



## Problema A

### Muralla de defensa

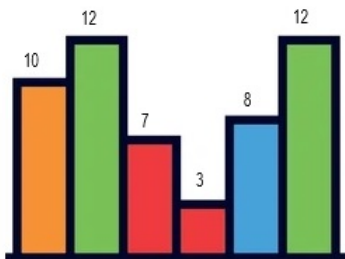
En un pequeño rincón de Galicia, en una aldea costera llamada Porto da Lúa, los habitantes enfrentan cada invierno la amenaza de grandes tormentas marítimas que erosionan la costa y ponen en peligro sus hogares. Para protegerse, han decidido construir una gran muralla de defensa junto al puerto. Esta muralla es su escudo contra las olas y está formada por **bloques de piedra de 1 metro de ancho** alineados uno al lado del otro.

Sin embargo, debido a la naturaleza rocosa de la región, no todos los bloques tienen la misma altura. Algunos son enormes y otros más pequeños, dependiendo del tipo de roca disponible. Eso significa que hay zonas más protegidas que otras. Por ese motivo en la aldea quieren calcular el área del tramo rectangular más grande que se ha construido utilizando bloques consecutivos de la muralla.

Por restricciones de diseño, la altura del tramo rectangular debe ser uniforme, lo que significa que estará determinada por el bloque más bajo en ese tramo.

Tu misión como ingeniero del proyecto es ayudar a los lugareños a **calcular el área máxima de un tramo rectangular construido con los bloques disponibles**. Esto les permitirá conocer la zona de la muralla que ofrece máxima protección.

Ejemplo: Se dispone de una muralla de 6 bloques de piedra de 1 metro de ancho y las alturas que se muestran en la siguiente figura (10, 12, 7, 3, 8, 12).



Para este ejemplo, el tramo rectangular más grande se forma utilizando los 3 primeros bloques con una altura mínima de 7 y un ancho de 3, es decir, un área de  $7 \times 3 = 21$ .

#### Entrada

La entrada consta de una serie de casos de prueba. Cada caso se compone de dos líneas.



La primera línea contiene un único entero  $n$ , que representa el número de bloques en la muralla, siendo  $1 \leq n \leq 100.000$ .

La segunda línea representa la secuencia de alturas de cada bloque, separadas por espacios. Cada altura será un valor entero entre 1 y 10.000.

El final de la entrada se reconoce con  $n = 0$ .

## Salida

Para cada caso de prueba, la salida es una línea con un único número entero que representa el área del tramo rectangular más grande de la muralla.

## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
6	21
10 12 7 3 8 12	420
7	12
100 80 60 70 60 75 85	910
8	796
1 3 6 5 4 2 1 2	
10	
25 530 120 910 330 50 50 102 55 75	
15	
56 67 45 2 46 57 68 79 80 90 189 200 398 560 72	
0	



## Problema B

### Planilandia

En un tranquilo pueblo llamado Planilandia, vivía una joven llamada Sofía, conocida por su pasión por aprender cosas nuevas. Cada semana, en el centro cultural del pueblo, se ofrecían cursos de todo tipo: desde arte y cocina hasta programación y astronomía. Sin embargo, había una pequeña complicación: los horarios de los cursos a menudo se solapaban, lo que hacía imposible asistir a todos.

Un día, Sofía se encontró frente al tablón de anuncios del centro cultural. Había una lista de cursos para el sábado, y todos parecían emocionantes:

1. **Pintura al óleo:** 09:00 - 11:00
2. **Yoga para principiantes:** 10:00 - 14:10
3. **Introducción a la astronomía:** 11:30 - 13:00
4. **Taller de panadería:** 11:00 - 14:00
5. **Robótica básica:** 13:30 - 15:30
6. **Fotografía con el móvil:** 13:10 - 15:40

Sofía miró los horarios con detenimiento. Se dio cuenta de que, si quería aprovechar el día al máximo, debía elegir cuidadosamente. Su objetivo era asistir al mayor número posible de cursos sin que sus horarios se solaparan.

Sofía contactó con los organizadores del Ada Byron para que los participantes implementasen un **algoritmo óptimo y eficiente** que le ayudase a planificar su día y le indicase a cuántos cursos como máximo podría asistir.

Por lo tanto, el planificador le aconsejó a Sofía que podía asistir a 3 cursos, por ejemplo:

1. Pintura al óleo (09:00 - 11:00)
2. Introducción a la astronomía (11:30 - 13:00)
3. Robótica básica (13:30 - 15:30)

El sábado, Sofía asistió a los tres cursos y disfrutó de cada minuto. Aprendió a mezclar colores para crear paisajes, entendió las maravillas del cosmos y dominó la robótica básica. Aunque no pudo asistir a todos los cursos, estaba feliz porque había aprovechado el día al máximo.

Desde entonces, Sofía compartió su algoritmo con otros vecinos de Planilandia, quienes comenzaron a aplicar la misma técnica para organizar su tiempo.



## Entrada

La entrada consta de varias líneas. La primera es el número de cursos organizados  $m$ , siendo  $1 \leq m \leq 500.000$ .

Las siguientes líneas detallan el nombre del curso (sin espacios en blanco), *hora de inicio* y *hora de fin* (las horas serán un entero que representan los milisegundos desde una fecha determinada, se garantiza que *hora de inicio* < *hora de fin*, y que no habrá dos cursos que finalicen a la misma hora). Estos elementos de información del curso (nombre, hora de inicio y hora de fin) están separados por un espacio en blanco.

## Salida

Número máximo de cursos a los que se puede asistir.

## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
10	3
Act_1 248974 489861	
Act_2 82560 486634	
Act_3 140280 758615	
Act_4 482461 730495	
Act_5 178363 802948	
Act_6 690642 754525	
Act_7 77710 340315	
Act_8 664232 887712	
Act_9 510747 920638	
Act_10 371699 549522	



## Problema C

### Dakar

El Rally Dakar es una de las competiciones más emocionantes y desafiantes del mundo. Los pilotos de los vehículos atraviesan terrenos hostiles, desiertos interminables, montañas y planicies áridas, enfrentándose no solo a la distancia, sino también al clima extremo y a sus propios límites físicos y mentales. De ahí que, para muchos participantes, el objetivo no es ganar, sino simplemente llegar al final. Sin embargo, ese no es el caso de Rápido Furioso, un piloto veterano de motos cuya única meta es acabar el Rally Dakar por delante de todos sus competidores.

Antes de cada etapa la organización del Rally Dakar le proporciona a cada piloto información sobre la salida y la meta de esa etapa. Además, les indica varios *waypoints*, puntos por los que tienen que pasar obligatoriamente. Tanto por los *waypoints* como por la salida y la meta, sólo pueden pasar una vez. Gracias a su experiencia, Rápido Furioso es capaz de estimar cuánto tiempo necesita para moverse de un punto del desierto a otro. Con esa información necesita planificar la ruta que debe hacer para llegar a la meta en el menor tiempo posible. Lo debe hacer rápido, ya que en su diccionario no existe la palabra lento. Como sus habilidades algorítmicas no son las mismas que encima de la moto, contacta contigo para que le eches una mano y puedas compartir con él la gloria de ganar el Rally Dakar.

Rápido Furioso te informa de la salida, la meta y los *waypoints*. Además, te indica los distintos puntos por los que puede pasar la etapa y el tiempo que tardaría en pasar por cada uno de los caminos que hay entre dos puntos. Es importante destacar que, en el desierto, el tiempo necesario para ir de un punto  $A$  a un punto  $B$  no siempre es el mismo que en la dirección opuesta. ¡Subir dunas presenta desafíos muy distintos al de descenderlas! Con esa información, debes ser capaz de calcular cuál es el camino más rápido para llegar a la meta pasando por los *waypoints*. Realmente, solo necesitarás indicarle cuánto tiempo tardará, con su veteranía, Rápido Furioso sabrá decidir cuál es la ruta correcta.

#### Entrada

La primera línea de entrada contiene dos enteros  $P$  ( $1 \leq P \leq 1.000$ ) y  $C$  ( $1 \leq C \leq 4.000.000$ ), representando el número de puntos y el número de caminos que hay entre esos puntos, respectivamente.

Las siguientes  $C$  líneas representan los caminos que unen dos puntos en una dirección. Cada línea contiene el punto origen  $O$ , el punto destino  $D$  y los segundos  $T$  ( $1 \leq T \leq 100$ ) que se tarda en llegar del origen al destino.

La siguiente línea contiene tres enteros que representan el punto de salida  $S$ , el punto de la meta  $M$  y el número de *waypoints*  $W$ .

Las siguientes  $W$  líneas contienen un entero cada una que representa los *waypoints*. La primera línea contiene el primer *waypoint*, las segunda corresponde al segundo *waypoint*, y así, sucesivamente.

#### Salida

La salida será un entero que indique los segundos que se tarda en completar la etapa por la ruta más rápida, respetando las reglas del Rally Dakar.



## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
5 6 1 2 2 2 3 1 2 5 2 3 4 2 3 5 6 4 5 2 1 5 1 3	7

Entrada de ejemplo	Salida de ejemplo
5 18 1 2 88 1 3 56 1 4 71 1 5 64 2 1 42 2 3 59 2 4 23 2 5 58 3 1 56 3 4 1 3 5 12 3 5 45 4 1 73 4 3 2 4 5 14 5 1 70 5 2 11 5 3 95 1 5 1 3	68



## Problema D

### La Torre Oscura

Saurito, el hijo de Sauron, hereda la imponente Torre Oscura (Barad-dûr) tras la trágica muerte de su padre, injustamente asesinado. La Torre Oscura representaba el mayor proyecto y sueño de Sauron: una prisión de lujo equipada con vistas incomparables al Monte del Destino. Hoy en día, gracias a la visión y legado de su progenitor, Saurito se ha consolidado como un hombre de éxito. La Torre, ahora completamente operativa, alberga a una peculiar clientela compuesta de hobbits y enanos, quienes, según el estilo distintivo de Saurito, disfrutan de rigurosas sesiones diarias de tortura.

Una mañana, al llegar a la Torre, Saurito descubre con sorpresa que uno de sus prisioneros, Frodo Bolsón, ha logrado escapar de su celda. Para su asombro, Frodo se encuentra en la cima de la Torre, dedicándose a la construcción de una cometa con la evidente intención de aprovechar los fuertes vientos que atraviesan el Valle de Gorgoroth y así consumir su huida.

Para evitar la fuga y preservar su reputación como el mejor carcelero de Mordor, Saurito se ve obligado a ascender hasta el último piso de la Torre. Allí, deberá enfrentarse a Frodo y frustrar su intento de escape antes de que logre surcar los cielos con su improvisada cometa.

La Torre Oscura tiene  $h$  pisos numerados de 1 a  $h$ . Para cada piso  $i$  ( $1 \leq i < h$ ) existe una escalera que puede ser usada para subir al piso  $i + 1$ . Adicionalmente, la torre posee  $n$  ascensores. Debido a la humedad, estos están en muy malas condiciones y por lo tanto el ascensor  $j$  solo sirve para subir desde el piso  $a_j$  hasta el  $b_j$ , sin parar entre medio. Además, cabe mencionar que estos solo son ascensores y no descendores, por lo que no es posible utilizarlos para bajar.

Después de tantos años vigilando la torre, la salud de Saurito ya no es la que era. La humedad y el moho han perjudicado su salud por lo que quiere minimizar la cantidad de escaleras que necesita subir y evitar a toda costa bajarlas por el reuma. Específicamente, usando los ascensores y las escaleras para subir, Saurito quiere encontrar una ruta desde el piso 1 al  $h$  que minimice la cantidad de escaleras que debe subir. Normalmente lo calcularía mentalmente, pero está demasiado ocupado planificando cómo impedir que Frodo escape, por lo que te pidió ayuda con esta tarea.

¿Podrás ayudar a Saurito a subir la torre para evitar que Frodo escape y así mantener su impoluta reputación en Mordor?

#### Entrada

La primera línea de la entrada contiene dos enteros  $n$  y  $h$  ( $0 \leq n \leq 10^5$ ,  $1 \leq h \leq 10^6$ ) correspondientes a la cantidad de ascensores y la cantidad de pisos en la torre respectivamente.

Luego, siguen  $n$  líneas describiendo cada uno de los ascensores.

La  $j$ -ésima línea contiene dos enteros  $a_j, b_j$  ( $1 \leq a_j < b_j \leq h$ ), que corresponden respectivamente al piso de inicio y al de llegada del ascensor  $j$ .



## Salida

La salida debe tener un único entero, correspondiente a la cantidad mínima de escaleras con la que es posible ir desde el piso 1 al  $h$ .

## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
1 5	2
1 3	

Entrada de ejemplo	Salida de ejemplo
2 10	1
1 4	
2 10	



## Problema H

### Rompecabezas 3D

Estas navidades, con la intención de que se olviden de sus teléfonos móviles durante el mayor tiempo posible, Luis ha regalado a sus dos hijos, Antía y José, un antiguo rompecabezas 3D que se ha transmitido de padres a hijos durante varias generaciones. Se trata de una caja de madera de castaño que alberga en su parte superior un rompecabezas tallado a mano. Esta caja tiene la peculiaridad de que esconde un compartimento secreto que solo se abrirá una vez que se resuelva el rompecabezas. Para ponerlo más interesante, Luis ha escondido dentro del rompecabezas dos entradas para ver la próxima película de la saga Star Wars, de la que sus dos hijos son muy fanes.

Junto a las instrucciones que acompañan al rompecabezas se encuentra una lámina en la que hay dibujada una cuadrícula de  $n$  filas y  $m$  columnas. Las filas están numeradas de arriba hacia abajo desde 1 a  $n$ , mientras que las columnas están numeradas de izquierda a derecha desde 1 a  $m$ . La casilla correspondiente a la fila  $i$  y columna  $j$  se identifica con el par  $(i, j)$ .

Algunas de las casillas dibujadas en lámina se han decorado con el escudo de la familia de Luis (la cabeza de un león) formando una *figura conexa*, es decir, todas las casillas decoradas son *adyacentes* a al menos otra casilla decorada. Dos casillas decoradas se consideran adyacentes si comparten un borde vertical u horizontal. La siguiente imagen muestra un ejemplo para una cuadrícula de  $3 \times 5$  con casillas decoradas en las posiciones  $(3, 3)$ ,  $(3, 4)$  y  $(3, 5)$ .




	1	2	3	4	5
1					
2					
3					




En la parte superior de la caja de madera se encuentra el tablero del rompecabezas formado también por una cuadrícula de  $n \times m$  casillas. Sobre el tablero, hay fichas que pueden moverse entre sus casillas. La cantidad de fichas es la misma que la de casillas decoradas que aparecen en la lámina que se encontraba junto a las instrucciones del rompecabezas, pero están dispuestas de forma distintas. La siguiente imagen muestra un ejemplo del tablero con fichas en las posiciones  $(1, 2)$ ,  $(1, 3)$  y  $(2, 3)$ .




	1	2	3	4	5
1					
2					
3					




De acuerdo a las indicaciones de su padre, para acceder al compartimento secreto y así poder ir al cine gratis, Antía y José tendrán que resolver el rompecabezas. Para ello, deben mover las fichas del tablero hasta que coincidan con las casillas de la lámina. A priori esto parece una tarea fácil, pero para poder acceder al compartimento secreto, las fichas deben moverse siguiendo una regla especial. Así, el único movimiento permitido es coger una ficha y colocarla adyacente a otra ficha de manera que en todo momento se forme una figura conexas.

Dada la cuadrícula dibujada en la lámina y el tablero inicial del rompecabezas en las imágenes anteriores, la siguiente imagen muestra una posible secuencia de movimientos que pueden usarse para que las fichas en el tablero coincidan con las de la lámina.

	1	2	3	4	5
1					
2					
3					

	1	2	3	4	5
1					
2					
3					

	1	2	3	4	5
1					
2					
3					

	1	2	3	4	5
1					
2					
3					

En el primer movimiento la ficha en la posición (1,2) se mueve a la posición (3,3). Posteriormente, la ficha en la posición (1,3) se mueve a la posición (3,4). Finalmente, la ficha en la posición (2,3) se mueve a la posición (3,5).

Antía y José están ansiosos por ir al cine y quieren resolver el rompecabezas lo más rápido posible, pero están teniendo problemas. En particular, les gustaría saber la mínima cantidad de movimientos en que es posible hacer coincidir las fichas del rompecabezas con la figura de la lámina y así acceder a su premio. ¿Podrías ayudarlos?

## Entrada

La primera línea de la entrada contiene dos enteros  $n$  y  $m$  ( $n \times m \leq 2 \times 10^5$ ) correspondientes a las dimensiones del tablero del rompecabezas y a la cuadrícula de la lámina. Luego vienen  $2 \times n$  líneas describiendo el tablero y la cuadrícula de la lámina.

Las primeras  $n$  líneas describen el tablero. Cada línea contiene  $m$  enteros iguales a 0 o 1. El  $j$ -ésimo entero en la línea  $i$  describe la casilla  $(i, j)$  del tablero. Un 1 indica que la casilla contiene una ficha y un 0 indica que está vacía.



Las últimas  $n$  líneas describen las casillas tal y como aparecen en la cuadrícula de la lámina usando el mismo formato.

Se garantiza que la entrada cumple las siguientes condiciones:

- La cantidad de unos en el tablero y en la lámina es la misma.
- La cantidad de unos en el tablero y en la lámina es mayor o igual que 2.
- Los unos en el tablero y en la lámina forman ambos una figura conexa.

## Salida

La salida debe contener un entero igual al número mínimo de movimientos requeridos para mover las casillas de forma que coincidan con casillas talladas en la lámina.

## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
1 4 1 1 0 0 0 0 1 1	2

Entrada de ejemplo	Salida de ejemplo
3 5 0 1 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1	3



## Problema F

### Festival de Streamers

En un gran evento de streamers, varios creadores de contenido han montado *stands* donde ofrecen experiencias únicas para sus seguidores. Jugar una partida con Ibai, aprender trucos de magia con ElRubius, o escuchar cómo El Xokas te explica por qué siempre tiene la razón, son sólo algunas de las apasionantes experiencias que puedes disfrutar en este festival (¿sigues prefiriendo estar en el AdaByron? ¡puesto a que sí!). En el festival se han organizado  $N$  actividades como fotos exclusivas, partidas en directo, o incluso charlas personalizadas. Cada actividad dura  $t[i]$  minutos y te da una recompensa de popularidad  $p[i]$  puntos. El tiempo que puedes dedicar al festival es limitado: solo tienes  $T$  minutos en total.

Tu objetivo es maximizar la popularidad obtenida sin exceder el tiempo del que dispones. Las actividades las puedes realizar por completo, o solo una fracción del tiempo que requieren (en cuyo caso, obviamente obtendrás la fracción correspondiente de puntos de popularidad). Por ejemplo, aguantar la explicación del Xokas te puede dar 10 puntos, pero requiere un total de 100 unidades de tiempo. Si solo estás dispuesto a aguantarlo 10 unidades de tiempo, obtendrás 1 punto.

#### Entrada

La entrada comienza con un entero  $C$  ( $1 \leq C \leq 50$ ), el número de casos de prueba.

Para cada caso de prueba, la primera línea contiene dos enteros  $N$  ( $1 \leq N \leq 10^5$ ) y  $T$  ( $1 \leq T \leq 10^6$ ), que representan el número de actividades y tu tiempo total. La segunda línea contiene  $N$  enteros  $t[i]$  ( $1 \leq t[i] \leq 10^6$ ), el tiempo que requiere cada actividad. La tercera línea contiene  $N$  enteros  $p[i]$  ( $1 \leq p[i] \leq 10^6$ ), la popularidad que obtienes al realizar cada actividad completa.

#### Salida

Para cada caso de prueba, imprime en una línea un único entero, la máxima popularidad que puedes ganar sin exceder el presupuesto. El valor absoluto de la diferencia entre la respuesta de tu programa y el valor óptimo debe ser como máximo  $10^{-3}$ . Para garantizar esto, muestra tu respuesta con al menos cuatro dígitos después del punto decimal (de lo contrario, tu respuesta, aunque se haya calculado correctamente, puede resultar incorrecta debido a problemas de redondeo).



## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
3	390.0000
4 125	238.0488
16 27 38 41	20.2083
30 75 105 180	
5 67	
11 22 31 41 54	
25 60 90 120 200	
3 5	
12 24 36	
47 97 140	



## Problema G

### Monedas

En la cueva del temido Dragón Dormido un montón de monedas mágicas esparcidas por el suelo. Pero, ¡cuidado! ¡algunas están malditas!. Cuando alguien se adentra en la cueva, el dragón abre un ojo y maldice todas las monedas que ve. Es más, maldice todas las monedas que tengan el mismo valor de alguna moneda que ve. Lo bueno es que las monedas malditas se vuelven brillantes y son fáciles de distinguir. Lo malo, que cuando se intentan coger las no malditas es posible que se despierte completamente y la vida de esa persona que intenta coger las monedas se pone en peligro.

El Consejo de Magos te ha encargado una tarea importante: sumar el valor total del tesoro, ignorando las monedas malditas, para que aquellas personas aventureras que quieran adentrarse en la cueva sepan cuánto oro pueden llevarse y si les compensa el riesgo.

#### Entrada

La primera línea de la entrada contiene un entero positivo  $T$  ( $1 \leq T \leq 10$ ) indicando el número de casos de test. Cada uno de los casos de test se describen a continuación de la siguiente manera:

- Un entero  $N$  ( $1 \leq N \leq 1000$ ): el número total de monedas en la cueva.
- Una línea con  $N$  enteros separados por espacios: los valores de cada una de las monedas la cueva, ordenados según se encuentran esparcidas en la cueva. El valor de las monedas está entre 1 y 1000.
- Un entero  $M$  ( $1 \leq M \leq 100$ ): el número de valores diferentes las monedas malditas.
- Una línea con  $M$  enteros: los valores de las monedas malditas, en orden creciente. De nuevo, estos valores están entre 1 y 1000.

#### Salida

La salida estará formada por una línea por cada caso de test, en el mismo orden de la entrada, en donde se debe indicar la suma total de las monedas buenas (no malditas) para ese caso de test.



## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
3	18
8	0
5 2 7 3 2 9 1 5	19
3	
2 3 9	
4	
4 4 4 4	
1	
4	
6	
10 5 7 5 9 2	
2	
5 9	



## Problema H

### El príncipe Shrek

A la intrépida guerrera Fiona se le ha encomendado la difícil misión de rescatar al indefenso príncipe Shrek de la torre infernal, donde la temida reina roja lo tiene cautivo.

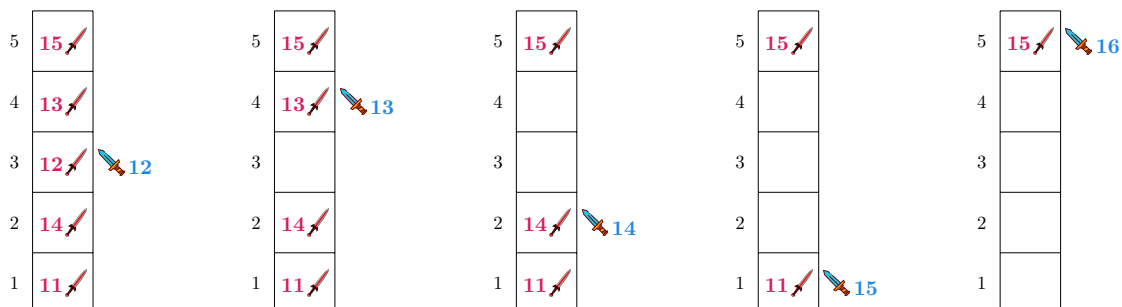
La torre infernal tiene  $N$  niveles numerados de 1 a  $N$ . En cada nivel hay un guerrero de la reina roja que dará su vida para que Fiona fracase en su misión. Dicho guerrero, después de realizar un cursillo de Tai Chi del SEPE, posee un nivel de energía  $P_i$ . Para que la misión de Fiona sea satisfactoria, esta debe derrotar a los diferentes enemigos que se encontrará en todos los niveles hasta llegar a lo más alto de la torre donde se encuentra el príncipe Shrek. Antes de llegar a la torre, Fiona ha decidido parar en un furancho a tomar un cocido bien completo y así cargar su nivel de energía en  $V$ .

Cada vez que Fiona se enfrenta a un guerrero de la reina roja solo puede derrotarlo si su poder  $V$  es mayor o igual que el poder  $P_i$  de dicho guerrero. Cada vez que Fiona sale victoriosa de un enfrentamiento con un guerrero, este es eliminado de la torre y el nivel queda *vacío*. Si en cualquier momento el enfrentamiento es con un guerrero con un poder mayor que el de ella, Fiona perderá el combate y por lo tanto será encarcelada en las mazmorras de la torre infernal. Afortunadamente, gracias a la filloa mágica que tomó de postre antes de empezar su aventura, cada vez que Fiona derrota a un guerrero su nivel de energía aumenta en 1.

Además, gracias a las catiúscas mágicas que le regaló su madre al iniciar la misión, Fiona puede comenzar en cualquier nivel de la torre y enfrentarse con el guerrero que custodia dicho nivel. Una vez que Fiona ha seleccionado un nivel, solo puede moverse al siguiente nivel no vacío ya sea hacia arriba o hacia abajo, pero siempre que derrote al aterrador guerrero que lo defiende. Nuestra heroína está interesada en saber en qué orden visitar los niveles y enfrentarse a los guerreros para que su misión sea exitosa.

La siguiente imagen muestra un ejemplo para una torre infernal con  $N = 5$  niveles y un poder inicial para Fiona de 12. Con esas condiciones iniciales, Fiona puede decidir comenzar su misión infiltrándose en el nivel 3 y derrotar al guerrero cuyo poder también es 12. Después de derrotarlo, el poder de Fiona pasa a ser de 13. El siguiente guerrero a enfrentar debe ser el del nivel 2 (abajo) o el del nivel 4 (arriba). El guerrero del nivel 2 tiene poder 14 y por lo tanto Fiona no podrá derrotarlo. Así, está forzada a enfrentarse al guerrero del nivel 4 cuyo poder es 13. Tras vencerlo, el poder de Fiona es ahora de 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 guerrero con poder 14. En este punto, Fiona puede decidir derrotar al guerrero del nivel 1 o al guerrero del nivel 5. Para este ejemplo, elige primero derrotar al guerrero del nivel 1 y luego al guerrero del nivel 5. La secuencia en la que Fiona visita todos los niveles es la siguiente 3, 4, 2, 1, 5. Nota: si inicialmente Fiona 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 infernal y el poder inicial de Fiona. La siguiente línea contiene  $N$  enteros. El  $i$ -ésimo entero corresponde al poder  $P_i$  ( $0 < P_i \leq 10^9$ ) del guerrero del nivel  $i$ .

### Salida

La salida debe contener una línea con  $N$  enteros entre 1 y  $N$  correspondientes a un posible orden en que Fiona puede visitar los niveles para derrotar a todos los guerreros. En caso de haber más de un posible orden solo será válido aquel que priorice la conquista de los niveles más bajos en primer lugar. Si no existe un orden en el que sea posible derrotar a todos los guerreros, la salida debe contener una línea con la palabra FALLIDA.

### 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 I

### Cimientos

A Gaellicus Maximus le han encargado construir una nueva torre a semejanza de la de Hércules. En un principio todo apuntaba a ser una construcción rutinaria más, hasta que Gaellicus vio el plano del terreno y se dio cuenta de que este era totalmente irregular.

El plano describe el terreno como una matriz de enteros de tamaño  $H \times W$  ( $H$  filas y  $W$  columnas). Cada elemento en la matriz representa la altura del terreno en esa posición. La nueva torre debe construirse sobre una submatriz de tamaño  $M \times N$ . Antes de construirla, Gaellicus tiene que nivelar la submatriz sobre la cual se construirá. El coste de nivelación de una submatriz es igual a la diferencia entre las alturas de la celda más alta y la más baja en la submatriz.

A continuación se muestra un ejemplo para un terreno de tamaño  $4 \times 5$  donde se ha marcado una submatriz de tamaño  $3 \times 2$ . Para esta submatriz, la celda más baja tiene altura 8 y la más alta tiene altura 80. Por lo tanto, el coste de nivelación de esta submatriz es  $80 - 8 = 72$ .

20	100	50	3	22
7	40	8	100	33
60	80	33	9	55
33	56	44	21	88

Gaellicus desea encontrar una submatriz de forma que el coste de nivelación sea mínimo. En el ejemplo anterior no hay ninguna otra submatriz de tamaño  $3 \times 2$  con un coste de nivelación menor y por lo tanto el coste mínimo para este tamaño de submatriz es 72.

### Entrada

La primera línea de la entrada contiene un entero positivo  $T$  ( $1 \leq T \leq 6$ ) indicando el número de casos de test. Cada uno de los casos de test se describen a continuación de la siguiente manera:

- La primera línea contiene cuatro enteros  $H$ ,  $W$ ,  $M$  y  $N$  ( $0 < M \leq H, 0 < N \leq W, H \times W \leq 10^6$ ). Estos enteros corresponden respectivamente al alto y ancho de la matriz que describe el terreno, y al alto y ancho de la submatriz donde se debe construir la torre.
- Las siguientes  $H$  líneas describen la matriz que representa el terreno. Cada línea contiene  $W$  enteros entre 0 y  $10^6$ . El  $j$ -ésimo entero en la línea  $i$ -ésima corresponde al entero en la fila  $i$ -ésima y columna  $j$ -ésima de la matriz.



## Salida

La salida debe contener para cada test un único entero correspondiente al coste de nivelación de la submatriz de tamaño  $M \times N$  de menor coste.

## Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
3	99
4 4 4 4	72
10 11 12 13	78
9 8 7 5	
10 8 9 1	
100 2 3 99	
4 5 3 2	
20 100 50 3 22	
7 40 8 100 33	
60 80 33 9 55	
33 56 44 21 88	
1 10 1 3	
88 22 100 2 200 101 0 500 100 0	



## Problema J

### Archipiélagos

A lo largo de la historia, navegantes y exploradores han cartografiado numerosos mares y océanos, pero aún quedan regiones inexploradas. En distintos puntos del mundo, antiguos mapas han revelado la posible existencia de archipiélagos olvidados, ocultos en océanos como el Atlántico, el Ártico, el Cantábrico y muchos más.

Cada uno de estos mapas está representado como una cuadrícula donde cada celda indica si hay agua (0) o tierra (1). Sin embargo, con el paso del tiempo, las marcas se han desvanecido, y no es posible determinar a simple vista cuántas islas hay en cada océano.

Se define una isla como un conjunto de tierras (1) conectadas en dirección horizontal, vertical o diagonal. Si dos porciones de tierra están separadas por agua (0), pertenecen a islas diferentes.

Para resolver este misterio, es necesario escribir un algoritmo que analice los mapas y cuente con precisión cuántas islas esconden estos océanos.

#### Ejemplo de entrada:

Matriz de  $5 \times 5$ :

```
1 1 0 0 0
1 1 0 0 1
0 0 0 1 1
0 0 0 0 0
1 1 0 0 1
```

#### Salida esperada:

Número de islas: 4.

#### Entrada

La primera línea de la entrada contiene un entero positivo  $T$  indicando el número de casos de test, siendo  $1 \leq T \leq 8$ . A continuación, cada caso de test tiene una primera línea con dos enteros positivos  $N$  y  $M$ ,  $1 \leq N, M \leq 250$ , que representan las dimensiones de la cuadrícula ( $N$  filas y  $M$  columnas). Las siguientes  $N$  líneas contienen  $M$  valores (0 o 1), separados por espacios, que representan el mapa del océano.

#### Salida

La salida debe contener  $T$  líneas, cada una correspondiente con un caso de test con un único número entero que indica la cantidad de islas presentes en el mapa.



## Ejemplos de Entrada y Salida

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



## Problema K

### Ciudad Futura

En un futuro lejano, la humanidad expandirá sus dominios más allá de la Tierra y comenzará a colonizar otros planetas. Para poder comunicarse con todos ellos, los científicos de **Ciudad Futura** han comenzado a desarrollar un avanzado sistema de comunicación interestelar. Para ello, han de tener en cuenta que cada civilización alienígena utiliza un sistema numérico diferente, con bases que van desde 2 hasta 9. Con el fin de facilitar esta comunicación, se necesita un programa que permita convertir cualquier número decimal a su equivalente en una base entre 2 y 9 ¿Podrías ayudarles?

#### Entrada

Un entero  $t$  ( $1 \leq t \leq 125000$ ) que indica cuántos números se van a transformar de base. A continuación, cada una de las  $i$  líneas siguientes ( $1 \leq i \leq t$ ) presentará un número natural  $n_i$  en base decimal, donde  $0 \leq n_i \leq 100000$  y un segundo número natural  $b_i$  que representa la base a la que se quiere convertir dicho número, siendo  $2 \leq b_i \leq 9$ .

#### Salida

Los resultados de conversión de cada número  $n_i$  proporcionados en la correspondiente base  $b_i$ , cada uno en una línea diferente y en el mismo orden de entrada.

#### Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
3	101
5 2	402
102 5	256701
89537 8	



## Problema L

### La fusión de los reinos digitales

En un mundo donde la informática domina la civilización, existen dos poderosos reinos: **Algorithm-landia** y **Datastructuria**.

Estos reinos han estado en competencia durante siglos, cada uno desarrollando su propio conjunto de procesadores ordenados según su eficiencia en operaciones computacionales. Sin embargo, han decidido unirse para crear una **supercomputadora definitiva**, combinando sus procesadores de manera justa.

Para lograrlo, necesitan encontrar el punto de equilibrio entre ambas arquitecturas: *el nivel de rendimiento mediano de la nueva red de cómputo*.

#### Tu misión

Dadas dos listas ordenadas de números enteros positivos, representando los índices de eficiencia de los procesadores de cada reino, escribe un programa eficiente que calcule la mediana de la combinación de ambos reinos.

#### Entrada

En la primera línea se indica el tamaño  $n$  de las dos listas, siendo  $1 \leq n \leq 1.000.000$  (se garantiza que ambas tienen el mismo tamaño). En la segunda y tercera línea se muestra cada una de las listas de enteros ordenadas de menor a mayor, representando la eficiencia de los procesadores en cada reino. Se garantiza que ambos reinos han ordenado correctamente sus datos antes de la fusión.

#### Salida

Un número decimal que representa la **mediana** del rendimiento de la nueva supercomputadora.

#### Ejemplos de Entrada y Salida

Entrada de ejemplo	Salida de ejemplo
3 1 3 5 2 4 6	3.5

Entrada de ejemplo	Salida de ejemplo
4 2 5 12 14 3 6 10 13	8.0