Կշեռք (Scales)

Ամինան ունի վեց մետաղադրամ՝ համարակալված 1-ից 6 թվերով: Նա գիտի, որ բոլոր մետաղադրամների քաշերը տարբեր են: Նա ցանկանում է դասավորել դրանք ըստ իրենց քաշի: Դրա համար նա ստեղծել է նոր տիպի նժարավոր կշեռք:

Ավանդական նժարավոր կշեռքն ունի երկու նժար: Այդ կշեռքից օգտվելու համար յուրաքանչյուր նժարին մեկական մետաղադրամ են դնում և պարզում, թե նրանցից որն է ավելի ծանր:

Ամինայի նոր կշեռքն ավելի բարդ է: Այն ունի չորս նժար, որոնք նշված են A, B, C և D տառերով: Կա կշեռքի չորս տարբեր կարգավորում, որոնցից յուրաքանչյուրի դեպքում կարելի է ստանալ մետաղադրամների վերաբերյալ մեկ հարցի պատասխան: Կշեռքն օգտագործելիս առաջին երեք կարգավորումների դեպքում Ամինան պետք է ճիշտ մեկական մետաղադրամ տեղադրի A, B և C նժարների վրա, իսկ չորրորդ կարգավորման դեպքում՝ նաև ճիշտ մեկ մետաղարդամ D նժարի վրա:

Այդ չորս կարգավորումների միջոցով կարելի է ստանալ հետևյալ չորս հարցերի պատասխանները.

1. A, B և C նժարների վրա դրված մետաղադրամներից ո՞րն է ամենածանրը:
2. A, B և C նժարների վրա դրված մետաղադրամներից ո՞րն է ամենաթեթևը:
3. A, B և C նժարների վրա դրված մետաղադրամներից ո՞րն է միջինը: (Միջին ասելով այստեղ հասկանում ենք այն մետաղադրամը, որը երեքից ո՛չ ամենածանրն է, ո՛չ էլ ամենաթեթևը):
4. A, B և C նժարների վրա դրված մետաղադրամներից դիտարկենք միայն նրանք, որոնք ավելի ծանր են, քան D նժարի վրա դրված կշռաքարը: Եթե կան այդպիսի մետաղադրամներ, ո՞րն է նրանցից ամենաթեթևը: Հակառակ դեպքում, այսինքն, եթե այդպիսի մետաղադրամներ չկան, A, B և C նժարներին դրված մետաղադրամներից ո՞րն է ամենաթեթևը:

Խնդիր

Գրեք ծրագիր, որը կդասավորի Ամինայի վեց մետաղադրամներն ըստ իրենց քաշի: Ծրագիրը կարող է մետաղադրամների քաշերի վերաբերյալ հարցումներ անել Ամինայի կշեռքին՝ մետաղադրամների կշիռները համեմատելու համար: Ձեր ծրագրին կտրվեն մի քանի թեստեր, որոնցից յուրաքանչյուրը կհամապատասխանի վեց մետաղադրամների մի նոր բազմության:

Ձեր ծրագրում պետք է իրականացված լինեն `init` և `orderCoins` ֆունկցիաները: Ամեն անգամ ձեր ծրագիրն աշխատացնելուց գրեյդերը նախ ճիշտ մեկ անգամ կկանչի `init` ֆունկցիան: Այն ձեզ կտա թեստերի քանակը և հնարավորություն՝

կատարել փոփոխականների սկզբնարժեքավորում: Դրանից հետո գրեյդերը յուրաքանչյուր թեստի համար կկանչի `orderCoins()` ֆունկցիան:

- `init(T)`
  - `T`: Ծրագրի կատարման ընթացքում տրվող թեստերի քանակը: `T`-ն $1, \ldots, 18$ տիրույթին պատկանող ամբողջ թիվ է:
  - Այս ֆունկցիան ոչինչ չի վերադարձնում:
- `orderCoins()`
  - Այս ֆունկցիան յուրաքանչյուր թեստի համար կանչվում է ճիշտ մեկ անգամ:
  - Այս ֆունկցիան պետք է դասավորի Ամինայի կշռաքարերը օգտագործելով գրեյդերի հետևյալ ֆունկցիաները՝ `getHeaviest()`, `getLightest()`, `getMedian()` և `getNextLightest()`:
  - Հենց որ ֆունկցիան պարզի մետաղադրամների ճիշտ կարգը, նա պետք է այդ մասին հաղորդի գրեյդարին `answer()` ֆունկցիայի միջոցով:
  - `answer()` ֆունկցիան կանչելուց հետո `orderCoins()` ֆունկցիան պետք է ավարտվի: Այն ոչինչ չի վերադարձնում:

Դուք ձեր ծրագրում կարող եք օգտագործել գրեյդերի հետևյալ ֆունկցիաները.

- `answer(W)` — ձեր ծրագիրը պետք է օգտագործի այս ֆունկցիանն գտնված պատասխանը գրեյդերին հաղորդելու համար:
  - `W`: 6 երկարության զանգված, որը պարունակում է մետաղադրամների ճիշտ դասավորությունը: `W[0]`-ից `W[5]`-ում պետք է պահել մետաղադրամների համարները (այսինքն, 1-ից 6 թվեր) մետաղադրամների քաշերի աճման կարգով:
  - Ձեր ծրագիրը պետք է կանչի այս ֆունկցիան միայն `orderCoins()` ֆունկցիայի ներսում, յուրաքանչյուր թեստի համար ճիշտ մեկ անգամ:
  - Այս ֆունկցիան վերադարձի արժեք չունի:
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — սրանք համապատասխանում են Ամինայի կշեռքի 1, 2 և 3 կարգավորումներին:
  - `A`, `B`, `C`: A, B և C նժարներին դրված մետաղադրամների համարները: `A`-ն, `B`-ն և `C`-ն պետք է լինեն զույգ առ զույգ իրարից տարբեր ամբողջ թվեր՝ 1-ից և 6-ը ներառյալ:
  - Նշված ֆունկցիաները վերադարձնում են `A`, `B` և `C` թվերից մեկը՝ համապատասխան մետաղադրամի համարը: Օրինակ, `getHeaviest(A, B, C)` ֆունկցիան վերադարձնում է երեք մետաղադրամներից մեծագույնի համարը:
- `getNextLightest(A, B, C, D)` — սա համապատասխանում է Ամինի կշեռքի 4-րդ կարգավորմանը:
  - `A`, `B`, `C`, `D`. Համապատասխանաբար A, B, C և D նժարներին դրված

մետաղադրամների համարները: `A`-ն, `B`-ն, `C`-ն և `D`-ն պետք է լինեն 1-ից 6 տիրույթին պատկանող իրարից տարբեր ամբողջ թվեր:

- Այս ֆունկցիան վերադարձնում է `A`, `B` և `C` թվերից մեկը՝ կշեռքի գործածման 4-րդ կարգավորման դեպքում ընտրված մետաղադրամի համարը: Այսինքն, եթե A, B և C նժարների վրա դրված մետաղադրամների մեջ կան D նժարի վրա դրված մետաղադրամից ծանր մետաղադրամներ, վերադարձնում է դրանցից ամենաթեթևի համարը: Եթե այդպիսի մետաղադրամ չկա, պարզապես վերադարձնում է A, B և C նժարների վրա դրված մետաղադրամներից թեթևագույնի համարը:

Միավորներ

Այս խնդրում ենթախնդիրներ չկան: Ձեր ծրագիրը միավոր կստանա կախված նրանից, թե քանի անգամ կշռում կկատարի (այսինքն, քանի անգամ կկանչի `getLightest()`, `getHeaviest()`, `getMedian()` և/կամ `getNextLightest()` ֆունկցիաները):

Ձեր ծրագիրը կաշխատացվի մի քանի անգամ, յուրաքանչյուր անգամ որոշակի քանակությամբ թեստերի վրա: Դիցուք, r-ը ցույց է տալիս, թե ձեր ծրագիրը քանի անգամ է աշխատացվել: Այս թիվը ֆիքսված է թեստային տվյալներում: Եթե ձեր ծրագիրն ամեն անգամ աշխատացնելուց գտնվի գոնե մեկ թեստ, որի դեպքում ծրագիրը սխալ կդասավորի մետաղադրամները կտրվի 0 միավոր: Հակառակ դեպքում միավորի հաշվարկը կատարվում է հետևյալ կերպ.

Դիցուք, Q-ն Ամինայի կշեռքի միջոցով ցանկացած վեց տարբեր մետաղադրամների հաջորդականություն կարգավորելու համար մինիմալ կշռումների քանակն է: Խնդիրն ավելի դժվար դարձնելու համար մենք այստեղ չենք բացահայտում Q-ի արժեքը:

Ենթադրենք ընդհանրապես բոլոր թեստերի վրա ձեր ծրագիրն աշխատացնելուց մաքսիմալ կշռումների քանակը $Q + y$ է, որտեղ y-ը ամբողջ թիվ է: Այդ դեպքում դիտարկենք ձեր ծրագրի մեկ գործարկում: Դիցուք այս անգամ ծրագիրը աշխատացվել է T թեստերի վրա, և դրանցից մաքսիմալ կշռումների քանակը $Q + x$ է, որտեղ x-ը ոչ բացասական ամբողջ թիվ է: (Եթե դուք բոլոր թեստերում կատարում եք Q-ից քիչ կշռումներ, ապա $x = 0$): Այդ դեպքում ծրագրի այս գործարկման համար կտրվի $\frac{100}{r((x+y)/5+1)}$ միավոր, կլորացրած դեպի *ներքև* ստորակետից հետո երկու նիշով:

Մասնավորապես, եթե ձեր ծրագիրը ամեն անգամ աշխատացնելուց յուրաքանչյուր թեստի համար կատարի առավելագույնը Q կշռում, դուք կստանաք 100 միավոր:

Օրինակ

Ենթադրենք մետաղադրամներն ըստ քաշերի աճման կարգի դասավորված են հետևյալ կերպ. $3\ 4\ 6\ 2\ 1\ 5$

Ֆունկցիայի կանչ	Վերադարձվող արժեք	Բացատրություն

Ֆունկցիայի կանչ	Վերադարձվող արժեք	Բացատրություն
getMedian(4, 5, 6)	6	6-րդ մետաղադրամը 4-րդ, 5-րդ և 6-րդ մետաղադրամներից միջինն է:
getHeaviest(3, 1, 2)	1	1-ին մետաղադրամը 1-ին, 2-րդ և 3-րդ մետաղադրամներից ամենածանրն է:
getNextLightest(2, 3, 4, 5)	3	2-րդ, 3-րդ և 4-րդ մետաղադրամները are all lighter than coin 5-րդ մետաղադրամից թեթև են, հետևաբար վերադարձվում է նրանցից ամենաթեթևը (3-րդը):
getNextLightest(1, 6, 3, 4)	6	1-ն և 6-րդ մետաղադրամներն ավելի ծանր են, քան 4-րդը: Նրանցից թեթևը 6-րդն է:
getHeaviest(3, 5, 6)	5	5-րդ մետաղադրամը 3-րդ, 5-րդ և 6-րդ մետաղադրամներից ամենածանրն է:
getMedian(1, 5, 6)	1	1-ն մետաղավրամը 1-ին, 5-րդ և 6-րդ մետաղադրամներից միջինն է:
getMedian(2, 4, 6)	6	6-րդ մետաղադրամը 2-րդ, 4-րդ և 6-րդ մետաղադրամներից միջինն է:
answer([3, 4, 6, 2, 1, 5])		Ծրագիրն այս թեստի համար գտավ ճիշտ պատասխան:

Sample grader

The sample grader reads input in the following format:

- line 1: T —- the number of test cases
- each of the lines from 2 to $T + 1$: a sequence of 6 distinct numbers from 1 to 6: the order of the coins from the lightest to the heaviest.

For instance, an input that consists of two test cases where the coins are ordered $1\ 2\ 3\ 4\ 5\ 6$ and $3\ 4\ 6\ 2\ 1\ 5$ looks as follows:

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

The sample grader prints the array that was passed as a parameter to the `answer()` function.