Laatikonetsintä

Tehtävän nimi	Laatikonetsintä
Aikaraja	1 sekunti
Muistiraja	1 gigatavu

Maj työskentelee robotiikkatutkijana Lundin teknillisessä yliopistossa. Hän on saanut tietää arvokkaasta aarteesta yliopiston kellarissa. Aarre on laatikossa, joka on tyhjässä huoneessa syvällä maan alla. Maj ei voi valitettavasti vain mennä etsimään laatikkoa, koska kellarissa on hyvin pimeää ja lampun ottaminen mukaan herättäisi epäilyksiä. Hänen ainoa keinonsa löytää aarre on etäohjata kellarissa asustelevaa robotti-imuria.

Kellari kuvataan $H \times W$ ruudukkona, jossa on rivit 0:sta $H-1$:een (ylhäältä alas) ja sarakkeet 0:sta $W-1$:een (vasemmalta oikealle), eli ylin solu vasemmalla on $(0,0)$ ja alin solu oikealla on $(H-1, W-1)$. Laatikko aarteineen on jossain tuntemattomassa solussa. Joka yö robotti-imuri aloittaa ylävasemmasta kulmasta ja liikkuu ympäri kellaria.

Joka ilta Maj antaa robotille sarjan komentoja merkkijonon muodossa, joiden mukaan robotti liikkuu. Merkkijono koostuu merkeistä: "<", ">", "^" ja "v". Jos robotti sijaitsee solussa (r, c), jonka millään sivulla ei ole estettä, "<" liikuttaa robottia vasemmanpuoleiseen soluun $(r, c-1)$, ">" liikuttaa robottia oikeanpuoleiseen soluun $(r, c+1)$, "^" liikuttaa robottia yläpuolella sijaitsevaan soluun $(r-1, c)$ ja "v" liikuttaa robottia alapuolella sijaitsevaan soluun $(r+1, c)$.

Kellarissa on tukevat seinät, joten jos robotti yrittää liikkua ruudukon ulkopuolelle, mitään ei tapahdu. Laatikko on myös tukevasti paikallaan ja sitä ei voi työntää. Jokaisen illan lopuksi robotti ilmoittaa sijaintinsa ja palaa vasempaan yläkulmaan.

Aikaa ei ole hukattavana, joten Maj päättää löytää laatikon mahdollisimman pienessä määrässä iltoja.

Interaktio

Tämä on interaktiivinen tehtävä.

- Ohjelmasi tulee aloittaa lukemalla kaksi kokonaislukua H ja W: ruudukon korkeus ja leveys.
- Tämän jälkeen ohjelmasi on vuorovaikutuksessa arvostelijan kanssa. Jokaisella vuorovaikutuskierroksella, tulee tulostaa kysymysmerkki "?", jota seuraa epätyhjä

merkkijono s, joka koostuu merkeistä "<", ">", "^", "v". Merkkijonon pituus saa olla enintään $20\,000$. Sitten ohjelmasi tulee lukea kaksi kokonaislukua r, c ($0 \le r \le H-1$, $0 \le c \le W-1$): robotin sijainti komentojen suorittamisen jälkeen. Ota huomioon, että robotti palaa aina komentosarjan suorittamisen jälkeen sijaintiin $(0, 0)$.
- Kun tiedät laatikon sijainnin, tulosta "`!`" ja kaksi kokonaislukua r_b, c_b: laatikon rivi ja sarake ($0 \le r_b \le H-1$, $0 \le c_b \le W-1$). Tämän jälkeen ohjelmasi tulee lopettaa itsensä tekemättä lisää kyselyitä. Tätä viimeistä tulostetta ei lasketa kyselyksi pisteytyksessä.

Varmista, että puskuri tyhjennetään jokaisen kysymyksen jälkeen, ettei ohjelmasi tulkita ylittävän aikarajaa (*Time Limit Exceeded*). Pythonissa `print()` tyhjentää puskurin automaattisesti. C++:ssa `cout << endl;` tyhjentää puskurin tulostamisen lisäksi. Jos käytät printf:ää, käytä lisäksi komentoa `fflush(stdout)`.

Arvostelija *ei ole* adaptiivinen, eli laatikon sijainti päätetään ennen interaktion alkamista.

Rajat ja pisteytys

- $1 \le H, W \le 50$.
- Laatikko ei koskaan sijaitse kohdassa $(0, 0)$. Tämä tarkoittaa, että $H + W \ge 3$.
- Jokainen kysely voi sisältää enintään $20\,000$ komentoa.
- Voit tehdä enintään $2\,500$ kyselyä (lopullisen vastauksen tulostusta ei lasketa kyselyksi).

Ohjelmasi testaukseen käytetään useita testitapauksia. Jos ratkaisusi toimii väärin *yhdessäkään* tapauksessa (esim. tulostaa virheellisen laatikon sijainnin (Wrong Answer), kaatuu (Runtime Error), ylittää aikarajan (Time Limit Exeeded), jne.), saat 0 pistettä ja vastaavan palautteen.

Jos ohjelma onnistuneesti löytää laatikon sijainnin *kaikissa* testitapauksissa, saat palautteen Accepted, ja pisteet lasketaan seuraavasti:

$$\text{pisteet} = \min\left(\frac{100\sqrt{2}}{\sqrt{Q}}, 100\right) \text{ pistettä}$$

Tässä Q on *suurin* kyselyjen määrä kaikista testitapauksista. Lopullisen vastauksen tulostamista ei lasketa kyselyksi. Pisteet pyöristetään lähimpään kokonaislukuun.

Toisin sanoen saadaksesi 100 pistettä, ohjelmasi tulee ratkaista jokainen testitapaus enintään kahdella kyselyllä, $Q = 2$. Taulukko alla näyttää muutamia Q:n arvoja ja niiden tuottamia pisteitä.

Q	2	3	4	5	...	20	...	50	...	2500
Score	100	82	71	63	...	32	...	20	...	3

Testaustyökalu

Ratkaisusi testaamiseksi on käytössäsi yksinkertainen ohjelma, joka on ladattavissa Kattiksen kautta. Ohjelma löytyy Kattiksen tehtäväsivun alalaidasta liitteistä kohdasta "attachments". Testausohjelmaa ei ole pakko käyttää ja sitä saa muuttaa. Muista että kilpailun virallinen testausohjelma on erilainen.

Esimerkki käytöstä arvoilla $H = 4$, $W = 5$, ja laatikko sijainnissa $r = 2$, $c = 3$:

Python-ohjelmille, kuten `solution.py` (tavallisesti suoritettuna `pypy3 solution.py`):

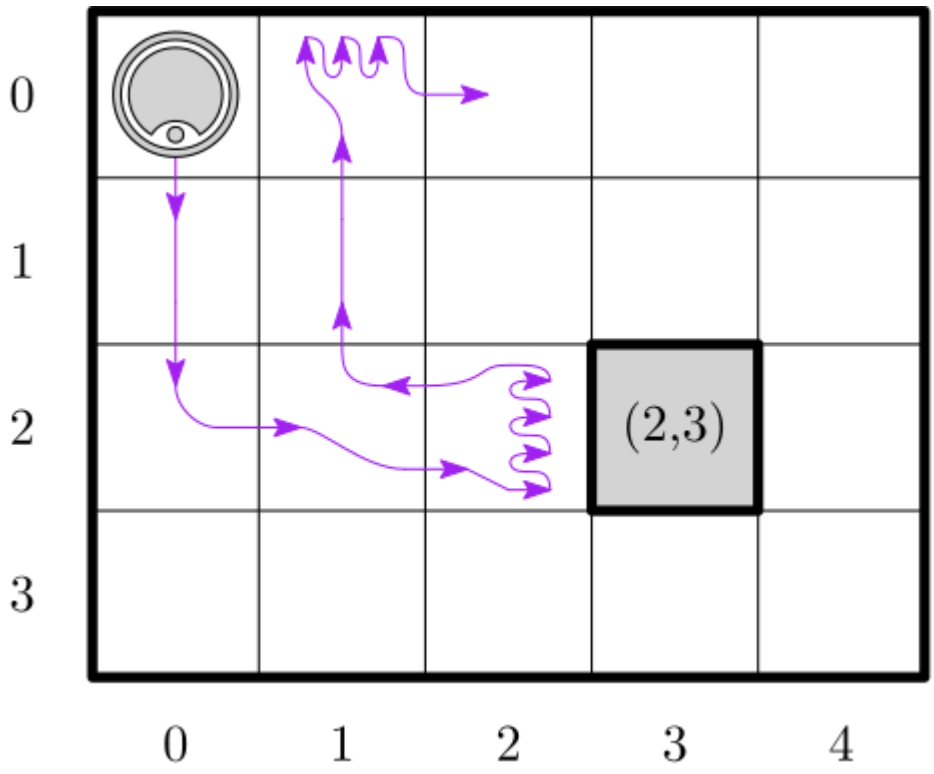
```
python3 testing_tool.py pypy3 solution.py <<<"4 5 2 3"
```

Jos käytät C++:aa, käännä ensin ohjelma (esim. `g++ -std=gnu++17 solution.cpp -o solution.out`) ja sitten suorita:

```
python3 testing_tool.py ./solution.out <<<"4 5 2 3"
```

Esimerkki

Olkoon testitapaus seuraavanlainen: ruudukon korkeus on $H = 4$ ja leveys on $W = 5$, ja laatikko on sijainnissa $(r, c) = (2, 3)$. Kuva alla näyttää robotin reitin, kun se suorittaa kyselyn "`? vv>>>>>> <^^^^^>`" komentoja, joiden johdosta robotti päätyy sijaintiin $(r, c) = (0, 2)$. Ennen toista kyselyä robotti palaa takaisin aloituspaikalleen vasempaan yläkulmaan $(0, 0)$. Sitten ohjelma tekee toisen kyselyn "`? >>>>>>>>>vvvvvvvvvv`", jolla robotti päätyy oikeaan alakulmaan $(r, c) = (3, 4)$. Ohjelma päättää arvata vastauksen "`! 2 3`", joka on laatikon oikea sijainti.



arvostelijan tuloste	ohjelmasi tuloste
4 5	
	? vv>>>>>><^^^^^>
0 2	
	? >>>>>>>>vvvvvvvvv
3 4	
	! 2 3