Wakacje (Holiday)

Jian-Jia planuje spędzić swoje następne wakacje na Tajwanie. Podczas wakacji będzie podróżował od miasta do miasta, starając się zwiedzić jak najwięcej atrakcji w różnych miastach.

Na Tajwanie znajduje się n miast usytuowanych wzdłuż jednej autostrady. Miasta są ponumerowane kolejno od 0 do $n-1$. Dla miasta i, gdzie $0 < i < n-1$, sąsiednimi miastami są $i-1$ oraz $i+1$. Jedynym sąsiednim miastem dla miasta 0 jest miasto 1, a jedynym sąsiednim miastem dla miasta $n-1$ jest miasto $n-2$.

W każdym mieście znajduje się pewna liczba atrakcji. Jian-Jia ma do dyspozycji d dni wakacji i chce w tym czasie zobaczyć możliwie najwięcej atrakcji. Już zdecydował, od którego miasta rozpocznie zwiedzanie Tajwanu. Każdego dnia podczas swoich wakacji Jian-Jia może albo przemieścić się do sąsiedniego miasta, albo zobaczyć wszystkie atrakcje w mieście, w którym się aktualnie znajduje. Jednego dnia nie może jednocześnie zwiedzać miasta i przemieszczać się między miastami. Jian-Jia *nigdy nie ogląda atrakcji w tym samym mieście dwukrotnie*, nawet jeśli znajdzie się tam wiele razy. Pomóż Jian-Jia zaplanować wakacje tak, żeby zobaczył możliwie najwięcej różnych atrakcji.

Przykład

Załóżmy, że Jian-Jia ma 7 dni wakacji, do zwiedzenia jest 5 miast (wymienionych w tabeli poniżej), a zwiedzanie rozpoczyna się od miasta o numerze 2. Pierwszego dnia Jian-Jia odwiedza 20 atrakcji w mieście 2. Drugiego dnia Jian-Jia przemieszcza się z miasta 2 do miasta 3, a trzeciego dnia zwiedza wszystkie 30 atrakcji w tym mieście. Następne trzy dni Jian-Jia spędza, podróżując z miasta 3 do miasta 0. Siódmego dnia Jian-Jia zwiedza 10 atrakcji w mieście 0. Łączna liczba atrakcji odwiedzonych przez Jian-Jia wynosi 20+30+10=60, co jest największą możliwą liczbą atrakcji, jakie Jian-Jia może odwiedzić w ciągu 7 dni, rozpoczynając zwiedzanie od miasta 2.

miasto	liczba atrakcji
0	10
1	2
2	20
3	30
4	1

dzień	akcja
1	zwiedza atrakcje w mieście 2
2	przemieszcza się z miasta 2 do miasta 3

dzień	akcja
3	zwiedza atrakcje w mieście 3
4	przemieszcza się z miasta 3 do miasta 2
5	przemieszcza się z miasta 2 do miasta 1
6	przemieszcza się z miasta 1 do miasta 0
7	zwiedza atrakcje w mieście 0

Zadanie

Napisz funkcję `findMaxAttraction`, która obliczy maksymalną liczbę atrakcji możliwą do odwiedzenia przez Jian-Jia.

- `findMaxAttraction(n, start, d, attraction)`
  - `n`: liczba miast.
  - `start`: indeks miasta, z którego rozpoczyna się zwiedzanie.
  - `d`: liczba dni.
  - `attraction`: tablica rozmiaru n; `attraction[i]` jest liczbą atrakcji w mieście i, dla $0 \leq i \leq n-1$.
  - Wynikiem funkcji powinna być maksymalna liczba atrakcji możliwych do odwiedzenia przez Jian-Jia.

Podzadania

We wszystkich podzadaniach zachodzi $0 \leq d \leq 2n + \lfloor n/2 \rfloor$. W każdym mieście znajduje się nieujemna liczba atrakcji.

Dodatkowe ograniczenia:

podzadanie	liczba punktów	n	max liczba atrakcji w jednym mieście	miasto startowe
1	7	$2 \leq n \leq 20$	1,000,000,000	dowolne
2	23	$2 \leq n \leq 100,000$	100	miasto 0
3	17	$2 \leq n \leq 3,000$	1,000,000,000	dowolne
4	53	$2 \leq n \leq 100,000$	1,000,000,000	dowolne

Implementacja

Powinieneś zgłosić dokładnie jeden plik o nazwie `holiday.c`, `holiday.cpp` lub `holiday.pas`. W pliku powinna znaleźć się implementacja funkcji opisanej powyżej, o następującej sygnaturze. W przypadku programu w C/C++ powinieneś także załączyć (*include*) plik nagłówkowy `holiday.h`.

Zauważ, że wynik może być duży, a do przechowywania wyniku funkcji `findMaxAttraction` jest używany 64-bitowy typ całkowity.

Programy w C/C++

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

Programy w Pascalu

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

Przykładowy program sprawdzający

Przykładowy program sprawdzający wczytuje dane w następującym formacie:

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

Przykładowy program sprawdzający wypisze na wyjście wynik funkcji `findMaxAttraction`.