De ideale stad

Leonardo was, net zoals vele andere Italiaanse wetenschappers en artiesten in zijn tijd, zeer geïnteresseerd in stedenplanning en stedenontwerp. Hij wilde een ideale stad: comfortabel, ruim, slim in gebruik, heel anders dan de nauwe, claustrofobische steden uit de middeleeuwen.

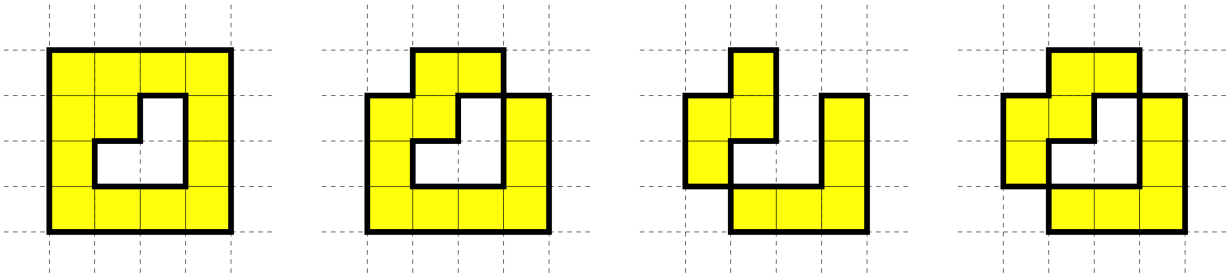
De ideale stad

De stad bestaat uit N wijken die gebouwd zijn op een denkbeeldig oneindig vierkant grid van cellen. Elke cel wordt aangeduid door een paar van coordinaten (rij, kolom). Als je een cel (i, j) bekijkt dan zijn de aangrenzende cellen: (i -1, j), (i+1, j), (i, j-1) en (i, j+1). Als je een wijk op het grid tekent beslaat deze precies één cel. Een wijk kan alleen in cel (i, j) geplaatst worden wanneer, en alleen wanneer, $1 \leq i, j \leq 2^{31} - 2$. We gebruiken de coordinaten van de cel om te refereren aan de wijk die erin ligt. Twee wijken zijn buren als ze in aangrenzende cellen zijn gelegen. In een ideale stad zijn alle wijken verbonden op een wijze dat er geen "'gaten" binnen de grenzen liggen; hieronder staat dit wat preciezer beschreven.

- Voor elk tweetal *lege* cellen is er minimaal een reeks van aangrenzende *lege* cellen die deze cellen met elkaar verbindt.
- Voor elk paar *niet-lege* cellen bestaat er minimaal een reeks van aangrenzende *niet-lege* cellen die deze twee cellen met elkaar verbindt.

Voorbeeld 1

Geen van de configuraties van wijken hieronder beschrijft een ideale stad: de twee aan de linkerzijde voldoen niet aan de eerste eis; de derde voldoet niet aan de tweede eis, de vierde voldoet aan geen van de eisen.



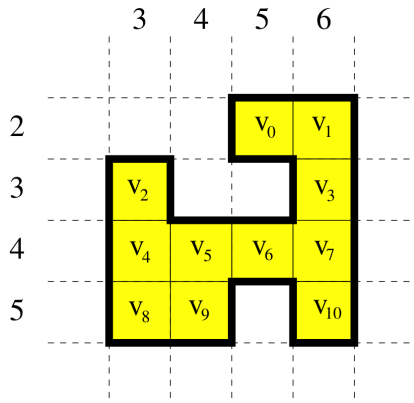
Afstand

Als je door de stad beweegt dan geeft een *hop* aan dat je van een wijk naar een buurwijk gaat. Je kunt niet door lege cellen bewegen. Als $v_0, v_1, \ldots, v_{N-1}$ de coordinaten zijn van N wijken die op

het grid zijn. Voor elk paar verschillende wijken op de coordinaten v_i en v_j, geldt dat de afstand $d(v_i, v_j)$ het minimale aantal hops dat nodig is om van de ene wijk naar de andere wijk te gaan.

Voorbeeld 2

De configuratie hieronder beschrijft een ideale stad die bestaat uit N = 11 wijken op de coordinaten $v_0 = (2, 5)$, $v_1 = (2, 6)$, $v_2 = (3, 3)$, $v_3 = (3, 6)$, $v_4 = (4, 3)$, $v_5 = (4, 4)$, $v_6 = (4, 5)$, $v_7 = (4, 6)$, $v_8 = (5, 3)$, $v_9 = (5, 4)$ en $v_{10} = (5, 6)$. Bijvoorbeeld: $d(v_1, v_3) = 1$, $d(v_1, v_8) = 6$, $d(v_6, v_{10}) = 2$ en $d(v_9, v_{10}) = 4$.



Opdracht

Jouw taak is om een programma te schrijven dat, voor een ideale stad, de som van alle paarsgewijze afstanden tussen wijken v_i en v_j waarbij $i < j$. Formeel moet je de volgende som bepalen:

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

Schrijf een routine `DistanceSum`(N, X, Y) die, gegeven N en twee arrays X en Y die de stad beschrijven, de bovenstaande formule uitrekent. De lengte van zowel X als Y is N. Wijk i ligt op de coordinaten (X[i], Y[i]), waarbij $0 \leq i \leq N - 1$, en $1 \leq X[i], Y[i] \leq 2^{31} - 2$. Omdat het resultaat groter zou kunnen zijn dan past in een 32 bits integer moet je je antwoord geven modulo 1 000 000 000 (een miljard).

In voorbeeld 2 zijn er $11 \times 10 / 2 = 55$ paren van wijken. De som van de paarsgewijze afstanden is 174.

Subtask 1 [11 punten]

Je kunt ervan uitgaan dat $N \leq 200$.

Subtask 2 [21 punten]

Je kunt ervan uitgaan dat $N \leq 2\,000$.

Subtask 3 [23 punten]

Je kunt ervan uitgaan dat $N \leq 100\,000$.

Bovendien gelden de volgende twee condities: gegeven elk paar niet-lege cellen i en j waarvoor geldt X[i] = X[j], elke cel tussen deze cellen is ook niet-leeg; gegeven elk paar niet-lege cellen i en j waarvoor geldt Y[i] = Y[j], elke cel tussen deze cellen is ook niet-leeg.

Subtask 4 [45 punten]

Je kunt ervan uitgaan dat $N \leq 100\,000$.

Details voor de implementatie

Je moet precies één bestand insturen met de naam `city.c`, `city.cpp` of `city.pas`. Dit bestand moet het bovengenoemde subprogramma implementeren conform de volgende specificaties.

C/C++ programma's

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

Pascal programma's

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

Dit subprogramma moet zich gedragen zoals hierboven beschreven. Het is vanzelfsprekend toegestaan om zelf ook nog andere subprogramma's te maken voor je interne gebruik. Je inzending mag op geen enkele wijze interactie hebben met standard input/output, en ook niet met andere bestanden.

Voorbeeld grader

De voorbeeld grader die je krijgt in de testomgeving verwacht invoer in het volgende formaat:

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

Tijds- en geheugenlimieten

- Tijdslimiet: 1 seconde.
- Geheugenlimiet: 256 MiB.