

Enunciados de Problemas

Nivel Libres

1 de noviembre de 2024

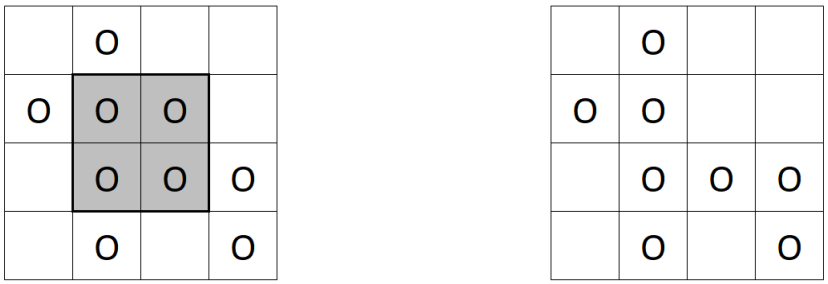
A. Agua Infinita

Tiempo límite: 1 segundo
Memoria límite: 256 megabytes

En el reconocido videojuego MinaCrack —donde los cracks minan metales preciosos en peligrosas cuevas— millones de jugadores alrededor del mundo construyen increíbles refugios y ambiciosos generadores de recursos. Desafortunadamente, muchas personas se han convencido de que MinaCrack es un videojuego diseñado únicamente para niños. Este problema pretende desafiar esa idea y eliminar este prejuicio.

En MinaCrack todo está construido utilizando bloques, que son la unidad mínima manipulable del videojuego. Dos bloques de agua se consideran conectados entre sí si y solo si uno de ellos es vertical u horizontalmente adyacente al otro. Cuando múltiples bloques de agua conectados se unen, adquieren la capacidad de formar patrones complejos. Estas estructuras se denominan *suministros de agua*.

Una de las muchas mecánicas de MinaCrack que los jugadores pueden aprovechar es la generación de suministros de agua infinitos. Un suministro de agua es infinito si y solo si dentro de su estructura contiene al menos un patrón cuadrado de dos bloques de agua de alto y ancho.



Dos suministros de agua, donde el carácter "O" indica la presencia de un bloque de agua. En la primera imagen, se visualiza un suministro de agua infinita y se destaca el patrón descrito. En la segunda imagen, se visualiza un suministro de agua que no es infinito.

En este problema, se le solicita que escriba un programa para determinar la cantidad de suministros de agua infinita en una disposición de bloques de *MinaCrack*.

Entrada

La entrada consta de múltiples lineas.

La primera línea contiene dos enteros n y m ($1 \leq n, m \leq 1000$), el alto y ancho de la disposición de bloques.

Las siguientes n líneas contienen m caracteres en mayúscula c_{ij} ($c_{ij} \in$ {"O", "X"}) , representando el tipo de bloque ubicado en la intersección entre la fila i y la columna j, donde el carácter "O" indica que el bloque es de agua y el carácter "X" indica que se trata de otro tipo de bloque.

Salida

La salida consta de una línea, la cantidad de suministros de agua infinita presentes en la disposición dada.

Ejemplos

entrada estándar	salida estándar
4 4 X O X X O O O X X O O O X O X O	1
4 4 X O X X O O X X X O O O X O X O	0
5 5 O O X X X O O X X X X O O O O X X O O O O X O X O	1

Nota

Si está utilizando el compilador integrado en *ZinjaI*, es posible que no pueda compilar su programa con $n = m = 1000$. Si este es el caso, realice pruebas en su entorno con valores menores, como $n = m = 100$, pero envíe la solución al juez con el valor correcto.

B. Bananas

Tiempo límite: 1 segundo
Memoria límite: 256 megabytes

El espectacular Coco -el mono araña- ha logrado infiltrarse en la celebración secreta del Gran Banquete de la Selva horas antes de que comience. Frente a él se encuentra una enorme variedad de platos exóticos de todos los tipos y colores. Sin embargo, lo que más le interesa a Coco son los r racimos de bananas. Como es un mono de refinados modales, Coco se dedicará a devorar cada racimo de bananas de principio a fin antes de pasar al siguiente. Además, Coco ha decidido que, para no sobrecargarse, cada minuto siempre comerá la misma cantidad de bananas, una cantidad que él mismo elige con mucho cuidado. Si en algún momento un racimo tiene menos bananas de las que ha decidido consumir, se comerá todas las que haya y no comerá más durante ese minuto.

A Coco le encantan las bananas, pero también es un mono prudente y no quiere sentirse mal después de comer. Por eso, siempre prioriza comer lo más lento posible sin exceder el tiempo disponible, es decir, busca la cantidad mínima de bananas por minuto que le permita terminar de comer todas las bananas antes de que lleguen los demás invitados.

Tu tarea es ayudar a Coco a encontrar esa cantidad mínima para que pueda disfrutar de todas las bananas y no ser sorprendido por el resto.

Entrada

La entrada consta de dos líneas.

La primer línea contiene dos enteros r y m ($1 \leq r \leq 10^4$, $r \leq m \leq 10^9$), que representan la cantidad de racimos y la cantidad de minutos que deben pasar antes de que lleguen el resto de los invitados.

La segunda línea contiene r enteros b_i ($1 \leq b_i \leq 10^9$), representando la cantidad de bananas del racimo i-ésimo.

Salida

La salida consta de una línea, la cantidad de bananas mínima que tiene que comer Coco por minuto para acabar todos los racimos antes de que lleguen los demás invitados.

Ejemplos

entrada estándar	salida estándar
4 8 7 5 10 3	4
5 5 21 29 10 17 5	29
5 6 21 29 10 17 5	21

Nota

En el caso que Coco disponga de dos racimos con 5 y 3 bananas y de 3 minutos antes de que lleguen los demás invitados, deberá comer a una tasa óptima de 3 bananas por minuto.

En el primer minuto, podrá comer 3 bananas del primer racimo, dejándolo con 2 restantes. En el segundo minuto, podrá comer las dos bananas restantes, no comiendo más hasta el tercer minuto, en el cual se comerá el segundo racimo. Finalmente, Coco podrá escapar satisfactoriamente ya que ha comido todas las bananas para el tercer minuto.

C. Caza Monedas

Tiempo límite: 1 segundo
Memoria límite: 400 megabytes

Bruno es fanático de los videojuegos. Hace unos días llegó al último nivel de su juego favorito, llamado Caza Monedas, pero no puede lograr ganarlo y necesita de tu ayuda.

El juego es muy sencillo y consiste simplemente en juntar un conjunto de monedas ubicadas en distintos puntos del plano cartesiano. Más específicamente, primero es posible visualizar dónde están ubicadas las monedas, luego se debe elegir un punto en el plano cartesiano donde comenzar a jugar, y luego una vez elegido el punto inicial, es posible moverse hacia la izquierda, derecha, arriba o abajo utilizando las flechas del teclado.

Para ganar cada nivel hay que lograr juntar todas las monedas utilizando la mínima cantidad de movimientos posibles, es decir que hay que elegir la posición inicial de forma óptima y luego apretar las flechas la menor cantidad de veces posible, logrando juntar todas las monedas.

Bruno es muy bueno en el juego, pero en el último nivel **está prohibido utilizar la flecha para moverse a la izquierda**, y esto lo tiene a maltraer. Él no quiere que le digas exactamente cómo ganar este último nivel, pero necesita que le ayudes un poco diciéndole cuál es la cantidad de movimientos que debe realizar para poder ganar.

Entrada

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

Luego siguen N líneas. La i-ésima de ellas contiene dos enteros X_i y Y_i ($-10^6 \leq X_i, Y_i \leq 10^6$), indicando la ubicación en el plano cartesiano de la i-ésima moneda.

Salida

Imprimir una línea con un entero, indicando la mínima cantidad de movimientos necesarios para juntar todas las monedas tras elegir el punto inicial de forma óptima.

Ejemplos

entrada estándar	salida estándar
5 2 3 6 5 4 7 8 9 2 2	17
4 0 1 1 0 -1 -1 3 3	10

Nota

En el segundo ejemplo, $(-1, -1)$ debe ser el punto inicial, y se pueden juntar todas las monedas con los siguientes 10 movimientos: `ADADaAAADD`. Donde el significado de las letras es el siguiente:

- `D`: moverse hacia la derecha, es decir incrementer la coordenada X.
- `A`: moverse hacia arriba, es decir incrementar la coordenada Y.
- `a`: moverse hacia abajo, es decir decrementar la coordenada Y.

D. Derrape

Autor:	Fernando Sebastián Lomazzi
Tiempo límite:	1 segundo
Memoria límite:	256 megabytes

Te diriges hacia Mar del Plata para disfrutar de unas vacaciones en familia. Todo va bien, comienzan su viaje con entusiasmo cantando una de sus canciones favoritas:

Vamos de paseo, pi pi pi,

En un auto feo, pi pi pi,

Pero no me importa, pi pi pi,

Porque llevo torta, pi pi ¡ruuum!

De repente, un pequeño carpincho cruza el camino y, al realizar una maniobra para evitarlo, el auto derrapa perdiendo el control. Te despiertas sin saber dónde se encuentran pero, afortunadamente, tu celular guardó las posiciones GPS durante el derrape.

Tienes disponibles tres ubicaciones: dos que indican tu posición antes del derrape y una que muestra tu posición actual, después de perder el control del vehículo. Sabes que las dos primeras ubicaciones corresponden a una trayectoria recta en dirección a Mar del Plata. Con esta información, tu objetivo es determinar si el derrape te desvió hacia la izquierda, continuó en línea recta o te llevó hacia la derecha en relación con la ruta que seguías.

Entrada

La entrada consta de múltiples líneas.

La primer línea contiene un entero t ($1 \leq t \leq 10^5$), la cantidad de casos de prueba.

Cada caso contiene seis enteros x_1, y_1, x_2, y_2, x_3 y y_3 ($-10^3 \leq x_1, y_1, x_2, y_2, x_3, y_3 \leq 10^3$), las ubicaciones ordenadas en forma temporal guardadas en tu celular.

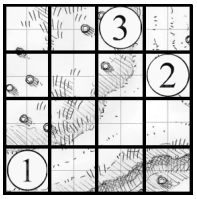
Salida

Por cada caso de prueba, deberás imprimir “IZQUIERDA”, “RECTO” o “DERECHA”, la dirección hacia la que ocurrió el derrape.

Ejemplos

entrada estándar	salida estándar
2 0 0 3 2 6 1 2 1 4 9 3 5	DERECHA RECTO
3 0 0 3 2 2 3 2 2 4 0 3 0 4 0 2 2 5 4	IZQUIERDA DERECHA DERECHA

Nota



Ejemplo del primer derrape del segundo caso de prueba

E. Césped

Tiempo límite: 1 segundo
Memoria límite: 256 megabytes

Fede y Juampi son dos hermanos que a veces se llevan bien y a veces se llevan mal. Pero cuando realmente discuten es cuando tienen que cortar el césped de su casa.

Para solucionar esto Mati, el padre de ambos, creó un juego para que ambos tengan que involucrarse en el corte del césped.

El patio de la casa de Fede y Juampi está dividido en cuadrados. El patio esta completamente sin el césped cortado. La cortadora de césped comienza en un cuadrado seleccionado por Mati que se considera que ya está cortado.

Los jugadores juegan por turnos, el primero en jugar es Fede, el siguiente jugador es Juampi. En cada turno, el jugador envía una señal a control remoto a la cortadora de césped que hace que la cortadora se mueva a uno de los cuadrados adyacentes de donde se encuentra la cortadora. La cortadora corta de manera inmediata el cuadrado de césped a donde fue movida y espera la señal del próximo jugador. Para que el movimiento hecho por un jugador sea válido la cortadora debe moverse a un cuadrado que no haya sido cortado. No es válido mover la cortadora afuera del patio y tampoco es válido moverla a un cuadrado que ya esté cortado. El jugador que no puede hacer un movimiento pierde y el otro gana.

Entrada

Cada caso contiene múltiples casos de prueba. La primera línea contiene un solo entero T ($1 \leq T \leq 100$) — el número de casos de prueba. Las siguientes T líneas contienen la información de cada caso de prueba. Cada caso de prueba contiene una sola línea con 4 enteros W, H, X, Y ($1 \leq W, H \leq 10^9, 1 \leq X \leq W, 1 \leq Y \leq H$). Estos valores expresan el ancho y el largo de el patio de la casa de Fede y Juampi, expresados en cuadrados, y las coordenadas de la posición inicial de la cortadora de césped.

Salida

Para cada caso de prueba, imprimí una sola línea con la palabra `Win` si Fede puede ganar sin importar que tan bien juegue Juampi, y si no imprimí `Lose`.

Ejemplo

entrada estándar	salida estándar
3 6 1 4 1 4 3 4 2 1 1 1 1	Win Win Lose

F. Fuga de memoria

Autor:	Lucio Rivera
Tiempo límite:	1 second
Memoria límite:	256 megabytes

Fernando es un programador muy famoso en su ciudad. La velocidad a la que este prodigioso joven programa es motivo de envidia para todos sus colegas, quienes tiemblan de miedo cada vez que lo ven acercarse —con teclado RGB y mouse óptico en mano— a las competencias de programación que se organizan en su ciudad. Si bien Fernando es un programador extremadamente veloz, la legibilidad de su código y la prevención de bugs no son aspectos que se le den muy bien.

En su tiempo libre, Fernando suele inspeccionar con especial orgullo el código que escribe en su trabajo. En una de estas inspecciones, se percató de que uno de sus programas tenía un peligroso bug: Fernando quería acceder k veces a la posición i-ésima de un vector indexado en cero; pero, incorrectamente, terminaba evaluando la expresión $v[_1\, v[_2 \cdots v[_{k-1}\, v[_k i]]]]$.

Mientras Fernando estaba haciendo la edición anterior de TecnoMate, descubrió un teorema que revolucionó el mundo del álgebra lineal y que lo llevo a convertirse en multimillonario. Por este motivo, ahora se encuentra vacacionando en *Miami*, lo que le impide corregir su error. Fernando le solicita a usted, a cambio de varios millones de dólares, que escriba un programa idéntico al defectuoso que codificó; pero que, dado un vector de entrada, advierta a sus colegas cuando se produzca una fuga de memoria.

Entrada

La entrada consta de múltiples líneas.

La primera línea contiene un entero n ($1 \leq n \leq 10^6$), que representa el tamaño del vector.

La siguiente línea contiene n números enteros v_i ($0 \leq v_i \leq 10^9$), representando el elemento en la posición i del vector.

La tercera línea contiene un entero q ($1 \leq q \leq 100$), representando la cantidad de consultas a responder. Cada consulta consiste en dos enteros k e i ($1 \leq k \leq 10^4$, $0 \leq i \leq 10^9$), la cantidad de veces y el índice inicial en el que el vector es evaluado, respectivamente.

Salida

Para cada consulta, deberá imprimir el resultado de la expresión descrita, si para su evaluación no se requiere acceder a una posición inválida del vector. Caso contrario deberá impirmir la frase *"Mi primera fuga de memoria"*.

Ejemplos

entrada estándar	salida estándar
1 0 1 1 0	0
6 2 0 4 3 5 7 4 1 5 3 1 2 3 5 0	7 4 3 Mi primera fuga de memoria

Nota

En la segunda consulta del caso de prueba, se debe evaluar la expresión $a[\,a[\,a[\,1]]]$. En la primera indexación, se obtiene $a[\,a[\,0]]$; en la segunda, $a[\,2]$ y finalmente en la tercera, se accede al número 4 ubicado en la tercera posición.

Si está utilizando el compilador integrado en *ZinjaI*, es posible que no pueda compilar su programa con $n = 10^6$. Si este es el caso, realice pruebas en su entorno con un valor menor, como $n = 10^5$, pero envíe la solución al juez con el valor correcto.

G. Seguridad en el Casino

Tiempo límite: 4 segundos
Memoria límite: 512 megabytes

El Casino de Santa Fe está teniendo muchos problemas de robos. Y aunque existen cámaras de seguridad en el mismo, los ladrones suelen escaparse sin mucho problema. Cansado de este problema, Daniel (el dueño del casino) decide contratar muchos guardias de seguridad. Pero no los distribuye de manera muy inteligente a través de la superficie del casino. Afortunadamente, la ubicación de los guardias se puede describir con coordenadas enteras en un plano 2D. Como los guardias de seguridad no están distribuidos de manera uniforme, es difícil para el supervisor de seguridad identificar que guardia está más cerca de un incidente. Y es aún más complicado teniendo en cuenta que el espacio en el casino está restringido por las máquinas y juegos. Estás limitaciones hacen que un guardia se mueva de una ubicación a otra de a pasos. En cada paso, un guardia puede cambiar cada una de sus coordenadas por *1, 0 o -1*. La distancia entre dos ubicaciones es igual a la cantidad mínima de pasos que tiene que hacer un guardia de un lugar a otro.

Teniendo la ubicación de los guardias y un conjunto de ubicaciones de incidentes, computar para cada incidente la menor distancia a un guardia. Esto ayudará a mejorar la seguridad en el casino.

Entrada

La primera línea de la entrada contiene dos enteros N y Q ($1 \leq N, Q \leq 3 \cdot 10^5$), el número de guardias y el número de incidentes. Luego le siguen N líneas. Cada una de esas líneas contiene dos enteros X e Y ($0 \leq X, Y \leq 5000$), que describen la ubicación de un guardia. Siguiente, Q líneas. Cada una de esas líneas contiene dos enteros A y B ($0 \leq A, B \leq 5000$), que describen la ubicación de un incidente de seguridad.

Salida

Para cada uno de los Q incidentes mostrar en la salida una línea conteniendo la distancia más corta a un guardia, medido en pasos.

Ejemplos

entrada estándar	salida estándar
2 3 0 1 4 0 5 0 4 3 1 2	1 3 1
2 4 0 0 3 3 1 1 0 3 1 2 3 3	1 3 2 0

Nota

Utilizar fast input/output

H. Tirada de la Muerte extrema

Autor: Fernando Sebastián Lomazzi
Tiempo límite: 1 segundo
Memoria límite: 256 megabytes

La Ciudadela de la Corona de Hielo se encuentra más gélida que nunca, ante ti y tu hermandad, se encuentra el Rey Exánime *Arthas Menethil*. Tras semanas, incluso meses de arduo trabajo finalmente tienen la oportunidad de cruzar miradas con el ex-paladín de la Mano de Plata. Ansiedad y nervios invaden al grupo, preparándolos mentalmente para la inminente batalla, no obstante, antes de enfrentar el desafío final, deciden tomarse un descanso para recobrar energías y fortalecer su determinación.

Mientras descansas, decides apostar un poco de tu oro con los miembros de tu hermandad, ya que esa montura mamut no se pagará sola. Decides jugar a la *Tirada de la Muerte*, un juego de dados que aprendiste con ellos.

La Tirada de la Muerte es un juego basado en dos jugadores, los cuales se turnan para arrojar un dado. El dado inicialmente tiene D caras y, a medida que se arroja el mismo, es reemplazado por uno con la misma cantidad de caras que el resultado obtenido. En el momento que uno de los jugadores obtenga un 1 como resultado de su tirada, pierde automáticamente el juego y debe entregar una cantidad de oro D a su rival.

Como además de un gran cazador, eres un genio de las matemáticas, no tardaste en darte cuenta que el juego se basa enteramente en el azar al aumentar la cantidad de caras del dado lanzado inicialmente, ya que la probabilidad de ganar se aproxima rápidamente al 50%. Por este motivo, decides optar por jugar su versión extrema.

Con base en la versión original, se complejiza el juego a través de la adición de dados. El juego comienza con 3 dados que se arrojarán de manera secuencial y cíclica por los jugadores, respetando las ideas de reducción de caras y condición de derrota.

Por ejemplo, si la disposición inicial de dados es de 4, 6 y 3 caras respectivamente, posibles juegos podrían ser:

- (4,6,3) el jugador A arroja un dado de 4 caras obteniendo un 2. (6,3,2) el jugador B arroja un dado de 6 caras y obtiene un 4. (3,2,4) el jugador A arroja un dado de 3 caras obteniendo un 3. (2,4,3) el jugador B arroja un dado de 2 caras obteniendo un 1, perdiendo el juego.
- (4,6,3) el jugador A arroja un dado de 4 caras obteniendo un 1, perdiendo el juego.

Como necesitas el oro, pero tampoco eres ningún necio, decides obtener la probabilidad de ganar siendo el jugador inicial. ¿Podrás lograrlo antes de que el descanso termine?

Entrada

La entrada consta de una línea "/roll D_1 D_2 D_3" siendo $2 \leq D_1\ D_2\ D_3 \leq 20$ la cantidad de caras iniciales de los dados que se lanzarán secuencialmente en la Tirada de la Muerte extrema.

Salida

La salida consta de una línea, la probabilidad de ganar el juego siendo el jugador inicial. La probabilidad deberá imprimirse con exactamente seis cifras decimales.

Ejemplos

entrada estándar	salida estándar
/roll 2 2 2	0.333333
/roll 6 2 8	0.597499
/roll 4 6 3	0.392438

I. Cuenta Impagable

Autor:	Lucio Rivera
Tiempo límite:	1 segundo
Memoria límite:	256 megabytes

El infame BurpChein, CEO de una compañía de ciberseguridad, ha cometido un grave error. En un acto de total imprudencia, BurpChein adquirió una licencia de software para su empresa en nombre de la universidad, que ahora debe pagar el costo de este imperdonable y, no precisamente barato, descuido.

Desafortunadamente para BurpChein, el departamento de sistemas, hará valer los 400 dólares que debió pagar por la licencia y ha decidido someter al vil estudiante a un juego denominado *secuencia 1 a 9*. El departamento determinó que, si BurpChein logra vencer a todos los profesores de sistemas en este juego, se le perdonará su error y podrá continuar estudiando en la universidad.

En una partida del juego, saldrá victorioso el jugador que, a partir de un vector de números enteros positivos indexados en cero, logre encontrar la secuencia de operaciones de menor costo total que transforme el vector dado en el vector [1, 2 , ... , 8, 9]. Las operaciones permitidas son las siguientes:

1. *Reemplazar*, el número v_i en la posición i por $v_i \ mod \ 10^{\phi(v_i)-k}$. Donde la función $\phi(x)$ es la cantidad de dígitos de un número entero positivo x y $0 \leq k < \phi(v_i)$. Costo: k.
2. *Eliminar*, el último dígito del número v_i en la posición i siempre y cuando $\phi(v_i) > 1$. Costo: 1.
3. *Eliminar*, el número v_i en la posición i. Costo i.
4. *Intercambiar*, el número v_i en la posición i con el número v_j en la posición j. Costo: 0.
5. *Copiar*, el número v_i en la posición i siempre y cuando v_i no haya sido alterado anteriormente por las operaciones 1 o 2. La copia es insertada al final del vector. Costo: i.

BurpChein sabe que, por algún motivo que aún desconoce, muchos profesores son imbatibles en este juego. Aterrorizado por la posibilidad de ser expulsado de la universidad, le solicita a usted que escriba un programa que le permita encontrar el costo total mínimo posible para asegurarse siempre salir victorioso en el desafío. BurpChein le promete un cargo en su compañía si logra ayudarlo, aunque le advertimos que quizás este sea un motivo más para no ayudarlo.

Entrada

La entrada consta de múltiples líneas.

La primera línea contiene un entero n ($2 \leq n \leq 10^5$), que representa el tamaño del vector inicial.

La segunda línea contiene n números enteros v_i ($1 \leq v_i \leq 10^5$), representando el número en la posición i del vector.

Se garantiza que todos los dígitos de 1 a 9 aparecen como mínimo una vez en algún número. Se puede demostrar que siempre existe una secuencia de operaciones ganadora.

Salida

La salida consta de una única línea, el costo mínimo posible para ganar la partida.

Ejemplos

entrada estándar	salida estándar
9 1 2 3 4 5 6 7 8 9	0
4 12 334 675 8889	18

J. Timba Kazaja

Tiempo límite: 3 segundos
Memoria límite: 200 megabytes

La Superliga de Kazajistán está siendo bastante aburrida últimamente, debido a que solo consiste de dos equipos. Para darle mas emoción, Mati decidió organizar un prode con sus amigos. Cada semana, los dos equipos se enfrentan entre si, y solamente uno es el ganador. Mati y sus amigos, en cada fecha, eligen quién va a ser el ganador del enfrentamiento, y todos los que acierten obtienen un punto.

Al ser el creador del prode, Mati, haciendo un poquito de trampa, puede elegir su ganador luego de ver que votó cada uno de sus amigos para la fecha actual.

En este momento, Mati va liderando la tabla, pero quiere saber por cuantos partidos más tiene asegurado el primer lugar (en el peor de los casos, dentro de todos los válidos). Para considerarlo líder, su puntuación tiene que ser mayor o igual al resto de los competidores.

Por ejemplo: Supongamos que inicialmente Mati tiene 5 puntos, y existen dos amigos que tienen 5 y 4 respectivamente. En este caso, Mati va a elegir el mismo equipo que el amigo que tiene 5, por lo tanto, si ese equipo pierde, los tres competidores tendrán 5 puntos al terminar la fecha. Para la segunda ronda, los dos amigos eligen equipos diferentes, y Mati no podrá evitar perder el liderazgo. Por lo tanto, solo logró mantener el primer lugar por una ronda.

Entrada

La entrada consiste de dos líneas:

- Una línea con un entero n ($3 \leq n \leq 10^5$), el número de amigos.
- Una línea con n enteros $a_1, \ldots, a_{n-1}, a_n$ ($0 \leq a_i \leq 10^{16}$ *para cada* i), los puntos de cada amigo que está jugando actualmente. El primer número siempre corresponde al puntaje de Mati, que inicialmente es estrictamente mayor o igual a cualquier otro puntaje.

Salida

La salida consiste de un solo número: la cantidad de fechas que Mati tiene garantizadas como líder.

Ejemplos

entrada estándar	salida estándar
3 5 5 4	1
5 7 3 2 4 1	6

K. Kamikaze

Tiempo límite: 1 segundo
Memoria límite: 256 megabytes

Fede está jugando su juego preferido, *Kamikaze*, en su *Play 2 Chipeada*. El objetivo del juego es determinar cuántos tipos de kamikazes pueden causar los cráteres observados en un mapa. Lo que distingue a un tipo de kamikaze es la dirección en la que lanza una de sus bombas, es decir, el vector V. Si está parado en la posición (X, Y) del mapa, lanza una bomba a $(X + X_V, Y + Y_V)$ y otra en su propia ubicación (X, Y). Ambas bombas explotan, generando cráteres.

Un tipo de kamikaze es válido si, al colocar una cierta cantidad de kamikazes del mismo tipo y hacer que exploten simultáneamente, se genera exactamente el mapa de cráteres.

Los mapas del juego se vuelven cada vez más complicados, con un gran número de cráteres. Fede necesita tu ayuda para determinar cuántos tipos de kamikazes pueden causar los cráteres observados. ¿Podrías darle una mano?

Entrada

La primera línea contiene un número entero T ($1 \leq T \leq 3$), la cantidad de casos de prueba. La primera línea de cada uno de los casos de prueba contiene un número entero C ($0 < C \leq 1000$), la cantidad de cráteres en el mapa. Las siguientes C líneas contienen dos enteros X_i e Y_i ($-1000 \leq X_i, Y_i \leq 1000$), la ubicación del i-ésimo cráter.

Salida

Por cada caso de prueba imprima una línea con un entero, la cantidad de diferentes tipos de kamikazes que pueden causar los cráteres observados en el mapa.

Ejemplo

entrada estándar	salida estándar
2 5 0 0 1 1 2 2 2 0 3 1 3 0 0 0 1 1 0	2 0

Nota

En el caso de prueba 1 los tipos de Kamikazes válidos son (1,1) y (-1,-1).