Atvaļinājums

Jian-Jia savu nākamo atvaļinājumu plāno pavadīt Taivānā un dažādās pilsētās apmeklēt atrakcijas.

Taivānā ir n pilsētas, kas izvietotas vienas šosejas malā. Pilsētas ir numurētas pēc kārtas ar skaitļiem no 0 līdz $n-1$. Katram i, kur $0 < i < n-1$, blakus pilsētas ir pilsētas $i-1$ un $i+1$. Pilsētai 0 blakus atrodas tikai pilsēta 1, un pilsētai $n-1$ blakus atrodas tikai pilsēta $n-2$.

Katrā pilsētā ir pieejams noteikts skaits atrakciju. Jian-Jia atvaļinājums ilgst d dienas. Viņš plāno apmeklēt pēc iespējas lielāku atrakciju skaitu un jau ir izvēlējies pilsētu, kurā sāksies viņa atvaļinājums. Katrā atvaļinājuma dienā Jian-Jia var vai nu doties uz blakus pilsētu, vai arī apmeklēt visas atrakcijas pilsētā, kurā atrodas (bet ne abas darbības vienlaicīgi!). Jian-Jia vienā un tajā pašā pilsētā atrakcijas divreiz nekad neapmeklēs, pat ja šajā pilsētā ierasties iznāks vairākkārt. Palīdziet Jian-Jia saplānot atvaļinājumu tā, lai viņš apmeklētu lielāko iespējamo atrakciju skaitu.

Piemērs

Pieņemsim, ka Jian-Jia atvaļinājums ilgst septiņas dienas, ir piecas pilsētas, kas pārskaitītas zemāk dotajā tabulā, un viņa atvaļinājums sākas pilsētā 2. Tad pirmajā dienā Jian-Jia var apmeklēt 20 atrakcijas pilsētā 2, otrajā dienā doties uz pilsētu 3, bet trešajā - apmeklēt 30 atrakcijas pilsētā 3. Ceturtajā dienā viņš var doties uz pilsētu 2, piektajā - uz pilsētu 1, bet sestajā - uz pilsētu 0. Septītajā dienā Jian-Jia var apmeklēt 10 atrakcijas pilsētā 0. Tātad, atvaļinājuma laikā Jian-Jia var apmeklēt 20 + 30 + 10 = 60 atrakcijas, kas arī ir lielākais atrakciju skaits, ko iespējams apmeklēt septiņu dienu laikā, atvaļinājumu uzsākot pilsētā 2.

Pilsēta	Atrakciju skaits
0	10
1	2
2	20
3	30
4	1

Diena	Darbība
1	Apmeklēt atrakcijas pilsētā 2
2	Doties no pilsētas 2 uz pilsētu 3
3	Apmeklēt atrakcijas pilsētā 3
4	Doties no pilsētas 3 uz pilsētu 2
5	Doties no pilsētas 2 uz pilsētu 1
6	Doties no pilsētas 1 uz pilsētu 0
7	Apmeklēt atrakcijas pilsētā 0

Uzdevums

Realizējiet fukciju `findMaxAttraction`, kas aprēķina lielāko atrakciju skaitu, ko Jian-Jia var apmeklēt.

- `findMaxAttraction(n, start, d, attraction)`
  - `n`: pilsētu skaits.
  - `start`: sākuma pilsētas numurs.
  - `d`: atvaļinājuma dienu skaits.
  - `attraction`: n elementu masīvs; `attraction[i]` vērtība ir atrakciju skaits pilsētā i, katram $0 \leq i \leq n-1$.
  - Funkcijas rezultātam jābūt lielākajam iespējamajam atrakciju skaitam, ko Jian-Jia var apmeklēt atvaļinājuma laikā.

Apakšuzdevumi

Visiem apakšuzdevumiem $0 \leq d \leq 2n + \lfloor n/2 \rfloor$ un atrakciju skaits katrā pilsētā ir nenegatīvs.

Papildus ierobežojumi:

Apakšuzdevums	Punkti	n	Maksimālais atrakciju skaits pilsētā	Sākuma pilsētas numurs
1	7	$2 \leq n \leq 20$	1,000,000,000	nav noteikts
2	23	$2 \leq n \leq 100,000$	100	Pilsēta 0
3	17	$2 \leq n \leq 3,000$	1,000,000,000	nav noteikts
4	53	$2 \leq n \leq 100,000$	1,000,000,000	nav noteikts

Realizācijas detaļas

Jums jāiesūta tieši viens fails `holiday.c`, `holiday.cpp` vai `holiday.pas`. Šim failam jārealizē iepriekš aprakstītā apakšprogramma, izmantojot norādīto signatūru. C/C++ realizācijai programmas tekstā jāiekļauj `holiday.h`.

Ievērojiet, ka rezultāts var būt liels skaitlis un `findMaxAttraction` rezultāta tips ir 64 bitu vesels skaitlis.

C/C++ programma

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

Pascal programma

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

Paraugtestētājs

Paraugtestētājs lasa ievaddatus šādā formātā:

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

Paraugtestētājs izvadīs funkcijas `findMaxAttraction` rezultātu.