Svarstyklės

Amina turi 6 monetas sunumeruotas nuo 1 iki 6. Visų monetų svoriai skirtingi. Monetas reikia surikiuoti pagal svorį. Tam Amina naudos neįprastas svertines svarstykles.

Įprastos svertinės svarstyklės turi dvi lėkšteles. Ant kiekvienos lėkštelės dedama po monetą ir svarstyklės parodo, kuri iš monetų sunkesnė.

Aminos svarstyklės sudėtingesnės, turinčios keturias lėkšteles, pažymėtas A, B, C ir D. Svarstyklės turi keturis skirtingus režimus. Norėdama pasinaudoti svarstyklėmis, Amina turi padėti po vieną monetą ant lėkštelių A, B ir C. Jei naudojamas ketvirtas režimas, viena moneta turi būti padėta ir ant lėkštelės D.

Keturi režimai leidžia gauti atsakymus į keturis klausimus:

1. Kuri iš monetų, esančių lėkštelėse A, B ir C yra sunkiausia?
2. Kuri iš monetų, esančių lėkštelėse A, B ir C yra lengviausia?
3. Kuri iš monetų, esančių lėkštelėse A, B ir C yra vidurinė? (t.y., nei sunkiausia, nei lengviausia iš trijų.)
4. Iš lėkštelėse A, B ir C esančių monetų išrenkamos sunkesnės už monetą, esančią lėkštelėje D. Jei tokių monetų yra, kuri iš jų lengviausia? Jei tokių monetų nėra, kuri iš monetų lėkštelėse A, B ir C yra lengviausia?

Užduotis

Parašykite programą, kuri surikiuotų Aminos monetas pagal svorį. Programa gali uždavinėti klausimus Aminos svarstyklėms. Programa turės išspręsti keletą testų, kiekvienas kurių yra šešių monetų rinkinys.

Programoje turi būti realizuotos funkcijos `init` ir `orderCoins`. Kiekvieną kartą vykdant programą, pirmiausia vieną kartą bus iškviečiama `init`. Tai grąžins testų skaičių ir leis inicializuoti kintamuosius. Po to kiekvienam testui bus kviečiama `orderCoins()`.

- `init(T)`
  - `T`: testų skaičius. `T` yra sveikasis skaičius iš intervalo $1, \ldots, 18$.
  - Funkcija nieko negrąžina.
- `orderCoins()`
  - Funkcija iškviečiama po vieną kartą kiekvienam testui.
  - Funkcija turi išrikiuoti monetas pagal svorius naudodama vertintojo funkcijas `getLightest()`, `getHeaviest()`, `getMedian()`, ir/arba `getNextLightest()`.

- Kai funkcija surikiuos monetas, ji turės pateikti sprendinį iškviesdama vertintojo funkciją `answer()`.
- Iškvietusi `answer()`, funkcija `orderCoins()` turi baigti darbą negrąžindama jokios reikšmės.

Šias vertintojo funkcijas galite naudoti savo programoje:

- `answer(C)` — ši funkcija naudojama pateikti rastam sprendiniui.
  - `C`: 6 elementų ilgio masyvas, kuriame yra pagal svorį surikiuotos monetos. Elementuose nuo `C[0]` iki `C[5]` turi būti įrašyti monetų numeriai (t.y., skaičiai nuo 1 iki 6) nuo lengviausios iki sunkiausios.
  - Jūsų programa turi kviesti šią funkciją iš `orderCoins()` po vieną kartą kiekvienam testui.
  - Funkcija nieko negrąžina.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — šios funkcijos atitinka tris pirmuosius Aminos svarstyklių režimus.
  - `A`, `B`, `C`: Ant lėkštelių A, B ir C padėtų monetų numeriai. `A`, `B` ir `C` turi būti trys skirtingi sveikieji skaičiai nuo 1 iki 6.
  - Kiekviena šių funkcijų grąžina vieną iš trijų skaičių: `A`, `B` arba `C` (atitinkamos monetos numerį). Pavyzdžiui, `getHeaviest(A, B, C)` grąžina sunkiausios iš trijų monetų numerį.
- `getNextLightest(A, B, C, D)` — atitinka ketvirtą Aminos svarstyklių režimą.
  - `A`, `B`, `C`, `D`: Ant lėkštelių A, B, C ir D padėtų monetų numeriai. `A`, `B`, `C` ir `D` turi būti keturi skirtingi sveikieji skaičiai nuo 1 iki 6 imtinai.
  - Funkcija grąžina vieną iš skaičių `A`, `B` arba `C`. Tai monetos, gautos atsižvelgiant į ketvirto režimo veikimą, numeris. Tai yra, grąžinamas arba lengviausios iš monetų, esančių ant lėkštelių A, B ir C ir sunkesnių nei moneta ant lėkštelės D, numeris; kitu atveju, jei nei viena šių monetų nėra sunkesnė už monetą ant lėkštelės D, grąžinamas lengviausios iš monetų, esančių ant lėkštelių A, B ir C, numeris.

Vertinimas

Dalinių užduočių nėra. Taškai priklausys nuo svėrimų, t.y., bendro funkcijų `getLightest()`, `getHeaviest()`, `getMedian()` ir/arba`getNextLightest()`, iškvietimo skaičiaus.

Programa bus vykdoma daug kartų, kiekvieną kartą — su daug testų. Duomenų rinkinių skaičių pažymėkime r. Šis skaičius yra fiksuotas (t.y., priklauso nuo duomenų). Jei programa monetų tinkamai nesurikiuoja, skiriama 0 taškų. Kitu atveju taškai paskiriami žemiau aprašytu būdu:

Pažymėkime Q mažiausią svėrimų skaičių, būtiną norint surikiuoti bet kurią 6 monetų seką. Sąlygoje nepateiksime, kam lygu Q.

Pažymėkime $Q + y$ (kur y yra kažkoks sveikasis skaičius) didžiausią svėrimų skaičių, gautą vykdant programą su visais duomenų rinkiniais (viename rinkinyje gali būti daug testų).

Analizuokime jūsų programos vykdymą su vienu duomenų rinkiniu. Pažymėkime didžiausią svėrimų skaičių duomenų rinkinyje $Q + x$, kur x yra sveikas neneigiamas skaičius. Jei svėrimų skaičius mažesnis nei Q, tuomet $x = 0$. Taškai, skirti už programos vykdymą su šiuo duomenų rinkiniu bus apskaičiuojami pagal formulę $\frac{100}{r((x+y)/5+1)}$, suapvalinti *į mažesniąją pusę* iki dviejų skaitmenų po kablelio.

Tai reiškia, kad jei Jūsų programa kiekvienam testui naudos ne daugiau Q svėrimų, surinksite 100 taškų.

Pavyzdys

Tarkime, kad monetų rinkinys toks: **3 4 6 2 1 5** (čia monetos surikiuotos nuo lengviausios iki sunkiausios).

Kreipinys	Grąžina	Paaiškinimas
getMedian(4, 5, 6)	6	Moneta **6** vidurinė pagal svorį iš monetų **4**, **5** ir **6**.
getHeaviest(3, 1, 2)	1	Moneta **1** yra sunkiausia iš monetų **1**, **2** ir **3**.
getNextLightest(2, 3, 4, 5)	3	Monetos **2**, **3** ir **4** yra lengvesnės už monetą **5**, taigi lengviausia iš jų yra **3**.
getNextLightest(1, 6, 3, 4)	6	Abi monetos **1** ir **6** yra sunkesnės nei moneta **4**. Iš monetų **1** ir **6**, moneta **6** yra lengvesnė.
getHeaviest(3, 5, 6)	5	Moneta **5** yra sunkiausia iš monetų **3**, **5** ir **6**.
getMedian(1, 5, 6)	1	Moneta **1** yra vidurinė iš monetų **1**, **5** ir **6**.
getMedian(2, 4, 6)	6	Moneta **6** yra vidurinė iš monetų **2**, **4** ir **6**.
answer([3, 4, 6, 2, 1, 5])		Programa rado teisingą sprendinį šiam testui.

Pavyzdinis vertintojas

Pavyzdinis vertintojas duomenis skaito tokiu formatu:

- eilutė 1: T — testų skaičius
- eilutės nuo 2 iki $T + 1$: 6 skirtingų sveikųjų skaičių nuo 1 iki 6 seka: tai monetų, išrikiuotų nuo lengviausios iki sunkiausios, numeriai.

Pavyzdžiui, pradinius duomenis sudaro du testai (**1 2 3 4 5 6** ir **3 4 6 2 1 5**):

```
2
1 2 3 4 5 6
3 4 6 2 1 5
```

Pavyzdinis vertintojas išveda masyvą, kuris buvo pateiktas kaip parametras funkcijai `answer()`.