Torneo Rosarino de Programación 2026 - Avanzado

Cuaderno de Problemas

6 de Junio de 2026

Información General

Este cuaderno contiene 10 problemas. Verifique que su cuaderno está completo.

A) 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 mediante una o más líneas dependiendo 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.

B) 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 carácter de final-de-línea.

Problema A - Obra Pública - Difícil

En el país de Grafonia hay N ciudades numeradas del 1 al N y M caminos bidireccionales que las conectan. Cada camino tiene un estado de conservación:

- $0 \rightarrow$ en buen estado, se puede usar libremente.
- $1 \rightarrow$ en mal estado, hay que repararlo antes de poder usarlo.

Vos sos el ministro de obras públicas y querés viajar desde la ciudad 1 hasta la ciudad N. Estás apurado, por lo que querés recorrer el mínimo número de caminos durante el trayecto. Solo es posible pasar por caminos en buen estado, pero por suerte tenes K unidades de presupuesto para destinar a las reparaciones. Cada reparación cuesta una unidad de presupuesto, sin importar cuántas veces se use ese camino después.

Entrada

La primera línea contiene dos enteros N, M y K ($2 \leq N \leq 10^5$, $0 \leq M \leq 2 \cdot 10^5$, $0 \leq K \leq 20$), el número de ciudades, el número de caminos y las unidades de presupuesto, respectivamente.
Las siguientes M líneas contienen tres enteros u, v y e ($1 \leq u, v \leq N$, $e \in \{0, 1\}$), indicando que existe un camino bidireccional entre la ciudad u y la ciudad v con estado e.

Salida

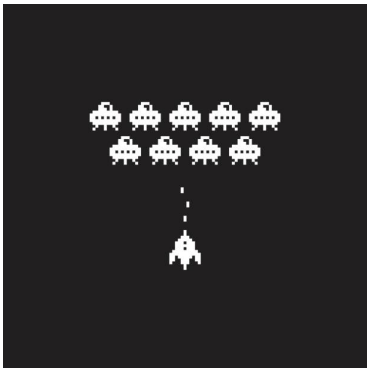
Imprimir un único entero que indique la mínima cantidad de caminos en mal estado que hay que reparar para ir de la ciudad 1 a la ciudad N, o imprimir -1 si no hay manera de llegar.

Ejemplo de entrada 1	Ejemplo de salida 1
5 7 1 1 2 0 2 3 0 1 3 1 1 4 1 4 3 0 4 5 1 3 5 0	2

Ejemplo de entrada 2	Ejemplo de salida 2
5 7 1 1 2 0 2 3 0 1 3 1 1 4 1 4 3 0 4 5 1 3 5 1	3

Problema B - Space Geometry

Estás programando el motor gráfico para un juego de naves retro estilo *Shoot 'em up* diseñado para consolas de 32 bits.
En este juego, cuando los enemigos son destruidos, dejan una estela de daño expansivo con forma perfectamente circular. Un problema conocido en el desarrollo de videojuegos 2D es calcular exactamente el área de la pantalla que está cubierta por estas explosiones, especialmente cuando dos explosiones ocurren muy cerca y sus áreas se superponen.



Para optimizar el *hitbox* del jugador, el motor del juego te envía las coordenadas en pantalla de dos explosiones simultáneas y sus respectivos radios de daño. Tu objetivo es escribir el algoritmo que calcule el área total cubierta por la unión de ambas explosiones.

Entrada

La primera línea contiene tres enteros X_A, Y_A y R_A ($-100 \leq X_A, Y_A \leq 100$, $1 \leq R_A \leq 100$), representando la coordenada y el radio de la primera explosión.
La segunda línea contiene tres enteros X_B, Y_B y R_B ($-100 \leq X_B, Y_B \leq 100$, $1 \leq R_B \leq 100$), representando la coordenada y el radio de la segunda explosión.

Salida

Imprimir el área exacta que abarcan ambas explosiones combinadas. El motor gráfico aceptará tu respuesta como válida si el error absoluto o relativo es como máximo 10^{-6}.

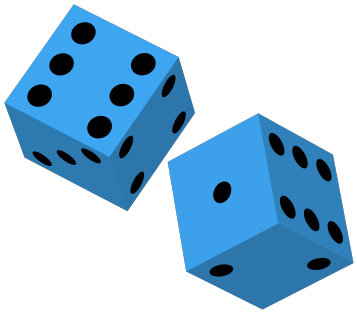
Ayuda para el formato de salida: Para imprimir un número de punto flotante r con precisión podés usar:
En Python: `print(f"{r:.10f}")`
En C++: `std::cout << std::fixed << std::setprecision(10) << r << "\n";`

Ejemplo de entrada 1	Ejemplo de salida 1
1 1 1 2 2 1	5.7123889804

Ejemplo de entrada 2	Ejemplo de salida 2
1 1 1 3 3 1	6.2831853072

Problema C - Tirando Dados

Alice y Bob están jugando una partida con dados. Alice tiene M dados, donde el i-ésimo dado tiene A_i caras. Por su parte, Bob tiene N dados, donde el i-ésimo dado tiene B_i caras. Todos los dados son justos y sus caras están numeradas desde el 1 hasta la cantidad de caras correspondientes.



En cada partida, ambos jugadores tiran todos sus dados simultáneamente y suman los resultados obtenidos. El jugador que consiga la mayor suma total se consagra como el ganador.
Sabiendo que Alice tiene una probabilidad P_A de ganar y Bob tiene una probabilidad P_B de ganar, tu tarea es determinar cuál de los dos tiene la ventaja estadística. ¿Quién de los dos tiene mayor probabilidad de ganar, o acaso tienen exactamente la misma probabilidad de hacerlo?

Entrada

La primera línea contiene dos enteros M y N ($1 \leq M, N \leq 10^5$), la cantidad de dados que tienen Alice y Bob, respectivamente.
La segunda línea contiene M enteros $A_1, A_2, \ldots, A_M$ ($1 \leq A_i \leq 10^9$), donde el i-ésimo número indica la cantidad de caras del i-ésimo dado de Alice.
La tercera línea contiene N enteros $B_1, B_2, \ldots, B_N$ ($1 \leq B_i \leq 10^9$), donde el i-ésimo número indica la cantidad de caras del i-ésimo dado de Bob.

Salida

Imprimir una única línea indicando el resultado del juego:

- `ALICE` si Alice tiene mayor probabilidad de ganar ($P_A > P_B$).
- `BOB` si Bob tiene mayor probabilidad de ganar ($P_A < P_B$).
- `EMPATE` si ambos tienen exactamente la misma probabilidad de ganar ($P_A = P_B$).

Ejemplo de entrada 1	Ejemplo de salida 1
8 1 4 4 4 4 4 4 4 4 6	ALICE

Ejemplo de entrada 2	Ejemplo de salida 2
2 2 6 4 4 6	EMPATE

Problema D - Concurso Maravilloso

Como una persona que ama las competencias, ahora necesitas participar en un maravilloso concurso. Este concurso tiene n problemas, cada uno con una puntuación máxima de 100. El problema i tiene a_i subtareas, y cada subtarea vale $100/a_i$ puntos. Se garantiza que a_i es un divisor de 100.
Varios concursantes participarán en este concurso. Supongamos que un concursante resuelve x_i ($0 \leq x_i \leq a_i$) subtareas del problema i; entonces su puntuación en ese problema será

$$x_i \times \frac{100}{a_i}$$

La puntuación total del concursante es la suma de las puntuaciones obtenidas en todos los problemas.
Para demostrar que el concurso es realmente maravilloso, debes determinar si es posible obtener **todos los puntajes enteros** desde 0 hasta $100 \times n$ (inclusive).
Formalmente, verifica si se cumple la siguiente afirmación:

- Para todo entero k tal que $0 \leq k \leq 100 \times n$, existe un arreglo x de longitud n ($0 \leq x_i \leq a_i$) que satisface

$$k = \sum_{i=1}^{n} x_i \times \frac{100}{a_i}$$

Entrada

La primera línea contiene un entero t ($1 \leq t \leq 100$), la cantidad de casos de prueba.
Para cada caso de prueba:

- La primera línea contiene un entero n ($1 \leq n \leq 10$), la cantidad de problemas.
- La segunda línea contiene n enteros $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 100$), la cantidad de subtareas de cada problema.

Se garantiza que cada a_i es un divisor de 100.

Salida

Para cada caso de prueba, imprime:

- `"Yes"` si es posible obtener todos los puntajes enteros desde 0 hasta $100 \times n$.
- `"No"` en caso contrario.

Ejemplo de entrada 1	Ejemplo de salida 1
5 2 100 20 2 10 10 3 50 100 25 4 1 2 5 20 10 100 1 2 4 5 10 20 25 50 100	Yes No Yes No Yes

Problema E - Kilos de desperdicio

Seba sabe exactamente la cantidad de alimento que van a necesitar sus gatos cada día. Sin embargo, se enfrenta a un gran problema: sus gatos son tan exquisitos que se niegan a comer cualquier alimento que no haya sido comprado ese mismo día, obligando a Seba a ir a la tienda diariamente.

Para complicar más las cosas, en la tienda solo venden el alimento en P tamaños distintos de bolsas, donde la i-ésima bolsa trae exactamente r_i kilos. Como no siempre puede comprar la cantidad exacta que necesita, hay días en los que Seba se ve obligado a tirar a la basura los kilos que sobran.

A Seba le gustaría diseñar un programa que, dadas Q consultas (una por cada día) con la cantidad N de kilos necesarios, calcule cuál es la mínima cantidad de alimento que será desperdiciada al comprar al menos N kilos.

Por ejemplo, si necesita comprar 25 kilos y solo hay bolsas de 2 y 15 kilos, la mejor opción es comprar una bolsa de 15 y seis bolsas de 2 (o cualquier otra combinación que sume 27 kilos), lo cual daría un desperdicio mínimo de solo 2 kilos.

Lamentablemente, Seba se va de viaje y no tiene tiempo de programarlo, ¡así que te pidió ayuda para resolverlo!

Entrada

La primera línea contiene dos enteros Q ($1 \leq Q \leq 10^5$) y P ($1 \leq P \leq 50$), la cantidad de días a consultar y la cantidad de tamaños de bolsas disponibles, respectivamente.

La segunda línea contiene P enteros distintos $r_1, r_2, \ldots, r_P$ ($1 \leq r_i \leq 100$), donde el i-ésimo número indica que existe una bolsa de r_i kilos de alimento.

Las siguientes Q líneas contienen un entero N ($1 \leq N \leq 5 \cdot 10^4$) cada una, indicando la cantidad mínima de kilos de alimento que se deben comprar ese día.

Salida

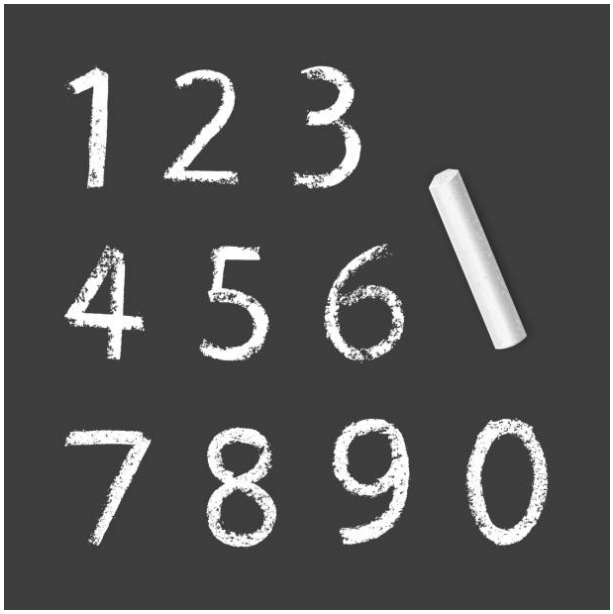
Para cada una de las Q consultas, imprimir una única línea con un entero que represente la mínima cantidad de kilos de alimento que será desperdiciada.

Ejemplo de entrada 1	**Ejemplo de salida 1**
2 3 2 15 7 10 5	0 1

Ejemplo de entrada 2	Ejemplo de salida 2
3 2 10 17 1 16 27	9 1 0

Problema F - Real Divido

En una pizarra están escritos los números enteros del 1 al N. Ana y Beto son muy competitivos y deciden utilizar estos números para jugar un juego por turnos.
En cada turno, el jugador activo debe elegir un número k que aún siga escrito en la pizarra y borrar tanto ese número k como todos sus divisores que permanezcan escritos. Por ejemplo, si un jugador elige el 6 y todos sus divisores están disponibles, borrará de la pizarra el 6, el 3, el 2 y el 1.



El jugador que en su turno no tenga números disponibles para elegir (porque la pizarra está vacía) pierde la partida.
Ana siempre es la primera en jugar. Si ambos jugadores juegan de forma completamente óptima, tu tarea es determinar quién será el ganador.

Entrada

La primera línea contiene un único entero T ($1 \leq T \leq 10^5$), la cantidad de casos de prueba.
Las siguientes T líneas contienen un único entero N ($1 \leq N \leq 10^9$) cada una, que representa el número máximo escrito inicialmente en la pizarra para ese caso de prueba.

Salida

Para cada caso de prueba, imprimir una única línea con el nombre del ganador: `ANA` si gana el primer jugador, o `BETO` si gana el segundo jugador.

Ejemplo de entrada 1	Ejemplo de salida 1
2 1 2	ANA ANA

Problema G - Poblado Colorido

El Pueblo Colorido es un popular destino turístico. Tiene $2n$ casas, numeradas del 1 al $2n$. Cada casa tiene uno de n colores, numerados del 1 al n. Casualmente, para cada uno de los n colores, exactamente dos casas están pintadas con ese color.

Hay $2n-1$ caminos bidireccionales en el Pueblo Colorido. Cada camino conecta dos casas distintas, y es posible llegar desde cualquier casa a cualquier otra usando estos caminos.
Catalina está planeando un viaje al Pueblo Colorido. Su tiempo es limitado, así que quiere elegir un conjunto S de n casas para visitar, con exactamente una casa de cada color. Sin embargo, como Catalina también necesita moverse entre las casas, el conjunto de casas que va a visitar debe ser **conexo**. En otras palabras, debe ser posible llegar desde cualquier casa en S a cualquier otra casa en S usando los caminos, visitando únicamente casas de S en el trayecto.
Ayudá a Catalina a encontrar un conjunto conexo S de n casas, una de cada color, o reportá que dicho conjunto no existe.

Entrada

Cada prueba contiene múltiples casos de prueba. La primera línea contiene el número de casos de prueba t ($1 \leq t \leq 10^5$). A continuación se describe cada caso de prueba.
La primera línea de cada caso de prueba contiene un único entero n ($1 \leq n \leq 10^5$).
La segunda línea contiene $2n$ enteros $c_1, c_2, \ldots, c_{2n}$, que denotan los colores de las casas del Pueblo Colorido ($1 \leq c_i \leq n$). Cada entero del 1 al n aparece exactamente dos veces en esta línea.
La i-ésima de las siguientes $2n-1$ líneas contiene dos enteros u_i y v_i, que denotan las casas conectadas por el i-ésimo camino ($1 \leq u_i, v_i \leq 2n$; $u_i \neq v_i$).
Se garantiza que la suma de n sobre todos los casos de prueba no excede 10^5.

Salida

Para cada caso de prueba, imprimí un único entero -1 si el conjunto requerido no existe. En caso contrario, imprimí n enteros distintos $s_1, s_2, \ldots, s_n$ en cualquier orden, que denoten un conjunto conexo S de n casas, una de cada color ($1 \leq s_i \leq 2n$). Si hay múltiples respuestas, imprimí cualquiera de ellas.

Ejemplo de entrada 1	Ejemplo de salida 1
2 4 1 3 1 3 4 4 2 2 1 6 5 3 2 4 7 1 7 5 5 8 2 5 3 1 1 2 2 3 3 1 2 2 3 3 4 4 5 5 6	2 3 5 7 -1

Problema H - No Me Alcanza La Ram

CyberLife quiere ser el monopolio de la IA en el mercado. Como están cansados de que las organizaciones LLM superen sus rendimientos en todos los Benchmarks y ya, hastiados de pensar en espacios multidimensionales y embeddings, tuvieron una idea tanto trivial, como brillante. ¿Qué hay de malo en comprar más RAM?
Supongamos que se espera que la IA sea superadora al resto de sus competidores, si al menos, ésta goza de M petabytes de RAM, (sí, esta narrativa pasa en 20 años). Además, CyberLife, tiene memorias RAM de alto y bajo voltaje. En su centro de datos, ellos preferirían tener, no solo algo que solo les garantice M petabytes, sino también, minimizar el consumo. Sí, no quieren ni pensar, ni gastar.
Luego de una exhaustiva búsqueda en el mercado, ellos están interesados en N memorias, con los siguientes parámetros: (llamemos R_i a la i-ésima memoria)

$$S_i : \text{Capacidad en Petabytes de } R_i$$
$$l_i : 1, \text{ si } R_i \text{ es de bajo voltaje; sino } 0$$

La empresa desea seleccionar un conjunto de memorias cuya capacidad total sea al menos M, minimizando la cantidad de memorias compradas.
Si hay múltiples formas de lograr esta suma utilizando la mínima cantidad de memorias posibles, se desea elegir la opción que maximice la cantidad de memorias de bajo voltaje.
Tu tarea es determinar una selección óptima de memorias.

Entrada

La primera línea contiene dos enteros N, la cantidad de memorias, y M, la capacidad requerida ($1 \leq N \leq 2 \times 10^5, 1 \leq M \leq 2 \times 10^{15}$).
Las siguientes N líneas contienen dos enteros S_i y l_i ($1 \leq S_i \leq 10^{10}, 0 \leq l_i \leq 1$), que representan respectivamente la capacidad de la memoria i y si es o no de bajo voltaje.
Se garantiza que la suma de las capacidades de todas las memorias es al menos M.

Salida

Imprimir 2 enteros r y w en la primera linea - la cantidad minima de memorias y la cantidad maxima de memorias de bajo voltaje que se pueden incluir en un conjunto optimo de tamaño r.
En la segunda línea imprimir r enteros distintos entre 1 y N, correspondientes a los índices de las memorias seleccionadas.

Ejemplo de entrada 1	**Ejemplo de salida 1**
4 10 3 1 7 0 5 1 4 1	2 1 4 2

Ejemplo de entrada 1	Ejemplo de salida 1
3 13 6 1 6 1 6 1	3 3 1 2 3

Problema I - Cocina con magia

Paula Paulina es una renombrada cocinera rosarina que alcanzó la fama gracias a sus recetas típicas y simples.
Sin embargo, últimamente empezó a atravesar una crisis: es tan famosa que sus ingredientes parecen no ser suficientes para toda la demanda que ella tiene. Paula acude a nosotros como programadores y nos dice algo entre dientes: *"Mirá, me quedé sin ingredientes y tuve que acudir a los poderes arcanos"*. Así es, Paula Paulina ya no cocina solo con ingredientes, sino también con magia.

Como quiere cocinar Carlitos con una cantidad absurda de ingredientes (10^5), nos ha enunciado los siguientes puntos clave:

- Para hacer sus Carlitos necesita N ingredientes distintos.
- De cada ingrediente i necesita una cantidad A_i de gramos.
- Para cada ingrediente i Paulina tiene B_i gramos en su cocina.
- Además, tiene K gramos de un ingrediente mágico que puede convertir en cualquier otro ingrediente.

Para hacerlo más claro, veamos la siguiente entrada:

i	1	2
A_i	8	9
B_i	5	6

$$K = 6$$

En este caso, vemos que Paulina solo puede hacer un Carlito debido a que cuenta con 5 gramos del ingrediente B_1, pudiendo complementarlo con 3 gramos del ingrediente mágico para llegar a 8 gramos, y luego, cuenta con 6 gramos del ingrediente B_2 pudiendo complementarlo con otros 3 gramos del ingrediente mágico, para llegar a 9 gramos.
Dado un N, un K, un arreglo $A_1, A_2, \ldots, A_N$ y otro arreglo $B_1, B_2, \ldots, B_N$, determine a cuántas personas puede saciar Paulina, si cada una se conforma con un Carlito.

Entrada

La primera línea contiene dos enteros N y K ($1 \leq N \leq 10^5$, $1 \leq K \leq 10^9$), la cantidad de ingredientes distintos a usar y la cantidad de gramos del ingrediente mágico, respectivamente.
La segunda línea contiene N enteros $A_1, A_2, \ldots, A_N$ ($1 \leq A_i \leq 10^9$ para cada $i = 1 \ldots n$), donde el i-ésimo número equivale a la cantidad de gramos que se necesitan del i-ésimo ingrediente para hacer un Carlito.
La última línea contiene N enteros $B_1, B_2, \ldots, B_N$ ($1 \leq B_i \leq 10^9$ para cada $i = 1 \ldots n$), donde el i-ésimo número equivale a la cantidad de gramos del i-ésimo ingrediente que hay disponibles.

Salida

Imprima una sola línea con un entero indicando la cantidad máxima de personas a las que Paulina puede satisfacer.

Ejemplo de entrada 1	**Ejemplo de salida 1**
2 6 8 9 5 6	1

Problema J - Speedrun de Zuma

A Agustin le gustan mucho los juegos de puzzles y los speedruns. El siguiente juego de puzzle del que planea hacer un speedrun es el Zuma. En el Zuma hay una fila de N gemas, donde la gema en la posición i tiene el color C_i. El objetivo del juego es destruir todas las gemas.
En un segundo Agustín puede elegir exactamente una subsecuencia continua de gemas de colores que sea un palíndromo y sacarla de la fila. Después de sacar esa subsecuencia, las gemas que quedan se juntan para volver a formar una fila continua, sin dejar huecos.
Para poder planear su speedrun Agustin quiere saber cual es la mínima cantidad de segundos que necesita para destruir todas las gemas
Una cadena (o subsecuencia) es un palíndromo si se lee exactamente igual de izquierda a derecha que de derecha a izquierda. En este caso particular, significa que el color de la primera gema tiene que ser igual al color de la última, el color de la segunda gema tiene que ser igual al de la anteúltima, y así sucesivamente.

Entrada

La primera línea de entrada tiene un solo número entero N ($1 \leq N \leq 500$), la cantidad total de gemas.
La segunda línea contiene N números enteros separados por un espacio, donde el número en la posición i es C_i ($1 \leq C_i \leq N$), que representa el color de la i-ésima gema en la fila.

Salida

Imprimí un único número entero que indique la cantidad mínima de segundos que se necesitan para destruir todas las gemas.

Ejemplo de entrada 1	**Ejemplo de salida 1**
3 1 2 1	1

Ejemplo de entrada 2	**Ejemplo de salida 2**
3 1 2 3	3

Ejemplo de entrada 3	**Ejemplo de salida 3**
7 1 4 4 2 3 2 1	2