Lanovka (Gondola)

Kabinová lanovka Mao-Kong je populární turistickou atrakcí v Taipei. Lanovkový systém se skládá z okružní trasy, jedné stanice a n kabinek očíslovaných postupně od 1 do n. Tyto kabinky jezdí po okružní trase v pevně stanoveném pořadí a směru jízdy. Když byla lanovka nová, tak vždy když stanicí projela kabinka i, hned po ní projela stanicí kabinka $i+1$ (pro $i < n$). Bezprostředně po kabince n projela stanicí kabinka 1.

Kabinky se občas porouchají. Naštěstí máme nekonečnou zásobu náhradních kabinek, které jsou očíslovány $n+1, n+2, \ldots$ Kdykoliv se nějaká kabinka rozbije, nahradíme ji (na stejné pozici) první dosud nepoužitou náhradní kabinkou, tedy tou s nejnižším číslem. Například pokud má lanovka pět kabinek a kabinka 1 se rozbije, nahradíme ji kabinkou 6.

Posloupnost kabinek je uspořádaná n-tice čísel kabinek, které projíždějí v tomto pořadí stanicí. Je možné, že se jedna nebo více kabinek porouchaly a byly nahrazeny náhradními kabinkami.

Všimněte si, že stejné uspořádání kabinek na trase může být popsáno více různými posloupnostmi kabinek. Například pokud se žádná kabinka nerozbila, jsou pětice (2, 3, 4, 5, 1) i (4, 5, 1, 2, 3) možnými posloupnostmi kabinek, zatímco (4, 3, 2, 5, 1) není, protože kabinky v ní mají špatné pořadí.

Pokud se kabinka 1 porouchala, je jednou z možných posloupností kabinek (4, 5, 6, 2, 3). Když se poté rozbila kabinka 4, byla nahrazena kabinkou 7 a možná posloupnost kabinek je (6, 2, 3, 7, 5). Jestliže se potom porouchala i kabinka 7, byla nahrazena kabinkou 8 a jednou z možných posloupností kabinek je (3, 8, 5, 6, 2).

porouchaná kabinka	nahrazující kabinka	možná posloupnost kabinek
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

Posloupnost poruch je posloupnost tvořená čísly všech rozbitých kabinek v takovém pořadí, v jakém se porouchaly. V předchozím příkladu je popsána posloupnost poruch (1, 4, 7). Posloupnost poruch r *vytváří* posloupnost kabinek g, pokud po rozbití kabinek v pořadí určeném posloupností r bude g jednou z možných posloupností kabinek.

Kontrola správnosti posloupnosti kabinek

V prvních třech podúlohách ověřte, zda je zadaná posloupnost čísel korektní posloupností kabinek. V tabulce níže vidíte ukázky posloupností, které jsou, resp. nejsou správnými posloupnostmi kabinek. Implementujte funkci `valid`.

- `valid(n, inputSeq)`

- `n`: délka zkoumané posloupnosti.
- `inputSeq`: pole délky n; `inputSeq[i]` je prvek zkoumané posloupnosti s pořadovým číslem i, pro $0 \leq i \leq n-1$.
- Funkce vrátí hodnotu 1, pokud je zkoumaná posloupnost správnou posloupností kabinek. V opačném případě vrátí hodnotu 0.

Podúlohy 1, 2, 3

podúloha	počet bodů	n	`inputSeq`
1	5	$n \leq 100$	obsahuje každé z čísel od 1 do n právě jednou
2	5	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100,000$	$1 \leq$ `inputSeq[i]` $\leq 250,000$

Příklady

podúloha	`inputSeq`	návratová hodnota	komentář
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	číslo 1 se v posloupnosti nemůže objevit těsně před číslem 5
1	(4, 3, 2, 1)	0	číslo 4 se v posloupnosti nemůže objevit těsně před číslem 3
2	(1, 2, 3, 4, 5, 6, 5)	0	dvě kabinky s číslem 5
3	(2, 3, 4, 9, 6, 7, 1)	1	posloupnost poruch (5, 8)
3	(10, 4, 3, 11, 12)	0	číslo 4 se v posloupnosti nemůže objevit těsně před číslem 3

Posloupnost poruch

V následujících třech podúlohách zkonstruujte jednu posloupnost poruch, která vytváří zadanou posloupnost kabinek. Libovolná taková posloupnost poruch bude považována za správnou.
Implementujte funkci `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` je délka posloupnosti kabinek.
  - `gondolaSeq`: pole délky n; `gondolaSeq` je vždy posloupnost kabinek, `gondolaSeq[i]` je prvek posloupnosti kabinek s pořadovým číslem i, pro $0 \leq i \leq n-1$.

- Funkce vrátí hodnotu l, což je délka výsledné posloupnosti poruch.
- `replacementSeq`: pole dostatečně velké na uložení posloupnosti poruch; funkce uloží na pozici i tohoto pole i-tý prvek nalezené posloupnosti poruch, pro $0 \leq i \leq l-1$.

Podúlohy 4, 5, 6

podúloha	počet bodů	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]` $\leq 250,000$

Příklady

podúloha	`gondolaSeq`	návratová hodnota	`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)

Počet posloupností poruch

V následujících čtyřech podúlohách určete počet možných posloupností poruch, které vytvářejí zadanou posloupnost, jež může (ale nemusí) být posloupností kabinek. Vraťte zbytek tohoto počtu po dělení číslem **1,000,000,009**. Implementujte funkci `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: délka vstupní posloupnosti.
  - `inputSeq`: pole délky n; `inputSeq[i]` je prvek na pozici i ve vstupní posloupnosti, pro $0 \leq i \leq n-1$.
  - Pokud je vstupní posloupnost správnou posloupností kabinek, spočítejte počet možných posloupností poruch, které vytvářejí tuto posloupnost kabinek. Tento počet může být velký, proto *vraťte výsledek modulo* ***1,000,000,009***. Pokud vstupní posloupnost není posloupností kabinek, funkce vrátí hodnotu 0. Jestliže vstupní posloupnost je posloupností kabinek, ale žádná kabinka se nerozbila, funkce vrátí hodnotu 1.

Podúlohy 7, 8, 9, 10

podúloha	počet bodů	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, a alespoň $n-3$ původních kabinek $1, \ldots, n$ se nerozbilo.

podúloha	počet bodů	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]` $\leq 1,000,000,000$

Příklady

podúloha	`inputSeq`	návratová hodnota	posloupnost poruch
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` není posloupností kabinek
10	(3, 4)	2	(1, 2) or (2, 1)

Upřesnění implementace

Odevzdejte právě jeden soubor pojmenovaný `gondola.c`, `gondola.cpp` nebo `gondola.pas`. Tento soubor implementuje všechny tři funkce popsané výše (i pokud se pokoušíte řešit pouze některé podúlohy) s následujícími parametry. Nezapomeňte v případě jazyka C/C++ vložit hlavičkový soubor `gondola.h`.

Programy v C/C++

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

Programy v Pascalu

```
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;
```

Ukázkový vyhodnocovač

Ukázkový vyhodnocovač čte vstup v následujícím formátu:

- řádek 1: `T`, číslo podúlohy, kterou se váš program pokouší řešit ($1 \leq T \leq 10$).
- řádek 2: `n`, délka vstupní posloupnosti.
- řádek 3: Pokud `T` je 4, 5, nebo 6, tento řádek obsahuje `gondolaSeq[0]`, ..., `gondolaSeq[n-1]`. V opačném případě tento řádek obsahuje `inputSeq[0]`, ..., `inputSeq[n-1]`.