Ciudad Ideal

Leonardo, al igual que otros científicos italianos y artistas de su época, estaba altamente interesado en la planificación de las ciudades y el diseño de urbanismos. El apuntaba a modelar una ciudad ideal: confortable, espaciosa y racional en el uso de sus recursos, muy lejos de las estrechas y claustrofóbicas ciudades de la Edad Media.

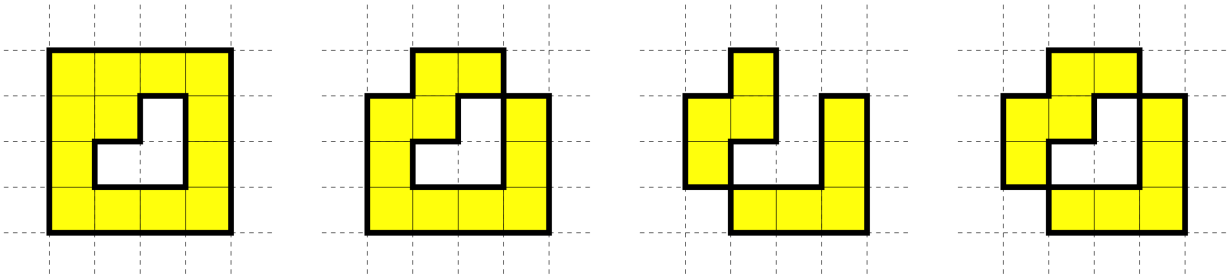
La ciudad ideal

La ciudad está hecha de N bloques ubicados en un mallado infinito de celdas. Cada celda está identificada por un par de coordenadas (fila, columna). Dada una celda (i,j), las celdas adyacentes son: (i - 1, j), (i + 1, j), (i, j - 1), y (i, j + 1). Cada bloque, cuando es ubicado en el mallado, cubre exactamente una de las celdas. Un bloque puede ser ubicado en la celda (i, j) si y sólo si $1 \leq i, j \leq 2^{31} - 2$. Utilizaremos las coordenadas de las celdas también para referirnos a los bloques que están sobre ellas. Dos bloques están adyacentes si están ubicados en celdas adyacentes. En una ciudad ideal, todos los bloques están conectados de tal forma que no existen "huecos" dentro de su borde, esto es, las celdas satisfacen las condiciones a continuación:

- Para cualesquiera dos celdas *vacías*, existe al menos una secuencia de celdas *vacías* adyacentes que las conectan.
- Para cualesquiera dos celdas *no-vacías*, existe al menos una secuencia de celdas *no-vacías* adyacentes que las conectan.

Ejemplo 1

Ninguna de las configurraciones de los bloques a continuación representa una ciudad ideal: las primeras dos en el lado izquierdo, no satisfacen la primera condición, la tercera no satisface la segunda condición y la cuarta no satisface ninguna de las dos condiciones.



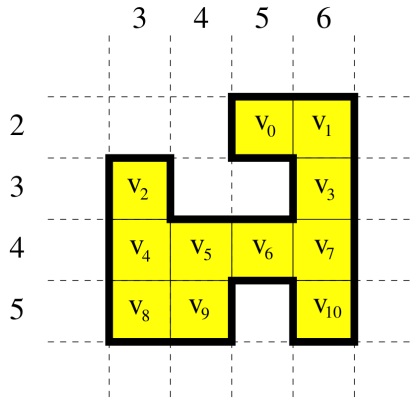
Distancia

Cuando se recorre una ciudad, un *salto* indica que se va de un bloque a uno adyacente. Las celdas

vacías no pueden ser recorridas. Sean v_0, v_1, …, v_{N-1} las coordenadas de los N bloques ubicados en el mallado. Para cualesquiera dos bloques diferentes en las coordenadas v_i and v_j, su distancia $d(v_i, v_j)$ es el menor número de saltos que son requeridos para ir de uno de los bloques hasta el otro.

Ejemplo 2

La siguiente configuración representa una ciudad ideal hecha de 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

Su tarea es, dada una ciudad ideal, escriba un programa que permita calcular la suma de todos las distancias entre pares de bloques v_i y v_j para los cuales $i < j$. Formalmente, su 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 usted debe implementar una función `DistanceSum`(N, X, Y) la cual, dados N y dos arreglos X e Y que describen la ciudad, calcule la fórmula descrita anteriormente. Ambos X e Y son de tamaño N; el bloque i se encuentra en las coordenadas (X[i], Y[i]) para $0 \le i \le N - 1$, y $1 \le X[i], Y[i] \le 2^{31} - 2$. dado que el resultado puede ser muy grande para poder calcularlo en un entero de 32 bits, usted debe retornar el resultado modulo 1 000 000 000 (mil millones).

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

Sub-tarea 1 [11 puntos]

Usted puede asumir $N \le 200$.

Sub-tarea 2 [21 puntos]

Usted puede asumir $N \le 2\,000$.

Sub-tarea 3 [23 puntos]

Usted puede asumir $N \leq 100\,000$.

Adicionalmente, las siguientes condiciones se cumplen: dadas cualesquiera dos celdas no-vacías i y j, tal que, X[i] = X[j], cada celda entre ellas es no-vacía también; dadas cualesquiera dos celdas no-vacías i y j tal que, Y[i] = Y[j], cada celda entre ellas es no-vacía también.

Sub-tarea 4 [45 puntos]

Usted puede asumir $N \leq 100\,000$.

Detalles de Implementación

Usted debe enviar un solo archivo llamado `city.c`, `city.cpp` or `city.pas`. Este archivo debe implementar el subprograma descrito anteriormente utilizando las firmas siguientes.

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;
```

Este subprograma debe comportarse como se describió anteriormente. Por supuesto, usted es libre de implementar otros subprogramas para su uso interno. Sus envíos no deben interactuar en ninguna manera con entrada/salida estándar, ni con ningún archivo.

Evaluador Ejemplo

El evaluador de ejemplo provisto con el ambiente de competición esperará la entrada en el siguiente formato:

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

Límites de Tiempo y Memoria

- Tiempo límite: 1 segundo.
- Memoria límite: 256 MiB.