Ваканция

Джан планира следващата си ваканция в Тайван. По време на ваканцията той смята да пътува от град на град и да посещава различни атракциони.

Тайван има n града, разположени по една магистрала. Градовете са номерирани последователно от 0 до $n-1$. За всеки град i, $0 < i < n-1$, съседни са градовете $i-1$ и $i+1$. Съседен на града 0 е само градът 1, а съседен на града $n-1$ е само градът $n-2$.

Във всеки от градовете има определен брой атракциони. Джан вече е определил града в който ще започне ваканцията си и би искал да посети колкото може повече атракциони. През всеки от отделените за ваканцията d дни Джан може или да разгледа всички атракциони в града, в който се намира или да се придвижи до някой съседен град, но не и двете. Джан не може да посещава атракционите на един град повече от един път, дори и да мине през съответния град повече от един път. Помогнете на Джан да планира ваканцията си така, че да посети колкото може повече атракциони.

Пример

Да предположим, че Джан има 7 дни на разположение, 5 града с атракциони (както е дадено в таблицата по-долу) и започва ваканцията си в града 2. През първия ден той посещава 20-те атракциони в този град. През втория ден се премества от град 2 в град 3 и през третия ден посещава 30 атракциона в този град. Следващите три дни Джан използва за да се премести от град 3 в град 0, където посещава 10 атракциона през седмия ден на ваканцията си. Така броят на посетените от Джан атракциони става 20 + 30 + 10 = 60 и това е максималният възможен брой атракциони, които той може да посети за 7 дни, започвайки ваканцията в града 2.

град	брой атракциони
0	10
1	2
2	20
3	30
4	1

ден	действие
1	посети атракционите в град 2
2	премести се от град 2 в град 3
3	посети атракционите в град 3
4	премести се от град 3 в град 2
5	премести се от град 2 в град 1
6	премести се от град 1 в град 0

ден	действие
7	посети атракционите в град 0

Задача

Напишете функция `findMaxAttraction`, която да определя максималния брой атракциони, които Джан може да посети:

- `findMaxAttraction(n, start, d, attraction)`
  - `n`: брой на градовете.
  - `start`: начален град.
  - `d`: брой на дните.
  - `attraction`: масив с n елемента; в `attraction[i]` е зададен броят на атракционите в града i, $0 \leq i \leq n-1$.
  - Функцията трябва да връща максималния брой атракциони, които Джан може да посети.

Подзадачи

Във всички подзадачи $0 \leq d \leq 2n + \lfloor n/2 \rfloor$, а броят на атракционите е неотрицателен.

Допълнителни ограничения:

подзадача	точки	n	брой атракциони (t) в град	начален град
1	7	$2 \leq n \leq 20$	$0 \leq t \leq 1000000000$	без ограничение
2	23	$2 \leq n \leq 100000$	$0 \leq t \leq 100$	град 0
3	17	$2 \leq n \leq 3000$	$0 \leq t \leq 1000000000$	без ограничение
4	53	$2 \leq n \leq 100000$	$0 \leq t \leq 1000000000$	без ограничение

Детайли на имплементацията за C/C++

Трябва да изпратите само един файл с име `holiday.c` или , `holiday.cpp`. Той трябва да съдържа имплементация на функцията `findMaxAttraction` такава, каквото е описана по-горе, със следната спецификация:

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

като не забравите да включите файла `holiday.h`. Забележете, че резултатът от работата на функцията може да бъде много голямо число и затова типът на функцията е 64-битово цяло число.

Примерен грейдър

Примерният грейдър чете вход във формат:

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

Примерният грейдър извежда резултата от извикването на функцията `findMaxAttraction`.