Ciudad ideal

Leonardo, como muchos otros científicos y artistas italianos de su época, estaba muy interesado en la planificación urbana. Para él, una ciudad ideal debería ser confortable, espaciosa y racional, muy lejos de las estrechas y claustrofóbicas ciudades de la Edad Media.

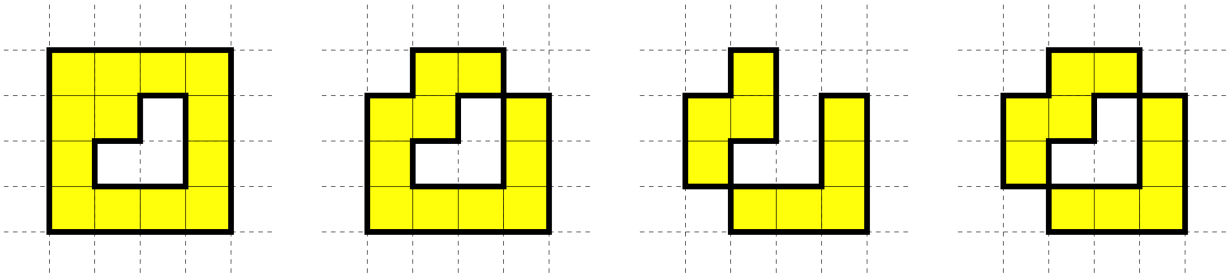
La ciudad ideal

La ciudad consiste en N bloques sobre un tablero infinito. Cada celda se identifica con un par de coordenadas (fila, columna). Dada una celda (i, j), sus celdas adyacentes son: (i - 1, j), (i + 1, j), (i, j - 1), y (i, j + 1). Cada bloque colocado en el tablero cubre exactamente una celda. Un bloque puede colocarse en la celda (i, j) si y sólo si $1 \leq i, j \leq 2^{31} - 2$. Usaremos las coordenadas de las celdas para referirnos también a los bloques sobre ellas. Dos bloques son adyacentes si están colocados sobre celdas adyacentes. En una ciudad ideal, todos los bloques están conectados de manera que no hay "agujeros" dentro de su borde. Esto es, las celdas deben satisfacer estas dos condiciones:

- Para cualquier par de celdas vacías, existe al menos una secuencia de celdas vacías adyacentes conectándolas.
- Para cualquier par de celdas ocupadas, existe al menos una secuencia de celdas ocupadas adyacentes conectándolas.

Ejemplo 1

Ninguna de las configuraciones siguientes es una ciudad ideal: las dos primeras de la izquierda no satisfacen la primera condición, la tercera no satisface la segunda, y la cuarta no satisface ninguna.

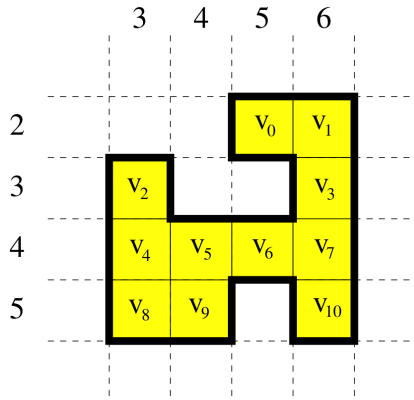


Distancia

Un *salto* es ir de una celda a otra adyacente. No se puede pasar por las celdas vacías. Sean v_0, v_1, …, v_{N-1} las coordenadas de los N bloques sobre el tablero. Para cada par de de bloques distintos en las coordenadas v_i y v_j, su distancia $d(v_i, v_j)$ es el número mínimo de saltos necesarios para ir de uno de los bloques al otro.

Ejemplo 2

Esta configuración representa una ciudad ideal con N = 11 bloques en las coordenadas 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), y v_{10} = (5, 6). Por ejemplo, d(v_1, v_3) = 1, d(v_1, v_8) = 6, d(v_6, v_{10}) = 2, y d(v_9, v_{10}) = 4.



Enunciado

Tu tarea es escribir un programa que, dada una ciudad ideal, calcule la suma de todos los pares de distancias entre bloques v_i and v_j para los cuales i < j. Formalmente, tu programa debe calcular el valor de la siguiente suma:

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

Específicamente, tienes que implementar una rutina `DistanceSum`(N, X, Y) que, dados N y dos arrays X e Y que describen la ciudad, calcula la fórmula dada más arriba. Tanto X como Y tienen tamaño N; el bloque i se encuentra en las coordenadas (X[i], Y[i]) para $0 \le i \le N - 1$, con $1 \le X[i], Y[i] \le 2^{31} - 2$. Como el resultado puede ser demasiado grande para un entero de 32 bits, debes devolverlo módulo 1 000 000 000 (mil millones).

En el Ejemplo 2, hay 11 × 10 / 2 = 55 pares de bloques. La suma de todas las distancias de todos los pares es 174.

Subtarea 1 [11 puntos]

Puedes asumir que $N \le 200$.

Subtarea 2 [21 puntos]

Puedes asumir que $N \le 2\,000$.

Subtarea 3 [23 puntos]

Puedes asumir que $N \le 100\,000$.

Adicionalmente, se cumplen las dos condiciones siguientes: para cualquier par de celdas ocupadas i y j tal que X[i] = X[j], cada celda entre ellas también está ocupada; y para cualquier par de celdas ocupadas i y j tales que Y[i]=Y[j], cada celda entre ellas también está ocupada.

Subtarea 4 [45 puntos]

Puedes asumir que $N \leq 100\,000$.

Detalles de implementación

Tienes que mandar exactamente un archivo con nombre `city.c`, `city.cpp` o `city.pas`. Este archivo debe implementar las rutinas descritas anteriormente con los siguientes cabeceras.

Programas en C/C++

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

Programas en Pascal

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

Esta rutina tiene que comportarse como se ha descrito anteriormente. Por supuesto, esta rutina puede llamar a otras rutinas que hayas programado. Tu envío no debe interactuar de ningún modo con la entrada/salida estándar, ni con ningún otro fichero.

Sample grader

El sample grader que tienes disponible espera un archivo de entrada con el formato siguiente:

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

Restricciones de tiempo y memoria

- Time limit: 1 second.
- Memory limit: 256 MiB.