

## Maratón de programación de la SBC 2019

El set de problemas de este cuaderno es utilizado simultáneamente en las siguientes competencias:

Maratona de Programação da SBC 2019  
Tercera Fecha Gran Premio de México 2019  
Tercera Fecha Gran Premio de Centroamérica 2019

*14 de Septiembre de 2019*

### Cuaderno de Problemas

#### Información General

Este cuaderno contiene 13 problemas; Las páginas están numeradas de 1 a 18, sin contar esta página. Verifique que su cuaderno está completo.

#### A) Sobre los nombres de los programas

- 1) Para soluciones en C/C++ y Python, el nombre del archivo de código fuente no es significativo, puede ser cualquier nombre.
- 2) Si su solución es en Java, el archivo debe ser llamado: *codigo\_de\_problema.java* donde *codigo\_de\_problema* es la letra mayúscula que identifica al problema. Recuerde que en Java el nombre de la clase principal debe ser igual que el nombre del archivo.
- 3) Si su solución es en Kotlin, el archivo debe ser llamado: *codigo\_de\_problema.kt* donde *codigo\_de\_problema* es la letra mayúscula que identifica al problema. Recuerde que en Kotlin el nombre de la clase principal debe ser llamado igual que el nombre del archivo

#### B) Sobre la entrada

- 1) La entrada de su programa debe ser leída de *entrada standard*.
- 2) La entrada está compuesta de un único caso de prueba, descrito en un número de línea que depende del problema.
- 3) Cuando una línea de entrada contiene varios valores, estos están separados por un único espacio en blanco; la entrada no contiene ningún otro espacio en blanco.
- 4) Cada línea, incluyendo la última, contiene exactamente un caracter de final-de-línea.
- 5) El final de la entrada coincide con el final del archivo.

#### C) Sobre la salida

- 1) La salida de su programa debe ser escrita en *salida standard*.
- 2) Cuando una línea de salida contiene varios valores, estos deben ser separados por un único espacio en blanco; la salida no debe contener ningún otro espacio en blanco.
- 3) Cada línea, incluyendo la última, debe contener exactamente un caracter de final-de-línea.

Promocional:



Sociedade Brasileira de Computação

## Problema A

# Arte Valiosa

La Mona Dura es una de las obras de arte más valiosas del museo de Nlogonia. La pintura famosa es exhibida en una sala rectangular de  $M$  por  $N$  metros. La entrada de la sala se encuentra en una esquina de la misma, mientras que la Mona está en la esquina diagonalmente opuesta a la entrada.

Para evitar robos, la sala tiene sensores de movimiento que son activados todas las noches cuando el museo cierra. Cada sensor tiene una sensibilidad  $S$ , tal que el sensor dispara una alarma si detecta algún movimiento a no más de  $S$  metros de distancia de su ubicación.

Un ladrón irrumpió en el museo esta noche con la intención de robar la Mona Dura. Para lograr su cometido, el ladrón necesita entrar a la sala y llegar a la pintura sin ser detectado por ningún sensor de movimiento, es decir, debe de mantener una distancia mayor a  $S_i$  metros del  $i$ -ésimo sensor todo el tiempo, para todos los sensores.

El ladrón ha obtenido acceso a los planos del museo, por lo tanto, conoce las dimensiones de la sala, las coordenadas, y las sensibilidades de cada uno de los sensores. Dada esta información, su tarea es, determinar si el robo es posible o no.

### Entrada

La primera línea contiene tres números enteros,  $M$ ,  $N$ , y  $K$ , representando las dimensiones de la sala, y el número de sensores de movimiento, respectivamente. ( $10 \leq M, N \leq 10^4$ ,  $1 \leq K \leq 1000$ ). La entrada de la sala se encuentra en la coordenada  $(0, 0)$ , y la pintura en la coordenada  $(M, N)$ .

Cada una de las siguientes  $K$  líneas corresponde a uno de los  $K$  sensores y contiene tres números enteros,  $X$ ,  $Y$ , y  $S$ , donde  $(X, Y)$  representa la coordenada del sensor y  $S$  indica su sensibilidad. ( $0 < X < M$ ,  $0 < Y < N$ ,  $0 < S \leq 10^4$ ). Todas las dimensiones y coordenadas de la entrada son en metros. Se garantiza que todos los sensores tienen coordenadas distintas.

### Salida

Su programa debe imprimir una única línea que contenga la letra 'S' en caso de que sea posible robar la pintura, o la letra 'N' en caso contrario.

<p><b>Ejemplo de entrada 1</b></p> <p>10 22 2 4 6 5 6 16 5</p>	<p><b>Ejemplo de salida 1</b></p> <p>S</p>
<p><b>Ejemplo de entrada 2</b></p> <p>10 10 2 3 7 4 5 4 4</p>	<p><b>Ejemplo de salida 2</b></p> <p>N</p>
<p><b>Ejemplo de entrada 3</b></p> <p>100 100 3 40 50 30 5 90 50 90 10 5</p>	<p><b>Ejemplo de salida 3</b></p> <p>S</p>

## Problema B

# Bufón de la corte

El reino de los emparejamientos es gobernado por un generoso comandante. La fama y grandes cualidades del comandante son conocidas por todos, inclusive en los reinos vecinos. Una de sus más famosas cualidades es su buen humor, que es nutrido todos los días por el bufón de la corte que es elegido anualmente en el Gran Concurso de Comedia (GCC) del reino. El bufón ayuda a aliviar las tensiones de las diversas reuniones políticas que demanda el trabajo, alegrando no solo al comandante sino a todo el reino.

El joven Carlos es un gran comediante cuyo sueño es convertirse en el bufón de la corte para la próxima temporada. El ha pasado los últimos meses escribiendo chistes y juegos de palabras de varios tipos, muchos de los cuales son sobre su propia (muy baja) estatura. Ha llegado el momento de las elecciones para bufón de la corte y se han inscrito un total de  $N$  candidatos. Cada uno de los candidatos tendrá cinco minutos para presentarse frente a la audiencia. Después de las presentaciones, cada ciudadano del reino de los emparejamientos puede votar por uno de los candidatos, y el candidato más votado será el nuevo bufón de la corte. En caso de haber un empate entre uno o más candidatos, el candidato que haya hecho la inscripción primero es elegido. Sabiendo esto, el joven Carlos pasó noches frente a la oficina electoral y se aseguró de que su inscripción fuera la primera en hacerse.

Después de la votación, solo queda esperar los resultados. La urna electrónica genera un informe con  $N$  números enteros, correspondientes al número de votos de cada candidato, ordenando por orden de registro. Tu misión es determinar si el joven Carlos fue elegido o no.

### Entrada

La primera línea de la entrada contiene un número entero  $N$ , ( $2 \leq N \leq 10^4$ ). Las siguientes  $N$  líneas contienen  $N$  enteros positivos  $v_1, \dots, v_N$ , uno en cada línea, correspondientes al número de votos recibidos por cada uno de los candidatos, en su orden de inscripción. Como la población del Reino de los emparejamientos es de 100,000 personas, el número total de votos no será superior a este valor, es decir,  $\sum_{i=1}^N v_i \leq 100,000$ .

### Salida

Su programa debe imprimir una única línea que contenga la letra ‘S’ en caso que el joven Carlos sea electo como bufón de la corte, o la letra ‘N’ en caso contrario.

<p><b>Ejemplo de entrada 1</b></p> <p>3 1000 1000 1000</p>	<p><b>Ejemplo de salida 1</b></p> <p>S</p>
<p><b>Ejemplo de entrada 2</b></p> <p>5 1 2 3 4 5</p>	<p><b>Ejemplo de salida 2</b></p> <p>N</p>

## Problema C

# Cruces peligrosos

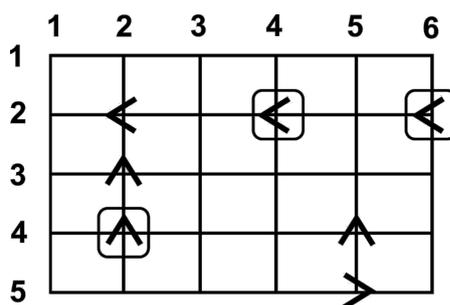
La Agencia de Simulación Vehicular (ASV), administrada por su fundador Crishna, está trabajando en un proyecto destinado a obtener datos relacionados con colisiones de vehículos en cruces de algunas regiones del país.

La primera simulación de la ASV consiste en colocar aleatoriamente  $C$  vehículos en cruces de una región determinada. Inicialmente habrá máximo un vehículo en cada cruce. Una región está representada por  $N$  carriles horizontales que cruzan con  $M$  carriles verticales.

Al comenzar la simulación, cada vehículo se desplazará siguiendo su dirección inicial (Norte, Sur, Este, u Oeste) a una velocidad constante de 1 cruce por segundo.

Si dos o más vehículos llegan al mismo cruce al mismo tiempo, colisionarán y ya no se moverán. Si un vehículo pasa por un cruce en el cuál ocurrió una colisión anteriormente, el vehículo también colisionará con los vehículos detenidos. Por razones desconocidas, cuando dos vehículos chocan horizontalmente **entre** dos cruces, ambos se detendrán en el cruce que se encuentra al este, y si dos vehículos chocan verticalmente **entre** dos cruces, ambos se detendrán en el cruce que se encuentra al norte.

A continuación se muestra un ejemplo de simulación con  $N = 5$ ,  $M = 6$  y  $C = 7$  los vehículos son representados por flechas indicando sus direcciones. Tenga en cuenta que los 3 vehículos resaltados colisionarán en algún momento.



Dado que el software de simulación de la ASV aún no es totalmente confiable, ellos proporcionarán ejemplos de configuraciones iniciales para contar la cantidad de vehículos que no colisionarán.

### Entrada

La primer línea de entrada contendrá tres números enteros  $N$ ,  $M$ , y  $C$  ( $2 \leq N, M \leq 10^5$  y  $1 \leq C \leq \min(10^5, N \times M)$ ), representando respectivamente la cantidad de carriles horizontales, la cantidad de carriles verticales, y la cantidad de vehículos en la simulación. Cada una de las siguientes  $C$  líneas contendrá dos números enteros  $A_i$  y  $B_i$  y una letra  $D$  ( $1 \leq A_i \leq N$  y  $1 \leq B_i \leq M$ ) representando que el  $i$ -ésimo vehículo está inicialmente en el cruce del  $A_i$ -ésimo carril horizontal con el  $B_i$ -ésimo carril vertical, con una dirección indicada por  $D$ : 'N' para el Norte, 'S' para el Sur, 'L' para el Este, y 'O' para el Oeste.

### Salida

Su programa debe imprimir una única línea con un número entero que representan la cantidad de vehículos que no colisionarán

<b>Ejemplo de entrada 1</b> 5 6 7 2 2 0 3 2 N 4 2 N 4 5 N 2 6 0 5 5 L 2 4 0	<b>Ejemplo de salida 1</b> 4
<b>Ejemplo de entrada 2</b> 2 2 3 1 1 L 1 2 0 2 2 N	<b>Ejemplo de salida 2</b> 0
<b>Ejemplo de entrada 3</b> 2 2 3 1 1 L 1 2 0 2 1 N	<b>Ejemplo de salida 3</b> 1

## Problema D

# Denuncia Premiada

La policía de Nlogonia está investigando a la mafia local. Ya conocen a todos los miembros y la estructura de la organización: la mafia nlogoniana tiene  $N$  miembros en total, y cada uno se identifica por un número entero entre 1 y  $N$ , donde 1 es la identificación del jefe. Además, cada miembro es subordinado directo de otro miembro, excepto el jefe.

Incluso después de meses de investigación, la policía aún no tiene suficiente información para arrestar a ningún miembro de la mafia por algún delito. Entonces decidieron pedir la ayuda de un vidente: dado un miembro de la mafia, el vidente puede adivinar mágicamente los crímenes que ha cometido, y la policía puede confirmarlos mediante un interrogatorio.

Además, cuando se interroga a un mafioso nlogoniano, este no solo admite sus crímenes, sino que a cambio de una sentencia más leve también informa los crímenes de su superior directo. Si el superior directo no ha sido arrestado aún, la policía también lo interrogará, y este informará los crímenes de su superior, y así sucesivamente, hasta que lleguen al jefe de la mafia.

Desafortunadamente, el vidente solo tiene suficiente energía para adivinar los crímenes de máximo  $K$  mafiosos, y la policía quiere usar sus poderes con cuidado para arrestar a tantos tipos malos como sea posible. Dado  $K$  y la estructura completa de la mafia, ¿Cuál es la cantidad máxima de mafiosos que la policía puede arrestar?

### Entrada

La primera línea de entrada contiene dos enteros,  $N$  y  $K$ , donde  $N$  es el número de miembros de la mafia y  $K$  es el número máximo de mafiosos cuyo vidente puede adivinar ( $3 \leq N \leq 10^5$ ,  $1 \leq K < N$ ). La segunda línea contiene  $N - 1$  enteros, donde el  $i$ -ésimo de ellos identifica al superior directo del mafioso de identificación  $i + 1$ . Se garantiza que todos los números enteros en la segunda línea de entrada tienen valores entre 1 y  $N$ , y que todos los miembros de la mafia son subordinados del jefe, directa o indirectamente.

### Salida

Su programa debe imprimir una única línea con un entero que representan el número máximo de mafiosos que la policía puede arrestar

<b>Ejemplo de entrada 1</b> 8 2 1 1 2 3 4 4 6	<b>Ejemplo de salida 1</b> 7
<b>Ejemplo de entrada 2</b> 10 3 1 1 2 2 3 3 4 4 5	<b>Ejemplo de salida 2</b> 8

## Problema E

# Exhibición de Peces

El Gran Acuario de Nlogonia recibe a miles de visitantes cada mes. Una de sus atracciones más populares es la exhibición de peces payaso, un salón con varios tanques de cardúmenes de esta hermosa especie blanca y naranja. Los visitantes tienen la oportunidad de aprender muchas curiosidades sobre el pez payaso, incluida su organización social: las cardúmenes de peces payaso son dirigidos por hembras, y cuando la última hembra muere o abandona el cardumen, uno de los machos del cardumen muta y se convierte en hembra y entonces pasa a liderar del cardumen!

¡Tan pronto como se enteró de esto, Zelius el Conserje decidió jugar un truco en el Acuario para hacer que todos los peces payaso en la exhibición se vuelvan hembras! Para esto, moverá los peces de un tanque a otro por la noche cuando el Acuario esté cerrado. Si al final de la noche queda un tanque con uno o más machos y ninguna hembra, al día siguiente uno de ellos ya se habrá convertido en hembra.

Para no despertar sospechas del resto del personal del acuario, Zelius solo moverá un pez payaso cada noche, y cada pez solo será movido entre los tanques de la exposición. Cada tanque es lo suficientemente grande como para contener una cantidad ilimitada de peces, y Zelius puede dejar tantos tanques vacíos como quiera. Podemos suponer que nadie más tocará a los peces, y que ningún pez nacerá, morirá, se agregará o se eliminará de los tanques.

Zelius contó cuántos peces machos y hembras viven actualmente en cada tanque de la exhibición. Ahora necesita tu ayuda para planificar sus movimientos para convertir a todos los peces payaso en hembras en el menor tiempo posible.

### Entrada

La primera línea contiene un número entero  $N$ , el número de tanques en la exhibición ( $2 \leq N \leq 3000$ ). Cada una de las siguientes líneas  $N$  corresponde a uno de los tanques y contiene dos enteros,  $M$  y  $F$ , las cantidades de peces machos y hembras en ese tanque, respectivamente ( $0 \leq M, F \leq 10^5$ ,  $M = 0$  o  $F > 0$ ).

### Salida

Su programa debe imprimir una única línea con un número entero que representan la cantidad mínima de movimientos necesarios.

<b>Ejemplo de entrada 1</b> 2 2 1 0 2	<b>Ejemplo de salida 1</b> 2
<b>Ejemplo de entrada 2</b> 2 2 5 1 3	<b>Ejemplo de salida 2</b> 7

<b>Ejemplo de entrada 3</b>	<b>Ejemplo de salida 3</b>
4 2 3 0 0 3 1 0 0	5

## Problema F

# Forestación en Riesgo

Debido al avance de la deforestación en las últimas décadas, los ríos de Nlogonia registraron una reducción significativa en su flujo. Como Nlogonia es una nación desarrollada que basa sus decisiones en datos técnicos, el líder de la nación ha encargado una serie de estudios para comprender qué pasos deben tomarse para garantizar el agua para las generaciones futuras.

El informe técnico preparado por los científicos involucrados en el proyecto fue categórico: un porcentaje del territorio del país debe conservar su vegetación. Más que eso, las áreas cercanas a las orillas del río deben ser las más conservadas.

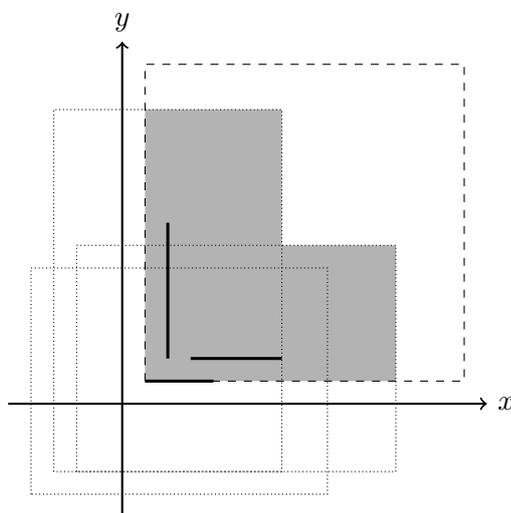
Entrará en vigor una nueva legislación ambiental, en la que las áreas que se encuentren a cierta distancia de las orillas del río formarán parte del área de conservación. El valor ideal de esta distancia aún se desconoce, pero el informe técnico ya ha determinado el porcentaje del territorio de la nación que debe preservarse.

Dadas sus capacidades técnicas, se le ha pedido que ayude a determinar la distancia alrededor de los ríos que debe preservarse para lograr el porcentaje requerido de área conservada.

Los ríos de Nlogonia se pueden representar en el plano como segmentos rectos paralelos a los ejes. Dada una distancia  $r$ , el área del territorio a preservar se determina de la siguiente manera: para cada río, el área preservada circundante corresponde al rectángulo más pequeño que contiene al segmento que representa el río, respetando una distancia mínima de  $r$  unidades entre cualquier punto del segmento y cualquier punto fuera del rectángulo. El territorio de Nlogonia se define como un rectángulo con lados paralelos a los ejes, de modo que cada río es paralelo a alguna frontera.

Dado un entero  $P$  entre 1 y 100, debe determinar el entero  $r$  más pequeño que garantice la preservación de  $P\%$  del territorio de Nlogonia.

La siguiente figura ilustra el primer ejemplo de la entrada. El territorio de Nlogonia está representado por la región discontinua, y el área preservada está representada por la región gris:



### Entrada

La primera línea de entrada contiene un entero  $N$  ( $1 \leq N \leq 10^4$ ) que indica el número de segmentos de línea que representan los ríos de Nlogonia. Cada una de las siguientes  $N$  líneas contiene 4 enteros:  $x_1$ ,  $y_1$ ,  $x_2$  y  $y_2$ , donde  $(x_1, y_1)$  y  $(x_2, y_2)$  son los extremos de un segmento de recta que representa a un río. Como los ríos de Nlogonia son paralelos a las fronteras, se garantiza que  $x_1 = x_2$  o  $y_1 = y_2$ .

La siguiente línea contiene un entero  $P$  ( $1 \leq P \leq 100$ ) que indica el porcentaje mínimo de territorio que debe preservarse. La última línea contiene 4 enteros  $x_1$ ,  $y_1$ ,  $x_2$ ,  $y_2$ , donde  $(x_1, y_1)$  es la esquina

inferior izquierda y  $(x_2, y_2)$  es la esquina superior derecha del rectángulo que representa el territorio de Nlogonia, con lados paralelos a los ejes coordenados.

Cada coordenada descrita en la entrada es un número entero entre 0 y  $10^5$ . Puede suponer que todos los ríos están completamente contenidos en el territorio de Nlogonia.

### Salida

Su programa debe imprimir una única línea con un entero  $r$  que representan el valor mínimo de  $r$  que puede ser usado para preservar el  $P\%$  del territorio de Nlogonia.

<p><b>Ejemplo de entrada 1</b></p> <p>3  1 1 4 1  2 2 2 8  3 2 7 2  50  1 1 15 15</p>	<p><b>Ejemplo de salida 1</b></p> <p>5</p>
<p><b>Ejemplo de entrada 2</b></p> <p>1  0 0 0 4  50  0 0 4 4</p>	<p><b>Ejemplo de salida 2</b></p> <p>2</p>

## Problema G

# Guardando los Adornos

Sicrana ama los adornos. En casa, ella tiene  $N$  adornos que se muestran alineados en un gran estante. Cada adorno se identifica por un entero distinto entre 1 y  $N$ .

Un día, mientras jugaba a la pelota en el interior, Fulano, el hijo de Sicrana, golpeó el estante de adornos de su madre y los tiró al suelo. Afortunadamente, no se dañó ningún adorno con la caída. Si Fulano puede volver a colocar todos los adornos en el estante exactamente como estaban antes de la caída es posible que su madre no se dé cuenta de que algo malo ha sucedido.

Dado que Fulano tiene mala memoria, no puede recordar el orden en que se encontraban originalmente los adornos, por lo que necesita tu ayuda. Por cada adorno  $i$ , Fulano te dirá  $N$  valores entre 1 y 100, donde el  $j$ -ésimo valor indica cuánta confianza tiene Fulano de que el adorno  $i$  estaba originalmente en la posición  $j$  del estante. Para maximizar su confianza de que no recibirá un regaño y será castigado, al elegir un orden de las decoraciones, Fulano multiplicará la confianza de que cada adorno está en el lugar correcto. Más formalmente, la confianza total de Fulano en un orden dado de adornos se calcula de la siguiente manera: si  $p_i$  es la posición ocupada por el  $i$ -ésimo adorno y  $a(i, j)$  es la confianza de Fulano de que el adorno  $i$  estaba originalmente en la posición  $j$ , entonces la confianza total de fulano está dada por :  $\prod_{i=1}^N a(i, p_i)$ .

Dado que hay muchas formas diferentes para colocar los adornos, tu misión, si desea aceptarla, es encontrar el orden en el que Fulano tenga más confianza.

### Entrada

La primera línea de la entrada contiene un número entero  $N$  ( $1 \leq N \leq 100$ ), que indica el número de decoraciones. Cada una de las siguientes  $N$  líneas contiene  $N$  enteros con valores entre 1 y 100. El  $j$ -ésimo entero de la  $i$ -ésima línea representa la confianza que tiene Fulano de que el adorno  $i$  estaba en la posición  $j$  del estante.

### Salida

Su programa debe imprimir una única línea con  $N$  enteros que representan el orden en el que Fulano debe colocar las decoraciones para maximizar su confianza. Si hay más de un orden que da como resultado la máxima confianza, se aceptará cualquier pedido.

<p><b>Ejemplo de entrada 1</b></p> <pre>3 1 15 37 42 8 25 77 2 1</pre>	<p><b>Ejemplo de salida 1</b></p> <pre>3 1 2</pre>
<p><b>Ejemplo de entrada 2</b></p> <pre>2 15 1 33 42</pre>	<p><b>Ejemplo de salida 2</b></p> <pre>1 2</pre>

## Problema H

# Hora de Correr

Vinicius se toma muy en serio su acondicionamiento físico, y todas las mañanas a las 6 de la mañana, llueva o haga sol, sea verano o sea invierno, él corre en una pista alrededor del lago. A lo largo de la pista de carreras hay  $N$  señales igualmente separadas. Para no desanimarse del ejercicio, Vinicius cuenta la cantidad de señales que ha pasado y comprueba si ha corrido al menos 10%, al menos 20%, ..., al menos 90% de su entrenamiento.

Ayudemos a Vinicius dada la cantidad de vueltas que desea correr en su entrenamiento y el número total de señales que hay a lo largo de la pista, a calcular la cantidad de señales que necesita contar para haber completado al menos 10%, 20%, ..., 90% de su entrenamiento.

Por ejemplo, supongamos que Vinicius quiere hacer un entrenamiento de 3 vueltas a la pista y en total hay 17 señales. Entonces, para asegurarse de que ha ejecutado al menos 30% del entrenamiento, necesita contar 16 señales. Para garantizar al menos 60%, necesita contar 31.

### Entrada

La entrada consiste de una única línea que contiene dos enteros,  $V$  y  $N$  ( $1 \leq V, N \leq 10^4$ ), donde  $V$  es el número deseado de vueltas en el entrenamiento y  $N$  es el número de señales en la pista.

### Salida

Su programa debe imprimir una única línea con nueve enteros que representan los números de señales que deben contarse para garantizar que se han corrido al menos 10%, 20%, ..., 90% del entrenamiento, respectivamente.

<b>Ejemplo de entrada 1</b> 3 17	<b>Ejemplo de salida 1</b> 6 11 16 21 26 31 36 41 46
<b>Ejemplo de entrada 2</b> 5 17	<b>Ejemplo de salida 2</b> 9 17 26 34 43 51 60 68 77
<b>Ejemplo de entrada 3</b> 3 11	<b>Ejemplo de salida 3</b> 4 7 10 14 17 20 24 27 30

## Problema I

# Interplanetario

Es el año 2306 y con el avance de la nanotecnología, los viajes interplanetarios son cada vez más accesibles. Bibika trabaja en la agencia de viajes interplanetarios más grande del universo y recibe diariamente a clientes interesados.

Los clientes de Bibika son exigentes y hacen varias demandas antes de cerrar su itinerario de viaje, como minimizar la distancia total recorrida. Pero las mayores restricciones están en las temperaturas de los planetas visitados en la ruta (excluyendo los planetas de origen y destino). La temperatura de un planeta, medida en grados Anidos, puede variar de  $10^9$  grados de Anidos negativos a  $10^9$  grados de Anidos positivos. Los clientes de Bibika provienen de planetas de climas variables y, en consecuencia, tienen diferentes preferencias de temperatura: algunos se preocupan por planetas muy fríos y otros por planetas muy calientes. Bibika necesita planear la ruta de viaje para evitarle a sus clientes cualquier molestia, incluso si la longitud total de la ruta no es lo más corta posible (o incluso si no hay una ruta: en este caso, Bibika simplemente informa a los clientes que el viaje es imposible).

Bibika te ha proporcionado la temperatura promedio histórica de cada uno de los  $N$  planetas así como las  $R$  rutas que conectan pares de planetas directamente (se garantiza que entre dos planetas hay como máximo una ruta directa), junto con sus respectivas distancias. Ella te entregará además las solicitudes de viaje de sus  $Q$  clientes. Cada solicitud consta de un planeta de origen  $A$ , un planeta de destino  $B$  y la restricción del cliente sobre la temperatura que deben tener los planetas intermedios en el viaje: cada cliente puede solicitar pasar por planetas que tengan una temperatura entre las  $K$  más frías o las  $K$  más calientes de entre las diferentes temperaturas de los  $N$  planetas.

Tu tarea es, para cada solicitud de viaje, encontrar la distancia más corta posible dadas las restricciones descritas, o decir que el viaje es imposible.

### Entrada

La primera línea de entrada contiene dos enteros  $N$  y  $R$  ( $2 \leq N \leq 400$  y  $0 \leq R \leq N \cdot (N - 1)/2$ ), que representan el número de planetas conocidos y cantidad de rutas directas entre ellos. El primer planeta está representado por el número 1, el segundo por el número 2, ..., hasta el  $N$ -ésimo por el número  $N$ . La segunda línea de entrada contiene  $N$  enteros  $T_i$  ( $-10^9 \leq T_i \leq 10^9$ ), que representa la temperatura promedio de cada uno de los planetas. Luego habrá  $R$  líneas, cada una con tres enteros  $X$ ,  $Y$  y  $D$  ( $1 \leq X, Y \leq N$  donde  $X \neq Y$  y  $1 \leq D \leq 10^3$ ), que representa una ruta directa de longitud  $D$  entre los planetas  $X$  y  $Y$ . Luego habrá un número entero  $Q$  ( $1 \leq Q \leq 10^5$ ), que representa el número de órdenes de viaje de los clientes. Finalmente, cada una de las siguientes  $Q$  líneas contendrá cuatro enteros  $A$ ,  $B$ ,  $K$  y  $T$  ( $1 \leq A, B, K \leq N$  donde  $A \neq B$  y  $T \in \{0, 1\}$ ), que representa a un cliente que quiere ir del planeta  $A$  al planeta  $B$  pasando en su ruta solo entre planetas tales que su temperatura está entre las  $K$  más frías si  $T = 0$  o las  $K$  más calientes si  $T = 1$ .

### Salida

Su programa debe imprimir una línea para cada cliente que contenga un número entero que represente la distancia de viaje total más corta entre los dos planetas dadas las restricciones del cliente, o  $-1$  si el viaje es imposible.

<b>Ejemplo de entrada 1</b> 7 9 -53 -180 456 420 -210 15 150 1 2 2 1 3 1 2 3 4 2 4 2 2 5 5 3 4 6 6 4 10 4 5 4 3 7 2 4 1 5 2 1 1 2 1 1 5 6 1 0 1 7 2 1	<b>Ejemplo de salida 1</b> 11 2 -1 3
<b>Ejemplo de entrada 2</b> 6 5 5 10 20 10 10 8 1 2 5 2 3 5 3 4 5 4 5 5 5 6 5 4 1 6 2 1 1 6 1 1 4 5 1 0 2 4 1 1	<b>Ejemplo de salida 2</b> 25 -1 5 10

## Problema J

# Juego de Cartas

El orden del día de las competencias de programación generalmente sigue el mismo patrón: calentamiento por la mañana, seguido de la hora del almuerzo, un tiempo de descanso, ajustes finales del entorno de la competencia y luego el inicio de la competencia.

En el tiempo de descanso, algunos competidores prefieren relajarse, otros prefieren socializar y algunos tienen la costumbre de jugar algún juego de cartas. Luciano y sus amigos disfrutaban jugando un juego conocido como “Vaso de agua”. Cansado de no ganar, Luciano quiere escribir un programa que, dadas las cartas iniciales de todos los jugadores (no me pregunte cómo lo sabe), determine si ganará o no. Si no va a ganar, puede inventar cualquier excusa y pedir no participar en esa ronda.

El juego funciona de la siguiente manera:

- El mazo usado tiene las cartas: “A23456789DQJK” (en este orden, del valor más bajo al más alto), donde se ignoran los palos. Además, el mazo tiene una carta extra más: el comodín.
- Los  $N$  competidores se sientan uno al lado del otro en un círculo. El competidor 1 está inmediatamente a la izquierda del 2, que está inmediatamente a la izquierda del 3, y así sucesivamente hasta completar el círculo con el  $N$ -ésimo concursante, que está inmediatamente a la izquierda del 1. Se hace un sorteo para elegir el competidor  $K$  que inicia el juego.
- En un juego con  $N$  competidores, habrá cuatro cartas de cada uno de  $N$  diferentes valores y un comodín. Al comienzo del juego, el competidor  $K$  recibe el comodín; Las cartas restantes se barajan y se reparten entre los jugadores para que cada jugador reciba cuatro de ellas.
- En cada ronda, el jugador en turno elige una de sus cartas y se la pasa al jugador a su derecha. El jugador que recibió la carta será el próximo jugador en turno.
- Decimos que un jugador está en un estado ganador si tiene exactamente cuatro cartas en la mano y todas son iguales. El juego termina tan pronto como al menos un competidor esté en el estado ganador. En este caso, el competidor con el número menor que se encuentre en estado ganador será declarado el ganador del juego .

La siguiente regla define la tarjeta que se pasará de un competidor al siguiente:

- El comodín no puede ser pasado justo después de haberlo recibido. Esto también se aplica al jugador inicial que recibió el comodín del repartidor de cartas justo antes de la primera ronda.
- El competidor, siempre que sea posible, pasará el comodín al siguiente competidor.
- Si no pasa el comodín, el competidor elegirá la carta que aparece menos en su mano y pasará a la siguiente. Si hay más de una carta que aparece con menos frecuencia, pasará, de entre ellas, la carta con menor valor según el orden descrito anteriormente.

Conociendo las reglas, ayuda a Luciano a escribir un programa que, dada la configuración inicial del juego, indique qué jugador será declarado ganador.

### Entrada

La primera línea de entrada contiene dos enteros  $N$  y  $K$  ( $2 \leq N \leq 13$  y  $1 \leq K \leq N$ ) representando, respectivamente, el número de competidores en el juego y el competidor que iniciará el juego. Cada una de las siguientes  $N$  líneas contiene cuatro caracteres, representando las cartas iniciales del  $i$ -ésimo competidor (con excepción del comodín).

## Salida

Su programa debe imprimir una única línea con un entero que representen el competidor que será declarado ganador del juego.

<b>Ejemplo de entrada 1</b> 2 1 33J3 JJJ3	<b>Ejemplo de salida 1</b> 2
<b>Ejemplo de entrada 2</b> 2 2 A2A2 22AA	<b>Ejemplo de salida 2</b> 2
<b>Ejemplo de entrada 3</b> 4 2 774Q JJQ7 44Q7 4QJJ	<b>Ejemplo de salida 3</b> 3
<b>Ejemplo de entrada 4</b> 3 1 JQAA JJJA QQQA	<b>Ejemplo de salida 4</b> 3

## Problema K

# Keep Calm y Vende Globos

Walter vende globos de puerta en puerta. Todos los días elige una calle en su ciudad y visita todas las casas de ahí, ofreciendo sus globos de colores.

Cada calle de la ciudad de Walter tiene el mismo número de casas en ambos lados, y todas las casas de la ciudad son del mismo tamaño. De esa manera, cada calle puede verse como una matriz de  $2 \times N$ , donde cada celda es una casa, y  $N$  es la cantidad de casas a lo largo de cada lado de la calle.

Después de elegir la calle del día, Walter visita cada casa en esa calle exactamente una vez. Puede comenzar su camino en cualquier casa, pero solo puede moverse entre casas adyacentes horizontalmente, verticalmente o en diagonal.

1	2	3	4	5	6
7	8	9	10	11	12

La tabla de arriba ilustra un ejemplo para  $N = 6$ . Después de visitar la casa número 1, Walter solo pudo pasar inmediatamente a las casas número 2, 7 y 8 (es decir, si no las ha visitado antes). Y después de visitar la casa número 11, la siguiente en el camino solo podría ser una de 4, 5, 6, 10 o 12.

Hoy, antes de salir de casa, Walter miró el mapa de la ciudad para contar el número de casas a cada lado de la calle elegida. Ahora quiere saber de cuántas maneras diferentes puede visitar todas las  $2N$  casas de la calle, siguiendo las reglas descritas. Dos formas de visitar las casas son diferentes si y solo si el orden de las casas varía: es decir, si hay dos casas  $A$  y  $B$  de modo que  $A$  se visita antes de  $B$  en un orden y  $B$  se visita antes de  $A$  en el otro.

### Entrada

La entrada consiste de una única línea que contiene un entero  $N$  ( $1 \leq N \leq 10^9$ ).

### Salida

Su programa debe imprimir una única línea con un entero que representan la cantidad formas distintas posibles de visitar todas las casas de la calle. Dado que este número puede ser muy grande, imprima el resultado de dividir este número por  $10^9 + 7$ .

<b>Ejemplo de entrada 1</b> 2	<b>Ejemplo de salida 1</b> 24
<b>Ejemplo de entrada 2</b> 3	<b>Ejemplo de salida 2</b> 96
<b>Ejemplo de entrada 3</b> 4	<b>Ejemplo de salida 3</b> 416
<b>Ejemplo de entrada 4</b> 61728	<b>Ejemplo de salida 4</b> 654783381

## Problema L

# Lanzando Monedas

Carla y Daniel han decidido jugar cara o cruz para decidir quién lavará los platos hoy. Jugarán con una de las monedas antiguas de la colección de Carla. Esto le preocupa a Daniel porque estas monedas están torcidas y desequilibradas: cuando se lanza una moneda, las probabilidades de obtener cara o cruz no son necesariamente iguales.

Carla conoce bien sus monedas y puede elegir una que maximice sus posibilidades de ganar. Por lo tanto, Daniel ideó un esquema para hacer el sorteo completamente justo, independientemente de la moneda elegida. Primero, a cada uno se le asignará un conjunto no vacío de cadenas binarias de tamaño  $N$ . Ninguna cadena puede pertenecer a ambos, y algunas cadenas pueden no estar incluidas en ninguno de los dos conjuntos. Por ejemplo, para  $N = 3$ , una forma válida de dividir cadenas sería:

- “010” y “110” para Carla;
- “001” y “011” para Daniel;
- “000”, “100”, “101” y “111” para ninguno de los dos.

Después de dividir las cadenas, Carla y Daniel jugarán la misma moneda  $N$  veces y escribirán la secuencia de resultados, donde cada cara equivale a un 0 y cada cruz equivale a 1. Si la cadena binaria resultante pertenece al conjunto de Carla, ella es la ganadora. Si pertenece al conjunto de Daniel, él es el ganador. Si la cadena no pertenece a ninguno de ellos, la moneda se arroja otras  $N$  veces para generar una nueva cadena. El proceso se repite tantas veces como sea necesario hasta que obtengan un ganador.

El funcionamiento adecuado de este esquema depende de la división de cadenas entre Carla y Daniel: la probabilidad de generar una cadena del conjunto de Carla debe ser igual a la probabilidad de generar una cadena del conjunto de Daniel. En otras palabras, digamos que  $P(S)$  sea la probabilidad de que una secuencia binaria  $S$  de longitud  $N$  sea generada por una secuencia de  $N$  lanzamientos de la misma moneda, posiblemente no balanceadas. El total de  $P$  para todas las cadenas en el conjunto de Carla debe ser el mismo que el total de  $P$  para todas las cadenas en el conjunto de Daniel.

Además de compartir las cadenas de manera justa, Carla y Daniel quieren evitar tener que repetir el lanzamiento de la moneda tanto como sea posible, por lo que quieren minimizar el número de cadenas que tampoco pertenecen. Dado  $N$ , determine la menor cantidad posible de cadenas sin asignar.

### Entrada

La entrada consiste de una única línea que contiene un entero  $N$ , el número de lanzamientos de monedas y la longitud de las cadenas binarias ( $2 \leq N \leq 10^{18}$ ).

### Salida

Su programa debe imprimir una única línea con un entero que representan el número mínimo de cadenas no usadas en la división.

<b>Ejemplo de entrada 1</b> 3	<b>Ejemplo de salida 1</b> 4
<b>Ejemplo de entrada 2</b> 5	<b>Ejemplo de salida 2</b> 4
<b>Ejemplo de entrada 3</b> 8	<b>Ejemplo de salida 3</b> 2

## Problema M

# Maratón Brasileño de Palomitas

El Maratón brasileño de comer palomitas es una competencia que se realiza anualmente para descubrir qué equipo es el más organizado, preparado y mejor capacitado en el arte de comer palomitas de maíz. Está organizado por la SBCp (Sociedad Brasileña de Comedores de Palomitas de Maíz), que se reúne periódicamente para discutir las reglas y el formato de la competencia.

La competencia consiste de  $N$  bolsas de palomitas de maíz colocadas una al lado de la otra, donde cada bolsa tiene una cantidad arbitraria de palomitas de maíz. Para mayor diversión, la competencia se lleva a cabo en equipos, cada uno compuesto por  $C$  competidores. Dado que el Maratón brasileño de palomitas es un evento serio que valora, sobre todo, la salud de los competidores, la comisión médica ha impuesto que cada competidor puede comer un máximo de  $T$  palomitas de maíz por segundo para evitar posibles malestares.

En su última reunión, la SBCp definió dos nuevas reglas para la edición de 2019:

- Cada competidor del equipo debe comer una secuencia contigua de bolsas de palomitas de maíz. Es perfectamente válido que un competidor no coma palomitas de maíz.
- Todas las palomitas de maíz en la misma bolsa deben ser comidas por un solo competidor.

El objetivo de la competencia es comer todas las palomitas de maíz en el menor tiempo posible, ya que los  $C$  competidores pueden comer en paralelo y cumplirán con todas las reglas impuestas por la SBCp.

### Entrada

La primera línea de entrada contiene tres enteros  $N$ ,  $C$  y  $T$  ( $1 \leq N \leq 10^5$ ,  $1 \leq C \leq 10^5$  y  $1 \leq T \leq 50$ ), que representa la cantidad de bolsas de palomitas de maíz, el número de competidores en el mismo equipo y la cantidad máxima de palomitas de maíz por segundo que un competidor puede comer. La segunda línea contendrá  $N$  enteros  $P_i$  ( $1 \leq P_i \leq 10^4$ ), que es la cantidad de palomitas de maíz en cada una de las bolsas de  $N$ .

### Salida

Su programa debe imprimir una única línea con un entero que representan la cantidad mínima de segundos que le toma al equipo comer todas las palomitas de maíz si se organizan de la mejor manera posible.

<b>Ejemplo de entrada 1</b> 5 3 4 5 8 3 10 7	<b>Ejemplo de salida 1</b> 4
<b>Ejemplo de entrada 2</b> 3 2 1 1 5 1	<b>Ejemplo de salida 2</b> 6
<b>Ejemplo de entrada 3</b> 3 2 1 1 1 5	<b>Ejemplo de salida 3</b> 5