

TRIMINOS

DESCRIPCIÓN

Un trimino en forma de L es una pieza compuesta por 3 cuadrados, cada uno de dimensiones 1x1 que forman cualquiera de las siguientes figuras:



Cierto día, paseando por la calle, viste a un par de personas jugando un juego de mesa, del cual no lograste entender las reglas, pero si notaste algunas cosas:

- Se juega en un tablero cuadrado de H cuadros de alto por B cuadros de ancho, las piezas son cuadrados de 1x1.
- En su turno, cada jugador coloca 3 piezas en el tablero, las piezas pueden colocarse en cualquier posición sin importar si ya está ocupada por otra pieza.

Como te gustan mucho los triminos quieres saber en qué turnos se colocó algún trimino. Además te gustaría saber cuántos cuadros del tablero quedaron libres (sin ninguna pieza) al final del juego.

PROBLEMA

Deberás escribir un programa que reciba como entrada las dimensiones del tablero, las jugadas que se realizan durante el juego y que determine en qué jugadas alguien colocó sus piezas en forma de un trimino como los de arriba y cuántos cuadros quedaron libres al final.

ENTRADA

Tu programa deberá leer del teclado de la PC los siguientes datos

- En la primera línea los números $0 < H, B \leq 100$ los cuales representan el alto y el ancho del tablero.
- En la segunda línea un número $0 < N \leq 100$ representando la cantidad de movidas que se hicieron durante el juego.
- Cada una de las siguientes N líneas representará una movida de un jugador y contendrá 6 números enteros positivos a, b, c, d, e y f separados por espacios, donde a y b representarán la fila y columna donde se colocó la primera pieza de ese movimiento, c y d la segunda pieza y e y f la tercera pieza. La esquina superior izquierda del tablero está representada por la coordenada (1,1)

SALIDA

Tu programa deberá escribir en la pantalla de la computadora $N+1$ líneas, en cada una de las primeras N líneas deberá haber un 1 si las fichas de esa jugada corresponden a un trimino con forma de L y un 0 en caso contrario. La última línea deberá contener un solo número que representa la cantidad de cuadros libres al final del juego.

EJEMPLO

ENTRADA	SALIDA
5 6	1
4	0
2 3 3 3 2 4	0
1 1 4 2 2 5	1
2 6 2 6 2 6	20
5 4 5 5 4 5	

REQUERIMIENTOS DE EJECUCION

Para obtener puntos en este problema, además de entregar la solución correcta tu programa deberá ejecutarse en un tiempo menor a 1 segundo.

TERCIAS

DESCRIPCIÓN

Se dice que 3 números a , b , y c están en sucesión aritmética si, $a < b < c$, y $c - b = b - a$. Por ejemplo los números 1, 2 y 3 están en sucesión aritmética, los números 2, 20 y 38 también lo están. Pero los números 2, 3 y 20 no lo están.

Determinar si 3 números están en sucesión aritmética es una tarea simple, pero dado un conjunto de números determinar cuántos de ellos están en sucesión aritmética es algo un poco más difícil.

PROBLEMA

Debes escribir un programa que lea un conjunto de números distintos e imprima en pantalla cuantas tercias de esos números están en sucesión aritmética.

ENTRADA

Tu programa deberá leer del teclado de la PC los siguientes datos:

- En la primera línea el número $0 < N \leq 7,000$ que representa la cantidad de números en el conjunto.
- Cada una de las siguientes N líneas contendrá un número del conjunto el cuál será un entero mayor a 0 y menor o igual a 1,000,000,000, además, estos números estarán ordenados de manera creciente.

SALIDA

Tu programa deberá escribir en la pantalla de la PC un solo número entero indicando cuántas tercias de los números hay en sucesión aritmética.

EJEMPLO

ENTRADA	SALIDA
5 1 2 3 20 38	2

EXPLICACION

En el ejemplo anterior se tiene un conjunto con 5 números, con estos 5 números se pueden formar las siguientes tercias:

- (1,2,3)
- (1,2,20)
- (1,2,38)
- (1,3,20)
- (1,3,38)
- (1,20,38)
- (2,3,20)
- (2,3,38)
- (2,20,38)
- (3,20,38)

De las tercias anteriores sólo las tercias (1,2,3) y (2,20,38) están en sucesión aritmética, por lo tanto la respuesta es 2.

REQUERIMIENTOS DE EJECUCION

Para obtener puntos en este problema, además de entregar la solución correcta tu programa deberá ejecutarse en un tiempo menor a 1 segundo.

CAMIONES

DESCRIPCIÓN

Recientemente se inauguró el Parque de los Clonosaurios, Juanito, el gerente del parque se enfrenta a un dilema difícil de resolver. Por lo que ha pedido tu ayuda.

El parque se encuentra retirado de la civilización, para llegar a el es necesario tomar un camión. Los camiones del parque funcionan de la siguiente manera:

- A cierta hora del día, llamémosle H , llega a la parada el primer camión de ese día.
- El camión espera t nanosegundos en la parada y se retira.
- En el instante en el que el camión se retira, llega un siguiente camión a la parada.
- El proceso se repite hasta que se retira el último camión del día.

Juanito renta los camiones, si al momento de retirarse el camión no tiene pasajeros inmediatamente regresa y no es necesario que se pague la corrida. Como buen gerente, Juanito desea cortar los gastos de su parque, por lo que necesita minimizar el número de camiones que se pagan en el día.

En base a estadísticas, Juanito sabe en qué momento llega gente a la parada. El tiempo t de nanosegundos que espera cada camión no es negociable, sin embargo Juanito tiene opción de recorrer la hora de llegada del primer camión hasta un máximo de t nanosegundos, de modo que puede decidir que el primer camión llegue en cualquier momento entre H y $(H + t)$.

PROBLEMA

Dados dos números t y n , donde t representa el número de nanosegundos que espera cada camión en la parada y n el número de personas que se esperan ese día y una lista de números que representan el nanosegundo del día en el que cada persona llega a la parada, escribe un programa que determine en qué momentos puede llegar el primer camión de modo que se minimice el número de camiones en donde subió al menos una persona.

ENTRADA

Tu programa debe leer del teclado de la PC los siguientes datos:

- En la primera línea los números t y n separados por un espacio, con $1 \leq t, n \leq 100,000$, donde t representa el tiempo en nanosegundos que espera cada camión y n el número de personas que se esperan ese día.
- La segunda línea contiene n números enteros separados por un espacio que representan el nanosegundo en que cada una de las n personas llegó a la parada, esta lista de números SIEMPRE ESTARÁ ORDENADA DE MANERA CRECIENTE.

CONSIDERACIONES

- No hay dos personas que lleguen en el mismo nanosegundo a la parada.
- Si una persona llega a la parada antes que el primer camión, la persona se va a su casa y presenta una queja contra el administrador del parque.
- Si alguien pone una queja contra Juanito, éste será despedido. Por lo que Juanito requiere que TODAS las personas de ese día lleguen al parque.
- El nanosegundo en que llega cada persona a la parada esta contado a partir de H , es decir H corresponde al nanosegundo 0.
- El nanosegundo de llegada de las personas está representado por un número mayor o igual a 0 y menor o igual a 2,000,000,000.
- Si una persona llega a la parada en el nanosegundo en el que se está retirando un camión, no puede subir a ese camión, debe hacerlo en el siguiente. Por ejemplo, si el tiempo de espera de cada camión es de 10 ns, y el camión llega en el nanosegundo 0, entonces a ese camión suben las personas que lleguen entre el nanosegundo 0 y el 9, una persona que llegue en el nanosegundo 10, tiene que subir al siguiente camión.
- La capacidad de cada camión es suficiente para alojar a todas las personas que llegan en ese día.

SALIDA

Tu programa deberá escribir en la pantalla de la PC los siguientes datos:

- En la primera línea un número entero indicando el número mínimo de camiones que se requieren para llevar a toda la gente.
- La segunda línea deberá contener una lista de enteros separados por un espacio que representan todos los posibles inicios que obtienen dicho mínimo.

Por ejemplo, si el tiempo de espera de cada camión es de 10 ns, esto quiere decir que Juanito puede decidir que el primer camión llegue en cualquier momento entre el nanosegundo 0 y el nanosegundo 10. En la salida deberán aparecer todos los nanosegundos entre 0 y 10 en donde el número de camiones utilizados sea igual al mínimo posible.

EJEMPLOS DE ENTRADA

ENTRADA	SALIDA
3 5	2
4 5 7 8 9	1

ENTRADA	SALIDA
4 3	2
7 14 15	1 2 4

ENTRADA	SALIDA
2 10	7
3 4 7 8 12 13 14 15 20 21	1 2

REQUERIMIENTOS DE EJECUCION

Para obtener puntos en este problema, además de entregar la solución correcta tu programa deberá ejecutarse en un tiempo menor a 3 segundos.