Viteški turnir

Za svoje venčanje sa caricom Milicom, Vojvoda od Niša Ivan Stošić tražio je od Demjana (ratara) da organizuje proslavu venčanja koja uključuje veliki viteški turnir koji će trajati cela tri dana. Međutim, najpopulariji vitez kasni...

Turnir

Na viteškom turniru, N vitezova se poređa u jednu vrstu i njihove pozicije u ovoj vrsti se numerišu, redom, od 0 do N - 1. Sudija turnira započinje jednu *rundu* tako što najavi dve pozicije S i E (gde je $0 \leq S < E \leq N - 1$). Svi vitezovi čije su pozicije između S i E (uključujući i S i E) se takmiče u toj rundi: pobednik runde nastavlja turnir i vraća se nazad na svoje mesto u vrsti, dok svi gubitnici ispadaju iz turnira i napuštaju vrstu. Nakon toga, svi preostali vitezovi se pomeraju (pakuju) ka početku vrste, zadržavajući originalni redosled i njihove nove pozicije su od 0 do N - (E - S) - 1. Nakon toga, sudija započinje sledeću rundu, ponavljajući ovaj postupak sve dok ne ostane samo jedan vitez.

Ratar Demjan zna da svi vitezovi imaju različitu snagu. Snaga viteza je predstavljena celim brojem od 0 (najslabiji) do N - 1 (najjači) i svi ovi brojevi su različiti. On takođe zna koje će naredbe izdavati sudija turnira za C rundi (on je ipak ratar na kralju kraljeva) i siguran je da će u svakoj rundi pobediti najjači vitez koji učestvuje u toj rundi.

Vitez koji kasni

N - 1 vitezova je već raspoređeno u vrstu, jedino nedostaje najpopularniji vitez. Taj vitez ima snagu R i dolazi nešto kasnije. Da bi turnir učinio zanimljivijim, Demjan želi da iskoristi popularnost ovog viteza i izabere za njega poziciju u vrsti tako da maksimizira broj rundi u kojima će "vitez koji kasni" pobediti. Primetimo da runde u kojima vitez koji kasni ne učestvuje nisu od interesa, nego samo runde u kojima on učestvuje i pobeđuje.

Primer

Za N = 5, N - 1 vitezova već je u vrsti i imaju, redom, snage [1, 0, 2, 4]. Jasno je da vitez koji kasni ima snagu R = 3. Sudija planira da u C = 3 rundi najavi pozicije (S, E) u sledećem redosledu po rundi: (1, 3), (0, 1), (0, 1).

Ako Demjan (ratar) ubaci viteza koji kasni na prvu poziciju (broj 0), snage vitezova u vrsti će biti [3, 1, 0, 2, 4]. Prva runda uključuje vitezove (na pozicijama 1, 2, 3) sa snagama 1, 0, 2, pa je pobednik vitez sa snagom 2. Raspored snaga vitezova posle formiranja nove vrste je [3, 2, 4]. Sledeća runda je 3 protiv 2 (na pozicijama 0 , 1), pa pobeđuje vitez sa snagom R = 3, što ostavlja vrstu [3, 4]. U poslednjoj rundi (vitezovi na pozicijama 0, 1) pobeđuje vitez sa snagom 4. Prema

tome, vitez koji kasni će pobediti u samo jednoj rundi (drugoj).

Međutim, ako Demajn (ratar po zanimanju) ubaci viteza koji kasni između vitezova sa snagama 1 i 0, vrsta će izgledati ovako: [1, 3, 0, 2, 4]. Sada, prva runda uključuje vitezove sa snagama 3, 0, 2, i vitez sa snagom R = 3 pobeđuje. Vrsta sada izgleda [1, 3, 4] i u sledećoj rundi (1 protiv 3), vitez sa snagom R = 3 ponovo pobeđuje. Krajnja vrsta je [3, 4] gde 4 pobeđuje. Prema tome, vitez koji kasni je pobedio u dve runde: ovo je zapravo najbolja moguća pozicija s obzirom na to da vitez koji kasni ne može nikako pobediti u više od dve runde.

Postavka problema

Vaš zadatak je da napišete program koji bira najbolju moguću poziciju za viteza koji kasni, tako da je broj rundi u kojima on pobeđuje najveći, kao što naš ratar želi. Konkretno, potrebno je implementirati funkciju `GetBestPosition`(N, C, R, K, S, E), gde:

- N je broj vitezova;
- C je broj rundi koje najavljuje sudija ($1 \leq C \leq N - 1$);
- R je snaga viteza koji kasni; snage svih vitezova (kako onih koji su već u vrsti tako i onog koji kasni) su međusobno različite i iz skupa 0, …, N - 1; snaga R viteza koji kasni data je eksplicitno iako se može i zaključiti na osnovu niza K;
- K je niz od N - 1 celih brojeva koji predstavljaju snage N - 1 viteza koji se već nalaze u vrsti;
- S i E su dva niza dužine C: za svako i između 0 i C - 1, (uključujući i 0 i C - 1), runda i + 1 koju najavljuje sudija će uključivati sve vitezove od pozicije S[i] do pozicije E[i], uključivo (u trenutnoj vrsti). Možete pretpostaviti da će za svako i važiti S[i] < E[i].

Svi parametri ove funkcije su ispravni: E[i] je manje od trenutnog broja vitezova preostalih u rundi i + 1, a nakon C rundi ostaće tačno jedan vitez.

`GetBestPosition`(N, C, R, K, S, E) mora vratiti najbolju poziciju P gde Demjan (gorepomenuti ratar) treba da ubaci viteza koji kasni ($0 \leq P \leq N - 1$). Ako postoji više takvih pozicija, *vratite najmanju*. (Pozicija P je 0-indeksirana. Drugim rečima, P je broj drugih vitezova koji stoje u vrsti ispred viteza koji kasni u optimalnom rešenju. Specijalno, P = 0 znači da je vitez koji kasni na početku vrste, dok P = N - 1 znači da je na kraju vrste.)

Podzadatak 1 [17 bodova]

Možete pretpostaviti da je $N \leq 500$.

Podzadatak 2 [32 boda]

Možete pretpostaviti da je $N \leq 5\,000$.

Podzadatak 3 [51 bodova]

Možete pretpostaviti da je $N \leq 100\,000$.

Detalji implementacije

Morate predati tačno jedan fajl, pod nazivom `tournament.c`, `tournament.cpp` ili `tournament.pas`. Ovaj fajl mora da sadržati implementaciju gore opisane funkcije koristeći sledeća zaglavlja:

C/C++ programi

```
int GetBestPosition(int N, int C, int R, int *K, int *S, int *E);
```

Pascal programi

```
function GetBestPosition(N, C, R : LongInt; var K, S, E : array of LongInt) : LongInt;
```

Ove funkcije se moraju ponašati tačno kako je gore opisano. Naravno, možete implementirati i druge funkcije. Vaš program koji predate ne sme ni na koji način da koristi standardni ulaz/izlaz, niti bilo koji drugi fajl.

Program za testiranje (sample grader)

Program za testiranje koji se nalazi u okviru zadatka očekuje ulaz u sledećem formatu:

- linija 1: N, C, R;
- linije 2, …, N: K[i];
- linije N + 1, …, N + C: S[i], E[i].

Vremenska i memorijska ograničenja

- Vremensko ograničenje: 1 sekunda.
- Memorijsko ograničenje: 256 MiB.