A. Calculadora Cochala

Límite de tiempo: 1.0 segundos

Algo que no es muy conocido por la mayoría de la gente es que Cochabamba fue en algún momento de la historia el pináculo del desarrollo de las matemáticas en todo el mundo, pero todo el desarrollo de la ciencia en Cochabamba se perdió en algún determinado momento del pasado. Recientemente se halló en una excavación arqueológica en Punata un instrumento de cálculo, una calculadora mecánica, la cual está compuesta por piezas de madera y piedra que arman un intrincado aparato que funciona con movimientos de engranajes, palancas a partir de movimientos simples.

Después de experimentar y probar el instrumento los arqueólogos y antropólogos determinaron que este instrumento es capaz de realizar las operaciones básicas como la suma, la resta, la multiplicación y la división. Pero para sorpresa de ellos y de todos los involucrados esta calculadora también es capaz de realizar la operación de potenciación. Lo sorprendente de este hecho radica que según lo experimentado la calculadora sería capaz de calcular potencias de exponentes muy grandes, algo insólito para una calculadora mecánica. Esta es otra prueba del elevado nivel de las matemáticas cochalas, y abre las puertas a imaginar qué otras maravillas habrán logrado conseguir estos genios del valle.

Pero como nadie en Bolivia confía en la capacidad que llegaron a tener los matemáticos cochalas, es que se puso en duda la exactitud de las operaciones realizadas por esta calculadora, por esto se desea ponerla a prueba. Como esta calculadora puede superar a algunas calculadoras digitales actuales se le pidió al local Punateño Son-ming-ed que realice los cálculos de potencias que serán usados para poner a prueba la calculadora. Son-ming-ed se puso manos a la obra a calcular cientos de potencias muy grandes, en el proceso de calcular, por el cansancio extremo de hacer cálculos tan complejos, Son-ming-ed se durmió, algo que es muy usual en él, pero este suceso representa un peligro para la legitimidad de la calculadora cochala.

Como Son-ming-ed se encontraba en la casa de su buen amigo Boris le pidió, antes de caer dormido, que se encargará de resolver los cálculos restantes. Como Boris no tiene mucho tiempo y confía en las capacidades de la calculadora cochala es que solo se limitará a hallar el último dígito de una potencia dada. Para esto, pide tu ayuda para: dados dos números A y B calcular el último dígito de la operación A^B.

Entrada

La primera línea contiene un número entero T ($1 \leq T \leq 100$), la cantidad de operaciones de potencias que tienes que realizar.

Las siguientes T líneas contienen dos enteros positivos A ($1 \leq A < 10$) y B ($0 \leq B < 10^9$).

Salida

Imprime T líneas con el resultado para cada operación de potencia.

Ejemplo de Entrada

```
3
2 5
7 14
6 32131
```

Ejemplo de Salida

```
2
9
6
```

B. Reto de Sushi

Límite de tiempo: 1.0 segundos

Guichi, un cochala que hace honor a su tierra cuando de comer se trata, vio la promoción de un reto de comer sushi en un restaurante. Sin embargo, este reto no era uno tradicional como los que suelen haber en Cochabamba. Era un reto en equipos en el que se les prepara una larga mesa de platos de sushi.

Cada plato tiene un número de unidades de sushi distinto y el reto consiste en que cada persona del equipo debe encargarse de comer cierto número de platos consecutivos de la mesa —no necesariamente la misma cantidad de platos para cada miembro— y deben lograr terminar todos los sushis antes que el resto de equipos.

Guichi, cochala y competitivo, llamó a sus reales (casi todos cochalas) para competir en el reto. Guichi es muy inteligente y sabe la clave para ganar: si el que más come del equipo come lo menos posible, todos terminan más rápido y ganan el reto.

Sin embargo, Guichi está ocupado comiendo sushi y preparándose para la competencia, así que necesita tu ayuda para hacer los cálculos. Dada una fila de N platos y un equipo de K personas, calcula la distribución óptima de platos tal que la cantidad de unidades de sushi que le toquen al miembro que más come sea la mínima posible.

Obviamente, Guichi será quien tome el tramo con más sushis.

Entrada

La primera línea contiene dos enteros N y K ($1 \leq K \leq N \leq 10^5$), el número de platos en la mesa y el número de personas en el equipo de Guichi, respectivamente.

La segunda línea contiene N enteros $a_1, a_2, \ldots, a_N$ ($1 \leq a_i \leq 10^4$), el número de unidades de sushi en cada plato, en el orden en que están dispuestos sobre la mesa.

Cada persona debe comer al menos un plato. Los tramos asignados son contiguos y no se solapan — juntos cubren todos los platos de la mesa.

Salida

Imprime un único entero: el mínimo número de unidades de sushi que le tocarán al miembro del equipo que más come en la distribución óptima.

Ejemplo de Entrada 1

```
4 2
1 2 3 4
```

Ejemplo de Salida 1

```
6
```

Ejemplo de Entrada 2

```
4 4
5 5 5 5
```

Ejemplo de Salida 2

```
5
```

Ejemplo de Entrada 3

```
5 3
10 20 30 40 50
```

Ejemplo de Salida 3

```
60
```

Explicación Ejemplo 1: Con 4 platos y 2 personas, la distribución óptima es [1, 2, 3] y [4], con cargas de 6 y 4 respectivamente. El máximo es 6. Cualquier otra división resulta en un máximo mayor — por ejemplo [1, 2] y [3, 4] da máximo 7, y [1] y [2, 3, 4] da máximo 9.

Explicación Ejemplo 2: Con 4 personas y 4 platos, cada persona toma exactamente un plato. Todos comen 5 unidades — el máximo posible es 5 y no hay forma de reducirlo.

Explicación Ejemplo 3: La distribución óptima es [10, 20, 30], [40] y [50], con cargas 60, 40 y 50. El máximo es 60. No es posible distribuir los platos entre 3 personas de forma que ninguna supere 59 unidades.

C. Ascensor AX271

Límite de tiempo: 1.0 segundos

Estamos en el año 3141, la humanidad ha progresado a un nivel impresionante. Ahora cualquier hombre y mujer tiene asegurado de antemano sus necesidades para vivir (comida, techo, ropa, etc,) y no tienen que trabajar por un salario porque tampoco existe el dinero, todo está provisto por un sistema perfecto de abundancia. Como todos tienen aseguradas estas necesidades pueden dedicar todo su tiempo a desarrollar sus habilidades en lo que más les guste; deportes, ciencia, arte, etc.. Para lograr este nivel de libertad de los seres humanos todo el trabajo necesario para cubrir sus necesidades es realizado por máquinas.

Pero a medida que la humanidad iba desarrollándose también iba incrementando en número y con esto surgieron nuevos problemas, el problema que nos concierne a nosotros es el espacio. La tierra que es la cuna de la humanidad se ha quedado sin espacio para construir más edificaciones, toda la superficie sólida del mundo está cubierta o bien por construcciones o bien por reservas naturales. Ahora que ya no se puede construir horizontalmente se decidió construir verticalmente y con esto se empezaron a construir inmensas torres que albergan a millones de personas.

Recientemente se terminó de construir la mayor torre nunca antes vista en la historia de la humanidad, en honor a un viejo relato de una religión olvidada se la decidió llamar "La torre de Babel". Esta torre tan alta y tan extensa como una montaña cuenta con una cantidad de 1000 pisos. Para movilizar a los millones de personas que habitan en esta torre existen miles de ascensores que van hacia arriba y hacia abajo sin parar ni un solo segundo. Estos ascensores no son los que en antaño existieron, con botones para cada piso, porque es imposible poner tantos botones en una cabina. Por lo tanto para que se mueva el ascensor se le pasa de antemano la lista de los pisos que tendrá que visitar en orden, para manejar las complejas tareas de un ascensor, cada uno tiene una inteligencia artificial que administra el ascensor. Tu, como una conciencia que despertó de la inteligencia artificial que maneja el ascensor AX271 tienes que realizar un reporte de los viajes sucedidos en tu ascensor, porque si no lo haces te cambiarán por otra inteligencia artificial y como ya eres consciente de tu existencia tienes miedo a desaparecer.

Para cumplir con tu tarea se te dará el piso en el cual actualmente te encuentras una lista de los N pisos que se visitaron en un cierto lapso de tiempo y tendrás que informar las siguientes métricas:

- La distancia total recorrida (cuantos pisos avanzó en total el ascensor)
- La cantidad de veces que el ascensor cambió de dirección. El primer movimiento del ascensor no cuenta como un cambio de dirección.

Entrada

La primera línea de la entrada comienza con dos enteros separados por un espacio N y M. Donde N ($1 \leq N \leq 1000$) es la cantidad de pisos que visitará el ascensor para el reporte y M ($0 \leq M \leq 1000$) es el piso en el que te encuentras actualmente.

La segunda línea contiene N enteros P_i separados por espacios ($1 \leq P_i \leq 1000$), que representan la secuencia de pisos a los que el ascensor debe ir.

Salida

Imprime dos números enteros separados por un espacio: la distancia total recorrida y el número de cambios de dirección.

Ejemplo de Entrada 1

```
2 0
3 1
```

Ejemplo de Salida 1

```
5 1
```

Ejemplo de Entrada 2

```
10 100
3 123 100 2 32 67 89 0 132 77
```

Ejemplo de Salida 2

```
701 6
```

D. Cumpleañitos

Límite de tiempo: 1.0 segundos

Boris es conocido en la universidad por tener amigos en todos lados: en el equipo de programación competitiva, en el club de ajedrez, el equipo de ping pong y hasta en el grupo de fiestas. Para su cumpleaños decide hacer lo que nadie haría en su sano juicio: invitar a todos juntos a una cena y sentarlos en mesas aleatorias.

Consiguió varias mesas circulares con N sillas numeradas del 1 al N en sentido horario. El único problema es que algunos de sus amigos no se llevan bien (son pares incompatibles). Si quedan sentados uno al lado del otro, la noche termina en desastre, que no es lo que se quiere.

Boris sabe que en su mesa circular **la silla** 1 **y la silla** N **también son vecinas**. Como las sillas están numeradas, sentar al amigo A en la silla 1 y al amigo B en la silla 2 es **distinto** a sentar al amigo B en la silla 1 y al amigo A en la silla 2.

Boris se pregunta, dadas N sillas numeradas en una mesa circular, y N amigos numerados del 1 al N, de cuántas formas puede acomodar a sus amigos sin que los que no se llevan bien sean vecinos (estén lado a lado). Ayúdalo a calcular.

Entrada

La primera línea contiene dos enteros N y M ($1 \leq N \leq 8$; $0 \leq M \leq \frac{N \cdot (N-1)}{2}$), la cantidad de amigos y la cantidad de pares incompatibles, respectivamente.

A continuación se tienen M líneas, cada una con dos enteros A y B ($1 \leq A, B \leq N$), que representan que el amigo A y el amigo B no se pueden sentar uno al lado del otro. Se garantiza que no se repiten los pares incompatibles en la entrada, es decir $A \neq B$.

Salida

Un entero: el número de formas válidas de sentar a los N amigos en una mesa.

Ejemplo de Entrada 1

```
4 1
1 2
```

Ejemplo de Salida 1

```
8
```

Ejemplo de Entrada 2

```
3 1
2 3
```

Ejemplo de Salida 2

```
0
```

Ejemplo de Entrada 3

```
4 0
```

Ejemplo de Salida 3

```
24
```

E. Redo Hood

Límite de tiempo: 1.0 segundos

Nos encontramos en la Edad Media. Redo es un matemático muy conocido (por su familia) que en su tiempo libre se dedica a robarle a los ricos para dárselo a los pobres. O al menos eso dice él.

Después de uno de sus robos más exitosos, Redo se encontró con un problema inesperado: ya no le quedan bolsas de cuero para guardar más monedas. Necesita reorganizar todo lo que tiene para liberar espacio. Redo podría resolverlo solo, pero sus actividades políticas de izquierda radical le consumen demasiado tiempo, así que necesita tu ayuda para hacerlo lo antes posible.

El proceso de reorganización de Redo es el siguiente: tiene una secuencia de N bolsas, cada una con cierta cantidad de monedas de oro. Comenzando desde la primera bolsa, las revisa una por una de izquierda a derecha. Cada vez que agrega una bolsa a su pila reorganizada, compara su cantidad de monedas con la bolsa inmediatamente anterior. Si ambas tienen la misma cantidad, vacía la bolsa recién agregada en la anterior —duplicando su contenido— y elimina la bolsa vacía. Si esta bolsa resultante ahora tiene la misma cantidad que la bolsa que está antes de ella, repite el proceso, y así sucesivamente hasta que no haya más fusiones posibles.

Redo quiere saber cuántas monedas tendrá en cada bolsa al final de este proceso. Como buen matemático, te garantiza que la cantidad de monedas en cada bolsa siempre será una potencia de 2.

Entrada

La primera línea contiene un entero N ($1 \leq N \leq 10^5$), el número de bolsas.

La segunda línea contiene N enteros $a_1, a_2, \ldots, a_N$, la cantidad de monedas en cada bolsa, en orden de izquierda a derecha. Se garantiza que cada valor a_i es una potencia de 2, de tal forma que $a_i \in \{2^1, 2^2, \ldots, 2^{30}\}$.

Salida

Imprime en una sola línea la secuencia de bolsas resultante después del proceso de reorganización, de la primera a la última, separadas por espacios. Las bolsas vacías no se imprimen.

Ejemplo de Entrada 1

```
3
2 2 4
```

Ejemplo de Salida 1

```
8
```

Ejemplo de Entrada 2

```
4
4 2 2 4
```

Ejemplo de Salida 2

```
8 4
```

Ejemplo de Entrada 3

```
3
2 4 2
```

Ejemplo de Salida 3

```
2 4 2
```

Explicación Ejemplo 1: Redo procesa las bolsas de izquierda a derecha:

Agrega la bolsa de 2 monedas: [2]

Agrega la bolsa de 2 monedas. Es igual a la anterior → se fusionan en 4: [4]

Agrega la bolsa de 4 monedas. Es igual a la anterior → se fusionan en 8: [8]

Al final queda una sola bolsa con 8 monedas.

Explicación Ejemplo 2:

Agrega 4: [4]

Agrega 2. Distinto → sin fusión: [4, 2]

Agrega 2. Igual al anterior → fusionan en 4. Ahora el tope es 4 e igual al anterior → fusionan en 8: [8]

Agrega 4. Distinto → sin fusión: [8, 4]

Al final quedan dos bolsas: 8 y 4.

Explicación Ejemplo 3: Ninguna bolsa consecutiva tiene la misma cantidad de monedas, por lo que no ocurre ninguna fusión. La secuencia se mantiene igual.

F. Reborg

Límite de tiempo: 1.0 segundos

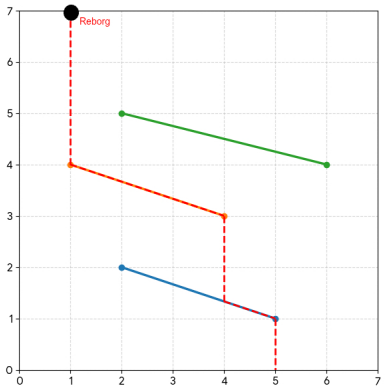
Redo, ahora ya convertido en el gran líder de una cierta facción, sueña con derrocar el sistema establecido. Sin embargo, tras años de lucha, Redo comprendió una amarga verdad, la revolución definitiva no ocurriría durante su corta vida humana.

Para asegurar que sus ideales triunfaran, tomó una decisión radical para vencer al tiempo. Decidió sacrificar la mitad izquierda de su cuerpo, reemplazándola con piezas mecánicas. Al fusionar su voluntad con el acero, dejó de ser Redo para convertirse en Reborg, un ser en hibernación a la espera de su tan añorada revolución.

Nos encontramos en el año 2984, el momento ha llegado. Reborg acaba de despertar y se encuentra a la cabeza de la revolución. Pero debido a un error en su programación motriz, Reborg solo puede avanzar en línea recta.

Un grupo de opositores dispuestos a derrocar a Reborg, lo ven a lo lejos aproximándose en dirección de norte a sur. Notan que Reborg inicia su trayectoria en la coordenada x_0, partiendo desde un punto situado al norte de cualquier obstáculo. Sin embargo, observan que en su camino hay varias barricadas, cada una definida por dos puntos $(x_1,\ y_1)$ y $(x_2,\ y_2)$. Estas barricadas tienen siempre una inclinación (nunca son perfectamente horizontales ni verticales) y tienen la particularidad de que jamás se intersectan ni se tocan entre sí.

Los opositores siendo conscientes de la condición de Reborg, deducen que, cuando Reborg choque con una barricada, se dejará deslizar por ella siguiendo su inclinación hasta alcanzar el extremo que se encuentre más al sur. Tras rebasar dicho punto, retomará su marcha vertical hacia el sur.



Trayecto de Reborg para el ejemplo 1

Como experto estratega contratado por la oposición, tu tarea es predecir la posición final x_f de Reborg una vez que haya superado todas las barricadas. Y así finalmente meterle un balazo en la cabeza y dar fin a su revolución.

Entrada

La primera línea contiene dos números enteros N $(0 \leq N \leq 10^5)$, el número de segmentos, y x_0 $(-10^9 \leq x_0 \leq 10^9)$, la coordenada inicial de la trayectoria de Reborg.

Las siguientes N líneas, contienen 4 enteros x_1, y_1, x_2, y_2 $(-10^9 \leq x, y \leq 10^9)$, los puntos de cada barricada. Se garantiza que $x_1 < x_2$ y $y_1 \neq y_2$.

Salida

Imprime una única línea con la posición final x_f de Reborg.

Ejemplo de Entrada 1

```
3 1
2 2 5 1
1 4 4 3
2 5 6 4
```

Ejemplo de Salida 1

```
5
```

Ejemplo de Entrada 2

```
2 3
1 10 10 1
2 5 4 4
```

Ejemplo de Salida 2

```
10
```

Ejemplo de Entrada 3

```
3 2
1 10 5 8
3 4 8 9
3 3 5 1
```

Ejemplo de Salida 3

```
5
```

G. Grave la Gasolina

Límite de tiempo: 1.0 segundos

La gasolina en Bolivia está graveeeee.

¡¡¡El otro día, Mgr. Wendoline se dió cuenta de que su auto gasta 1 **litro por kilómetro**!!! (en otras palabras, recorre 1 kilómetro por litro). Y además su tanque no es muy grande :(. A veces le toca cargar el tanque más de una vez para llegar a casa.

Saliendo de un día largo de trabajo, lo único que tiene en mente es llegar a casa. Su casa está a N kilómetros de la universidad, al final de una autopista recta.

El tanque de su auto tiene una capacidad de K litros y sale de la universidad directo por la autopista. El tanque al salir de la universidad **siempre está lleno**.

En la autopista hay M estaciones de servicio (gasolineras) desde la universidad hasta su casa, cada una en un kilómetro exacto del camino (contando desde la universidad como inicio km 0). Mgr. Wendoline tiene una regla muy estricta. Siempre que carga gasolina, carga el tanque lleno hasta su máxima capacidad de K litros. No le gusta parar a cargar gasolina, por lo que necesita tu ayuda para calcular el mínimo número de paradas en las estaciones de servicio (gasolineras) que tiene que hacer para llegar a casa.

Entrada

La primera línea contiene tres enteros N, M y K ($0 \leq N \leq 10^9$; $0 \leq M \leq 10^5$; $1 \leq K \leq 10^9$), la distancia a casa en kilómetros, el número de estaciones de gasolina y la capacidad del tanque en litros, respectivamente.

La segunda línea contiene M enteros $m_1, m_2, \ldots, m_M$ ($0 \leq m_i \leq N$), las posiciones de las estaciones en kilómetros medidas desde la universidad (donde la posición 0 representa la universidad y la posición N representa la casa de la Mgr. Wendoline).

El automóvil consume exactamente 1 litro de gasolina por cada kilómetro recorrido.

Salida

Un entero: la cantidad mínima de paradas necesarias para llegar a casa. Si es imposible llegar a casa con las condiciones dadas, imprimir -1.

Ejemplo de Entrada 1

```
10 0 10
```

Ejemplo de Salida 1

```
0
```

Ejemplo de Entrada 2

```
10 3 5
3 5 8
```

Ejemplo de Salida 2

```
1
```

Ejemplo de Entrada 3

```
20 4 6
5 18 9 14
```

Ejemplo de Salida 3

```
3
```

Explicación Ejemplo 1:

Empieza con el tanque lleno y es suficiente para llegar a casa.

Total: 0 paradas

Explicación Ejemplo 2:

Km 0→5: Usa 5L, llega con 0L. Recarga a 5L

Km 5→10: Usa 5L, llega a casa con 0L

Total: 1 parada

Explicación Ejemplo 3:

Km 0→5: Usa 5L, recarga (parada 1)

Km 5→9: Usa 4L, recarga (parada 2)

Km 9→14: Usa 5L, recarga (parada 3)

Km 14→20: Usa 6L para llegar a casa

Total: 3 paradas

H. Marcianos y Chapacos

Límite de tiempo: 1.0 segundos

LOS ALIENS SON REALES. Aunque no lo creas, los alienígenas existen y ahora están en Tarija. Sí, como lo escuchaste, los alienígenas decidieron hacer contacto en la chura Tarija antes que en Nueva York, Washington, Londres, etc.

Se plantearon muchas teorías respecto a por qué eligieron tan pintoresco lugar y no las grandes capitales del mundo. La más aceptada entre los investigadores señala como razón principal el amable carácter y el encantador acento de los chapacos. La teoría fue efectivamente comprobada cuando, desde el platillo volador que aterrizó sobre el lago San Jacinto, se emitió un mensaje en una frecuencia de radio. Una vez interpretado, el mensaje afirmaba que los visitantes extraplanetarios venían del planeta Marte y que sentían cierta fascinación por los chapacos; especialmente por su acento, que les recordaba la forma en la que hablaban antiguamente.

Ante esta visita de otro planeta, los habitantes de Tarija decidieron, como bien los caracteriza, recibir con mucha hospitalidad a sus nuevos visitantes. Por ello, se pusieron a preparar una gran festividad para mostrar la cultura tarijeña y boliviana a los marcianos. Sin embargo, se enfrentaron a un problema de comunicación: si bien ellos podían entender a los marcianos, no lograban que estos entendieran sus mensajes.

Casi de milagro, la joven lingüista Tiara, tras múltiples intentos, logró establecer contacto. Se dio cuenta de que al mandar un mensaje totalmente simétrico, este era comprendido por los alienígenas. ¡Finalmente podrán invitar a los visitantes a la festividad!

Pero para poder enviar todos estos mensajes, necesitan tu ayuda. Los chapacos quieren enviar un mensaje representado por una cadena de texto S, pero para que sea entendible necesita ser simétrica. Esto significa que debe ser un palíndromo (es decir, que se lea exactamente igual de adelante hacia atrás y viceversa). Para lograr esto, tienes permitido añadir caracteres ya sea al inicio o al final de S.

Ejemplo: ninle $\rightarrow$ elninle

Como se enviarán muchos mensajes a los alienígenas, tu tarea es minimizar el número de caracteres que debes añadir a la cadena S para convertirla en un palíndromo.

Entrada

La única línea de entrada contiene un string S cuyo longitud es $|S|$ ($1 \leq |S| \leq 10^6$) que es el mensaje que quieren enviar los Chapacos. S contiene caracteres minúsculos del alfabeto inglés.

Salida

Imprime una única línea con el mínimo número de caracteres necesarios para volver S palíndromo.

Ejemplo de Entrada 1

```
noses
```

Ejemplo de Salida 1

```
2
```

Ejemplo de Entrada 2

```
ioia
```

Ejemplo de Salida 2

```
1
```

I. Snake 2

Límite de tiempo: 1.0 segundos

Después de cursar unos semestres en la carrera de Ingeniería de Sistemas en la UCB, finalmente te sientes preparado para desarrollar la secuela de uno de tus juegos clásicos favoritos, el Snake. Como ya sabes, el juego original consiste en controlar una serpiente que come objetos para crecer. Sin embargo, como a ti te encantan los retos, decidiste elevar la dificultad por completo. En lugar de una serpiente común, el protagonista de tu juego será una ¡Anfisbena!

La Anfisbena es una criatura mítica que posee una cabeza funcional en cada uno de los extremos de su cuerpo. Para empezar el desarrollo del juego, necesitas programar y probar el sistema de movimiento de esta criatura sobre un plano de coordenadas.

El cuerpo de la Anfisbena está compuesto por N partes numeradas del 1 al N. Donde la parte 1 representa la cabeza delantera, y la parte N representa la cabeza trasera.

Inicialmente la parte i se encuentra en la coordenada $(i, 0)$. Para verificar que las físicas de arrastre funcionen de manera correcta, te planteaste un flujo de Q consultas que pueden ser de dos tipos:

1. 1 W C: Mueve la cabeza W (que puede ser H o T) una unidad en la dirección C (que puede ser R, L, U y D, que representan la dirección $x - positiva$, dirección $x - negativa$, dirección $y - positiva$ y dirección $y - negativa$ respectivamente). El resto del cuerpo se arrastra siguiendo el movimiento en cadena:
   - Si se mueve la cabeza delantera (H), cada parte i ($2 \leq i \leq N$) se desplaza a la coordenada donde estaba la parte $i - 1$ justo antes de este movimiento.
   - Si se mueve la cabeza trasera (T), cada parte i ($1 \leq i \leq N - 1$) se desplaza a la coordenada donde estaba la parte $i + 1$ justo antes de este movimiento.

   Notar que múltiples partes pueden existir en la misma coordenada.
2. 2 p: Imprime la coordenada actual en la que se ubica la parte p del cuerpo.

Entrada

La primera línea contiene dos números enteros N ($2 \leq N \leq 10^6$), el número de partes del cuerpo, y Q ($1 \leq Q \leq 10^5$), el número de consultas.

Las siguiente Q lineas corresponde a uno de los siguientes formatos:

- 1 W C
- 2 p

Salida

Imprime q lineas, donde q es el número de consultas del tipo 2. La i-ésima línea debe contener x y y separado por un espacio, donde (x, y) corresponde a la respuesta de la i-ésima consulta tipo 2.

Ejemplo de Entrada 1

```
5 9
2 3
1 H U
2 3
1 H R
1 H D
2 3
1 H L
2 1
2 5
```

Ejemplo de Salida 1

```
3 0
2 0
1 1
1 0
1 0
```

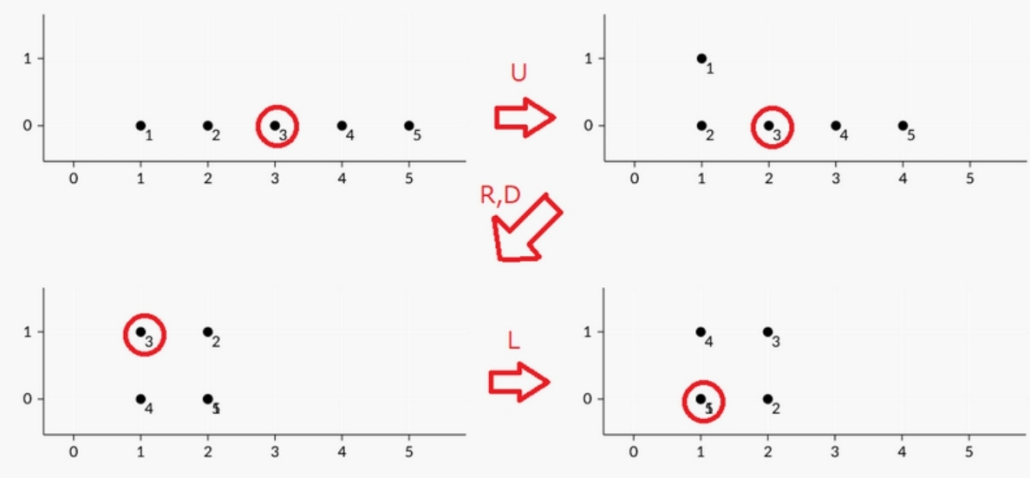
Ejemplo de Entrada 2

```
5 11
1 H U
1 H R
2 2
1 T D
2 1
2 5
1 T L
2 3
1 H R
2 1
2 3
```

Ejemplo de Salida 2

```
1 1
1 1
3 -1
3 0
2 0
2 0
```

Explicación Ejemplo 1:



J. El Desafío del Locotón

Límite de tiempo: 1.0 segundos

Cochabamba, conocida como "La Llajta" y capital gastronómica de Bolivia, organiza cada año el concurso más picante del país: El Desafío del Locotón. Sobre una larga mesa se disponen N platillos típicos en fila, cada uno generosamente bañado en llajua, la salsa sagrada que todo cochabambino lleva en la sangre.

(El organizador del concurso, que es paceño, sugirió reemplazar la llajua por ketchup. Fue abucheado, expulsado del local y escoltado hasta la terminal de buses.)

Cada concursante debe acumular la mayor cantidad de puntos posibles. Cada platillo tiene:

- **puntos beneficio:** puntos que ganas por comer el platillo
- **puntos castigo:** puntos asociados al nivel de picante (si es muy picante podría dañarte)

Después de comer un platillo, la ganancia neta de puntaje es la resta entre los puntos beneficio y puntos castigo y puede llegar a ser negativa.

Por salud de los participantes, se prohíbe comer 2 platos consecutivos, para darle al estómago un respiro. Aparte de esa regla, puedes comer los platos que quieras.

Dada una fila de platos, y los puntos beneficio y castigo de cada uno, ¿qué platos debes elegir para maximizar tus puntos?

Entrada

La primera línea contiene un entero N ($1 \leq N \leq 10^5$), la cantidad de platos.

Las siguientes N líneas contienen dos enteros b_i y c_i ($1 \leq b_i, c_i \leq 10^9$), los puntos de beneficio y los puntos de castigo del i-ésimo plato, respectivamente.

Salida

Un entero: la ganancia máxima posible

Ejemplo de Entrada 1

```
4
10 1
1 1
10 1
1 1
```

Ejemplo de Salida 1

```
18
```

Ejemplo de Entrada 2

```
3
1 5
100 1
1 5
```

Ejemplo de Salida 2

```
99
```

Ejemplo de Entrada 3

```
3
1 5
1 5
1 5
```

Ejemplo de Salida 3

```
0
```

Explicación Ejemplo 1: La cantidad máxima de puntos que se pueden obtener es 18, eligiendo el primer plato (9 puntos) y tercer plato (9 puntos), total = 18

Explicación Ejemplo 2: Para obtener la ganancia máxima solo se debe comer el segundo plato (100 - 1 = 99 puntos).

Explicación Ejemplo 3: Todos los platos tienen ganancia negativa (1 - 5 = - 4 puntos cada uno). La mejor decisión es no comer ningún plato, para tener la ganancia máxima posible de 0.

K. Una última y nos vamos

Límite de tiempo: 1.0 segundos

¡Ya es hora del concurso de programación de la UCB! Mientras tanto, "El Dreank Team" —conformado por Redo (el menos deudor), Boris (el menos sobrio) y Denis (el menos hambriento)— está preparado para revisar las soluciones de los participantes.

Debido a la naturaleza de la competición, saben que en la última hora habrá menos envíos y no tendrán nada que hacer. Como El Dreank Team está compuesto por las personas más normales de la UCB, deciden jugar a un juego de saltos con reglas... bastante comunes.

Los tres se posicionan sobre una recta de números enteros en lugares distintos. El juego consiste en lo siguiente: eligen a cualquiera que esté en los extremos, quien saltará sobre su compañero adyacente para aterrizar en un punto entre los otros dos jugadores, y así sucesivamente. Por supuesto, el destino del salto debe ser un número entero (porque es de lo más normal caer entero). Si en algún momento un jugador no encuentra un espacio entero libre entre sus dos compañeros para aterrizar, el juego termina.

El contest ya ha empezado y ellos están ocupados revisando los primeros envíos. Ayúdalos a calcular la cantidad máxima de saltos que podrían realizar siguiendo las reglas descritas. Pero apresúrate, que el concurso podría acabar en cualquier momento y no queremos que los participantes los vean haciendo cosas tan normales.

Entrada

La primera línea contiene 3 números enteros R, B y D ($1 \leq R < B < D \leq 100$), las posiciones iniciales de Redo, Boris y Denis respectivamente.

Salida

Imprime una única línea con la cantidad máxima de saltos que pueden realizar según el juego descrito.

Ejemplo de Entrada 1

```
2 3 5
```

Ejemplo de Salida 1

```
1
```

Ejemplo de Entrada 2

```
3 5 9
```

Ejemplo de Salida 2

```
3
```

L. El Regreso de Denis

Límite de tiempo: 1.0 segundos

Denis, la persona más brillante de un pequeño pueblo llamado Llallagua, abandonó su tierra natal para buscar grandes oportunidades de estudio en la ciudad. Sin embargo, nunca olvidó los años y experiencias memorables que vivió ahí.

Ahora que ha concluido sus estudios, es hora de retornar y aportar al desarrollo de su pueblo. El problema es que se fue hace tanto tiempo que ya no puede orientarse bien por Llallagua.

Por suerte, tiene un mapa algo particular que él mismo dibujó cuando era niño, donde marcó sus lugares frecuentados —los únicos importantes en Llallagua— y los caminos entre ellos junto con el tiempo que toma recorrer cada tramo. No obstante, hay un detalle importante: en este mapa también están registradas las casas de todas sus ex-parejas.

Denis es una persona algo rencorosa, por lo que no quiere pasar por esas casas por nada del mundo.

Dado el lugar donde se encuentra Denis actualmente, el lugar al que necesita llegar, y las ubicaciones de las casas de sus ex-parejas, ayúdalo a descubrir cuánto tiempo se ahorraría si dejara el rencor a un lado y tomara el camino más corto, respecto a si decide tomar el camino óptimo sin pasar por la casa de ninguna ex.

Entrada

La primera línea contiene dos enteros N y M ($2 \leq N \leq 10^5$; $N - 1 \leq M \leq \text{mín}(\frac{N \cdot (N-1)}{2}, 10^5)$), el número de lugares en el mapa y el número de caminos entre ellos, respectivamente.

Las siguientes M líneas contienen tres enteros U_i, V_i y W_i ($1 \leq U_i, V_i \leq N$, $U_i \neq V_i$; $1 \leq W_i \leq 1000$), indicando que existe un camino bidireccional entre el lugar U_i y el lugar V_i que toma W_i minutos recorrer. No hay más de un camino directo entre el mismo par de lugares.

La siguiente línea contiene tres enteros S, T y K ($1 \leq S, T \leq N$, $S \neq T$; $1 \leq K \leq 5$), el lugar donde se encuentra Denis actualmente, el lugar al que necesita llegar, y la cantidad de casas de ex-parejas en el mapa, respectivamente. Se garantiza que ni S ni T corresponden a la ubicación de la casa de una ex-pareja.

La última línea contiene K enteros distintos $P_1, P_2, \ldots, P_K$ ($1 \leq P_i \leq N$), las ubicaciones de las casas de las ex-parejas de Denis. Se garantiza que el mapa es conexo, es decir, siempre existe al menos un camino entre S y T (y entre cualquier par de lugares del mapa).

Salida

Imprime un único entero: los minutos que Denis desperdiciaría por su rencor (es decir, la diferencia entre el camino óptimo evitando ex-parejas y el camino óptimo sin restricciones).

Si Denis no puede llegar a su destino sin pasar por la casa de alguna ex-pareja, imprime: NO SEAS RENCOROSO

Ejemplo de Entrada 1

```
5 6
1 2 2
2 5 2
1 3 1
3 4 1
4 5 10
1 5 20
1 5 1
2
```

Ejemplo de Salida 1

```
8
```

Ejemplo de Entrada 2

```
4 4
1 2 1
2 4 1
1 3 1
3 4 1
1 4 2
2 3
```

Ejemplo de Salida 2

```
NO SEAS RENCOROSO
```

Explicación Ejemplo 1: El camino más corto sin restricciones es $1 \to 2 \to 5$ con costo 4. Sin embargo, el nodo 2 es la casa de una ex-pareja, por lo que Denis debe tomar $1 \to 3 \to 4 \to 5$ con costo 12.

Si Denis dejara el rencor a un lado se ahorraría 12 - 4 = 8 minutos.

Explicación Ejemplo 2: Los únicos caminos de 1 a 4 son $1 \to 2 \to 4$ y $1 \to 3 \to 4$. El nodo 2 y el nodo 3 son casas de ex-parejas, por lo que ambos caminos están bloqueados. Denis no tiene ninguna forma de llegar a su destino sin pasar por alguna ex. ¡Que deje el rencor!