Ideálne mesto

Leonardo, rovnako ako veľa iných talianských vedcov a umelcov v jeho období, mal veľký záujem o plánovanie výstavby miest a ich urbanizáciu. Rozhodol sa preto navrhnúť ideálne mesto: pohodlné, priestranné a racionálne pri využívaní prostriedkov, nie ako vtedajšie (stredoveké) úzke až klaustrofobické mestá.

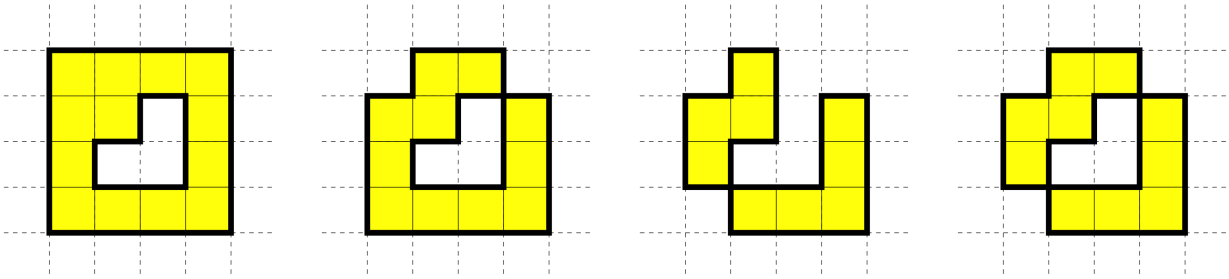
Ideálne mesto

Mesto sa skladá z N blokov. Každý blok stojí na jednom z políčok nekonečnej štvorcovej siete. Každé políčko siete je určené dvojicou súradníc (riadok, stĺpec). Susedné políčka pre políčko (i,j) sú štyri políčka (i - 1, j), (i + 1, j), (i, j - 1), a (i, j + 1). Blok môže byť umiestnený na políčko (i,j) vtedy a len vtedy, ak $1 \leq i, j \leq 2^{31} - 2$. V tejto úlohe budeme označovať súradnicami políčok aj bloky, ktoré sú umiestnené na týchto políčkach. Budeme hovoriť, že dva bloky sú susedné, ak sú umiestnené na susedných políčkach. V ideálnom meste sú všetky bloky pospájané tak, aby vo vnútri mesta neboli "diery". To znamená, že ideálne mesto musí spĺňať obe nasledujúce podmienky:

- Pre ľubovolné dve *prázdne* políčka existuje aspoň jedna postupnosť susedných *prázdnych* políčok, ktorá ich spája.
- Pre ľubovolné dve *neprázdne* políčka existuje aspoň jedna postupnosť susedných *neprázdnych* políčok, ktorá ich spája.

Príklad 1

Žiadna z dole zobrazených konfigurácií blokov nepredstavuje ideálne mesto. Prvé dve konfigurácie nespĺňajú prvú podmienku, tretia nespĺňa druhú podmienku a štvrtá nespĺňa ani jednu z našich dvoch podmienok.



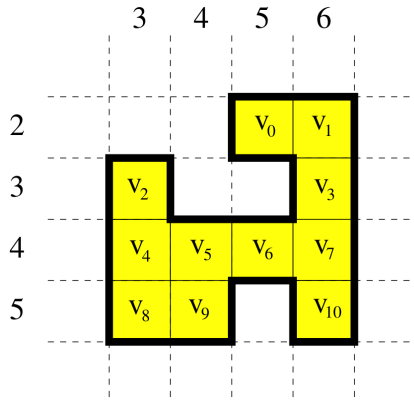
Vzdialenosť

Pri prechádzaní mesta budeme pod *krokom* rozumieť prechod medzi dvoma susednými blokmi.

Cez prázdne políčka sa prechádzať nedá. *Vzdialenosť* medzi dvoma blokmi je najmenší počet krokov potrebný na to, aby sme sa dostali z jedného bloku do druhého. Označme si súradnice našich N blokov v miežke postupne ako v_0, v_1, …, v_{N-1}. Vzdialenosť dvojice blokov so súradnicami v_i a v_j budeme označovať $d(v_i, v_j)$.

Príklad 2

Dole je uvedená konfigurácia ideálneho mesta pre N=11 blokov so súradnicami $v_0 = (2, 5)$, $v_1 = (2, 6)$, $v_2 = (3, 3)$, $v_3 = (3, 6)$, $v_4 = (4, 3)$, $v_5 = (4, 4)$, $v_6 = (4, 5)$, $v_7 = (4, 6)$, $v_8 = (5, 3)$, $v_9 = (5, 4)$, a $v_{10} = (5, 6)$. Platí napríklad $d(v_1, v_3) = 1$, $d(v_1, v_8) = 6$, $d(v_6, v_{10}) = 2$, a $d(v_9, v_{10}) = 4$.



Úloha

Vašou úlohou je napísať program, ktorý vypočíta sumu vzdialeností všetkých dvojíc blokov v_i a v_j pre $i < j$ pre dané ideálne mesto. Formálne váš program má vypočítať hodnotu nasledovnej sumy:

$$\sum d(v_i, v_j), \text{ kde } 0 \leq i < j \leq N - 1$$

Presnejšie, vašou úlohou je implementovať funkciu `DistanceSum`(N, X, Y), ktorá pre dané N a dve polia X a Y popisujúce mesto, vypočíta formulu uvedenú vyššie. Obe polia X a Y majú veľkosť N. Pre $0 \leq i \leq N - 1$ platí, že blok číslo i je umiestnený na súradniciach (X[i], Y[i]), pričom $1 \leq X[i], Y[i] \leq 31 - 2$. Kedže výsledná suma môže byť priveľká na to, aby sa zmestila do 32 bitového čísla, výsledok vypíšte modulo 1 000 000 000 (jedna miliarda).

V príklade 2 je $11 \times 10 / 2 = 55$ dvojíc blokov. Výsledná suma vzdialeností všetkých 55 dvojíc blokov je 174.

Podúloha 1 [11 bodov]

Môžete predpokladať, že $N \leq 200$.

Podúloha 2 [21 bodov]

Môžete predpokladať, že $N \leq 2\,000$.

Podúloha 3 [23 bodov]

Môžete predpokladať, že $N \leq 100\,000$.

Naviac budú platiť nasledujúce dve podmienky:

- Pre každú dvojicu neprázdnych políčok i a j platí: ak X[i] = X[j], tak každé políčko medzi políčkami i a j je tiež neprázdne.
- Pre každú dvojicu neprázdnych políčok i a j platí: ak Y[i] = Y[j], tak každé políčko medzi políčkami i a j je tiež neprázdne.

Podúloha 4 [45 bodov]

Môžete predpokladať, že $N \leq 100\,000$.

Implementačné detaily

Musíte odovzdať práve jeden súbor, ktorý sa volá `city.c`, `city.cpp` alebo `city.pas`. Tento súbor musí obsahovať implementáciu niektorej hore uvedenej podúlohy použitím nasledovných funkcií.

C/C++ programy

```
int DistanceSum(int N, int *X, int *Y);
```

Pascalové programy

```
function DistanceSum(N : LongInt; var X, Y : array of LongInt) : LongInt;
```

Tento podprogram sa musí správať tak, ako bolo popísané hore. Samozrejme, môžete implementovať ďalšie podprogramy pre vaše interné použitie. Vaš submit nesmie interagovať žiadnym spôsobom so štandardným vstupom/výstupom alebo s iným súborom.

Ukážkový testovač

Ukážkový testovač poskytnutý prostredím úlohy očakáva vstup v nasledujúcom formáte:

- riadok 1: N;
- riadok 2, …, N + 1: X[i], Y[i].

Časové a pamäťové limity

- Časový limit: 1 sekunda.
- Pamäťový limit: 256 MiB.