Holiday

Jian-Jia esta planeando su siguiente vacación en Taiwan. Durante su vacación, Jian-Jia se mueve de ciudad a ciudad y visita las atracciones en cada ciudad.

Hay n ciudades en Taiwan, todas localizadas a lo largo de una autopista. Las ciudades son numeradas consecutivamentes de 0 a $n-1$. Para la ciudad i, donde $0 < i < n-1$, las ciudades adyacentes son $i-1$ y $i+1$. La única ciudad adyacente a la ciudad 0 es 1, y la única ciudad adyacente a la ciudad $n-1$ es la ciudad $n-2$.

Cada ciudad contiene un número de atracciones. Jian-Jia planea visitar cuantas atracciones sea posible. Jian-Jia ha seleccionado una ciudad donde comenzar su vacación. Cada día de su vacacion JianJia puede moverse a la ciudad adyacente, o en su defecto visitar todas la atracciones de la ciudad que la que está, pero no ambas cosas. Jian-Jia *nunca visitará las atracciones de la misma ciudad más de una vez* aun cuando se quede en la ciudad varias veces. Por favor ayuda a Jian-Jia a planear su vacación tal que visite cuantas diferentes atracciones sea posible.

Example

Supon que Jian-Jia tiene 6 días de vacación, hay 5 ciudades (listadas en la tabla de abajo), y el inicia desde la ciudad 2. En el primer día Jian-Jia visita las 2 atracciones de la ciudad 2. En el segundo día Jian-Jia se mueve de la ciudad 2 a la ciudad 3, y en el día 3 visita las 30 atracciones de la ciudad 3. Entonces Jian-Jia gasta los siguientes 2 días moviendose de la ciudad 3 a la ciudad 1, y visita las 10 atracciones en la ciudad 1 en el día 6. El número total de atracciones que Jian-Jia visita es 20 + 30 + 10 = 60, el cual es el máximo número de atracciones que Jian-Jia puede visitar en 6 días cuando el inicie su vacación desde la ciudad 2.

city	number of attractions
0	2
1	10
2	20
3	30
4	1

day	action
1	visita las atracciones en la ciudad 2
2	se mueve de la ciudad 2 a la ciudad 3
3	visita las atracciones en la ciudad 3
4	se mueve de la ciudad 3 a la ciudad 2
5	se mueve de la ciudad 2 a la ciudad 1
6	visita las atracciones en la ciuda 1

Task

Por favor implementa una función `findMaxAttraction` que computa el máximo numero de atracciones que Jian-Jia puede visitar.

- `findMaxAttraction(n, start, d, attraction)`
  - `n`: número de ciudades.
  - `start`: el indice de la ciudad a iniciar.
  - `d`: número de días.
  - `attraction`: array de longitud n; `attraction[i]` es el número de atracciones en la ciudad i, para $0 \leq i \leq n-1$.
  - La función debe retornar el máximo número de atraccions que Jian-Jia puede visitar.

Subtasks

In all subtasks $0 \leq d \leq 2n + \lfloor n/2 \rfloor$.

Additional constraints:

subtask	points	n	number of attractions in a city (t)	starting city
1	7	$2 \leq n \leq 20$	$0 \leq t \leq 1,000,000,000$	no constraints
2	23	$2 \leq n \leq 100,000$	$0 \leq t \leq 100$	city 0
3	17	$2 \leq n \leq 3,000$	$0 \leq t \leq 1,000,000,000$	no constraints
4	53	$2 \leq n \leq 100,000$	$0 \leq t \leq 1,000,000,000$	no constraints

Implementation details

You have to submit exactly one file, called `holiday.c`, `holiday.cpp` or `holiday.pas`. This file should implement the subprogram described above using the following signatures. You also need to include a header file `holiday.h` for C/C++ implementation.

Note that the result may be large, and the return type of `findMaxAttraction` is a 64-bit 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;
```

Sample grader

The sample grader reads the input in the following format:

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

The sample grader will print the return value of `findMaxAttraction`.