Keltuvų trasa (Gondolos)

Mao-Kong keltuvų trasa yra žinoma pramoga Taipėjuje. Keltuvų trasa – apskritimo formos. Joje yra viena stotis ir trasoje yra n vagonėlių, iš eilės sunumeruotų nuo 1 iki n, judančių viena kryptimi. Po to, kai vagovėlis i pravažiuoja stotį, kito stotį privažiuosiančio vagonėlio numeris bus $i + 1$, jei $i < n$, arba 1, jei $i = n$.

Vagonėliai gali sugesti. Laimei, turime neribotą kiekį atsarginių vagonėlių, sunumeruotų $n + 1, n + 2$, t.t. Kai vagonėlis sugenda, jį pakeičiame (toje pačioje pozicijoje, kaip ir sugedęs) nauju, t.y., nauju vagonėliu, kurio numeris mažiausias galimas. Pavyzdžiui, jei yra penki vagonėliai ir vagonėlis nr. 1 sugenda, jį pakeičiame vagonėliu nr. 6.

Tau patinka stovėti stotyje ir žiūrėti į keltuvu judančius vagonėlius. *Vagonėlių seka* yra n skaičių seka, nurodanti, kurie vagonėliai pravažiuoja stotį. Gali būti, kad vienas arba daugiau vagonėlių sugedo (ir buvo pakeistas) prieš tau atvykstant. Tačiau žinoma, kad nei vienas vagonėlis nesugedo stebėjimo metu.

Ta pati vagonėlių konfigūracija trasoje gali nurodyti skirtingas vagonėlių sekas priklausomai nuo to, kuris vagonėlis pirmasis pravažiuoja iš karto po tavo atėjimo į stotį. Pavyzdžiui, jei nei vienas vagonėlis nesugedo, tuomet (2, 3, 4, 5, 1) ir (4, 5, 1, 2, 3) yra galimos vagonėlių sekos. Tačiau (4, 3, 2, 5, 1) nėra, nes vagonėliai surašyti netinkama tvarka.

Jei vagonėlis nr. 1 sugenda, galime stebėti vagonėlių seką (4, 5, 6, 2, 3). Jei po to sugenda vagonėlis nr. 4, pakeičiame jį vagonėliu nr. 7 ir galime stebėti seką (6, 2, 3, 7, 5). Jei po to sugenda vagonėlis nr. 7, pakeičiame jį vagonėliu nr. 8 ir galime stebėti seką (3, 8, 5, 6, 2).

sugedęs vagonėlis	naujas vagonėlis	galima vagonėlių seka
1	6	(4, 5, 6, 2, 3)
4	7	(6, 2, 3, 7, 5)
7	8	(3, 8, 5, 6, 2)

Pakaitinė seka yra seka, sudaryta iš sugedusių vagonėlių numerių jų sugedimo tvarka. Praeitame pavyzdyje pakeitimų seka yra (1, 4, 7). Pakaitinė seka r *sukuria* vagonėlių seką g jei, po to, kai vagonėliai sulūžta pagal pakeitimų seką r, vagonėlių seka g yra galima.

Vagonėlių sekų tikrinimas

Pirmosiose trijose dalinėse užduotyse reikia patikrinti, ar duotoji seka yra vagonėlių seka. Žemiau pavaizduota keletas pavyzdinių neteisingų sekų. Reikia parašyti funkciją `valid`.

- `valid(n, inputSeq)`

- `n`: sekos ilgis.
- `inputSeq`: n ilgio masyvas; `inputSeq[i]` yra i-asis įvesties sekos elementas; galioja $0 \leq i \leq n-1$.
- Funkcija turi grąžinti 1, jei įvesties seka yra gondolų seka; 0 kitu atveju.

Dalinės užduotys 1, 2, 3

dalinė užduotis	taškai	n	`inputSeq`
1	5	$n \leq 100$	kiekvienas skaičius nuo iki n pasikartoja lygiai 1 kartą
2	5	$n \leq 100\,000$	$1 \leq$ `inputSeq[i]` $\leq n$
3	10	$n \leq 100\,000$	$1 \leq$ `inputSeq[i]` $\leq 250\,000$

Pavyzdžiai

dalinė užduotis	`inputSeq`	grąžinama reikšmė	pastaba
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 negali būti prieš pat 5
1	(4, 3, 2, 1)	0	4 negali būti prieš pat 3
2	(1, 2, 3, 4, 5, 6, 5)	0	dvi gondolos turi numerį 5
3	(2, 3, 4, 9, 6, 7, 1)	1	pakaitinė seka (5, 8)
3	(10, 4, 3, 11, 12)	0	4 negali būti prieš pat 3

Pakaitinė seka

Kitose trijose dalinėse užduotyse reikia sukonstruoti galimą pakaitinę seką, kuri sukuria duotą vagonėlių seką. Bet kokia tokia seka bus priimta. Reikia parašyti funkciją `replacement`.

- `replacement(n, gondolaSeq, replacementSeq)`
  - `n` yra vagonėlių sekos ilgis.
  - `gondolaSeq`: n ilgio masyvas; `gondolaSeq` tikrai yra vagonėlių seka, ir `gondolaSeq[i]` yra i-asis sekos elementas; galioja $0 \leq i \leq n-1$.
  - Funkcija turi grąžinti l, pakaitinės sekos ilgį.
  - `replacementSeq`: pakankamai didelis masyvas pakaitinės sekos laikymui; grąžinkite savo seką su įkeltu pakaitinės sekos elementu i į `replacementSeq[i]`; galioja $0 \leq i \leq l-1$.

Dalinės užduotys 4, 5, 6

dalinė užduotis	taškai	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]` $\leq 250\,000$

Pavyzdžiai

dalinė užduotis	`gondolaSeq`	grąžinama reikšmė	`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)

Pakaitinių sekų skaičiavimas

Kitose keturiose dalinėse užduotyse reikia suskaičiuoti duotos sekos galimą pakaitinių sekų (kurios gali būti arba nebūti vagonėlių sekos) skaičiaus liekaną iš $1\,000\,000\,009$. Reikia parašyti funkciją `countReplacement`.

- `countReplacement(n, inputSeq)`
  - `n`: sekos ilgis.
  - `inputSeq`: n ilgio masyvas; `inputSeq[i]` yra i-asis įvesties sekos elementas; galioja $0 \leq i \leq n - 1$.
  - jei įvesties seka yra vagonėlių seka, suskaičiuokite pakaitinių sekų kiekį, kurios padaro šią vagonėlių seką (tai gali būti labai didelis skaičius), *ir grąžinkite šio skaičiaus liekaną iš* $1\,000\,000\,009$. Jei įvesties seka nėra vagonėlių seka, ši funkcija turi grąžinti 0. Jei įvesties seka yra vagonėlių seka, tačiau nei vienas vagonėlis nesugedo, ši funkcija turi grąžinti 1.

Dalinės užduotys 7, 8, 9, 10

dalinė užduotis	taškai	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, ir bent $n - 3$ iš pradinių vagonėlių $1, \ldots, n$ nesugedo.
9	15	$n \leq 100\,000$	$1 \leq$ `inputSeq[i]` $\leq 250\,000$
10	10	$n \leq 100\,000$	$1 \leq$ `inputSeq[i]` $\leq 1\,000\,000\,000$

Pavyzdžiai

dalinė užduotis	`inputSeq`	grąžinama reikšmė	pakaitinė seka
7	(1, 2, 7, 6)	2	(3, 4, 5) arba (4, 5, 3)
8	(2, 3, 4, 12, 6, 7, 1)	1	(5, 8, 9, 10, 11)
9	(4, 7, 4, 7)	0	`inputSeq` yra ne vagonėlių seka
10	(3, 4)	2	(1, 2) arba (2, 1)

Reikalavimai

Pateikite vieną failą, pavadintą `gondola.c`, `gondola.cpp` arba `gondola.pas`. Jame turi būti visos trys aprašytos procedūros (net jei planuojate išspręsti tik kai kurias dalines užduotis). Jei naudojate C/C++, įtraukite antraštinį failą `gondola.h`.

Programuojantiems C/C++

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

Programuojantiems Paskaliu

```
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;
```

Pavyzdinis vertintojas

Pavyzdinis vertintojas skaito duomenis tokiu formatu:

- 1-oji eilutė: `T`, dalinės užduoties numeris ($1 \leq T \leq 10$).
- 2-oji eilutė: `n`, sekos ilgis.
- 3-oji eilutė: jei `T` yra 4, 5 arba 6, šioje eilutėje yra `gondolaSeq[0]`, ..., `gondolaSeq[n-1]`. Kitu atveju – `inputSeq[0]`, ..., `inputSeq[n-1]`.