



## Vage

Amina ima šest novčića, označenih od **1** do **6**. Ona zna da nikoja dva novčića nemaju istu težinu. Željela bi da ih sortira prema njihovoj težini. Za tu priliku osmislila je novu vrstu vage.

Tradicionalna vaga ima dva tasa. Koristi se tako što se postavi novčić na svaki tas i vaga odredi koji je novčić teži.

Aminina nova vaga je komplikovanija. Ona ima četiri tasa, označena sa **A**, **B**, **C** i **D**. Vaga ima četiri različita podešavanja, od kojih svako odgovora na različito pitanje u vezi sa novčićima. Da bi koristila vagu Amina mora postaviti tačno jedan novčić na svaki od tasova **A**, **B** i **C**. Dodatno, pri četvrtom podešavanju takođe mora postaviti jedan novčić na tas **D**.

Četiri podešavanja daju naredbu vagi da odgovori na sljedeća pitanja:

1. Koji od novčića na tasovima **A**, **B** i **C** je najteži?
2. Koji od novčića na tasovima **A**, **B** i **C** je najlakši?
3. Koji od novčića na tasovima **A**, **B** i **C** je srednji po težini?
4. Među novčićima na tasovima **A**, **B** i **C**, posmatrajmo samo one novčiće koji su teži od novčića na tasu **D**. Ukoliko postoji bar jedan takav, koji od njih je najlakši? U suprotnom, ukoliko ne postoji takvih novčića, koji od novčića na tasovima **A**, **B** i **C** je najlakši?

## Zadatak

Napisati program koji će sortirati Amininih šest novčića prema njihovoj težini. Program može zadavati upite Amininoj vagi da bi poredio težine novčića. Vašem programu će biti dato da riješi nekoliko test primjera, pri čemu svakom odgovara novi skup novčića.

Vaš program treba da implementira funkcije `init` i `orderCoins`. Pri svakom izvršavanju vašeg programa, grader će prvo pozvati `init` tačno jednom. To će vam dati broj test primjera i mogućnost da inicijalizirate neke promenljive. Grader će nakon toga pozivati `orderCoins()`, po jednom za svaki test primjer.

- `init(T)`
  - `T`: Broj test primjera koji vaš program treba da riješi prilikom ovog izvršavanja. `T` je cijeli broj iz intervala **1, ..., 18**.
  - Ova funkcija nema povratnu vrijednost.
- `orderCoins()`
  - Ova funkcija se poziva tačno jednom za svaki test primjer.
  - Ova funkcija treba da odredi tačan redoslijed Amininih novčića pozivajući funkcije gradera

`getHeaviest()`, `getLightest()`, `getMedian()` i/ili `getNextLightest()`.

- Jednom kada ova funkcija shvati koji je tačan redoslijed, treba da ga saopšti pozivajući funkciju `gradera answer()`.
- Nakon pozivanja `answer()`, funkcija `orderCoins()` treba da se završi. Ova funkcija nema povratnu vrijednost.

U vašem programu možete koristiti sljedeće funkcije `gradera`:

- `answer(W)` — vaš program treba da koristi ovu funkciju da saopšti odgovor koji je pronašao.
  - `W`: Niz dužine 6 koji sadrži tačan redoslijed novčića. `W[0]` do `W[5]` treba da budu oznake novčića (dakle, brojevi od **1** do **6**) u redoslijedu od najlakšeg do najtežeg novčića.
  - Ovu funkciju vaš program smije da poziva samo iz `orderCoins()`, jednom za svaki test primjer.
  - Ova funkcija nema povratnu vrijednost.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — ove funkcije odgovaraju podešavanjima 1, 2 i 3 Aminine vage, respektivno.
  - `A, B, C`: Novčići koji su stavljani na tasove **A**, **B** i **C**, respektivno. `A, B` i `C` moraju biti tri različita cijela broja, svaki između **1** i **6** uključujući i **1** i **6**.
  - Svaka funkcija vraća jedan od brojeva `A, B` i `C`: oznaku odgovarajućeg novčića. Na primjer `getHeaviest(A, B, C)` vraća oznaku najtežeg od tri data novčića.
- `getNextLightest(A, B, C, D)` — ovo odgovara podešavanju 4 Aminine vage.
  - `A, B, C, D`: Novčići stavljani na tasove **A**, **B**, **C** i **D**, respektivno. `A, B, C` i `D` moraju biti četiri različita broja, svaki između **1** i **6** uključujući i **1** i **6**.
  - Funkcija vraća jedan od brojeva `A, B` i `C`: oznaku novčića koji je vaga dala kao odgovor pri podešavanju 4. Tj. vraćeni novčić je najlakši od onih novčića na tasovima **A**, **B** i **C** koji su teži od novčića na tasu **D**; ili, ukoliko nijedan od njih nije teži od novčića na tasu **D**, vraćeni novčić je najlakši od tri novčića na tasovima **A**, **B**, and **C**.

## Bodovanje

U ovom problemu ne postoje potproblemi. Umjesto toga vaš broj bodova će biti baziran na broju vaganja (ukupan broj poziva funkcija `gradera` `getLightest()`, `getHeaviest()`, `getMedian()` i/ili `getNextLightest()`) koje vaš program napravi.

Vaš program će biti izvršavan više puta sa više test primjera u svakom izvršavanju. Neka je  $r$  broj izvršavanja vašeg programa. Ova broj je fiksiran u test podacima. Ako vaš program ne nađe ispravan redoslijed novčića u bilo kom test primjeru bilo kog izvršavanja, dobit ćete 0 poena. U suprotnom, svako izvršavanje se boduje pojedinačno na sljedeći način.

Neka je  $Q$  najmanji broj takav da je moguće sortirati bilo koji niz šest novčića sa  $Q$  vaganja na Amininoj vagi. Da bi vas dodatno namučili, nećemo vam ovdje otkriti vrijednost broja  $Q$ .

Pretpostavimo da je najveći broj vaganja među svima test primjerima u svim izvršavanjima jednak  $Q + y$  za neki cijeli broj  $y$ . Sada posmatrajmo pojedinačno izvršavanje vašeg programa. Neka je

najveći broj vaganja među svih  $T$  test primjera u tom izvršavanju jednak  $Q + x$  za neki nenegativan cijeli broj  $x$ . (Ako koristite manje od  $Q$  vaganja u svakom test primjeru, onda je  $x = 0$ .) Konačno, broj bodova za ovo izvršavanje iznosit će  $\frac{100}{r((x+y)/5+1)}$ , zaokružen *na dole* na dvije decimale.

Na primjer, ako vaš program napravi najviše  $Q$  vaganja u svakom test primjeru svakog izvršavanja, dobijate 100 bodova.

## Primjer

Pretpostavimo da su novčići poredani u redosljedu **3 4 6 2 1 5** od najlakšeg do najtežeg.

Poziv funkcije	Povratna vrijednost	Objašnjenje
getMedian(4, 5, 6)	6	Novčić <b>6</b> je srednji po težini među novčićima <b>4, 5</b> i <b>6</b> .
getHeaviest(3, 1, 2)	1	Novčić <b>1</b> je najteži među novčićima <b>1, 2</b> i <b>3</b> .
getNextLightest(2, 3, 4, 5)	3	Novčići <b>2, 3</b> i <b>4</b> su svi lakši od novčića <b>5</b> , pa je najlakši među njima ( <b>3</b> ) vraćen.
getNextLightest(1, 6, 3, 4)	6	Novčići <b>1</b> i <b>6</b> su oba teža od novčića <b>4</b> . Među novčićima <b>1</b> i <b>6</b> , novčić <b>6</b> je najlakši.
getHeaviest(3, 5, 6)	5	Novčić <b>5</b> je najteži među novčićima <b>3, 5</b> i <b>6</b> .
getMedian(1, 5, 6)	1	Novčić <b>1</b> je srednji po težini među novčićima <b>1, 5</b> i <b>6</b> .
getMedian(2, 4, 6)	6	Novčić <b>6</b> je srednji po težini među novčićima <b>2, 4</b> i <b>6</b> .
answer([3, 4, 6, 2, 1, 5])		Program je pronašao tačan odgovor za ovaj test primjer.

## Lokalni grader

Lokalni grader čita ulazne podatke u sljedećem formatu:

- linija **1**:  $T$  — broj test primjera
- svaka od linija **2** do  $T + 1$ : niz od **6** različitih brojeva od **1** do **6**: redosljed novčića od najlakšeg do najtežeg.

Na primjer, ulaz koji se sastoji od dva test primjera u kojima su novčići u redosljedima **1 2 3 4 5 6** i **3 4 6 2 1 5** izgleda na sljedeći način:

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

Grader ispisuje niz koji je proslijeđen kao parametar funkciji `answer()`.