Идеален град

Леонардо, както много други италиански учени и артисти от онези времена, много се интересувал от планиране и проектиране на градската среда. Имал намерение да изработи модел на идеалния град: удобен, просторен и умно използващ ресурсите; колкото може по-далеч от тесните, клаустрофобични средновековни градове.

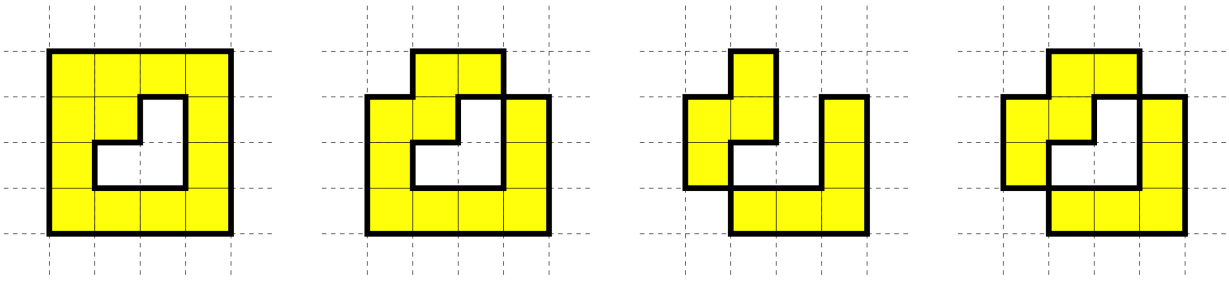
Идеалният град

Град е съставен от N блока, разположени върху безкрайна квадратна решетка от клетки. Всяка клетка се идентифицира с координатите си (номерата на реда и стълба). Съседни на клетката с координати (i, j) са клетките с координати (i - 1,j), (i + 1, j), (i, j - 1) и (i, j + 1). Всеки блок поставен върху решетката покрива точно една клетка. Блок може да бъде поставен върху клетката с координати (i, j), когато $1 \leq i, j \leq 2^{31} - 2$. Всеки блок се идентифицира с координатите на клетката, върху която е поставен. Два блока са съседни, ако лежат върху съседни клетки. В идеалния град блоковете са свързани така, че няма „дупки“ по границите им, т.е. те трябва да удовлетворяват условията:

- Всеки две *празни* клетки са свързани с поне една последователност от съседни *празни* клетки.
- Всеки две *непразни* клетки са свързани с поне една последователност от съседни *непразни* клетки.

Пример 1

Нито една от показаните на фигурите по-долу конфигурации от блокове не е идеален град: първите две отляво не изпълняват първото условие (празните клетки от вътрешността не са свързани с тези отвън), третата не изпълнява второто условие, а четвъртата - и двете.

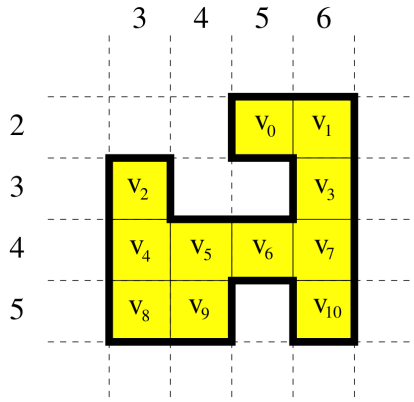


Разстояние

Ходенето из града става с *подскоци* - от блок, на съседен на него блок. Празните клетки не могат да се използват за придвижване. Нека v_0, v_1, …, v_{N-1} са координатите на N-те блока, поставени на решетката. За всеки два различни блока с координати v_i и v_j, разстояние между тях $d(v_i, v_j)$ е най-малкият брой подскоци, необходими да се стигне от единия блок до другия.

Пример 2

Конфигурацията показана по-долу е идеален град съставен от N = 11 блока с координати $v_0 = (2, 5)$, $v_1 = (2, 6)$, $v_2 = (3, 3)$, $v_3 = (3, 6)$, $v_4 = (4, 3)$, $v_5 = (4, 4)$, $v_6 = (4, 5)$, $v_7 = (4, 6)$, $v_8 = (5, 3)$, $v_9 = (5, 4)$ и $v_{10} = (5, 6)$. например, $d(v_1, v_3) = 1$, $d(v_1, v_8) = 6$, $d(v_6, v_{10}) = 2$, and $d(v_9, v_{10}) = 4$.



Задача

Напишете програма, която, за зададен идеален град, да намира сумата на всички разстояния между двойки блокове v_i и v_j, за които $i < j$. Т.е., програмата трябва да пресмята следната сума:

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

Още по-точно, вие трябва да напишете функция `DistanceSum`(N, X, Y), която, по зададени N и двата масива X и Y, които описват града, пресмята стойността на дадената по-горе формула. Както X, така и Y, са с по N елемента; блокът i е с координати (X[i], Y[i]), за $0 \le i \le N - 1$, $1 \le X[i], Y[i] \le 2^{31} - 2$. Тъй като резултатът може да се окаже твърде голям за да се събере в 32 бита, програмата трябва да го пресметне като остатък по модул 1 000 000 000 (един милион).

В Пример 2 има 11 × 10 / 2 = 55 двойки от блокове. Сумата от разстоянията между всяка двойка блокове е 174.

Подзадача 1 [11 точки]

Може да разчитате, че $N \le 200$.

Подзадача 2 [21 точки]

Може да разчитате, че $N \leq 2\,000$.

Подзадача 3 [23 точки]

Може да разчитате, че $N \leq 100\,000$.

Освен това, в сила са следните две твърдения: ако за непразните клетки i и j е в сила X[i] = X[j], всяка клетка между тях е също непразна; ако за непразните клетки i и j е в сила Y[i] = Y[j], всяка клетка между тях е също непразна.

Подзадача 4 [45 точки]

Може да разчитате, че $N \leq 100\,000$.

Детайли на имплементацията

В оценяващата система трябва да изпратите точно един файл с име `city.c`, `city.cpp` или `city.pas`. Файлът трябва да съдържа функция, която решава задачата със следния прототип:

C/C++ програми

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

Pascal програми

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

Изпратената функция трябва да има описаното по-горе поведение. Разбира се, може да включите във файла и други функции за вътрешна употреба. Изпратените функции не бива да има нищо общо със стандартния вход/изход нито с какъвто и да било друг файл.

Прост грейдер

Примерният грейдър, който ще бъде предоставен, очаква вход в следния формат:

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

Ограничения по време и памет

- Ограничение по време: 1 секунда.
- Ограничение по памет: 256 MiB.