La Presa (ejemplo del problema de etapa 1° base de programación)

Desgraciadamente ha llovido mucho y se requiere abrir las compuertas de la presa pero no se desea inundar al pueblo más cercano, el camino de la presa al pueblo está marcado por **n** trozos de terreno cada uno de cierta altura y el pueblo es el de mayor altura.

Tarea

Conociendo la altura del pueblo y la altura de cada trozo del camino de la presa al pueblo debes escribir un programa que nos indique cual es la cantidad de agua que se puede liberar de la presa sin afectar al pueblo.

Entrada

En la primera línea un número entero representando la altura del pueblo, nunca será mayor a 50,000.

En la segunda línea un número entero **n** que representa la cantidad de trozos de terreno de la presa al pueblo el número máximo de trozos será de 50.

En la tercera línea las alturas de cada uno de los trozos del terreno del camino de la presa al pueblo, la altura de cada uno de los trozos del camino siempre es menor a la altura del pueblo.

Salida

Un solo valor indicando la cantidad de agua que se puede liberar sin inundar el pueblo.

Ejemplos

Entrada ejemplo 1	**Respuesta**	**Explicación**
10 4 3 5 7 7	18	La altura del pueblo es 10 Existe 4 trozos de terreno de la presa al pueblo, el primero tiene altura de 3 el segundo altura de 5 el tercero y cuarto altura de 7. Por tanto podemos liberar 18 unidades de agua. Sin inundar el pueblo (en el primer trozo colocamos 7 unidades de agua, en el segundo trozo 5 unidades de agua en el tercero y cuarto trozo colocamos en cada uno 3 unidades de agua).

Entrada ejemplo 2	**Respuesta**	**Explicación**
60 5 20 10 30 50 47	143	El la altura del pueblo es 60. Existen 5 trozos de terreno en el trayecto de la presa al pueblo el primero con altura 20, el segundo con altura 10,... el quinto con altura 47. Por tanto se puede liberar 143 unidades de agua sin inundar el pueblo (colocamos 40 unidades en el 1° trozo, 50 en el 2°, 30 en el 3°, 10 en el 4° y 13 en el 5° trozo).

Campamento (ejemplo del problema de etapa 2° lenguaje de programación)

Puntos	27.89	Límite de memoria	32 MiB
Límite de tiempo (caso)	1s	Límite de tiempo (total)	1m0s
Tamaño límite de entrada (bytes)	10 KiB		

Descripción

En un parque recreativo se tiene un campamento, el cual es un área rectangular de 1<=R<=100 renglones y de 1<=C<=100 columnas y tiene numerados los lugares con valores del 1 al 100,000, con base a la siguiente regla: en un mismo renglón los lugares están de forma ascendente y en cada columna también están numerados de forma ascendente, como se puede apreciar en la figura un área de 3 renglones por 4 columnas.

r/c	1	2	3	4
1	10	17	20	30
2	16	20	25	40
3	17	25	37	41

Los boletos válidos son los que aparecen en la tabla, por ejemplo los números 16, 25 y 41 son válidos porque aparecen en la tabla. El 16 aparece en la posición (r=2,c=1), el 25 aparece en las posiciones (r=3,c=3) y (r=2,c=3) y el 41 aparece en la posición (r=3,c=4).
Los boletos 22 y 33 son inválidos porque no aparecen en la tabla.

Problema

Escribe una función que determina si un boleto es válido (indicando su posición en forma de renglón y columna). Si el boleto es inválido deberá ser indicado con los valores de renglón = 0 y columna = 0.

Implementación

Deberás implementar la función llamada:

```
void  ticket(int R, int C, int B)
```

Donde las variables “R” y “C” representan el total de renglones y columnas que existen en el campamento, respectivamente, y la variable “B” representa el número de boleto que debes validar.

Para conocer el número que tiene cada lugar de campamento:

Debes utilizar la función “consulta”:

```
int consulta(int r, int c)
```

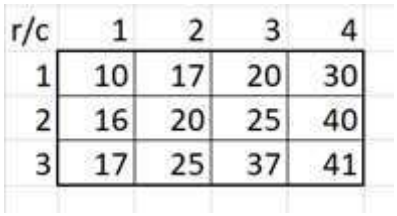
Qué recibe 2 parámetros de entrada: el renglón “r” y la columna “c”, devolverá el valor que existe en esa posición. Regresará un -1 en caso de consultar una coordenada que no exista.

Para enviar tu respuesta:

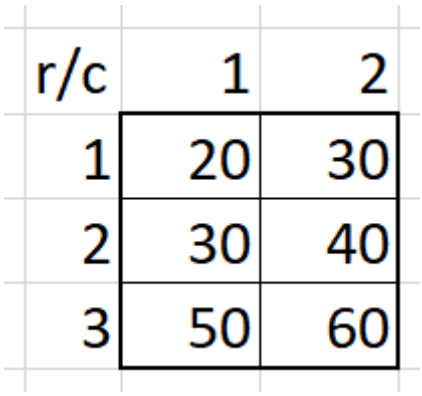
Debes hacerlo a través de la función respuesta: void respuesta (int r, int c)
en donde el parámetro “r” representa el renglón y el parámetro “c” representa la columna.

Ejemplo 1

Entrada	Salida	Descripción
`Ticket(3,4,17)`		El evaluador invoca mi función con un terreno de renglones 3 x 4 columnas y me pide que verifique si el boleto 17 es válido. El evaluador tiene este mapa del terreno.

Entrada	Salida	Descripción
		
`consulta(1,1)`	`Regresa 10`	Consulto el valor del renglón 1 y columna 1 y el evaluador devuelve 10.
`consulta(1,2)`	`Regresa 17`	Consulta la posición del renglón 1 y columna 2 y el evaluador devuelve el valor 17.
`respuesta(1,2)`		He encontrado una posición correcta dentro del terreno de campamento y se utilizó la función consulta menos de 3+4-1=6 veces (menos de R+C-1).

Ejemplo 2

Entrada	Salida	Descripción
`Ticket(3,2,100)`		El evaluador invoca mi función con un terreno de renglones 3 x 2 columnas y me pide que verifique si el boleto 100 es correcto. El evaluador tiene esté mapa del terreno 
`consulta(3,2)`	`Regresa 60`	Consulto el valor del renglón 2 y columna 4 y el evaluador devuelve 60.
`respuesta(0,0)`		Como el boleto 100 no está en la tabla es inválido y debo invocar a respuesta con valor de (0,0). Se realizaron menos de R+C-1 llamados a “consulta”.

Consideraciones de evaluación

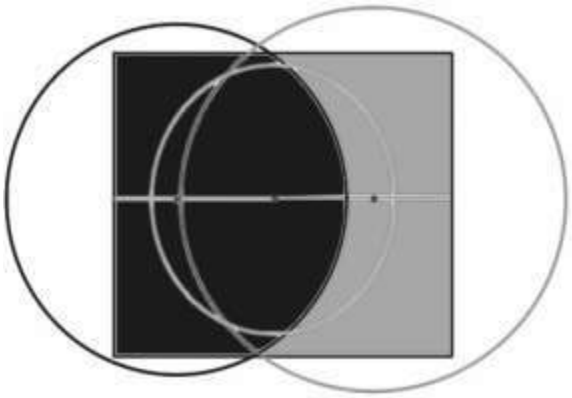
Los casos de boletos inválidos estarán agrupados a casos con boletos válidos.
Para el 40% de los puntos, podrás hacer uso de la función consulta R*C veces.
Para los demás casos, podrás hacer uso de la función consulta R+C-1 veces.

Circulismo (ejemplo del problema de etapa 3° Estructuras de datos)

Puntos	35.62	Límite de memoria	32 MiB
Límite de tiempo (caso)	1s	Límite de tiempo (total)	1m0s
Tamaño límite de entrada (bytes)	10 KiB		

Descripción

Te has inscrito a una escuela de arte y la clase del día de hoy es circulism o, donde todo lo que se dibuja sobre el lienzo son círculos.
Tu primera tarea de la clase es poder pintar un lienzo de altura H y base B, solo puedes dibujar círculos, pero con el detalle de que cada círculo tiene que tener su centro sobre la línea horizontal que está en medio, el maestro te dio una descripción de una serie de círculos (su posición y su radio) que son los únicos círculos que puedes dibujar, tienes que pintar cada círculo con un color diferente.
Pero la pintura es muy cara por eso quieres comprar la menor cantidad de colores y como tienes que cubrir cada rincón de cuadro con al menos un color, quieres saber cuál es el menor número de colores necesarios para realizar tu primer obra maestra de circulismo.



Entrada

Primer línea: contiene 3 números H, B y N, donde H es la altura del cuadro, B es la longitud de la base y N es el número de círculos que tienes permitido dibujar.
Siguientes N líneas: cada línea representa la descripción de cada círculo y contiene dos números X y R, donde X es la posición del centro del círculo y R es el radio del círculo.

Salida

Tu programa debe decir cuál es el mínimo número de colores que se necesita para realizar tu pintura o imprimir -1 si no es posible pintar todo el cuadro.

Ejemplo

Entrada	Salida	Descripción
30 30 3 10 19 15 13 20 20	2	Vea la imagen de arriba en la descripción

Límites
1≤N≤10,000
1≤H,B≤20,000

Fuente: OMI CDMX-EDOMEX Selectivo 2014

Defendiendo el Castillo (ejemplo del problema de Etapa 3° estructuras de datos)

Puntos	20	Límite de memoria	32 MiB
Límite de tiempo (caso)	1s	Límite de tiempo (total)	1m0s
Tamaño límite de entrada (bytes)	10 KiB		

Descripción

Legolas lidera un grupo de elfos arqueros para defender el castillo que está siendo atacado por una armada de orcos furiosos. Tres lados del castillo son protegidos por montañas impenetrables y el lado sobrante está ocupado por una larga muralla divididad en n secciones. En este momento hay exactamente ai arqueros localizados en la i-ésima sección de la muralla. Se sabe que un arquero situado en la sección i puede disparar a los orcos que atacan las secciones localizadas a distancia no mayor a r, esto es todas aquellas secciones j tales que $|i-j| \leq r$. En particular, r=0 significa que los arqueros pueden tirar únicamente a los arqueros que atacan la sección i.

Se define el *nivel de defensa* de la sección i como el número de arqueros que pueden disparar a los orcos que atacan esa sección. La *seguridad* del plan de defensa es el valor mínimo del *nivel de defensa* de todas las secciones.

Hay poco tiempo para que los orcos lleguen a atacar el castillo, por lo que no se puede reacomodar los arqueros que ya están posicionados en la muralla. Sin embargo, aún hay k arqueros restantes que se pueden acomodar de cualquier forma a lo largo de la muralla. Ayuda a Legolas a alcanzar la máxima *seguridad* posible del plan de defensa.

Entrada

La primera línea contiene tres enteros n, r y k, el número de secciones de la muralla, la distancia máxima a otra sección que los arqueros pueden disparar y el número de arqueros restantes para acomodar a lo largo de la muralla.

La segunda línea contiene n enteros a1,a2,...,an, el número actual de arqueros en cada sección.

Salida

Imprime el máximo valor posible de *seguridad*; es decir, el máximo valor posible del mínimo *nivel de defensa* si acomodamos k arqueros adicionales de manera óptima.

Ejemplo

Entrada	Salida
5 0 6 5 4 3 4 9	5
4 2 0 1 2 3 4	6
5 1 1 2 1 2 1 2	3

Límites

$1 \leq n \leq 500{,}000$
$0 \leq r \leq n$
$0 \leq k \leq 1018$
$0 \leq ai \leq 109$

Fuente: OMI CDMX-EDOMEX Selectivo 2019

PANTANO (ejemplo del problema de etapa 4° algoritmos)

DESCRIPCION

Natalia se encuentra en una tumba dentro de la selva maya y ha visto el tesoro, sin embargo hay una leyenda escrita que dice “al momento que alguien tome el tesoro se activaran las trampas”, posteriormente encuentras un códice con lo siguiente las siguientes indicaciones, La zona protegida del tesoro abarca una cuadricula de renglones por columnas en donde cada celda es representada un valor de humedad. El comportamiento de las trampas es el siguiente, en todos los lugares donde existe un número primo en el primer segundo si alguien pisa esa celda morirá, en el siguiente segundo se desactiva la trampa en el tercer segundo se vuelve activar y así sucesivamente, Natalia solo puede caminar hacia una celda continua en un segundo en cualquiera de las 8 direcciones (horizontales, verticales y diagonales), Natalia como linda mujer prefiere salir sana y salva y lo más seca posible para que no se estropee su maquillaje. Ella siempre debe camina, ósea no puede permanecer 2 unidades de tiempo en una misma casilla.

PROBLEMA

Escribe un programa que ayude a Natalia a salir de la zona protegida del tesoro en el menor tiempo posible pasando por el recorrido menos húmedo. En caso de que existan varios caminos para salir de la zona en la misma cantidad de tiempo se debe elegir el camino que tenga menos humedad.

ENTRADA

Primer renglón

Las coordenadas (renglón y columna) en donde se ubican Natalia y el tesoro en 1<=Y<= R y 1<= X<=C.

Segundo renglón

Los números de renglones 3<=R<=200 y columnas 3<=C<=200 que representan el mapa

Siguientes R renglones y C columnas representado la humedad 1<=H <=1000 en cada celda todos los valores separados por un espacio.

SALIDA

Un solo renglón con 2 valores el primer valor es el tiempo en que tarda en salir de la zona protegida y el segundo valor es el la humedad acumulada por la que tuvo que pasar en caso de existir empate en el tiempo se debe elegir el camino con menor humedad.

CONSIDERACIONES

Tu programa deberá ejecutarse como máximo en 1 segundo.

EJEMPLO DE ENTRADA

```
2 3
3 4
1 5 7 10
4 4 8 7
3 11 2 2
```

EJEMPLO DE SALIDA

```
2 10
```