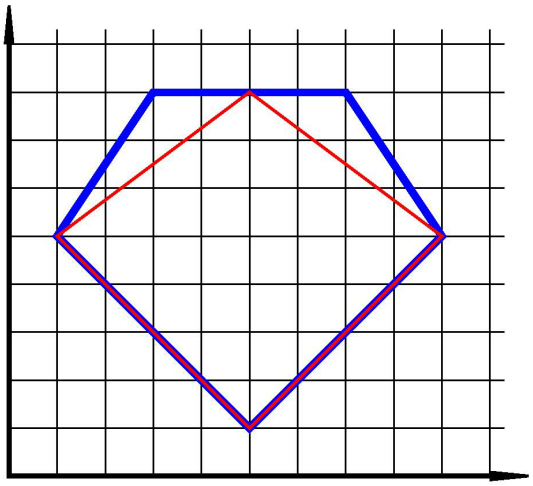
Mejorando la vigilancia de un barrio

Contribución de Hugo Ryckeboer

Descripción del problema

El dueño de un terreno quiere convertirlo en un barrio privado. Su terreno tiene forma de un polígono convexo. Es consciente que para tener una buena vigilancia del perímetro debería colocar torres de observación en todos los vértices pero sabe que esta alternativa encarecerá en gran medida las expensas del condominio. Decide entonces inscribir un polígono con un número menor de lados y donar a la municipalidad el excedente. Por supuesto intentará conservar la mayor superficie posible sabiendo además que la municipalidad, para simplificar su catastro, pide que todas las propiedades tengan vértices de coordenadas enteras.



Para ayudarlo se te pide que elabores una función **barrio(N, vertices, M)** que recibiendo como dato los vértices del terreno disponible y el número de vértices del polígono buscado determine la superficie que entregará a la municipa¬lidad, la cual deberá ser la menor posible. Para simplificar, la función deberá devolver el área truncada a su parte entera. Sus parámetros son:

N: cantidad de vértices del polígono dato (**3** ≤ **N** ≤ **1.000**).

vertices: arreglo de tamaño **N** que contiene las coordenadas de cada vértice. Se encuentran ordenadas siguiendo el perímetro en sentido horario. Todas las coordenadas son no-negativas.

M: cantidad de vértices del polígono a diseñar (**3** ≤ **M** ≤ **N**).

Ejemplo

Si se presenta el siguiente caso:

N=5; M=4

Vértices:

5 1

1 5

3 8

7 8

9 5

La función debería devolver 6

Detalles de implementación

En un único archivo llamado **barrio.c**, **barrio.cpp** o **barrio.pas** debes enviar una función que implemente el análisis descrito anteriormente, usando los siguientes prototipos:

En C/C++:

long long **barrio**(long N, long vertices[][2], long M) ;

En Pascal:

type verticesA = array[1..1000][0..1] ;

function **barrio**(N : longint ; var vertices : verticesA ; M : longint): int4 ;

Evaluador local

El evaluador local (programa para probar ejemplos propios) lee la entrada por stdin en el siguiente formato:

- Línea 1: N y M separados por un blanco
- Línea 2+i (0 ≤ i < N): 2 enteros separados por blanco indicando las coordenadas del polígono dato.

La función entrega el resultado por consola.

Para el caso del ejemplo la entrada podría ser:

```
5 4
5 1
1 5
3 8
7 8
9 5
```

Y la salida

```
6
```

Subtareas

Habrá casos de prueba por un total de **10** puntos donde (**3** ≤ **M** ≤ **N** ≤ **20**).
Habrá casos de prueba por un total de **30** puntos donde (**3** ≤ **M** ≤ **N** ≤ **50**).
Habrá casos de prueba por un total de **30** puntos donde (**N-20** ≤ **M** ≤ **N** ≤ **100**)
Habrá casos de prueba por un total de **30** puntos donde (**N-4** ≤ **M** ≤ **N** ≤ **1000**).

Todos los casos de prueba corresponden a alguna de las cuatro subtareas mencionadas.