módszert is találhatunk N elemből K elem választására, ha felírjuk B(n.k) és B(n,k-1) értékét:

$$B(n,k) = \frac{n!}{k!*(n-k)!} = \frac{n*(n-1)*...*(k+1)}{(n-k)*(n-k-1)*...*1}$$

$$B(n,k-1) = \frac{n!}{(k-1)!*(n-k+1)!} = \frac{n*(n-1)*...*(k+1)*k}{(n-k+1)*(n-k)*...*1}$$

Ha az alsó képlet számlálójából elhagyjuk a K értékét, a nevezőjéből pedig az (n-k+1)-et, akkor éppen a felső képletet kapjuk. A módszer másképp megfogalmazva:

- először kiválasztunk K-1 elemet, majd
- a maradék N-K+1 elemből kell egyet választani (de így minden kombináció pontosan K-féleképpen áll elő), tehát jön még egy K-val osztás.

$$B(n,k) = \begin{cases} 1 & ha\ \ k = 0 \\ B(n,k-1)*\dfrac{n-k+1}{k} & ha\ \ 0 < k \le n \end{cases}$$

```
Bin(n,k):
  Ha k=0 akkor Bin:=1
  különben Bin:=Bin(n,k-1)*(n-k+1)/k
Eljárás vége.
```

Eddig közvetlen, szimpla rekurzióval foglalkoztunk. Ideje megnézni a dupla, illetve a közvetett rekurziót is.

7. McCarthy-féle 91-es függvény:

Zohar Manna, Amir Pnueli és John McCarthy 1970-ben találta ki elméleti informatikai célokra ezt a rekurzív függvényt. Értéke 91 lesz minden 100-nál kisebb vagy egyenlő n természetes számra. 100-nál nagyobb n-ekre az értéke n-10 lesz.

$$M(n)=\begin{cases} n-10 & ha \quad n>100 \\ M(M(n+11)) & ha \quad n\leq 100 \end{cases}$$

1. példa:

```
M(99) = M(M(110)) mert 99 ≤ 100

      = M(100)    mert 110 > 100

      = M(M(111)) mert 100 ≤ 100

      = M(101)    mert 111 > 100

      = 91        mert 101 > 100
```

2. példa:

```
M(87) = M(M(98))

      = M(M(M(109)))

      = M(M(99))

      = M(M(M(110)))

      = M(M(100))

      = M(M(M(111)))

      = M(M(101))

      = M(91)

      = M(M(102))

      = M(92)
```

```
      = M(M(103))
      = M(93)
      …
      = M(99)
```

innen ugyanaz, mint az 1. példa

```
      = 91
```

A függvény duplán rekurzívan:

```
M(n):
  Ha n>100 akkor M:=n-10 különben M:=M(M(n+11))
Eljárás vége.
```

A függvényben a dupla rekurzió kifejtve:

```
M(n):
  Ha n>100 akkor M:=n-10
        különben x:=M(n+11); M:=M(x)
Eljárás vége.
```

Tehát a dupla rekurzió algoritmikus szinten nem okoz semmilyen gondot!

8. Döntsük el egy számról, hogy páros-e!

Tegyük fel, hogy nincs maradék-számítás műveletünk! A megoldás egy közvetett rekurzív számítás: a párosságot visszavezethetjük eggyel kisebb szám páratlanságára, a páratlanságot pedig eggyel kisebb szám párosságára. Ez a közvetett rekurzió

```
Páros(n):
  Ha n=0 akkor Páros:=igaz
  különben ha n=1 akkor Páros:=hamis
      különben Páros:=Páratlan(n-1)
Függvény vége.
```

```
Páratlan(n):
  Ha n=0 akkor Páratlan:=hamis
  különben ha n=1 akkor Páratlan:=igaz
      különben Páratlan:=Páros(n-1)
Függvény vége.
```

Ugyanez összevonva egyetlen rekurzív függvénybe, közvetlen rekurzióvá alakítva:

```
Páros(n):
  Ha n=0 akkor Páros:=igaz
  különben ha n=1 akkor Páros:=hamis
  különben Páros:=Páros(n-2)
Függvény vége.
```

9. Hatványozás

Két szám hatványozását (A^B) visszavezethetjük szorzásokra és kettővel osztásra a következőképpen:

$$A^B = \begin{cases} 1 & \text{ha} \quad B = 0 \\ (A * A)^{B/2} & \text{ha} \quad B \quad \text{páros} \\ A * A^{B-1} & \text{ha} \quad B \quad \text{páratlan} \end{cases}$$

```
Hatvány(A,B):
  Ha B=0 akkor Hatvány:=1
  különben Ha B páros akkor Hatvány:=Hatvány((A*A),(B/2))
  különben Hatvány:=A*Hatvány(A,(B-1))
Függvény vége.
```

Példa:

$$2^{10}=4^5=4*4^4=4*16^2=4*256^1=4*256$$