Gondola

Telegondola Mao-Kong este o atracție turistică faimoasă în Taiwan. Sistemul telegondolei constă dintr-o șină circulară, o singură stație, și n gondole numerotate consecutiv de la 1 la n care merg circular pe linie într-o direcție fixată. După ce gondola cu numărul i trece prin stație, următoarea gondolă care trece prin stație va fi gondola cu numărul $i + 1$ dacă $i < n$, sau gondola cu numărul 1 dacă $i = n$.

Gondolele pot să se defecteze. Din fericire avem la dispoziție oricâte gondole de rezervă, care sunt numerotate $n + 1$, $n + 2$, și așa mai departe. Când o gondolă se defectează, o înlocuim (în aceeași poziție pe șină) cu prima gondolă de rezervă disponibilă, adică cu cea care are cel mai mic număr. De exemplu, dacă pe șină există 5 gondole și gondola cu numărul 1 se defectează prima, o înlocuim cu gondola cu numărul 6.

Ție îți place să stai în stație și să te uiți cum trec gondolele. O *secvență de gondole* este o secvență de n numere ale gondolelor în ordinea în care trec ele prin stație. Este posibil ca una sau mai multe din gondole să se fi defectat (și să fi fost înlocuite) înainte să fi ajuns tu în stație, dar niciuna din gondole nu se defectează în timpul în care tu te uiți la gondole.

Observați că aceeași configurație a gondolelor pe șină poate produce mai multe secvențe de gondole, în funcție de care gondolă trece prima oară prin stație când ajungi tu acolo. De exemplu, dacă nicuna din gondole nu s-a defectat, atunci (2, 3, 4, 5, 1) și (4, 5, 1, 2, 3) sunt secvențe posibile de gondole, în timp ce (4, 3, 2, 5, 1) nu este (deoarece gondolele apar în ordinea greșită).

Dacă se defectează gondola cu numărul 1, atunci am putea observa secvența de gondole (4, 5, 6, 2, 3). Dacă următoarea care se defectează este gondola cu numărul 4, o înlocuim cu gondola numărul 7 și am putea observa secvența de gondole (6, 2, 3, 7, 5). Dacă apoi se defectează gondola cu numărul 7, o înlocuim cu gondola numărul 8 și am putea observa secvența de gondole (3, 8, 5, 6, 2).

gondola stricată	gondola nouă	secvență posibilă de gondole
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

O *secvență de înlocuiri* este o secvență care conține numerele gondolelor care s-au stricat, în ordinea în care s-au stricat. Pentru exemplul precedent, secvența de înlocuiri este (1, 4, 7). O secvență de înlocuiri r *produce* o secvență de gondole g dacă, după ce gondolele se strică conform cu secvența de înlocuiri r, secvența de gondole g ar putea fi observată.

Verificarea unei secvențe de gondole

Pentru primele trei subprobleme (subtask-uri) trebuie să verificați dacă o secvență dată este o secvență de gondole. În tabelul de mai jos puteți vedea exemple de secvențe care sunt sau nu secvențe de gondole. Trebuie să impementați funcția `valid`.

- `valid(n, inputSeq)`
  - `n`: lungimea secvenței date.
  - `inputSeq`: tablou unidimensional de lungime n; `inputSeq[i]` este elementul i al secvenței date, pentru $0 \leq i \leq n-1$.
  - Funcția trebuie să retuneze 1 dacă secvența dată este o secvență de gondole, sau 0 în caz contrar.

Subproblemele 1, 2, 3

subproblemă	puncte	n	`inputSeq`
1	5	$n \leq 100$	fiecare număr de la 1 la n apare exact o dată
2	5	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$

Exemple

subproblemă	`inputSeq`	valoarea returnată	note
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 nu poate apărea imediat înaintea lui 5
1	(4, 3, 2, 1)	0	4 nu poate apărea imediat înaintea lui 3
2	(1, 2, 3, 4, 5, 6, 5)	0	există două gondole cu numărul 5
3	(2, 3, 4, 9, 6, 7, 1)	1	secvența de înlocuiri este (5, 8)
3	(10, 4, 3, 11, 12)	0	4 nu poate apărea imediat înaintea lui 3

Secvența de înlocuiri

Pentru următoarele trei subprobleme trebuie să contruiți o secvență de înlocuiri care produce o secvență de gondole dată. Orice secvență corectă de înlocuiri va fi acceptată. Trebuie să implementați funcția `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` este lungimea secvenței de gondole.
  - `gondolaSeq`: tablou unidimensional de lungime n; se garantează că `gondolaSeq` este o secvență de gondole, și `gondolaSeq[i]` este elementul cu numărul i din secvență,

pentru $0 \leq i \leq n-1$.

- Funcția trebuie să returneze numărul l, lungimea secvenței de înlocuiri.
- `replacementSeq`: un tablou unidimensional care este suficient de mare pentru a stoca elementele secvenței de înlocuiri; trebuie să transmiteți secvența prin plasarea elementului i al secvenței de înloucuiri găsite de voi în `replacementSeq[i]`, pentru $0 \leq i \leq l-1$.

Subproblemele 4, 5, 6

subproblemă	puncte	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]` $\leq 250,000$

Exemple

subproblemă	`gondolaSeq`	valoare returnată	`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)

Numărarea secvențelor de înlocuiri

Pentru următoarele patru subprobleme trebuie să determinați numărul de secvențe posibile de înlocuiri care produc o secvență (care poate sau nu să fie o secvență de gondole), modulo **1,000,000,009**.
Trebuie să implementați funcția `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: lungimea secvenței date.
  - `inputSeq`: tablou unidimensional de lungime n; `inputSeq[i]` este elementul cu numărul i din secvența dată, pentru $0 \leq i \leq n-1$.
  - Dacă secvența dată este o secvență de gondole, atunci trebuie să determinați numărul (care poate fi foarte mare) de secvențe de înlocuiri posibile care produc această secvență, *și să returnați acest număr modulo **1,000,000,009***. Dacă secvența dată nu este o secvență de gondole, funcția trebuie să returneze 0. Dacă secvența dată este o secvență de gondole dar nicio gondolă nu s-a stricat, funcția trebuie să retuneze 1.

Subproblemele 7, 8, 9, 10

subproblemă	puncte	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, și cel puțin $n - 3$ din gondolele inițiale $1, \ldots, n$ nu s-au stricat.
9	15	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 250,000$
10	10	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 1,000,000,000$

Exemple

subproblemă	inputSeq	valoare returnată	secvență de înlocuiri
7	(1, 2, 7, 6)	2	(3, 4, 5) sau (4, 5, 3)
8	(2, 3, 4, 12, 6, 7, 1)	1	(5, 8, 9, 10, 11)
9	(4, 7, 4, 7)	0	inputSeq nu este o secvență de gondole
10	(3, 4)	2	(1, 2) or (2, 1)

Detalii de implementare

Trebuie să încărcați exact un fișier, numit gondola.c, gondola.cpp sau gondola.pas. Acest fișier trebuie să implementeze toate cele trei subprograme descrise mai sus (chiar dacă plănuiți să rezolvați doar unele din subprobleme), folosing următoarele antete. De asemenea trebuie să includeți fișierul header gondola.h pentru implementări C/C++.

pentru programe C/C++

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

pentru programe Pascal

```
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;
```

Grader-ul de pe computerul vostru

Graderul de pe computerul vostru citește datele de intrare în următorul format:

- linia 1: `T`, subproblema pe care programul vostru trebuie să o rezolve ($\mathbf{1 \leq T \leq 10}$).
- linia 2: `n`, lungimea secvenței date.
- linia 3: Dacă `T` este 4, 5, sau 6, atunci această linie conține `gondolaSeq[0]`, ..., `gondolaSeq[n-1]`. În caz contrar această linie conține `inputSeq[0]`, ..., `inputSeq[n-1]`.