Odometer

Leonardo fedezte fel az odometer eredeti változatát, amely egy forgó kerék és kavicsok alkalmazásával tudott távolságot mérni. Egy ilyen eszköz számítógépes változatát kell elkészítened!

Négyzetrács

Az odometer egy 256*256-os négyzetrácson mozog. Minden cella legfeljebb 15 kavicsot tartalmazhat. A mezőket a 0 és 255 közötti koordinátáikkal azonosítjuk. Az (i,j) mező szomszédjai az (i-1,j), az (i+1,j), az (i,j-1) és az (i,j+1) mezők, ha léteznek. Az első és utolsó sor, valamint az első és utolsó oszlop a határ. Az odometer a (0,0) mezőről indul (ami a bal felső sarok), kezdetben felfelé néz.

Elemi utasítások

Utasítások:

- `left` — balra fordul 90 fokot (órajárással ellenkező irányban) és helyben marad (azaz ha például lefelé nézett korábban, akkor az utasítás után jobbra fog nézni).
- `right` — jobbra fordul 90 fokot (órajárással megegyező irányban) és helyben marad (azaz ha például balra nézett korábban, akkor az utasítás után felfelé fog nézni).
- `move` — egy mezőt előre lép a szomszédos mezőre (amerre éppen nézett). Ha nincs ilyen mező (a határról ki akar lépni), az utasítás hatástalan.
- `get` — felvesz egy kavicsot az aktuális mezőről. Ha itt nincs kavics, akkor az utasítás hatástalan.
- `put` — hozzáad egy kavicsot az aktuális mezőhöz. Ha több mint1 15 kavics van ott, akkor az utasítás hatástalan.
- `halt` — befejezi a végrehajtást

Az odometer a leírás sorrendjében hajtja végre a programot, soronként 1 utasítást. Az üres sorok hatástalanok. A # jel megjegyzést jelöl, innen a sor végéig írt szöveg megjegyzés lesz. A végrehajtás befejeződik, ha a program utolsó utasítását is végrehajtotta.

1. példa

A program odometert a (0,2) mezőre viszi és ott jobbra néz. (Megjegyzés: az első `move` utasítás hatástalan, mert az odometer kezdetben a bal felső sarokban áll és felfelé néz).

```
move  # hatástalan
right
# most az odometer jobbra néz
move
move
```

Címkék, határmezők és kavicsok

Az utasítások sorrendje megváltoztatható címkék és ugró utasítások használatával. A címke legfeljebb 128 karakteres lehet, a használható karakterek: `a`, …, `z`, `A`, …, `Z`, `0`, …, `9`, a kis és nagybetűk különbözők. Az alábbi utasítások használhatók, ahol L érvényes címke.

- `L:` (azaz az `L` után kettőspont van '`:`') — az `L` címke helyét adja meg. Minden címke egyetlen deklarációban lehet. A címke utasítás az odometerre hatástalan.
- `jump L` — feltétel nélküli ugrás az `L` címkéjű sorra.
- `border L` — ugrás az `L` címkéjű sorra, ha az odometer a határon van és a `move` utasítást nem tudná végrehajtani; egyébként az utasítás hatástalan.
- `pebble L` — ugrás az `L` címkéjű sorra, ha az aktuális mezőn van legalább 1 kavics; egyébként az utasítás hatástalan.

2. példa

A program az odometer a 0. sor első kavicsához viszi. Ha nincs, akkor a 0. sor határmezőjén áll meg. A `leonardo` és a `davinci` címkéket használja.

```
right
leonardo:
pebble davinci  # ugrás, ha van kavics
border davinci  # ugrás, ha sor végén van
move
jump leonardo
davinci:
halt
```

Az odometer először jobbra fordul. A `leonardo:` címkénél ciklus kezdődik, aminek utolsó utasítása a `jump leonardo` utasítás. A ciklusban ellenőrzi, van-e kavics az aktuális mezőn vagy a határon van-e. Ha egyik sem, akkor egyet lép a (0,j) mezőről a (0,j+1) mezőre. (A `halt` nem feltétlenül szükséges a program végére.)

Feladat

Az odometer nyelvén írt programokat kell beküldened, amelyek a megadott módon működnek. Minden részfeladat megad egy kívánt működést, korlátozó feltételekkel:

- *Program size* — a program mérete (azaz az utasítások száma) maximum ekkora lehet. Nem számítanak bele a címkék, megjegyzések és üres sorok.
- *Execution length* — a végrehajtási idő (a végrehajtott utasítások száma) maximum ennyi lehet. A hatástalan utasítás is számít, de nem számítanak bele a címkék, megjegyzések és üres sorok.

Az 1. példában a program mérete 4, a végrehajtási idő is 4. A 2. példában a program mérete 6, ha a (0,10) mezőn van kavics, akkor végrehajtási idő 43 lépés: `right`, 10 ismétlése a ciklusmagnak, mindegyikben 4 lépés (`pebble davinci`; `border davinci`; `move`; `jump leonardo`), és végül, `pebble davinci` és `halt`.

1. részfeladat [9 pont]

Kezdetben x kavics van a (0,0) és y kavics a (0,1) mezőn, a többi mező üres. Írj programot, amely a (0,0) mezőre viszi az odometert, ha x<y, egyébként pedig a (0,1) mezőre. (A végén az odometer akármilyen irányba nézhet és minden mezőn akárhány kavics lehet.)

Határok: program méret ≤ 100, végrehajtási idő ≤ 1 000.

2. részfeladat [12 pont]

A feladat ugyanaz, mint az előzőben, de a program végén a (0,0) mezőn pontosan x, a (0,1) mezőn pontosan y kavicsnak kell lenni.

Határok: program méret ≤ 200, végrehajtási idő ≤ 2 000.

3. részfeladat [19 pont]

Pontosan 2 kavics van valahol a 0. sorban. Az egyik a (0,x), a másik a (0,y) mezőkön, ahol x és y különböző, x+y páros. Írj programot, amely az odometert (0, (x + y) / 2) mezőre viszi, vagyis a két kavics közötti középső mezőre. A végén bárhol bármennyi kavics lehet.

Határok: program méret ≤ 100, végrehajtási idő ≤ 200 000.

4. részfeladat [32 pont]

Legfeljebb 15 kavics van a mezőkön, minden mezőn legfeljebb 1. Írj programot, amely összegyűjti az összeset a bal felső sarokba, a végén máshol nem lehet kavics.

A részfeladat pontszáma a beküldött program végrehajtási idejétől függ. Ha a maximális végrehajtási idő az összes tesztesetre L, akkor a pontszámod:

- 32 pont ha $L \leq 200\,000$;
- $32 - 32 \log_{10} (L / 200\,000)$ pont, ha $200\,000 < L < 2\,000\,000$;
- 0 pont, ha $L \geq 2\,000\,000$.

Határok: program méret ≤ 200

5. részfeladat [28 pont]

Bármely mezőn akárhány (0 és 15 közötti) kavics lehet. Írj programot, amely az odometert a legkevesebb kavicsot tartalmazó mezőre viszi (ha több ilyen van, akkor bármelyikre)! A végén minden mezőn ugyanannyi kavicsnak kall lenni, mint kezdetben volt.

A pontszámod a program P méretétől függ:

- 28 pont, ha $P \leq 444$;
- $28 - 28 \log_{10} (P / 444)$ pont, ha $444 < P < 4\,440$;
- 0 pont, ha $P \geq 4\,440$.

Határok: végrehajtási idő ≤ 44 400 000.

Megvalósítás

Részfeladatonként 1-1 szabályos programot tartalmazó file-t kell beküldened. A méretük legfeljebb 5 MiB lehet. Mindegyiket több tesztesettel tesztelik és visszajelzést kapsz a futási időről és a program méretről. Szintaktikusan hibás program esetén hibajelzést kapsz.

Nem kell egyszerre az összes részfeladat megoldását beküldeni. Hiányzó részfeladat esetén a legutolsó beküldést veszi az értékelő. Ha nem volt, akkor 0 pontot kapsz rá.

A pontszám a részpontszámok összege, a végső pontszám a "release"-beküldések és az utolsó beküldés maximuma.

Szimulátor

Kapsz egy odometer szimulátort, amivel odometer programjaid tesztelheted.

A négyzetrács leírása a következő formájú: minden sorban három szám van: R, C és P. Jelentése: az (R,C) mező P kavicsot tartalma. Másutt nincs kavics.

```
0 10 3
4 5 12
```

A négyzetrács 15 kavicsot tartalmaz: 3-at a (0,10), 12-t a (4,5) mezőn.

A teszt szimulátort a feladat könyvtárban a `simulator.py` paranccsal kell hívni. A szimulátornak az alábbi parancssori paramétereket lehet adni:

- `-h` a használható utasítások listázása;

```
-g GRID_FILE betölti a GRID_FILE file-ban levő négyzetrács leírást. (Alapértelmezés az üres négyzetrács.)
```

- `-s GRID_SIDE` a négyzetrács méretét `GRID_SIDE x GRID_SIDE`-ra állítja (alapértelmezésben 256); kisebb négyzetráccsal könnyebben tesztelheted a programodat.
- `-m STEPS` legfeljebb `STEPS` számú lépést engedélyez a szimulátornak.
- `-c` C nyelvű programot készít, amelyet lefordíthatsz és futtathatsz. Ez hasznos lehet, ha 10 millió körüli lépésszámú a programod.

Beküldések száma

Legfeljebb 128.