Počitnice

Jian-Jia načrtuje svoje naslednje počitnice, ki si jih bo privoščil kar v Tajvanu. Tekom dopusta Jian-Jia potuje od mesta do mesta in v vsakem obiskuje njegove znamenitosti.

V Tajvanu je n mest, vsa pa ležijo vzdovž iste avtoceste. Mesta so oštevilčena zaporedno od 0 do $n-1$. Mesto i (za $0 < i < n-1$) je sosednje mestoma $i-1$ in $i+1$. Mestu 0 je sosednje samo mesto 1 in mesto $n-1$ samo mestu $n-2$.

V vsakem mestu je določeno število znamenitosti. Jian-Jia ima d dni dopusta in rad bi si ogledal kar največje število le-teh. Izbral si je že tudi začetno mesto svojega potovanja. Vsak dan svojega potovanja se Jian-Jia bodisi premakne v novo mesto bodisi si ogleda vse znamenitosti v trenutnem mestu. Jian-Jia *v nekem mestu nikoli ne obišče znamenitosti dvakrat*, tudi če se v tem mestu ustavi večkrat. Pomagaj Jian-Jia načrtovati njegove počitnice, da bo obiskal največje možno število znamenitosti.

Primer

Denimo da ima Jian-Jia 7 dni dopusta, na razpolago ima pet mest (podanih v spodnji tabeli), potovanje pa začne v mestu 2. Prvi dan obišče vseh 20 znamenotosti v mestu 2. Drugi dan se premakne iz mesta 2 v mesto 3 in tretji dan obišče vseh 30 znmenitosti v mestu 3. Naslednje tri dni porabi za potovanje iz mesta 3 do mesta 0, kjer sedmi dan obišče 10 znamenitosti. Skupno število obiskanih znamenitosti je torej 20+30+10=60, kar je tudi največje možno število za 7 dni potovanja z začetkom v mestu 2.

mesto	št. znamenitosti
0	10
1	2
2	20
3	30
4	1

dan	aktivnost
1	obisk znamenitosti mesta 2
2	premik iz mesta 2 v mesto 3
3	obisk znamenitosti mesta 3
4	premik iz mesta 3 v mesto 2
5	premik iz mesta 2 v mesto 1
6	premik iz mesta 1 v mesto 0
7	obisk znamenitosti mesta 0

Naloga

Napišite funkcijo `findMaxAttraction`, ki izračuna največje število znamenitosti, ki jih Jian-Jia lahko obišče na svojih počitnicah.

- `findMaxAttraction(n, start, d, attraction)`
  - `n`: število mest.
  - `start`: indeks začetnega mesta.
  - `d`: število dni dopusta.
  - `attraction`: tabela dolžine n; `attraction[i]` vsebuje število znamenitosti mesta i, za $0 \leq i \leq n-1$.
  - Funkcija vrača največje možno število znamenitosti, ki jih Jian-Jia lahko obišče.

Podnaloge

Za vse podnaloge velja: $0 \leq d \leq 2n + \lfloor n/2 \rfloor$ in število znamenitosti v nekem mestu je vedno nenegativno.

Dodatne omejitve:

podnaloga	točke	n	največje število znamenitosti v enem mestu (t)	začetno mesto
1	7	$2 \leq n \leq 20$	$0 \leq t \leq 1,000,000,000$	brez omejitev
2	23	$2 \leq n \leq 100,000$	$0 \leq t \leq 100$	mesto 0
3	17	$2 \leq n \leq 3,000$	$0 \leq t \leq 1,000,000,000$	brez omejitev
4	53	$2 \leq n \leq 100,000$	$0 \leq t \leq 1,000,000,000$	brez omejitev

Podrobnosti implementacije

Oddati morate natanko eno datoteko poimenovano `holiday.c`, `holiday.cpp` ali `holiday.pas`. V tej datoteki implementirajte funkcijo, ki je opisana zgoraj, imeti pa mora natako tak podpis, kot je podan spodaj. V jeziku C/C++ morate obvezno vključiti tudi header datoteko `holiday.h`.

Pozor: rezultat je zelo velik, zato ta funkcija vrača 64-bitno celo število.

Programski jezik C/C++

```
long long int findMaxAttraction(int n, int start, int d,
int attraction[]);
```

Programski jezik Pascal

```
function findMaxAttraction(n, start, d : longint;
attraction : array of longint): int64;
```

Vzorčni ocenjevalnik

Vzorčni ocenjevalnik bere vhod v sledeči obliki:

- vrstica 1: `n`, `start`, `d`.
- vrstica 2: `attraction[0]`, ..., `attraction[n-1]`.

Vzorčni ocenjevalnik bo izpisal vrednost, ki jo vrne funkcija `findMaxAttraction`.