Gondola

Mao-Kong gondola je poznata turistička atrakcija u Taipeiju. Konstrukcija gondole sastoji se od kružne trase, jedne stanice i n gondola označenih redom brojevima od 1 do n koje putuju trasom stalno u istom smjeru. Nakon što gondola s brojem i prođe stanicu, iduća gondola koja će proći bit će gondola $i+1$ osim ako je $i = n$, onda će to biti gondola s brojem 1. Drugim riječima, gondole se voze u krug.

Osim što se voze, gondole se mogu i kvariti. Na svu sreću, prljavo smo bogati i imamo beskonačno zamjenskih gondola koje su označene brojevima $n+1$, $n+2$, $n+3$ i tako do beskonačnosti. Kada se gondola pokvari zamjenimo je sa prvom neiskorištenom zamjenskom gondolom, odnosno, onom koja ima najmanji broj. Na primjer, ako imamo 5 gondola i gondola 1 se pokvari, zamijenit ćemo je gondolom 6.

Vi ste malo čudni, volite stajati na stanici i gledati gondole kako prolaze. Moguće je da se nekoliko gondola pokvarilo i da su zamijenjene prije no što ste došli, ali nijedna se nije pokvarila dok ste vi gledali kako prolaze.

Gondolični niz je niz od n brojeva koji predstavljaju redne brojeve uzastopnih gondola koja prolaze kraj stanice.

Vrijedi primijetiti da od iste konfiguracije gondola na trasi možemo dobiti više različitih gondoličnih nizova, ovisno o tome koja je gondola prva prošla kraj stanice otkad ste vi tamo. Na primjer, ako se niti jedna gondola nije pokvarila onda su $(2, 3, 4, 5, 1)$ i $(4, 5, 1, 2, 3)$ mogući gondolični nizovi, ali niz $(4, 3, 2, 5, 1)$ nije (jer se gondole javljaju u krivom poretku).

Ako se gondola broj 1 pokvari, onda možemo vidjeti gondolični niz $(4, 5, 6, 2, 3)$. Ako se gondola broj 4 pokvari nakon nje, zamjenjujemo je s gondolom broj 7 i tada možemo uočiti gondolični niz $(6, 2, 3, 7, 5)$. Ako se gondola broj 7 pokvari nakon ovoga, zamjenjujemo je s gondolom broj 8 i možemo uočiti gondolični niz $(3, 8, 5, 6, 2)$.

pokvarena gondola	nova gondola	mogući gondolični niz
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

Pokvareni niz je niz koji se sastoji od rednih brojeva gondola koje su se pokvarile, u redoslijedu kojim su se kvarile. U prethodnom primjeru, pokvareni niz je $(1, 4, 7)$. Pokvareni niz r generira gondolični niz g ako nakon kvarova opisanih nizom r možemo uočiti gondolični niz g.

Provjera gondoličnog niza

U prva tri podzadatka morate provjeriti je li zadani niz mogući gondolični niz. Pogledajte tablicu ispod za primjere nizova koji jesu ili nisu mogući gondolični niz. Morate implementirati funkciju `valid`.

- `valid(n, inputSeq)`
  - `n`: duljina zadanog niza
  - `inputSeq`: niz duljine n n; `inputSeq[i]` je i-ti element zadanog niza, za $i \in \{0, 1, \ldots, n-1\}$.
  - Funkcija treba vratiti 1 ako je zadani niz mogući gondolični niz, a 0 ako nije.

Podzadaci 1, 2 i 3

podzadatak	broj bodova	n	`inputSeq`
1	5	$n \leq 100$	svaki broj od 1 do n pojavljuje se točno jednom
2	5	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$

Primjeri

podzadatak	`inputSeq`	povratna vrijednost	napomena
1	(1, 2, 3, 4, 5, 6, 7)	1	
1	(3, 4, 5, 6, 1, 2)	1	
1	(1, 5, 3, 4, 2, 7, 6)	0	1 se ne može pojaviti drito ispred 5
1	(4, 3, 2, 1)	0	4 se ne može pojaviti drito ispred 3
2	(1, 2, 3, 4, 5, 6, 5)	0	dvije gondole označene brojem 5
3	(2, 3, 4, 9, 6, 7, 1)	1	pokvareni niz je (5, 8)
3	(10, 4, 3, 11, 12)	0	4 se ne može pojaviti drito ispred 3

Pokvareni niz

U sljedeća tri podzadatka morate konstruirati neki pokvareni niz koji generira zadani gondolični niz. Bilo koji ispravan pokvareni niz bit će prihvaćen. Morate implementirati funkciju `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` je duljina zadanog gondoličnog niza.
  - `gondolaSeq`: niz duljine n; `gondolaSeq` će sigurno biti gondolični niz, `gondolaSeq[i]` je i-ti element gondoličnog niza, za $i \in \{0, 1, \ldots, n-1\}$.
  - funkcija treba vratiti l, duljinu ispravnog pokvarenog niza.
  - `replacementSeq`: niz koji je dovoljno velik da se u njega spremi ispravan pokvareni niz; vaš odgovor trebate vratiti tako da zapišete i-ti element vašeg pokvarenog niza u `replacementSeq[i]`, za $i \in \{0, 1, \ldots, l-1\}$.

Podzadaci 4, 5 i 6

podzadatak	broj bodova	n	`gondolaSeq`
4	5	$n \leq 100$	$1 \leq$ `gondolaSeq[i]` $\leq n+1$
5	10	$n \leq 1,000$	$1 \leq$ `gondolaSeq[i]` $\leq 5,000$
6	20	$n \leq 100,000$	$1 \leq$ `gondolaSeq[i]` $\leq 250,000$

Primjeri

podzadatak	`gondolaSeq`	povratna vrijednost	`replacementSeq`
4	(3, 1, 4)	1	(2)
4	(5, 1, 2, 3, 4)	0	( )
5	(2, 3, 4, 9, 6, 7, 1)	2	(5, 8)

Prebrojavanje pokvarenih nizova

U iduća četiri podzadatka morate izračunati koliko ima mogućih pokvarenih nizova koji generiraju zadani niz (koji može, ali ne mora biti gondolični niz) modulo $1,000,000,009$. Implementirajte funkciju `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: duljina zadanog niza
  - `inputSeq`: duljina niza n; `inputSeq[i]` je i-ti element zadanog niza, za $i \in \{0, 1, \ldots, n-1\}$.
  - Ako zadani niz je gondolični niz, izračunajte broj pokvarenih nizova koji ga generiraju (što može biti vrlo velik broj) i *vratite ga modulo* $1,000,000,009$.
  - Ako zadani niz nije gondolični niz, vaša funkcija treba vratiti 0.
  - Ako zadani niz je gondolični niz, ali se nijedna gondola nije pokvarila, vaša funkcija treba vratiti 1.

Podzadaci 7, 8, 9 i 10

podzadaci	broj bodova	n	`inputSeq`
7	5	$4 \leq n \leq 50$	$1 \leq$ `inputSeq[i]` $\leq n+3$
8	15	$4 \leq n \leq 50$	$1 \leq$ `inputSeq[i]` ≤ 100, i barem $n-3$ od početnih gondola $1, \ldots, n$ se nije pokvarilo.
9	15	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$
10	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 1,000,000,000$

Primjeri

podzadatak	`inputSeq`	povratna vrijednost	mogući pokvareni nizovi
7	(1, 2, 7, 6)	2	(3, 4, 5) ili (4, 5, 3)
8	(2, 3, 4, 12, 6, 7, 1)	1	(5, 8, 9, 10, 11)
9	(4, 7, 4, 7)	0	`inputSeq` nije gondolični niz
10	(3, 4)	2	(1, 2) ili (2, 1)

Implementacijski detalji

Morate *submitati* točno jednu datoteku, `gondola.c`, `gondola.cpp` ili `gondola.pas`. U ovoj datoteci moraju biti implementirani gore opisani potprogrami (čak i ako želite riješiti samo neki od podzadataka) sa dolje navedenim prototipovima. Također, morate *includeati header file* `gondola.h` za C/C++.

C/C++ program

```
int valid(int n, int inputSeq[]);
int replacement(int n, int gondolaSeq[], int replacementSeq[]);
int countReplacement(int n, int inputSeq[]);
```

Paskal program

```
function valid(n: longint; inputSeq: array of longint): integer;
function replacement(n: longint; gondolaSeq: array of longint;
var replacementSeq: array of longint): longint;
function countReplacement(n: longint; inputSeq: array of longint):
longint;
```

Sample grader

Sample grader prima ulaz sljedećeg oblika:

- 1. linija: `T`, redni broj podzadatka koji želite testirati ($\mathbf{1 \leq T \leq 10}$).
- 2. linija: `n`, duljina zadanog niza.
- 3. linija: ako je `T` 4, 5, ili 6, ova linija sadrži `gondolaSeq[0]`, ..., `gondolaSeq[n-1]`. Inače, ova linija sadrži `inputSeq[0]`, ..., `inputSeq[n-1]`.