Olimpiada Chilena de Informática 2021

Final

22 de Enero, 2022

Información General

Esta página muestra información general que se aplica a todos los problemas.

Envío de una solución

1. Los participantes deben enviar **un solo archivo** con el código fuente de su solución.
2. El nombre del archivo debe tener la extensión `.cpp` o `.java` dependiendo de si la solución está escrita en `C++` o `Java` respectivamente. Para enviar una solución en Java hay que seguir algunos pasos adicionales. Ver detalles más abajo.

Casos de prueba, subtareas y puntaje

1. La solución enviada por los participantes será ejecutada varias veces con distintos casos de prueba.
2. A menos que se indique lo contrario, cada problema define diferentes subtareas que lo restringen. Se asignará puntaje de acuerdo a la cantidad de subtareas que se logre solucionar de manera correcta.
3. A menos que se indique lo contrario, para obtener el puntaje en una subtarea se debe tener correctos todos los casos de prueba incluídos en ella.
4. Una solución puede resolver al mismo tiempo más de una subtarea.
5. La solución es ejecutada con cada caso de prueba de manera independiente y por tanto puede fallar en algunas subtareas sin influir en la ejecución de otras.

Entrada

1. Toda lectura debe ser hecha desde la **entrada estándar** usando, por ejemplo, las funciones `scanf` o `std::cin` en C++ o la clase `BufferedReader` en Java.
2. La entrada corresponde a un solo caso de prueba, el cual está descrito en varias líneas dependiendo del problema.
3. **Se garantiza que la entrada sigue el formato descrito** en el enunciado de cada problema.

Salida

1. Toda escritura debe ser hecha hacia la **salida estándar** usando, por ejemplo, las funciones `printf`, `std::cout` en C++ o `System.out.println` en Java.
2. El formato de salida es explicado en el enunciado de cada problema.
3. **La salida del programa debe cumplir estrictamente con el formato indicado**, considerando los espacios, las mayúsculas y minúsculas.
4. Toda línea, incluyendo la última, debe terminar con un salto de línea.

Envío de una solución en Java

1. Cada problema tiene un *nombre clave* que será especificado en el enunciado. Este nombre clave será también utilizado en el sistema de evaluación para identificar al problema.
2. Para enviar correctamente una solución en Java, el archivo debe contener una clase llamada igual que el nombre clave del problema. Esta clase debe contener también el método `main`. Por ejemplo, si el nombre clave es `marraqueta`, el archivo con la solución debe llamarse `marraqueta.java` y tener la siguiente estructura:

```
public class marraqueta {
  public static void main (String[] args) {
    // tu solución va aquí
  }
}
```

3. Si el archivo no contiene la clase con el nombre correcto, el sistema de evaluación reportará un error de compilación.
4. La clase no debe estar contenida dentro de un *package*. Hay que tener cuidado pues algunos entornos de desarrollo como Eclipse incluyen las clases en un *package* por defecto.
5. Si la clase está contenida dentro de un package, el sistema reportará un error de compilación.

Problema A
La torre de la maldad

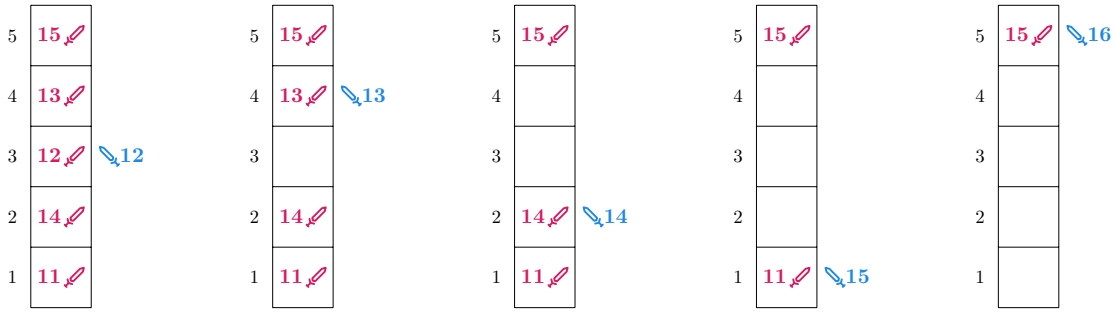
nombre clave: `torre`

Como prueba final de su valentía, Valeria ha sido encomendada con la misión de infiltrarse en la torre de la maldad y poner fin al reinado de la liga del mal.

La torre de la maldad tiene N niveles numerados de 1 a N. En cada nivel hay un enemigo con poder P_i. Para que la misión sea satisfactoria, Valeria debe derrotar a todos los enemigos. El poder inicial de Valeria es V. Valeria solo puede derrotar a un enemigo si su poder V es mayor o igual que el poder P_i del enemigo. Después de derrotar a un enemigo, este es eliminado de la torre y el nivel queda *vacío*. Si en cualquier momento se enfrenta a un enemigo con un poder mayor que el de ella, Valeria perderá la misión. Afortunadamente, gracias a la gema de la experiencia, cada vez que Valeria derrota a un enemigo su poder aumenta en 1.

Al comenzar la misión, Valeria puede infiltrarse en cualquier nivel de la torre y enfrentarse con el enemigo en aquel nivel. Una vez adentro, solo puedo moverse al siguiente nivel no vacío ya sea hacia arriba o hacia abajo. Nuestra heroína está interesada en saber en qué orden visitar los niveles y enfrentarse a los enemigos para que su misión sea exitosa.

La siguiente imagen muestra un ejemplo para una torre con $N = 5$ niveles. El poder inicial de Valeria es 12. Por lo tanto puede decidir inicialmente infiltrarse en el nivel 3 y derrotar al enemigo cuyo poder también es 12. Después de derrotarlo, el poder de Valeria es 13. El siguiente enemigo a enfrentar debe ser el del nivel 2 (abajo) o el del nivel 4 (arriba). El enemigo en el nivel 2 tiene poder 14 por lo tanto Valeria no puede derrotarlo. Por lo tanto está forzada a enfrentarse al enemigo en nivel 4 cuyo poder es 13. El poder de Valeria es ahora 14 y las posibilidades son moverse al nivel 2 (abajo) o al 5 (arriba). En este caso la única alternativa es moverse al nivel 2 y enfrentar al enemigo con poder 14. En este punto, Valeria puede decidir derrotar al enemigo en el nivel 1 o el 5. Para este ejemplo, elige primero derrotar al enemigo en el nivel 1 y luego al enemigo en el nivel 5. La secuencia en la que Valeria visita todos los niveles es la siguiente 3, 4, 2, 1, 5. Notar que si inicialmente Valeria hubiese decidido infiltrarse en el nivel 1, hubiese fallado la misión.



Entrada

La primera línea de la entrada contiene dos enteros N y V ($0 < N \leq 500$ y $0 < V \leq 10^9$) correspondientes respectivamente a la cantidad de niveles en la torre y el poder inicial de Valeria. A continuación siguen N líneas cada una conteniendo un entero mayor que cero. El entero en la i-ésima línea corresponde al poder P_i del enemigo en el nivel i.

Salida

La salida debe contener una línea con N enteros entre 1 y N correspondientes a un posible orden en que Valeria puede visitar los niveles para derrotar a todos los enemigos. En caso de haber más de un posible orden cualquiera de ellos es una respuesta válida. Si no existe un orden en el que sea posible derrotar a todos los enemigos, la salida debe contener una línea con la palabra `FALLIDA`.

Subtareas y puntaje

Subtarea 1 (100 puntos)

Este problema solo contiene una subtarea.

Ejemplos de entrada y salida

Entrada de ejemplo	Salida de ejemplo
5 12 11 14 12 13 15	3 4 2 1 5

Entrada de ejemplo	Salida de ejemplo
5 12 14 14 11 14 14	FALLIDA

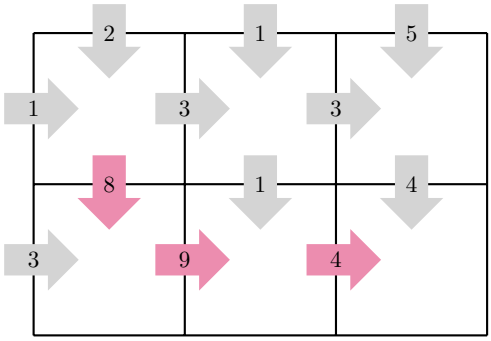
Problema B
El mejor camino

nombre clave: `camino`

Aburrida en su casa, Maiki acaba de inventar un nuevo juego. El juego se juega en una matriz de M filas y N columnas. Las filas de la matriz se enumeran de arriba a abajo entre 1 y M. Las columnas se enumeran de izquierda a derecha entre 1 y N. Identificamos con un par (i, j) a la casilla en la fila i y columna j.

Partiendo de la casilla $(1, 1)$, en cada turno uno puede moverse desde la casilla actual a la casilla inmediatamente abajo o inmediatamente a la derecha. Cada vez que uno se mueve a una nueva casilla se asigna un puntaje. El puntaje es distinto para cada casilla y depende de si uno se mueve a la casilla viniendo desde la izquierda o desde arriba. El objetivo del juego es llegar a la casilla (M, N) sumando la mayor cantidad de puntaje.

La siguiente figura muestra un ejemplo para $M = 2$ y $N = 3$. Las flechas representan los puntajes asociados a cada casilla. Notar que las casillas del borde superior tienen un puntaje asociado a llegar desde arriba. Estos puntajes son irrelevantes pues uno nunca puede moverse a estas casillas desde arriba. Similarmente, las casillas del borde izquierdo tienen un puntaje asociado a llegar desde la izquierda a pesar de ser irrelevante.



Las flechas marcadas de color rojo representan un posible camino. Partiendo desde la casilla (1, 1) primero se mueve hacia la casilla (2, 1) obteniendo 8 puntos. Posteriormente se mueve dos veces a la derecha obteniendo respectivamente en cada movimiento 9 y 4 puntos. El puntaje total es entonces $8 + 9 + 4 = 21$. Notar que cualquier otro camino obtiene un menor puntaje y por lo tanto 21 es el puntaje máximo que es posible obtener.

Dada una matriz con los puntajes asociados a cada casilla, tu tarea es encontrar el puntaje máximo que es posible obtener en el juego.

Entrada

La primera línea de la entrada contiene dos enteros M y N ($1 \leq N \leq 500$ y $1 \leq M \leq 500$) correspondientes respectivamente a la cantidad de filas y columnas en la matriz.

A continuación siguen M líneas cada una conteniendo N enteros mayores o iguales que cero y menores que 10^6. El j-ésimo entero de la línea i-ésima corresponde al puntaje asociado a moverse **desde arriba** a la casilla (i, j).

Finalmente, vienen M líneas más cada una conteniendo N enteros mayores o iguales que cero y menores que 10^6. El j-ésimo entero de la línea i-ésima corresponde al puntaje asociado a moverse **desde la izquierda** a la casilla (i, j).

Salida

Subtareas y puntaje

Subtarea 1 (10 puntos)

Se probarán varios casos en que $M = 1$ (ver primer caso de ejemplo).

Subtarea 2 (10 puntos)

Se probarán varios casos en que el puntaje asociado a moverse desde arriba es igual al puntaje de moverse desde la izquierda y además este es igual para todas las celdas (ver segundo caso de ejemplo).

Subtarea 3 (30 puntos)

Se probarán varios casos en que para cada celda el puntaje de moverse desde arriba es igual al puntaje de moverse desde la izquierda. Este valor puede ser distinto para celdas distintas (ver tercer caso de ejemplo).

Subtarea 4 (50 puntos)

Se probarán varios casos sin restricciones adicionales (ver cuarto caso de ejemplo).

Ejemplos de entrada y salida

Entrada de ejemplo	Salida de ejemplo
1 6 0 0 0 0 0 0 0 2 3 1 4 1	11

Entrada de ejemplo	Salida de ejemplo
3 5 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2	12

Entrada de ejemplo	Salida de ejemplo
4 2 1 2 4 2 3 4 1 3 1 2 4 2 3 4 1 3	14

Entrada de ejemplo	Salida de ejemplo
2 3 2 1 5 8 1 4 1 3 3 3 9 4	21

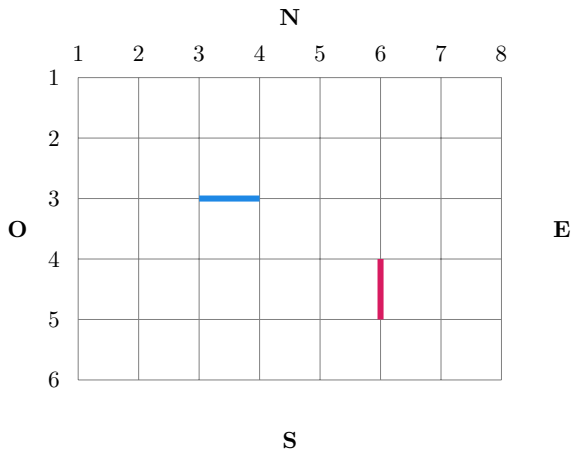
Problema C

Por una ciudad más caminable

nombre clave: `acceso`

Décadas de malas políticas de desarrollo han condenado a los habitantes de Grilland a depender del automóvil para poder desplazarse por la ciudad. Dado el indiscutible impacto negativo que esto tiene en el medio ambiente, los habitantes de Grilland han decidido poner un alto a la dependencia al automóvil. Los expertos saben que la mejor forma de lograrlo es hacer que la ciudad sea más caminable. Como primer paso, las autoridades están interesadas en saber qué puntos de la ciudad son actualmente los más caminables. Un punto es más caminable mientras más puntos de interés sea posible *acceder* caminando desde él.

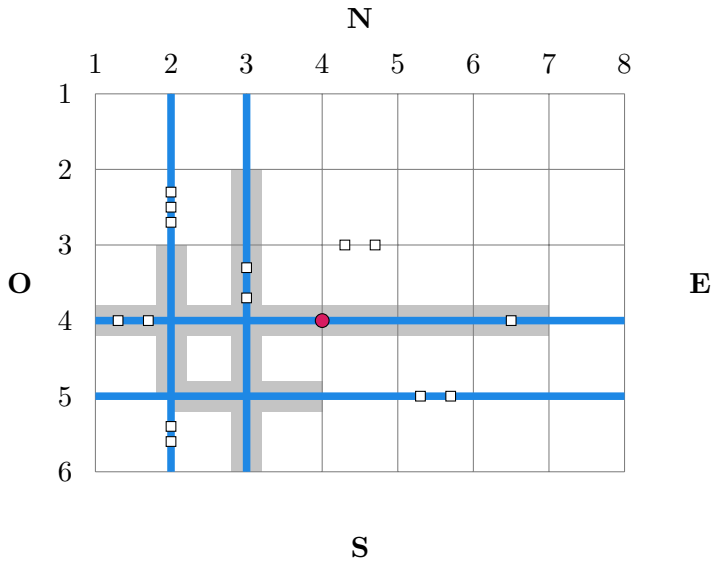
La geometría de Grilland es muy especial. Sus calles forman un grilla perfecta con M calles en dirección horizontal y N calles en dirección vertical. Las calles en dirección horizontal son numeradas de norte a sur entre 1 y M. Las calles en dirección vertical son numeradas de oeste a este entre 1 y N. Dada una calle i en dirección horizontal y una calle j en dirección vertical, identificamos con el par (i, j) a la intersección entre ambas calles. Denominamos *cuadra* al segmento de una calle contenido entre dos intersecciones consecutivas. La siguiente imagen muestra un ejemplo de la geometría de Grilland para $M = 6$ y $N = 8$, donde se ha marcado con **azul** la cuadra contenida entre las intersecciones (3, 3) y (3, 4), y con **rojo** la cuadra entre las intersecciones (4, 6) y (5, 6).



Solo algunas de las calles en Grilland tienen vereda. Si una calle tiene vereda, esta se extiende por todas sus cuadras. Una persona puede caminar libremente solo por las calles que tienen vereda.

Algunas de las cuadras contienen *puntos de interés*. Dada una intersección inicial estamos interesados en calcular el *nivel de acceso* para esa intersección. El nivel de acceso corresponde a la cantidad de puntos de interés que pueden ser accedidos desde la intersección inicial caminando no más de F cuadras.

La siguiente imagen muestra un ejemplo donde las calles con vereda han sido marcadas de color **azul**, los puntos de interés están representados con pequeños cuadrados blancos y la intersección inicial con un círculo **rojo**. El área marcada con gris muestra todas las cuadras que es posible acceder caminando a lo más $F = 3$ cuadras. Hay 5 puntos de interés en esta área y por lo tanto el nivel de acceso para la intersección inicial es 5. Notar que no importa donde están ubicados exactamente los puntos de interés sino solo la cuadra en la que están ubicados.



Dadas las calles que tienen vereda, las cuadras en las que están ubicados los puntos de interés y la intersección inicial, tu tarea es calcular el nivel de acceso para la intersección inicial.

Entrada

La primera línea de la entrada contiene dos enteros M y N ($0 < M \leq 100$ y $0 < N \leq 100$) correspondientes respectivamente a la cantidad de calles en dirección horizontal y la cantidad de calles en dirección vertical. La segunda línea contiene M enteros describiendo si las calles en dirección horizontal contienen o no una vereda. El entero i-ésimo será un 1 si la calle i tiene vereda o un 0 en caso contrario. Similarmente, la tercera línea contiene N enteros describiendo si las calles en dirección vertical contienen veredas.

La siguiente línea contiene 3 enteros U, V y F ($1 \leq U \leq M$, $1 \leq V \leq N$ y $1 \leq F \leq 500$). El par (U, V) describe la intersección inicial. El entero F describe la cantidad máxima de cuadras. La intersección (U, V) no necesariamente estará contenida en una calle con vereda.

A continuación sigue una línea con un entero E ($0 < E \leq 1000$) correspondiente a la cantidad de puntos de interés. Las siguientes E líneas describen cada una la cuadra en que se encuentra ubicado un punto de interés. Cada línea contiene tres enteros a, b, c. El par (a, b) representa la intersección de inicio de la cuadra. El entero c representa la orientación de la cuadra. Si c es 0, la cuadra está en orientada en dirección horizontal, es decir la cuadra está contenida entre las intersecciones (a, b) y

$(a, b+1)$. Si c es 1, la cuadra está en dirección vertical, es decir, la cuadra está contendida entre las intersecciones (a, b) y $(a+1, b)$. Se garantiza que la cuadra siempre estará contenida en la grilla.

Salida

La salida debe contener un único entero mayor o igual que cero indicando el nivel de acceso para la intersección inicial, es decir, la cantidad de puntos de interés a una distancia menor o igual que F cuadras.

Subtareas y puntaje

Subtarea 1 (30 puntos)

Se probarán varios casos donde todas las calles tienen vereda, es decir, la segunda y tercera línea de la entrada contendrán solo unos.

Subtarea 2 (70 puntos)

Se probarán varios casos sin restricciones adicionales.

Ejemplos de entrada y salida

Entrada de ejemplo	Salida de ejemplo
6 8 0 0 0 1 1 0 0 1 1 0 0 0 0 0 4 4 3 14 4 1 0 4 1 0 2 2 1 2 2 1 2 2 1 5 2 1 5 2 1 3 3 1 3 3 1 3 4 0 3 4 0 5 5 0 5 5 0 4 6 0	5

Problema D
Ocilandia

nombre clave: `ocilandia`

Pedrito acaba de cumplir su sueño de visitar Ocilandia, el parque de diversiones más famoso y popular del mundo. Pedrito estará en el parque solo por un día y quiere usar su tiempo de la forma más eficiente posible.

Ocilandia no es un parque de diversiones común y corriente. El parque tiene N atracciones y cada una entrega una experiencia individual adaptada a las necesidades de cada visitante. Por esta razón, el tiempo que tarda visitar una atracción varía de persona a persona.

Cada una de las atracciones tiene una fila de personas esperando poder visitarla. La i-ésima atracción tiene una fila de largo M_i. Adicionalmente, para cada atracción, sabemos el tiempo que tardará cada persona en la fila en visitar la atracción. El *tiempo de espera* de una atracción es igual a la suma de los tiempos que tardarán cada una de las personas en la fila en visitar la atracción.

Pedrito quiere usar bien su tiempo y está interesado en saber el menor tiempo de espera entre todas las atracciones. ¿Podrías ayudarlo?

Entrada

La primera línea de la entrada contiene un entero N ($0 < N \leq 1000$) correspondiente a la cantidad de atracciones.

La siguiente línea contiene N enteros indicando el largo de la fila para cada atracción. El entero i-ésimo corresponde al largo M_i ($0 < M_i \leq 1000$) de la fila en la i-ésima atracción.

Posteriormente, vienen N líneas describiendo las filas para cada una de las atracciones. La i-ésima línea contiene M_i enteros cada uno indicando el tiempo que tardará cada persona en visitar la atracción.

Salida

La salida debe contener una línea con un entero indicando el tiempo mínimo de espera entre todas las atracciones.

Subtareas y puntaje

Este problema no contiene subtareas. Se entregará puntaje proporcional a la cantidad de casos de prueba correctos, siendo 100 el puntaje máximo.

Ejemplos de entrada y salida

Entrada de ejemplo	Salida de ejemplo
3 1 3 2 10 1 3 4 5 9	8