Gondola

Gondola Mao-Kong adalah sebuah hal menarik yang terkenal di Taipei. Sistem gondola terdiri dari sebuah rel yang melingkar, satu stasiun, dan n gondola yang dinomori dari 1 sampai dengan n berkeliling rel pada satu arah yang tetap. Mula-mula, setelah gondola i melalui stasiun, gondola berikutnya melalui stasiun adalah gondola $i+1$ jika $i < n$, atau gondola 1 jika $i = n$.

Gondola bisa saja rusak. Untungnya, kita memiliki gondola cadangan yang banyaknya tidak terbatas, yang dinomori $n+1$, $n+2$, dan seterusnya. Saat sebuah gondola rusak, kita menggantinya (pada posisi yang sama pada jalur) dengan gondola cadangan pertama yang tersedia, yaitu yang nomornya paling kecil. Misalnya, jika ada lima gondola dan gondola 1 rusak, maka kita akan menggantinya dengan gondola 6.

Anda suka berdiri di stasiun dan mengamati gondola yang lewat. Sebuah sekuens gondola (*gondola sequence*) adalah sebuah deretan yang terdiri dari n buah gondola yang lewat stasiun. Mungkin saja satu atau lebih gondola rusak (dan telah digantikan) sebelum Anda datang, sehingga tidak ada gondola yang rusak ketika Anda sedang mengamati.

Perhatikan bahwa konfigurasi gondola yang sama pada rel dapat memberikan beberapa *gondola sequence*, tergantung pada gondola mana yang lewat pertama kali saat Anda tiba di stasiun. Misalnya, jika tak ada gondola yang rusak maka (2, 3, 4, 5, 1) dan (4, 5, 1, 2, 3) adalah *gondola sequence* yang mungkin, tetapi (4, 3, 2, 5, 1) adalah bukan *gondola sequence* (sebab gondola muncul dengan urutan yang salah).

Jika gondola 1 rusak, maka kita mungkin mengamati *gondola sequence* (4, 5, 6, 2, 3).
Jika gondola 4 rusak, maka kita menggantinya dengan gondola 7 dan kita sekarang mungkin mengamati *gondola sequence* (6, 2, 3, 7, 5).
Jika gondola 7 rusak setelah itu, kita mengganti dengan gondola 8 dan kita sekarang mungkin mengamati *gondola sequence* (3, 8, 5, 6, 2).

gondola rusak	**gondola baru**	***gondola sequence* yang mungkin**
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

Sebuah *replacement sequence* adalah sebuah sekuens yang terdiri dari nomor-nomor gondola yang rusak, sesuai dengan urutan rusaknya. Pada contoh sebelumnya, *replacement sequence* adalah (1, 4, 7). Sebuah *replacement sequence* r *menghasilkan gondola sequence* g jika, setelah gondola rusak menurut *replacement sequence* r, sekuens gondola g mungkin teramati.

Gondola Sequence Checking

Pada tiga subtask pertama, Anda harus memeriksa apakah sebuah *input sequence* adalah *gondola sequence*.

Lihat tabel di bawah sebagai berikut untuk contoh sekuens yang merupakan sebuah *gondola sequence* dan yang bukan *gondola sequence*. Anda harus mengimplementasi sebuah fungsi `valid`.

- `valid(n, inputSeq)`
  - `n`: the length of the input sequence.
  - `inputSeq`: array of length n; `inputSeq[i]` is element i of the input sequence, for $0 \leq i \leq n-1$.
  - The function should return 1 if the input sequence is a gondola sequence, or 0 otherwise.

Subtasks 1, 2, 3

subtask	points	n	`inputSeq`
1	5	$n \leq 100$	has each number from 1 to n exactly once
2	5	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$

Examples

subtask	`inputSeq`	return value	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 cannot appear just before 5
1	(4, 3, 2, 1)	0	4 cannot appear just before 3
2	(1, 2, 3, 4, 5, 6, 5)	0	two gondolas numbered 5
3	(2, 3, 4, 9, 6, 7, 1)	1	replacement sequence (5, 8)
3	(10, 4, 3, 11, 12)	0	4 cannot appear just before 3

Replacement Sequence

Untuk tiga subtask selajutnya Anda harus membangun salah satu *replacement sequence* yang mungkin menghasilkan *gondola sequence* yang diberikan. Sebarang *Replacement sequence* yang menghasilkan *gondola sequence* yang diberikan akan diterima. Anda harus mengimplementasikan fungsi `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` is the length of the gondola sequence.
  - `gondolaSeq`: array of length n; `gondolaSeq` is guaranteed to be a gondola sequence, and `gondolaSeq[i]` is element i of the sequence, for $0 \leq i \leq n-1$.

- The function should return l, the length of the replacement sequence.
- `replacementSeq`: array that is sufficiently large to store the replacement sequence; you should return your sequence by placing element i of your replacement sequence into `replacementSeq[i]`, for $0 \leq i \leq l-1$.

Subtasks 4, 5, 6

subtask	points	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]` $\leq 250,000$

Examples

subtask	`gondolaSeq`	return value	`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)

Count Replacement Sequences

Pada 4 subtask berikutnya, Anda harus menghitung banyaknya kemungkinan *replacement sequence* yang menghasilkan sebuah sekuens yang diberikan (yang mungkin adalah *gondola sequence* atau bukan *gondola sequence*), modulo **1,000,000,009**.
Anda harus mengimplementasi sebuah fungsi `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: the length of the input sequence.
  - `inputSeq`: array of length n; `inputSeq[i]` is element i of the input sequence, for $0 \leq i \leq n-1$.
  - If the input sequence is a gondola sequence, then count the number of replacement sequences that produce this gondola sequence (which could be extremely large), *and return this number modulo **1,000,000,009***. If the input sequence is not a gondola sequence, the function should return 0. If the input sequence is a gondola sequence but no gondolas broke down, the function should return 1.

Subtasks 7, 8, 9, 10

subtask	points	n	`inputSeq`
7	5	$4 \leq n \leq 50$	$1 \leq$ `inputSeq[i]` $\leq n+3$

subtask	points	n	inputSeq
8	15	$4 \leq n \leq 50$	$1 \leq$ inputSeq[i] ≤ 100, and at least $n - 3$ of the initial gondolas $1, \ldots, n$ did not break down.
9	15	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 250,000$
10	10	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 1,000,000,000$

Examples

subtask	inputSeq	return value	replacement sequence
7	(1, 2, 7, 6)	2	(3, 4, 5) or (4, 5, 3)
8	(2, 3, 4, 12, 6, 7, 1)	1	(5, 8, 9, 10, 11)
9	(4, 7, 4, 7)	0	inputSeq is not a gondola sequence
10	(3, 4)	2	(1, 2) or (2, 1)

Implementation details

You have to submit exactly one file, called gondola.c, gondola.cpp or gondola.pas. This file should implement the subprograms described above, using the following signatures. You also need to include a header file gondola.h for C/C++ implementation.

C/C++ programs

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

Pascal programs

```
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

The sample grader reads the input in the following format:

- line 1: T, the subtask number your program intends to solve ($1 \leq T \leq 10$).
- line 2: n, the length of the input sequence.
- line 3: If T is 4, 5, or 6, this line contains inputSeq[0], ..., inputSeq[n-1]. Otherwise this line contains gondolaSeq[0], ..., gondolaSeq[n-1].