Gondola

La Gondola de Mao-Kong es una famosa atracción en Taipei. El sistema de gondola consiste en un riel circular, una estación, y n gondolas numeradas en forma consecutiva desde la 1 a la n circulando por el riel en una unica dirección fija. Despues que la gondola i pasa por la estación, la proxima gondola en pasar la estación sera la gondola $i+1$ si $i < n$, o la gondola 1 si $i = n$.

Las gondolas pueden dejar de funcionar. Por suerte tenemos un numero infinito de gondolas de repuesto, las cuales estan numeradas como $n+1$, $n+2$, y asi sucesivamente. Cuando una gondola se descompone se reemplaza (en la misma posición en el riel) con la primera góndola de repuesto disponible, es decir, la que tiene el número más bajo. Por ejemplo, si hay cinco góndolas y la góndola 1 se rompe, entonces la reemplazaremos con la góndola 6.

A ti te gusta estar en la estación y ver las góndolas pasar. Una *secuencia de góndola* es una serie de n números de góndolas que pasan por la estación. Es posible que una o más góndolas se rompan (y fueron sustituidos) antes de que llegaras, pero ninguna de las góndolas se descomponen, mientras tu está viendo.

Tenga en cuenta que la misma configuración de góndolas en el riel puede dar múltiples secuencias de góndolas, dependiendo de qué góndola pasa primero al tu llegar a la estación. Por ejemplo, si ninguna de las góndolas se ha roto entonces (2, 3, 4, 5, 1) y (4, 5, 1, 2, 3) son posibles secuencias de góndola, pero (4, 3, 2, 5 , 1) no es (porque las góndolas aparecen en el orden equivocado).

Si la góndola 1 se rompe, entonces ahora podríamos observar la secuencia de góndolas (4, 5, 6, 2, 3). Si la góndola 4 se rompe luego, la reemplazamos con la góndola 7 y nosotros podríamos observar la secuencia de góndola (6, 2, 3, 7, 5). Si la góndola 7 se descompone después de esto, se la reemplaza con la góndola 8 y podemos ahora observar la secuencia de góndola (3, 8, 5, 6, 2).

gondola rota	gondola nueva	posible secuencia de gondola
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

Una *secuencia de reemplazo* es una secuencia que consta de los números de las góndolas que se han roto, en el orden en que se descomponen. En el ejemplo anterior la secuencia de reemplazo es (1, 4, 7). Una secuencia de reemplazo r *produce* una secuencia de góndola g si, después que las góndolas se rompen de acuerdo a la secuencia r, la secuencia de góndola g se puede observar.

Comprobacion de la secuencia de gondola

En las primeras tres subtareas debes comprobar si una secuencia de entrada es una secuencia de góndola. Consulte la tabla siguiente para ver ejemplos de secuencias que son y no son secuencias de góndola. Debes implementar la función `valid`.

- `valid(n, inputSeq)`
  - `n`: es el largo de la secuencia de entrada.
  - `inputSeq`: array de tamano n; `inputSeq[i]` es el elemento i de la secuencia de entrada, para $0 \leq i \leq n-1$.
  - La función debe devolver 1 si la secuencia de entrada es una secuencia de góndola, o 0 en caso contrario.

Subtareas 1, 2, 3

subtarea	puntos	n	`inputSeq`
1	5	$n \leq 100$	tiene cada número de 1 a n exactamente una vez
2	5	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$

Ejemplos

subtarea	`inputSeq`	valor retornado	nota
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 no puede aparecer justo antes del 5
1	(4, 3, 2, 1)	0	4 no puede aparecer justo antes del 3
2	(1, 2, 3, 4, 5, 6, 5)	0	dos gondolas numeradas 5
3	(2, 3, 4, 9, 6, 7, 1)	1	secuencia de reemplazo (5, 8)
3	(10, 4, 3, 11, 12)	0	4 no puede aparecer justo antes del 3

Secuencia de Reemplazo

En las próximas tres subtareas debes construir una posible secuencia de reemplazo que genera una secuencia de góndola dada. Se aceptará cualquier secuencia de sustitución. Debes implementar una función `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` es el largo de la secuencia de gondola.
  - `gondolaSeq`: array de tamanio n; `gondolaSeq` garantiza que sea una secuencia de gondola, y `gondolaSeq[i]` es el elemento i de la secuencia, para $0 \leq i \leq n-1$.
  - La funcion debe retornar l, el largo de la secuencia de reemplazo.

- `replacementSeq`: vector que es lo suficientemente grande para almacenar la secuencia de reemplazo; usted debe devolver su secuencia mediante la colocación de elementos i de su secuencia de reemplazo en `replacementSeq[i]` , $0 \leq i \leq l-1$.

Subtareas 4, 5, 6

subtarea	puntos	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]` $\leq 250,000$

Ejemplos

subtarea	`gondolaSeq`	valor de retorno	`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)

Contando Secuencias de Reemplazo

En las próximas cuatro subtareas debes contar el número de posibles secuencias de reemplazo que producen una secuencia dada (que puede o no ser una secuencia de góndola), modulo **1,000,000,009**. Debes implementar la funcion `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: es el largo de la secuencia de entrada.
  - `inputSeq`: array de tamanio n; `inputSeq[i]` es el elemento i de la secuencia de entrada, para $0 \leq i \leq n-1$.
  - Si la secuencia de entrada es una secuencia de góndola,entonces contar el número de secuencias de reemplazo que producen esta secuencia de góndola (que podría ser extremadamente grande), *y retornar un numero modulo **1,000,000,009***. Si la secuencia de entrada no es una secuencia de góndola, la función debe devolver 0. Si la secuencia de entrada es una secuencia de góndola, pero no hay góndolas rotas, la función debe devolver 1.

Subtareas 7, 8, 9, 10

subtarea	puntos	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, y al menos $n-3$ de las gondolas iniciales $1, \ldots, n$ no se rompen.

subtarea	puntos	n	inputSeq
9	15	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 250,000$
10	10	$n \leq 100,000$	$1 \leq$ inputSeq[i] $\leq 1,000,000,000$

Ejemplos

subtarea	inputSeq	valor de retorno	secuencia de reemplazo
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 no es una secuencia de gondola
10	(3, 4)	2	(1, 2) or (2, 1)

Detalles de la implementacion

Debes enviar exactamente un archivo, llamado `gondola.c`, `gondola.cpp` o `gondola.pas`. Este archivo debiera implementar el subprograma descripto mas arriba, utilizando los siguientes encabezamiento. Necesitas incluir tambien un archivo en la cabecera gondola.h para la implementacion en C/C++.

Programacion en C/C++

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

Programacion en 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;
```

Sample grader

El sample grader (programa evaluador para la prueba local) lee su entrada con el formato siguiente:

- linea 1: `T`, el mumero de subtarea que tu programa intenta resolver ($1 \leq T \leq 10$).
- linea 2: `n`, el largo de la secuencia de entrada.
- linea 3: Si `T` es 4, 5, o 6, esta linea contiene `gondolaSeq[0]`, ..., `gondolaSeq[n-1]`. De lo contrario esta line contiene `inputSeq[0]`, ..., `inputSeq[n-1]`.