Torneo Argentino de Programación

- Departamento de Ciencias e Ingeniería de la Computación
  Universidad Nacional del Sur
- Facultad de Ciencias Exactas, Físico-Químicas y Naturales
  Universidad Nacional de Río Cuarto
- Facultad de Ciencias Exactas y Naturales
  Universidad de Buenos Aires
- Facultad de Informática
  Universidad Nacional de La Plata
- Facultad de Informática
  Universidad Nacional del Comahue
- Facultad de Ingeniería y Ciencias Hídricas
  Universidad Nacional del Litoral
- Facultad de Matemática, Astronomía y Física
  Universidad Nacional de Córdoba
- Unidad Académica Río Turbio
  Universidad Nacional de la Patagonia Austral

SESIÓN DE COMPETENCIA

5 de octubre de 2013

Este conjunto contiene 10 problemas; las páginas están numeradas de 1 a 20.

Información General

Salvo indicación en contrario, lo siguiente vale para todos los problemas.

Entrada

1. La entrada se debe leer de la entrada estándar (standard input).
2. La entrada contiene un único caso de prueba, el cual se describe utilizando una cantidad de líneas que depende del problema. No hay otros datos en la entrada.
3. Cuando una línea de datos contiene varios valores, éstos se separan utilizando exactamente *un* espacio entre ellos. Ningún otro espacio aparece en la entrada. No hay líneas en blanco.
4. No hay letras con tildes, acentos, diéresis, ni otros signos ortográficos (ñ, Ã, é, Ì, ô, Ü, ç, etcétera).
5. Todas las líneas, incluyendo la última, tienen la marca usual de fin de línea.

Salida

1. La salida se debe escribir en la salida estándar (standard output).
2. El resultado del caso de prueba debe aparecer en la salida utilizando una cantidad de líneas que depende del problema. No debe haber otros datos en la salida.
3. Cuando una línea de resultados contiene varios valores, éstos se deben separar utilizando exactamente *un* espacio entre ellos. Ningún otro espacio debe aparecer en la salida. No debe haber líneas en blanco.
4. No debe haber letras con tildes, acentos, diéresis, ni otros signos ortográficos (ñ, Ã, é, Ì, ô, Ü, ç, etcétera).
5. Todas las líneas, incluyendo la última, deben tener la marca usual de fin de línea.
6. Para escribir números reales, redondearlos al racional más cercano con la cantidad de dígitos luego del punto decimal que se especifica en el enunciado. El caso de prueba es tal que no va a haber empates en el redondeo.

Tiempo límite

1. El tiempo límite informado corresponde a la entrada descripta en el enunciado, y *no* a múltiples instancias de la misma.

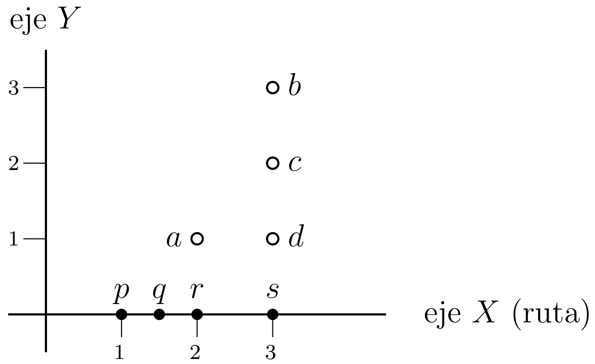
Problema A – Al costado del camino

Al costado del camino, hay palmeras, hay un bar, hay sombra, hay algo más. En este problema, nos interesan particularmente las palmeras.

Ana, Adán, Alan y Amanda organizaron un viaje. Mientras Ana y Adán se encargaban de nimiedades como hacer revisar el auto, preparar el equipaje y conseguir lugar para hospedarse, Alan y Amanda se dedicaban a lo más importante: estudiar los paisajes de palmeras a los que iban a tener acceso a lo largo del camino.

La ruta que ahora recorren es totalmente recta, y para los propósitos de este problema la modelamos como la recta $Y = 0$ del plano XY. Al costado de la ruta con coordenada $Y > 0$ hay palmeras, de modo que modelamos cada una de ellas como un punto diferente del plano XY con coordenada $Y > 0$. Alan y Amanda notaron que desde cada punto de la ruta son visibles determinadas palmeras, y que en general las palmeras visibles varían a lo largo de la ruta. Una palmera es visible desde un punto de la ruta si y sólo si el segmento que une ambos puntos no pasa por ninguna otra palmera.

En la siguiente figura los círculos sin relleno representan a las palmeras de la primera entrada de ejemplo, mientras que los círculos con relleno indican puntos posibles de la ruta.



Desde el punto p son visibles las palmeras a, b y d, ya que la palmera c queda oculta detrás de la palmera a. Desde el punto q son visibles las palmeras a, c y d, ya que la palmera b queda oculta detrás de la palmera a. Desde el punto r son visibles todas las palmeras. Desde el punto s son visibles las palmeras a y d, ya que las palmeras b y c quedan ocultas detrás de la palmera d.

Mientras Ana y Adán se turnan para manejar el auto, Alan y Amanda comentan entre ellos lo bueno que sería saber cuántas cantidades visibles de palmeras hay. Dado un conjunto de palmeras, un entero m es una cantidad visible de palmeras si y sólo si existe al menos un punto de la ruta (es decir, un punto con coordenada $Y = 0$) desde el cual exactamente m palmeras son visibles.

En el ejemplo ilustrado más arriba, 2, 3 y 4 son cantidades visibles de palmeras, como lo testifican respectivamente los puntos s, p y r de la ruta. Por otro lado, 0 y 1 no son cantidades visibles, porque desde todo punto de la ruta al menos 2 palmeras son visibles. Finalmente, ningún $m > 4$ es una cantidad visible, ya que hay 4 palmeras en total. Como resultado, en nuestro ejemplo hay 3 cantidades visibles de palmeras. Notar que si m es una cantidad visible de palmeras, podría haber más de un punto de la ruta que testifique tal situación; en el ejemplo eso ocurre con los puntos p y q para la cantidad visible 3, así como para infinitos otros puntos además del punto r para la cantidad visible 4.

Ana y Adán ya están cansados. Quieren que Alan y Amanda dejen las palmeras y al menos repartan la comida. Para eso es necesario que ustedes hagan un programa que calcule cuántas cantidades visibles de palmeras hay.

Entrada

La primera línea contiene un entero N que indica la cantidad de palmeras que están al costado de la ruta ($1 \leq N \leq 1000$). Cada una de las N líneas siguientes describe una palmera distinta utilizando dos enteros X e Y que representan las coordenadas de la palmera en el plano XY ($1 \leq X, Y \leq 10^5$). No hay dos palmeras en la misma posición (que coincidan en sus dos coordenadas).

Salida

Imprimir en la salida una línea conteniendo un entero que representa cuántas cantidades visibles de palmeras hay.

Entrada de ejemplo	**Salida para la entrada de ejemplo**

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

Entrada de ejemplo	**Salida para la entrada de ejemplo**

`7`
`2 1`
`3 1`
`4 1`
`1 2`
`3 2`
`5 2`
`3 3`
	`4`

Problema B – Boca de urna

Justo en este momento se están desarrollando las elecciones presidenciales en Nlogonia. Para que un candidato gane en primera vuelta debe obtener más votos que cada uno de los otros candidatos. Pero con eso no alcanza: además, debe obtener al menos el 45 % de todos los votos, o al menos el 40 % de todos los votos y al menos un 10 % más de votos que cada uno de los otros candidatos. Si ningún candidato gana en primera vuelta, se realiza una nueva elección en segunda vuelta.

Benicio es un periodista político de Nlogonia que siempre quiere tener la primicia. Por eso recolectó información de las encuestas de boca de urna, y quiere saber si de acuerdo a esos datos algún candidato gana en primera vuelta o, por el contrario, hay segunda vuelta. Benicio necesita decidir esto con urgencia antes de que alguien le saque la primicia. ¿Pueden ayudarlo?

Entrada

La primera línea contiene un entero N que indica la cantidad de candidatos ($2 \leq N \leq 10$). La segunda línea contiene N enteros V_i que representan las cantidades de votos obtenidos por cada uno de los candidatos ($0 \leq V_i \leq 1000$ para $i = 1, 2, \ldots, N$). Al menos un candidato obtuvo al menos un voto y no hay dos candidatos con la misma cantidad de votos.

Salida

Imprimir en la salida una línea conteniendo un dígito que representa si hay o no ganador en primera vuelta. Si hay ganador en primera vuelta el dígito debe ser "1"; caso contrario (es decir, si hay segunda vuelta) el dígito debe ser "2".

Entrada de ejemplo	Salida para la entrada de ejemplo
`2` `60 40`	`1`

Entrada de ejemplo	Salida para la entrada de ejemplo
`3` `16 28 21`	`1`

Entrada de ejemplo	Salida para la entrada de ejemplo
`3` `42 23 35`	`2`

Entrada de ejemplo	Salida para la entrada de ejemplo
`3` `297 302 401`	`2`

Página dejada casi totalmente en blanco en forma absolutamente intencional.

Problema C – Caperucita Roja

Érase una vez una niña muy alegre a la que llamaban Caperucita Roja porque siempre vestía una caperuza de color rojo. Caperucita disfrutaba mucho de sus paseos por el bosque, durante los cuales recolectaba frutos en su canastita para llevárselos a su abuelita, quien era conocida por preparar las tartas más deliciosas de toda la región. De lo que definitivamente no disfrutaba Caperucita era de los peligros del bosque, y en particular del malvado lobo que siempre estaba hambriento y al acecho.

Un día, Caperucita decide ir desde su casa hasta la de su abuelita, recolectando frutos del bosque en el camino y tratando de hacer el paseo de la manera más segura posible. La casa de Caperucita está en un claro del extremo oeste del bosque, la casa de su abuelita en un claro del extremo este, y dentro del bosque entre ambas hay algunos otros claros con árboles frutales. El bosque es muy espeso, por lo que la única forma de atravesarlo es utilizando senderos entre los distintos claros, los cuales por suerte Caperucita conoce muy bien. Para no perderse Caperucita se mueve siempre por senderos que la llevan a un punto estrictamente al este del punto en el que se encuentra. Para no ser atrapada por el lobo Caperucita encuentra imprescindible evitar una emboscada, y para ello siempre tiene en cuenta la cantidad de caminos distintos que la llevan desde su posición actual hasta la casa de su abuelita.

Un camino en el bosque es una sucesión de claros ordenados de oeste a este, de manera que cada claro está conectado con el siguiente por un sendero. Un camino hasta la casa de la abuelita es simplemente un camino cuyo último claro contiene dicha casa. Para cada claro, su *nivel de alternativas* es la cantidad de caminos que van desde él hasta la casa de la abuelita. Para cada camino, su *nivel de alternativas* es la suma de los niveles de alternativas de cada claro que compone ese camino. Para no ser capturada por el lobo, Caperucita desea encontrar el camino con máximo nivel de alternativas que empieza por su casa y termina en la de su abuelita. Como miembros de la sociedad Ayuda a Caperucita a Moverse (ACM), están llamados a ayudarla a encontrar ese valor máximo, y cuando lo hayan hecho, colorín, colorado, este problema habrá terminado.

Entrada

La primera línea contiene dos enteros N y S que indican respectivamente la cantidad de claros y de senderos en el bosque ($3 \leq N \leq 3 \times 10^4$ y $2 \leq S \leq 10^5$). Los claros son identificados por enteros diferentes entre 1 y N y están ordenados de oeste a este, de modo que si $1 \leq i < j \leq N$ el claro i está más al oeste que el claro j. La casa de Caperucita está en el claro 1, y la casa de la abuelita está en el claro N. Cada una de las S líneas siguientes describe un sendero utilizando dos enteros I y J que representan que hay un sendero entre entre el claro I y el claro J ($1 \leq I < J \leq N$). Hay al menos un camino desde la casa de Caperucita hasta la casa de la abuelita, y el máximo nivel de alternativas dentro del conjunto de tales caminos es menor o igual que 10^{18}.

Salida

Imprimir en la salida una línea conteniendo un entero que representa el máximo nivel de alternativas de un camino desde la casa de Caperucita hasta la de su abuelita.

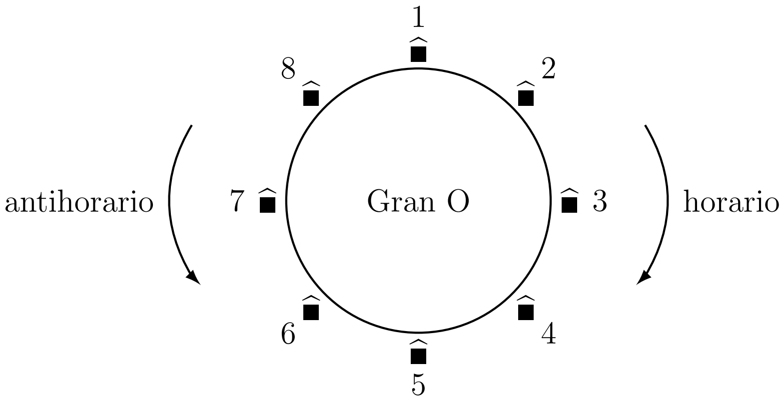
Entrada de ejemplo	Salida para la entrada de ejemplo
3 2 1 2 2 3	3

Entrada de ejemplo	Salida para la entrada de ejemplo
4 6 1 2 2 3 3 4 1 2 2 3 3 4	15

Entrada de ejemplo	Salida para la entrada de ejemplo
9 9 1 3 2 3 3 4 4 5 1 5 3 4 3 9 7 8 4 9	8

Problema D – ¿Dónde vemos el partido?

En el reino de Nlogonia hay un lago conocido como el "Gran O" por su forma perfectamente circular. A la orilla de este lago hay N casas, separada cada una de sus vecinas por una unidad nlogónica de distancia. Las casas están numeradas de 1 a N en sentido horario, como se muestra en la siguiente figura para el caso $N = 8$.



De este modo, si $i < j$, la distancia en sentido horario desde la casa número i hasta la casa número j es $j - i$, mientras que la distancia en sentido antihorario es $N - j + i$. La distancia desde una casa hasta sí misma es N en ambos sentidos.

Es bien sabido que la gente de Nlogonia es fanática del fútbol, así que cuando una familia se muda a una casa que está a la orilla del lago, es muy importante que pueda saber quiénes son sus vecinos más cercanos que son fanáticos del mismo equipo. Esto no siempre es fácil, siendo que puede haber muchas casas alrededor del lago, muchos equipos de fútbol distintos en Nlogonia, y gran cantidad de mudanzas. Dada una sucesión de M mudanzas, la gente que vive a la orilla del lago quiere darle la bienvenida a cada nueva familia que llega a una casa y decirle a qué distancia están las casas más cercanas cuyas familias simpatizan con su mismo equipo, tanto en sentido antihorario como horario. Notar que si no hay ninguna otra casa alrededor del lago cuya familia siga al mismo equipo que los recién llegados, la distancia en ambos sentidos sería N, ya que la casa más cercana sería la casa de la mudanza. ¿Quieren formar parte del comité de bienvenida?

En Nlogonia hay F equipos de fútbol, identificados por enteros diferentes entre 0 y $F-1$. Como no queremos que pierdan tiempo preguntando casa por casa a qué equipo se sigue allí, vamos a suponer que inicialmente la familia que vive en la casa número i es fanática del equipo número e_i, siendo este número generado en forma pseudo-aleatoria por la fórmula recursiva

$$e_1 = A \qquad \text{y} \qquad e_i = (B \times e_{i-1} + C) \text{ mód } F \quad \text{para } i = 2, 3, \ldots, N,$$

donde A, B y C son constantes y la expresión x mód y representa el resto de la división entera de x por y.

Entrada

La primera línea contiene dos enteros N y F que indican respectivamente la cantidad de casas alrededor del lago y la cantidad de equipos de fútbol que hay en Nlogonia

($3 \leq N \leq 10^5$ y $1 \leq F \leq 10^6$). La segunda línea contiene tres enteros A, B y C que determinan el equipo que sigue inicialmente la familia en cada casa según se describe en el enunciado ($0 \leq A, B, C < F$). La tercera línea contiene un entero M que indica la cantidad de mudanzas que se suceden a la orilla del lago ($1 \leq M \leq 10^5$). Cada una de las M líneas siguientes describe una mudanza utilizando dos enteros I y E que representan que a la casa número I se ha mudado una familia que sigue al equipo E ($1 \leq I \leq N$ y $0 \leq E < F$). Las mudanzas aparecen en el orden en que se suceden, y deben ser tenidas en cuenta por el comité para las posteriores bienvenidas.

Salida

Imprimir en la salida M líneas. La i-ésima línea debe indicar el resultado de la i-ésima mudanza utilizando dos enteros d_a y d_h que representan las distancias en unidades nlogónicas desde la casa involucrada en la mudanza hasta la primera casa cuya familia sigue al mismo equipo, en sentido antihorario y horario respectivamente.

Entrada de ejemplo	**Salida para la entrada de ejemplo**
`5 10`	`5 5`
`1 1 1`	`5 5`
`6`	`2 3`
`1 1`	`2 3`
`2 2`	`2 1`
`3 1`	`2 2`
`4 2`	
`5 1`	
`3 1`	

Problema E – Escape al escape

Un protocolo de comunicaciones es un conjunto de reglas para transmitir información en un sistema de comunicaciones. El trabajo de Elisa es escribir programas para implementar partes de dichos protocolos. Muchas veces es necesario transmitir secuencias de campos, y para saber dónde termina un campo y empieza el siguiente se intercala un separador entre cada par de campos consecutivos. Usar un separador simple como espacio, coma, o punto y coma, tiene el inconveniente de que a veces los campos a transmitir contienen esos mismos carácteres. La solución estándar para estos casos es insertar un carácter de "escape" justo antes de cada separador que aparece dentro de un campo, de modo de poder distinguirlo de un separador verdadero. Elisa cree que esta solución va a incrementar mucho la longitud de los datos a transmitir, por lo que decidió usar un separador lo suficientemente complejo como para que nunca esté contenido en los datos. Así piensa escapar de la ineficiente alternativa de escapar los separadores.

Para elegir el separador ideal, Elisa recopiló un *log*, que no es otra cosa que una cadena muy larga de carácteres representativa de los datos que su protocolo va a manejar. Luego de pensar un rato en el asunto, Elisa llegó a la conclusión de que cualquier cadena de carácteres no vacía que no aparezca dentro del log sería un separador aceptable para utilizar en su protocolo. Pero como a ella le interesa minimizar la longitud de los datos a transmitir, quiso saber la longitud mínima que un separador aceptable puede tener. De inmediato se puso a escribir un programa para calcular tal longitud, y lo está probando para el caso particular en el que tanto el log como los separadores aceptables contienen únicamente dígitos binarios ("`0`" y "`1`"). ¿Pueden ustedes anticipar los resultados?

Entrada

Una única línea que contiene el log, que es una cadena no vacía de a lo sumo 10^5 dígitos binarios.

Salida

Imprimir en la salida una línea conteniendo un entero que representa la longitud mínima de un separador aceptable para el log dado.

Entrada de ejemplo	**Salida para la entrada de ejemplo**
`011101001`	`3`

Entrada de ejemplo	**Salida para la entrada de ejemplo**
`100010110011101`	`4`

Entrada de ejemplo	**Salida para la entrada de ejemplo**
`11111`	`1`

Página dejada casi totalmente en blanco en forma absolutamente intencional.

Problema F – Flores

En Babilonia crecen unas plantas con flores que son muy apreciadas entre los habitantes. Florencio es un habitante de Babilonia que tiene un jardín con N plantas de esta especie y quiere recolectar algunas de sus flores. Como Florencio es bastante perezoso no quiere esforzarse mucho para recolectar las flores. Es así que ha decidido caminar hasta algún punto de su jardín y mediante un movimiento circular de su guadaña cortar una buena cantidad de plantas para luego recolectar sus flores. Florencio es muy hábil usando la guadaña y con ella abarca un círculo perfecto centrado en donde él está parado, lo que le permite cortar todas las plantas que estén en ese círculo, incluyendo su borde. Cuanto más alto levanta Florencio su guadaña, mayor es el radio del círculo que con ella abarca. Florencio quiere cortar al menos P plantas, pero su pereza es tal que quiere hacerlo levantando la herramienta lo menos posible.

Florencio consiguió una imagen satelital de su jardín donde aparecen todas sus plantas, y luego consiguió que alguien la convirtiera en una lista donde cada planta está representada por sus coordenadas en el plano XY. Ahora está sentado afuera, con su guadaña en mano, esperando que tu equipo le diga el radio mínimo de un círculo que abarca al menos P plantas.

Entrada

La primera línea contiene dos enteros N y P que indican respectivamente la cantidad de plantas que hay en el jardín y la cantidad mínima de plantas que Florencio quiere cortar ($1 \leq P \leq N \leq 500$). Cada una de las N líneas siguientes describe una planta distinta utilizando dos enteros X e Y que representan las coordenadas de la planta en el plano XY ($1 \leq X, Y \leq 10^5$). No hay dos plantas en la misma posición (que coincidan en sus dos coordenadas).

Salida

Imprimir en la salida una línea conteniendo un racional que representa el radio mínimo de un círculo que abarca al menos P plantas. Imprimir el resultado utilizando exactamente 4 dígitos luego del punto decimal, redondeando de ser necesario.

Entrada de ejemplo	**Salida para la entrada de ejemplo**

`3 2`
`10000 10000`
`10000 9999`
`9999 10000`
	`0.5000`

Entrada de ejemplo	**Salida para la entrada de ejemplo**

`2 1`
`1 1`
`10000 10000`
	`0.0000`

Página dejada casi totalmente en blanco en forma absolutamente intencional.

Problema G – Guerra

La guerra, evento solamente digno de aparecer en la literatura, películas o quizás problemas de competencias de programación, llegó al imperio de Nlogonia, que se enfrenta al vecino imperio de Quadradonia.

Los protocolos de guerra pautados por ambos imperios indican que la guerra será desarrollada en sucesivas batallas, y que en cada una de estas batallas se enfrentará un soldado distinto de cada imperio, de modo que cada soldado participará en exactamente una batalla. El imperio que gane la mayor cantidad de batallas ganará la guerra.

Cada imperio posee un ejército conformado por S soldados, y cada soldado posee cierta habilidad para el combate. En cada batalla entre dos soldados, aquel con mayor habilidad para el combate gana la batalla. Si ambos soldados poseen la misma habilidad, la batalla resulta un empate y técnicamente ningún bando se adjudica la victoria. Los espías de Nlogonia interceptaron información secreta sobre la habilidad de cada soldado de Quadradonia, y la reina de Nlogonia requiere la ayuda de ustedes para saber la cantidad máxima de batallas que puede ganar en la guerra si envía a sus soldados en el orden adecuado.

Entrada

La primera línea contiene un entero S que indica la cantidad de soldados de cada imperio ($1 \leq S \leq 10^5$). La segunda línea contiene S enteros Q_i que representan las habilidades de los distintos soldados de Quadradonia, en el orden en que se sucederán las batallas ($1 \leq Q_i \leq 10^9$ para $i = 1, 2, \ldots, S$). La tercera línea contiene S enteros N_i que representan las habilidades de los distintos soldados de Nlogonia, en un orden cualquiera ($1 \leq N_i \leq 10^9$ para $i = 1, 2, \ldots, S$).

Salida

Imprimir en la salida una línea conteniendo un entero que representa la cantidad máxima de batallas que puede ganar Nlogonia en la guerra.

Entrada de ejemplo	Salida para la entrada de ejemplo
3 2 1 1000000000 1 1 2	1

Entrada de ejemplo	Salida para la entrada de ejemplo
4 6 3 1 4 2 7 4 3	3

Página dejada casi totalmente en blanco en forma absolutamente intencional.

Problema H – Horacio y sus primos

A Horacio le gusta jugar escribiendo números naturales en el pizarrón que tiene en su dormitorio. Uno de sus juegos preferidos consiste en escribir primero un número n, luego la suma de todos los primos distintos que dividen a n, y así siguiendo hasta que el número escrito en el pizarrón sea un número primo. Por ejemplo, si Horacio comienza escribiendo el número $n = 90$, como $90 = 2 \times 3^2 \times 5$ el siguiente número escrito será $2 + 3 + 5 = 10$; a continuación, como $10 = 2 \times 5$, Horacio escribirá el número $2 + 5 = 7$; finalmente, como 7 es un número primo, aquí terminará el juego.

Formalmente, en este juego cada número natural $n \geq 2$ define una secuencia cuyo primer elemento es n, y cada nuevo elemento es la suma de todos los números primos que dividen al elemento anterior de la secuencia. El *orden* del juego es la posición del primer número primo de la secuencia, y coincide con la cantidad de números que quedan escritos en el pizarrón una vez que termina el juego. Para el ejemplo del párrafo anterior, con $n = 90$ el orden del juego es $K = 3$, ya que los números escritos serán 90, 10 y 7.

Ahora bien, no todos los juegos divierten por igual a Horacio, y en este caso resulta que él prefiere comenzar escribiendo un número n tal que el orden del juego correspondiente sea un valor K en particular. A Horacio le gustaría saber cuántos valores distintos de n entre A y B inclusive cumplen con esa condición, pero como no sabe programar necesita que alguien lo haga por él. ¿Ustedes pueden ayudarlo?

Entrada

La primera línea contiene un entero P que indica la cantidad de preguntas que Horacio quiere hacerles ($1 \leq P \leq 10^5$). Cada una de las P líneas siguientes describe una pregunta utilizando tres enteros A, B y K que representan que Horacio quiere saber cuántos valores distintos de n cumplen que $A \leq n \leq B$ y el orden del juego que comienza con n es K ($2 \leq A \leq B \leq 10^6$ y $1 \leq K \leq 10^6$).

Salida

Imprimir en la salida P líneas, cada una conteniendo un entero, con las respuestas a las preguntas hechas por Horacio en el orden en el que aparecen en la entrada.

Entrada de ejemplo	Salida para la entrada de ejemplo
`1` `90 90 3`	`1`

Entrada de ejemplo	Salida para la entrada de ejemplo

`5`
`2 9 1`
`2 9 2`
`800 810 4`
`999999 1000000 2`
`100000 1000000 1000000`
	`4`
	`4`
	`5`
	`2`
	`0`

Página dejada casi totalmente en blanco en forma absolutamente intencional.

Problema I – Isla del tesoro

Encontrar los tesoros escondidos hace siglos por piratas en las islas del Caribe no es tarea fácil, pero más difícil aún es vivir para contarlo. Esto es porque, como todos saben, los piratas tenían poderes sobrenaturales que usaban para maldecir a la persona que tomara su tesoro sin permiso.

Una maldición muy común entre los piratas más poderosos, y para la cual siempre conviene estar bien preparado, se conoce hoy como la *neblina mortífera*. Esta maldición consiste en que, cuando el tesoro de un pirata es encontrado, una especie de neblina venenosa se eleva desde la tierra hasta finalmente cubrir la isla entera. Todo ser vivo que sea alcanzado por la neblina muere al instante, algo especialmente poco deseable para quien acaba de encontrar un tesoro. La única forma de salvarse es volver al barco pasando siempre por zonas que no hayan sido todavía alcanzadas por la neblina, y así huir con la porción del tesoro que haya podido rescatarse. En este problema nos interesa saber cuál es el tiempo máximo que se puede demorar recolectando el tesoro de manera tal de poder volver al barco con vida.

Para simplificar el problema, vamos a considerar que una isla puede representarse con una grilla de F filas y C columnas, en la que la celda de la i-ésima fila y la j-ésima columna tiene altura H_{ij} sobre el nivel del mar. Más aún, vamos a suponer que el tesoro siempre se encuentra en la celda de la fila 1 y la columna 1, por ser esta la más alejada del único lugar donde puede atracar un barco, que es la celda en la fila F y la columna C. La neblina mortífera aparece al nivel del mar en el instante en que se descubre el tesoro, y sube por toda la isla a razón de una unidad de altura por segundo, de modo que luego de t segundos no se puede estar en ninguna celda de altura menor o igual que t. Para volver al barco se puede pasar de una celda a otra únicamente si comparten un lado, es decir que estando en determinada celda sólo es posible moverse horizontalmente a la celda anterior o siguiente en la misma fila, o verticalmente a la celda anterior o siguiente en la misma columna, pero no está permitido moverse en diagonal ni salir de los límites de la isla. Cada movimiento de una celda a otra demora un segundo.

Entrada

La primera línea contiene dos enteros F y C que indican respectivamente la cantidad de filas y columnas de la grilla que representa a la isla, la cual está formada por al menos dos celdas ($1 \leq F, C \leq 500$ y $F \times C \geq 2$). Cada una de las F líneas siguientes contiene C valores. En la i-ésima de estas F líneas, el j-ésimo valor es un entero H_{ij} que indica la altura de la celda en la fila i y la columna j ($1 \leq H_{ij} \leq 10^6$ para $i = 1, 2, \ldots, F$ y $j = 1, 2, \ldots, C$).

Salida

Imprimir en la salida una línea conteniendo un entero que representa el tiempo máximo en segundos que se puede demorar recolectando el tesoro de manera tal de poder volver al barco sin ser alcanzado por la neblina mortífera. Imprimir el número "`-1`" si es imposible volver al barco incluso emprendiendo el regreso en el instante en que se descubre el tesoro.

Entrada de ejemplo	Salida para la entrada de ejemplo
3 3 2 3 4 3 4 5 4 5 6	1

Entrada de ejemplo	Salida para la entrada de ejemplo
3 3 1 2 3 2 2 3 2 4 5	-1

Entrada de ejemplo	Salida para la entrada de ejemplo
3 2 1000000 1000000 1000000 1000000 1000000 314	310

Problema J – Jugando con piedras

A Jaimito le encanta jugar con las N piedras idénticas que le regalaron para su cumpleaños apilándolas para formar con ellas montañas de diversos tamaños. Su felicidad sería absoluta de no ser por su madre, Jimena, quien siempre le recuerda que al final de cada día llega el Tiempo de Acomodar las Piedras (TAP). Es en ese momento cuando Jaimito debe desarmar las montañas de piedras que con tanto esfuerzo construyó.

Como Jimena sabe cuánto le molesta a su hijo el TAP, le propone un juego para que la tarea se vuelva más divertida. Jaimito y su madre juegan alternadamente por turnos, con Jaimito comenzando el juego por ser el menor. Inicialmente hay varias montañas, cada una de las cuales está compuesta por una determinada cantidad de piedras. En su turno, cada jugador elige una montaña que tenga más de una piedra y la divide para formar dos montañas no necesariamente del mismo tamaño. El juego continúa hasta que alguno de los dos jugadores no puede realizar una jugada válida, momento en el que se declara a dicho jugador como el perdedor, y al otro como ganador.

Jaimito es muy inteligente, y se ha dado cuenta de que puede distribuir las N piedras para formar las montañas de manera estratégica, de modo tal que al comenzar a jugar con esa distribución él ya tenga la victoria asegurada en el TAP. Por cómo se desarrolla el juego, Jaimito no considera que dos distribuciones iniciales son diferentes si únicamente difieren en el orden en el que se encuentran las montañas. Esto significa que para considerar que dos distribuciones iniciales son diferentes, las mismas deben tener distinta cantidad de montañas, o si la cantidad de montañas es la misma entonces las piedras deben estar distribuidas de manera distinta dentro de las montañas. Por ejemplo, si Jaimito tiene $N = 4$ piedras, hay cinco distribuciones iniciales diferentes: cuatro montañas de una piedra cada una; dos montañas de una piedra y otra de dos piedras; una montaña de una piedra y otra de tres piedras; dos montañas de dos piedras; y, por último, una única montaña de cuatro piedras.

Como Jaimito no quiere que su madre se dé cuenta de que está haciendo trampa, quiere cambiar la distribución inicial de las N piedras todos los días. Está convencido de que hay muchas distribuciones iniciales diferentes que le aseguran ganar el juego, pero aún no sabe con certeza cuántas. Por ejemplo, en el caso de $N = 4$ piedras, Jaimito tiene sólo dos elecciones posibles: una única montaña de cuatro piedras, o dos montañas de una piedra y otra de dos piedras. La tarea de tu equipo en este problema es ayudar a Jaimito a contar de cuántas maneras diferentes puede distribuir sus N piedras en montañas de modo tal de tener asegurada la victoria en el juego contra Jimena. Así Jaimito podrá quedarse tranquilo sabiendo cuántos días puede ganar el juego sin que su madre sospeche de sus buenas intenciones.

Entrada

Una única línea que contiene un entero N, que indica la cantidad de piedras que posee Jaimito ($2 \leq N \leq 1000$).

Salida

Imprimir en la salida una línea conteniendo un entero que representa la cantidad de maneras diferentes de distribuir N piedras en montañas de modo tal que Jaimito tenga asegurada la victoria en el juego contra Jimena. Como la respuesta puede ser un número muy grande, sólo deben imprimir el resto de su división por $10^9 + 7$.

Entrada de ejemplo	**Salida para la entrada de ejemplo**
4	2

Entrada de ejemplo	**Salida para la entrada de ejemplo**
239	465766207