



Odmor (Holiday)

Marko planira da provede svoj sljedeći odmor na Tajvanu. Tokom odmora, Marko putuje iz grada u grad i posjećuje atrakcije (znamenitosti) u tim gradovima.

Na Tajvanu postoji n gradova, koji su smješteni duž jednog autoputa. Gradovi su numerisani od 0 do $n - 1$. Gradu i , gdje je $0 < i < n - 1$, susjedni su gradovi $i - 1$ i $i + 1$. Jedini susjedan gradu 0 je grad 1, a jedini susjedan gradu $n - 1$ je grad $n - 2$.

Svaki grad ima određeni broj atrakcija. Marko za odmor ima na raspolaganju d dana i želi da posjeti što je moguće više tih atrakcija. Marko je unaprijed odabrao grad u kom će početi odmor. Tokom jednog dana odmora Marko može ili da otputuje u susjedni grad ili da posjeti sve atrakcije u gradu u kojem se trenutno nalazi, ali ne može i jedno i drugo. Marko *neće posjetiti više puta atrakcije u jednom istom gradu* čak i u slučaju da u tom gradu boravi više puta. Potrebno je da pomognete Marku da isplanira svoj odmor tako da posjeti što je moguće više različitih atrakcija.

Primjer

Pretpostavimo da Marko ima 7 dana odmora, da postoji 5 gradova (navedeni su u donjoj tabeli) i da odmor počinje u gradu 2. Prvog dana on će posjetiti 20 atrakcija u gradu 2. Drugog dana Marko putuje iz grada 2 u grad 3, pa trećeg dana posjećuje 30 atrakcija u gradu 3. Sljedeća tri dana Marko koristi da otputuje iz grada 3 u grad 0, pa sedmog dana posjećuje 10 atrakcija u gradu 0. Ukupan broj atrakcija koje je Marko posjetio je $20 + 30 + 10 = 60$, što je i maksimalni broj atrakcija koje on može posjetiti za 7 dana kada počne odmor u gradu 2.

grad	broj atrakcija
0	10
1	2
2	20
3	30
4	1

dan	akcija
1	posjeti atrakcije u gradu 2
2	putuj iz grada 2 u grad 3
3	posjeti atrakcije u gradu 3
4	putuj iz grada 3 u grad 2
5	putuj iz grada 2 u grad 1
6	putuj iz grada 1 u grad 0
7	posjeti atrakcije u gradu 0

Zadatak

Potrebno je da implementirate funkciju `findMaxAttraction` koja izračunava maksimalni broj atrakcija koje Marko može posjetiti.

- `findMaxAttraction(n, start, d, attraction)`
 - `n`: broj gradova.
 - `start`: indeks početnog grada.
 - `d`: broj dana odmora.
 - `attraction`: niz dužine `n`; `attraction[i]` je broj atrakcija u gradu `i`, za $0 \leq i \leq n - 1$.
 - Funkcija treba da vrati maksimalni broj atrakcija koje Marko može posjetiti.

Podzadaci

U svim podzadacima važi $0 \leq d \leq 2n + \lfloor n/2 \rfloor$, i broj atrakcija u svakom gradu je nenegativan.

Dodatna ograničenja:

podzadatak	poeni	n	maksimalni broj atrakcija u gradu	početni grad
1	7	$2 \leq n \leq 20$	1,000,000,000	nema ograničenja
2	23	$2 \leq n \leq 100,000$	100	grad 0
3	17	$2 \leq n \leq 3,000$	1,000,000,000	nema ograničenja
4	53	$2 \leq n \leq 100,000$	1,000,000,000	nema ograničenja

Implementacioni detalji

Neophodno je da predate tačno jedan fajl, koji se naziva `holiday.c`, `holiday.cpp` ili `holiday.pas`. Ovaj fajl mora da implementira opisani potprogram (funkciju) koristeći sljedeće signature. Dodatno, morate uključiti header fajl `holiday.h` u slučaju C/C++ implementacije.

Uočite da rezultat može biti velik, zbog čega je povratna vrijednost funkcije `findMaxAttraction` 64-bitni integer.

C/C++ program

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

Pascal program

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

Ocjenjivač

Ocjenjivač čita ulaz u sljedećem formatu:

- linija 1: n , $start$, d .
- linija 2: $attraction[0], \dots, attraction[n-1]$.

Ocjenjivač će odštampati povratnu vrijednost funkcije `findMaxAttraction`.