# *4ª. Olimpiada de Informática del Estado de Guanajuato*

Guanajuato, Gto. a 12 de Abril de 2003

**Problema 1. Atentados**

El Comando Internacional de Maniobras Anti-Terroristas (CIMAT) ha estado siguiendo los pasos de un grupo terrorista muy peligroso. Después de un tiempo logró interceptar una transmisión que, se presume, proviene del cuartel general de este grupo. Después de analizar la señal, se encontró que se trataba de una serie de coordenadas que señalan la ubicación de los siguientes atentados.

Aunque se sabe la ubicación de los siguientes atentados no se sabe cuál de ellos será el primero en ocurrir, cuál el segundo, y así sucesivamente. Lo que el CIMAT sabe es que este grupo terrorista opera de una manera eficiente, es decir, se sabe que buscará la manera de realizar los atentados en el menor tiempo posible, entonces la estrategia del CIMAT es encontrar una secuencia en la que la distancia recorrida entre todos los lugares donde se realizarán los atentados sea mínima.

Aunque se hizo un esfuerzo por obtener esta información en el menor tiempo posible, el primer atentado acaba de ocurrir. Tu misión, si decides aceptarla, es ayudar al CIMAT a determinar la secuencia en la que ocurrirán los siguientes atentados para así poder evitarlos.

## Entrada

Tu programa debe leer del archivo **input.txt** los siguientes datos: en la primera línea el número N de atentados que ocurrirán (N<7). En las siguientes N líneas las coordenadas **x, y, (x e y son enteros)** de cada lugar donde ocurrirá un atentado **(todas las coordenadas son distintas)**. En la última línea las coordenadas **(a, b)** del atentado que acaba de ocurrir.

## Salida

Basta con que tu programa muestre la distancia mínima que tendrán que recorrer los terroristas para llevar a cabo todos los atentados comenzando desde el lugar donde se realizó el atentado que acaba de ocurrir.

## Ejemplos de entrada y salida

| Input.txt | Output.txt |
|---|---|
| 2<br>0 10<br>10 0<br>0 10 | 14.1421 |

| Input.txt | Output.txt |
|---|---|
| 4<br>1 2<br>1 0<br>0 1<br>2 1<br>1 0 | 4.2426 |

| Input.txt | Output.txt |
|---|---|
| 6<br>1 5<br>0 10<br>3 5<br>2 4<br>4 2<br>3 3<br>0 10 | 11.3416 |

# *4ª. Olimpiada de Informática del Estado de Guanajuato*

Guanajuato, Gto. a 12 de Abril de 2003

**Problema 1. Zonas**

Las Fuerzas Auxiliares de Maniobras Anti-Terroristas (FAMAT), han sido alertadas de los "posibles" atentados. Éstos ocurrirán dentro de un fraccionamiento dividido en manzanas. Aunque se sabe de los lugares donde supuestamente ocurrirán, no se sabe a ciencia cierta si se trata de un señuelo o un verdadero atentado. La lista de lugares es sorprendentemente larga, por lo que se ha decidido evacuar las "**manzanas de riesgo"**. Una manzana se considera "**de riesgo**" si está **habitada** y existe la amenaza de que un atentado suceda en ella o en una de sus manzanas vecinas. Una manzana **deshabitada nunca** se considera "**de riesgo**". El trabajo de FAMAT es encontrar y reportar las manzanas de riesgo.

## Entrada

Se ha codificado el fraccionamiento en una cuadrícula de **M** filas por **N** columnas, cada cuadro representa una manzana del fraccionamiento. En cada cuadro puede haber un **1** o un **0**, donde **1** significa que la manzana está habitada y **0** significa que la manzana está deshabitada.

Cada cuadro tiene asociadas unas coordenadas, de tal forma que el cuadro de la esquina superior izquierda de la cuadrícula es el cuadro **(0, 0)** y el de la esquina inferior derecha es **(M, N)**.

El archivo **input.txt** contiene los enteros **M** (1<=M<=20) y **N**, (1<=N<=20) que son las dimensiones del fraccionamiento. En cada una de las siguientes **M** líneas, están **N** números, separados por un espacio, los **M** renglones de **N** columnas cada uno, forman la representación del fraccionamiento.

A continuación el número **P** de posibles atentados, seguido por P líneas, cada una de las cuales contiene dos enteros **f** y **c**, separados por un espacio en blanco, que indican que la manzana con coordenadas **(f, c)** está **amenazada.**

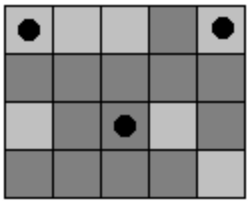
## Salida

Tu programa debe escribir en el archivo "output.txt" un arreglo de dígitos de **M** filas por **N** columnas, donde cada dígito puede ser **1** si la manzana es de riesgo o **0** en caso contrario.

### Ejemplo de entrada y salida

| Input.txt | Output.txt |
|---|---|
| 4 5<br>0 0 0 1 0<br>1 1 1 1 1<br>0 1 1 0 1<br>1 1 1 1 0<br>3<br>0 0<br>2 2<br>0 4 | 0 0 0 1 0<br>1 1 1 1 1<br>0 1 1 0 0<br>0 1 1 1 0 |

Este ejemplo se ilustra en la siguiente figura:



Las zonas claras representan manzanas **NO HABITADAS**, mientras que las zonas obscuras representan manzanas **HABITADAS**, los puntos negros marcan las manzanas donde posiblemente ocurrirá un atentado.

# *4ª. Olimpiada de Informática del Estado de Guanajuato*

Guanajuato, Gto. a 12 de Abril de 2003

### Problema 1. Códigos

Luego de evacuar las zonas de riesgo, y gracias a la ayuda de la Organización de Informática y Estudios Generales (OIEG), se desarrolló una estrategia óptima para llegar al lugar del siguiente atentado. Se trata de un dispositivo explosivo colocado por los terroristas y consta de un mecanismo más flexible que los convencionales. La forma de desactivarlo es ingresando un código de K dígitos mediante un pequeño teclado que se encuentra a la vista del dispositivo. En caso de ingresar un código incorrecto, el dispositivo **no** estalla inmediatamente, sino que lleva la cuenta de los códigos incorrectos que hayan sido ingresados, el dispositivo estallará si se ingresan más de 4000 códigos incorrectos.

Agentes de la OIEG saben que cuando se ingresa una serie de dígitos al dispositivo, éste responde mostrando en una pequeña pantalla, la suma de las diferencias de los dígitos ingresados con los del código verdadero. Por ejemplo, si el código verdadero es **1234** y se ingresa el código **4321**, el dispositivo mostrará en la pantalla el número **8**, ya que:

**|4-1| + |3-2| + |2-3| + |1-4| = 3 + 1 + 1 + 3 = 8.**

Afortunadamente, uno de los agentes lleva con sigo una computadora portátil que puede conectar al dispositivo para ingresar combinaciones de dígitos automáticamente. Tu misión es escribir un programa capaz de ingresar combinaciones de dígitos para encontrar el código que desactive el dispositivo lo más pronto posible.

## Instrucciones para programadores en C/C++

1. En el directorio **c:\tc\bin\codigos** encontrarás un archivo llamado **codigos.h**, debes copiar este archivo en tu directorio de trabajo.
2. Al principio de tu programa principal incluye la siguiente línea de código: ***#include "codigos.h"***
3. Ahora dentro de tu programa puedes utilizar las siguientes funciones:
   - ? void inicializar(void);
   - ? int ingresarCodigo(int k, int *codigo);
   - ? void finalizar(int k, int *codigo);

## Instrucciones para programadores en Pascal

1. En el directorio **c:\tc\bin\codigos** encontrarás un archivo llamado **codigos.tpu**, debes copiar este archivo en tu directorio de trabajo.
2. Al principio de tu programa principal incluye la siguiente línea de código: ***uses codigos;***
3. Ahora dentro de tu programa puedes utilizar las siguientes funciones:
   - ? procedure inicializar();
   - ? function ingresarCodigo(k:integer, codigo:array of integer):integer;
   - ? procedure finalizar(k:integer, codigo:array of integer);

# *4ª. Olimpiada de Informática del Estado de Guanajuato*

Guanajuato, Gto. a 12 de Abril de 2003

## Instrucciones generales:

Lo primero que debe hacer tu programa es llamar a la función **inicializar**, que no recibe parámetros.
La función **ingresarCódigo** ingresa un código de k dígitos al dispositivo, y devuelve la suma de las diferencias de los dígitos ingresados con los del código verdadero. El código es simplemente un arreglo de k enteros, cada entero en el arreglo se considera un dígito.

La función **finalizar** debe ser llamada cuando estés seguro de que conoces el código verdadero. Esta función escribirá en el archivo "resultado.txt" si tu solución es correcta o incorrecta y el número de intentos (número de llamadas a "**ingresarCodigo**") que hizo tu programa.

## Entrada

En el archivo **input.txt** está en la primera linea el número K (K<200) de dígitos que compone un código.

## Salida

En algún momento tu programa debe llamar a la función **finalizar,** que recibe como parámetros el número k de dígitos y un arreglo de K dígitos (enteros), el arreglo debe contener el código correcto.