



**Maratona
PPCI de
Programação
</SAET.26>**

Maratona PPCI de Programação SAET 2026

Caderno de Problemas

2026

Informações Gerais

Este caderno de tarefas tem 19 páginas (sem contar a capa), numeradas de 1 a 19. Verifique se o caderno está completo.

Entrada

- A entrada deve ser lida da entrada padrão (standard input).
- Todas as linhas da entrada, incluindo a última, terminam com o caractere fim-de-linha (`\n`).
- Quando a entrada contiver múltiplos valores separados por espaço em uma mesma linha, existirá exatamente um espaço em branco entre dois valores consecutivos na linha.

Saida

- A saída deve ser impressa na saída padrão (standard output).
- A saída deve seguir fielmente o formato especificado no enunciado do problema.
- Todas as linhas da saída, incluindo a última, devem terminar com o caractere fim-de-linha (`\n`).
- Quando vários valores devem ser impressos em uma mesma linha, deve haver exatamente um espaço em branco entre dois valores consecutivos na linha.

Problema A. A Vaca e o Camaleão

Tempo limite: 1000 ms
Memória limite: 256 MiB



Quando começou a estudar números binários, a vaca MU ficou fascinada com o assunto!

Ela escolheu dois números inteiros positivos A e B , com $A \leq B$, e escreveu a representação binária, sem zeros à esquerda, de todos os números de A até B (inclusive e nesta ordem) em seu quadro branco, formando uma string $S_{A,B}$. Depois, ela contou quantos dígitos 1 (um) apareceram em $S_{A,B}$ e anotou este número no quadro, o chamando de $f(A, B)$.

Como exemplo, considere $A = 5$ e $B = 10$. A string $S_{5,10}$ é 101110111100010011010. Como há 12 dígitos 1 (um) em $S_{5,10}$, tem-se que $f(5, 10) = 12$.

O camaleão Kurt resolveu pregar uma peça na vaca MU e, durante a noite, apagou várias informações que estavam no quadro! Os únicos valores que o camaleão deixou escritos são B e $f(A, B)$.

Sua tarefa é ajudar a vaca MU a lembrar qual era o valor de A .

Entrada

A primeira linha contém um inteiro T ($1 \leq T \leq 1000$), o número de casos de teste.

Cada caso de teste é dado por uma linha contendo dois inteiros B e f ($1 \leq B \leq 10^{16}$, $0 \leq f \leq 10^{18}$), o valor de B e de $f(A, B)$, respectivamente.

Saída

Para cada caso de teste, imprima uma linha com o valor de A .

Caso não exista um valor de A condizente com os valores de B e $f(A, B)$ dados na entrada, imprima a mensagem dada no exemplo de saída abaixo. Note que a mensagem deve ser impressa em *três* linhas.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
4 10 12 15 4 7 9 7 12	5 15 Eu aposto que voce nao vai digitar todas essas linhas e vai copiar e colar do PDF. Por isso, esta saida eh uma critica. 1

Problema B. Barril de Bananas

Tempo limite: 1000 ms
Memória limite: 256 MiB



Na Ilha dos Kongs tem N macacos que adoram muito bananas, mas tem o pavio muito curto. Todos na ilha tem um barril com a exata mesma quantidade de bananas que os barris dos outros macacos, pois se um macaco tem mais banana que outro é motivo para confusão!

Uma bela noite enquanto todos os macacos estavam dormindo, o crocodilo *King K. Rool* invadiu a ilha e roubou o barril de alguns macacos. Na manhã seguinte começou o maior alvoroço pois os macacos cujos barris foram roubados agora tinham menos bananas que outros.

Para evitar a briga de se intensificar ainda mais, o ancião da ilha, *Cranky Kong*, propôs a seguinte solução: pegar o total de B bananas que estão distribuídas entre todos os barris que ainda estão na ilha e redistribuí-las igualmente entre todos os macacos. Para isso, *Cranky Kong* precisa de sua ajuda para saber se é possível dividir B bananas entre N macacos de modo que todos tenham a mesma quantidade de bananas.

Entrada

A entrada consiste de dois números inteiros, N e B ($1 \leq N \leq B \leq 10^5$), o número de macacos e de bananas respectivamente.

Saída

A saída deve consistir de apenas um número inteiro, a quantidade de bananas que cada macaco vai ter se elas forem divididas igualmente entre eles, se não for possível esse rearranjo, imprima -1 .

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
3 9	3
Exemplo de entrada 2	Exemplo de saída 2
5 5	1
Exemplo de entrada 3	Exemplo de saída 3
9 14	-1

Problema C. Corredor Infinito

Tempo limite: 1000 ms
Memória limite: 256 MiB



Você está jogando o clássico *Super Mario 64*, e chegou em uma parte estranha: há um corredor com uma escada que parece ser infinita... Mas você descobriu um jeito de passar pela escada, explorando uma técnica avançada de pulo. Se você conseguir acertar um pulo bem específico K vezes seguidas, o final da escada finalmente aparece! Acertar o pulo, entretanto, não é nada fácil.

Considere uma sequência de N tentativas do pulo, feitas no instante 1, 2, etc., até o instante N . A sequência é representada por uma string S , de tamanho N , onde $S_i = o$ se a tentativa no instante i foi acertada, e $S_i = x$ se a tentativa no instante i foi errada. O final da escada aparece no primeiro instante em que se completa K acertos seguidos.

Processe Q operações; em cada operação, troque (de acerto para erro e vice-versa) a tentativa em um instante i dado. As mudanças feitas em uma operação se mantêm para as operações seguintes. Após cada operação, determine em que instante o final da escada aparece.

Entrada

A primeira linha contém três inteiros N , K e Q ($1 \leq K \leq N \leq 10^5$, $1 \leq Q \leq 10^5$). A próxima linha contém a string S inicial. As próximas Q linhas descrevem as operações a serem feitas. Cada operação é descrita por um inteiro i ($1 \leq i \leq N$) indicando o instante a se mudar a tentativa.

Saída

Após cada operação, imprima uma linha indicando em qual instante o final da escada aparece. Se o final da escada não aparecer em nenhum instante, imprima -1.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
9 3 6 ooxoxooxx 5 4 7 3 7 1	6 7 -1 3 3 7

Explicação do exemplo 1

Após a primeira operação, a string se torna ooxooooxx, e o final aparece no instante 6. Após a segunda operação, a string se torna ooxxooooxx, e o final aparece no instante 7. Após a terceira operação, a string se torna ooxxooxxx, e o final da escada não aparece.

Problema D. Dados Gêmeos

Tempo limite: 1000 ms
Memória limite: 256 MiB

Gabriel e André são irmãos gêmeos que cursam juntos Engenharia de Computação na UTFPR e também são grandes fãs de laranjas.

Todos os dias em que a sobremesa do Restaurante Universitário é laranja são dias mais felizes para os dois. Porém, certo dia, quando foram pegar a sobremesa, perceberam que havia apenas uma laranja restante.

Como meia laranja não seria suficiente para nenhum dos dois, eles decidiram fazer um sorteio para determinar quem ficaria com a fruta. Para isso, pegaram um par de dados que tinham consigo e lançaram os dois.

As regras são simples: se André tirar um número maior que Gabriel, ele fica com a laranja. Da mesma forma, se Gabriel tirar um número maior que André, ele fica com a laranja. Caso os dois tirem o mesmo número, o critério de desempate será o seguinte: se o número obtido for ímpar, Gabriel fica com a laranja; se for par, a laranja fica com André.

Dados os números obtidos nos dois dados, determine quem ficou com a laranja.

Entrada

A entrada consiste em dois inteiros A e G ($1 \leq A, G \leq 6$), o número que André e Gabriel tiraram no dado respectivamente.

Saída

A saída deve ser uma linha com a frase “Andre ficou com a laranja” ou “Gabriel ficou com a laranja”, de acordo com quem ganhou a laranja.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
1 5	Gabriel ficou com a laranja
Exemplo de entrada 2	Exemplo de saída 2
6 4	Andre ficou com a laranja
Exemplo de entrada 3	Exemplo de saída 3
3 3	Gabriel ficou com a laranja

Problema E. Encanador

Tempo limite: 1000 ms
Memória limite: 256 MiB



Odracir é um grande fã de jogos retrô e de computação. Em uma tarde, ele estava jogando seu jogo favorito, *Super Mario Bros.*, em seu Nintendinho.

No jogo, Mario pode estar em dois tamanhos: Alto ou Baixo. Quando Mario está Alto e recebe dano de um inimigo, ele diminui de tamanho e passa a ficar Baixo. Caso Mario já esteja Baixo ao receber dano, ele morre.

Mario também pode encontrar um cogumelo, que faz com que ele cresça. Assim, caso esteja Baixo, ele passa a ficar Alto. Se já estiver Alto, o cogumelo não altera seu tamanho.

Além disso, Mario pode encontrar uma estrela, que o torna invulnerável a danos pelos próximos 5 obstáculos. Enquanto estiver invulnerável, inimigos não causam dano a Mario, mas os obstáculos ainda são contabilizados.

Certo dia, devido a um curto-circuito em seu console, o *save* do jogo de Odracir foi corrompido. Agora ele não sabe qual foi o destino de Mario na fase em que estava. Felizmente, usando suas habilidades computacionais, conseguiu recuperar parte do arquivo corrompido, contendo apenas a sequência de obstáculos que Mario encontraria ao longo da fase.

Cada obstáculo pode ser:

- um inimigo, que causa dano a Mario;
- um cogumelo, que faz Mario crescer caso ele esteja Baixo;
- uma estrela, que torna Mario invulnerável a danos aos próximos 5 obstáculos.

Considere que Mario começa a fase Alto e que os obstáculos são encontrados na ordem em que aparecem na sequência.

Dada uma string representando todos os obstáculos da fase, determine se Mario morrerá ou sobreviverá à fase. Caso sobreviva, determine também se ele terminará a fase Alto ou Baixo.

Entrada

A entrada consiste de apenas uma string S representando os obstáculos da fase com pelo menos 1 e no máximo 10^5 letras, cada letra pode ser D , C ou E , representando um obstáculo de *Dano*, *Cogumelo* e *Estrela* respectivamente.

Saída

A saída deve ser uma linha com uma das seguintes 3 palavras: “MORTO” caso Mario morra no

percurso, “ALTO” caso Mario sobreviva com sua estatura inicial, ou “BAIXO” caso Mario sobreviva com a estatura menor que sua inicial.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
DCCEDCEDDCDCDC	ALTO

Explicação do exemplo 1

Observe um fluxograma representando a mudança no estado do Mario a cada obstáculo do exemplo:

$$\begin{array}{cccccccccccccccccccc} \text{Alto} & \xrightarrow{\text{D}} & \text{Baixo} & \xrightarrow{\text{C}} & \text{Alto} & \xrightarrow{\text{C}} & \text{Alto} & \xrightarrow{\text{E}} & \text{Alto} & \xrightarrow{\text{D}} & \text{Alto} & \xrightarrow{\text{C}} & \text{Alto} & \xrightarrow{\text{E}} & \text{Alto} & \xrightarrow{\text{D}} & \text{Alto} \\ & & & & & & & & & & & & & & & & & \xrightarrow{\text{D}} & \text{Alto} & \xrightarrow{\text{C}} & \text{Alto} & \xrightarrow{\text{D}} & \text{Alto} & \xrightarrow{\text{C}} & \text{Alto} & \xrightarrow{\text{D}} & \text{Baixo} & \xrightarrow{\text{C}} & \text{Alto} \end{array}$$

Exemplo de entrada 2	Exemplo de saída 2
ECCDCDCDCD	BAIXO

Exemplo de entrada 3	Exemplo de saída 3
CDDEDCCEDCCDCDEDDCEDC	MORTO

Explicação do exemplo 3

Neste exemplo no terceiro obstáculo Mario já está baixo e recebe dano, logo ele morre neste momento.

Observações

Caso Mario morra ele permanece neste estado mesmo que ainda há mais obstáculos a frente de onde ele recebeu dano.

Problema F. Fazendinha

Tempo limite: 1500 ms
Memória limite: 384 MiB

Pete tem uma pequena fazenda onde ele cultiva e colhe batatas.

A fazendinha de Pete é descrita como uma matriz de N linhas e M colunas. Na posição (i, j) da fazenda ($1 \leq i \leq N, 1 \leq j \leq M$) existem $B_{i,j}$ batatas maduras, prontas para a colheita.

Pete tem uma colheitadeira mágica, que pode ser acionada uma única vez. Ao acionar a colheitadeira em uma posição (l, c) e configurá-la para uma distância não negativa D , serão colhidas, de uma só vez, todas as batatas em todas as posições (i, j) tais que $|i - l| + |j - c| \leq D$.

Pete precisa colher pelo menos T batatas ao todo. Pete pode acionar a colheitadeira na posição que quiser, mas, para economizar com os gastos de seu uso, precisa configurá-la com a menor distância D possível.

Qual é a menor distância D necessária para configurar a colheitadeira de forma que pelo menos T batatas sejam colhidas?

Entrada

A primeira linha contém três inteiros N , M e T ($1 \leq N, M \leq 1000, 1 \leq T \leq 10^8$), o número de linhas e de colunas da fazenda, e o número mínimo de batatas a serem colhidas.

As próximas N linhas contém M inteiros $B_{i,j}$ cada ($0 \leq B_{i,j} \leq 99$), indicando o número de batatas em cada posição da fazenda.

Saída

Se não for possível colher pelo menos T batatas, imprima -1.

Caso contrário, imprima o menor valor de D necessário para se colher pelo menos T batatas.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
5 4 70 3 0 2 4 8 5 7 1 4 7 4 9 8 5 8 0 2 9 1 5	2

Explicação do exemplo 1

A figura abaixo apresenta o primeiro exemplo de entrada. Ao acionar a colheitadeira na posição $(3, 2)$, com $D = 2$, Pete consegue colher 74 batatas ao todo. É impossível colher pelo menos 70 batatas com $D < 2$.



Exemplo de entrada 2

Exemplo de saída 2

2 2 8 0 1 2 3	-1
---------------------	----

Problema G. Grande Dúvida

Tempo limite: 1000 ms
Memória limite: 256 MiB



Após uma feroz batalha contra o malvado dr. Robotnik, Sonic ficou gravemente ferido e foi internado no hospital de Mobius. Assim que soube da notícia, seu amigo Knuckles foi lhe visitar.

Vendo o estado do herói, Knuckles resolveu que chegou a hora de tirar sua grande dúvida: “Sonic me diga... me diga onde fica o **h** em **corinthians**!”. No que Sonic respondeu: “**h** é a sétima letra de **corinthians**, portanto, fica na posição 7”.

Knuckles então resolveu tirar outra dúvida: “E onde fica a letra **C** na palavra **P**?”. Mas desta vez, Sonic não teve forças para responder. Por isso, responda você a grande dúvida de Knuckles.

Entrada

A primeira linha contém **C**, uma letra minúscula.

A segunda linha contém a palavra **P** ($1 \leq |P| \leq 10^5$), sem espaços e contendo apenas letras minúsculas.

Saída

Imprima a posição da letra **C** na palavra **P**. Se a letra aparecer em mais de uma posição, imprima apenas a primeira em que aparece. Se a letra não aparecer em **P**, imprima -1 .

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
h corinthians	7
Exemplo de entrada 2	Exemplo de saída 2
h athlehticoh	3
Exemplo de entrada 3	Exemplo de saída 3
x hunterxhunter	7
Exemplo de entrada 4	Exemplo de saída 4
p saet	-1

Problema H. Horário

Tempo limite: 1000 ms
Memória limite: 256 MiB

Tanaka ganhou de presente de aniversário um relógio analógico, mas percebeu que ele não funcionava corretamente. Depois de observá-lo por um tempo, descobriu que sua função não era mostrar as horas, e sim ser um quebra-cabeça matemático. Tanaka adorou o presente, pois ama quebra-cabeças.

O desafio funciona da seguinte forma: o ponteiro das horas (desconsidere os demais ponteiros) começa apontando para as 12h, em um mostrador cíclico numerado de 1 a 12. Um painel do relógio exibe uma hora H e, em seguida, dois números X e Y .

O quebra-cabeça consiste em duas perguntas independentes, ambas partindo do ponteiro na posição 12: é possível, movendo-o exatamente X vezes (cada movimento avança o ponteiro em uma hora, podendo ser em sentido horário ou anti-horário), fazê-lo parar sobre a hora H ? E, separadamente, é possível fazer o mesmo movendo-o exatamente Y vezes?



Entrada

A entrada consiste em três inteiros H , X e Y ($1 \leq H \leq 12$, $1 \leq X, Y \leq 10^{18}$): a hora que se deseja alcançar e as quantidades de movimentos independentes a serem realizados com o ponteiro, respectivamente.

Saída

Imprima “SIM” caso seja possível levar o ponteiro, inicialmente na posição 12, até a hora H tanto realizando X movimentos quanto Y movimentos; caso contrário, imprima “NAO”.

Exemplos

Exemplo de entrada 1

6 6 6

Exemplo de saída 1

SIM

Explicação do exemplo 1

Neste exemplo podemos mover o ponteiro 6 vezes seguidas em qualquer sentido de modo que ele fique exatamente nas 6h.

Exemplo de entrada 2**Exemplo de saída 2**

7 15 4	NAO
--------	-----

Explicação do exemplo 2

Neste exemplo para 4 movimentos é impossível chegar nas 7h pois a menor distância entre 7 e 12 no relógio é 5.

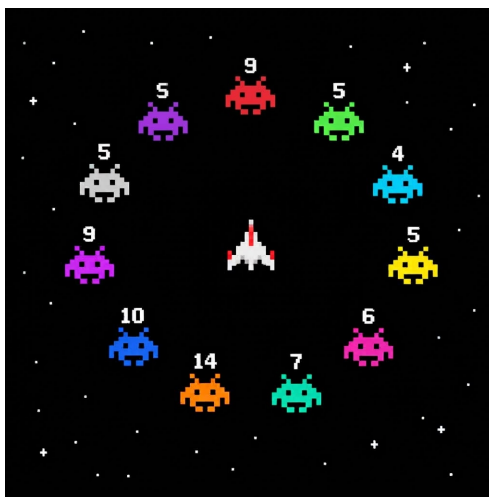
Exemplo de entrada 3**Exemplo de saída 3**

1 676767676767 676767676	NAO
--------------------------	-----

Problema I. Invasores Espaciais

Tempo limite: 1000 ms
Memória limite: 256 MiB

Kurt estava em uma missão espacial em sua nave quando foi atacado por uma tropa de N alienígenas invasores! Os alienígenas fizeram um cerco ao redor da nave de Kurt para impedir que ele consiga escapar para seu planeta natal. O plano deles é fazer com que Kurt fique preso em sua nave até que eventualmente fique sem oxigênio.



O que os alienígenas não contavam é que a nave de Kurt possui um poderoso mecanismo de defesa chamado *Raio Pulverizador*. Kurt conhece as defesas dos alienígenas. Cada alienígena possui um escudo de defesa de um determinado tipo. O i -ésimo alienígena possui um escudo do tipo A_i , onde A_i é um número inteiro.

Apesar de poderoso, o Raio Pulverizador possui algumas limitações. O raio consegue pulverizar um conjunto de alienígenas de, no máximo, K tipos distintos de escudo. Caso o disparo atinja um conjunto de alienígenas que possuam mais de K tipos distintos de escudo, o raio irá ricochetear e retornar para a nave de Kurt, destruindo-a. Além disso, o raio só pode ser disparado uma única vez. Depois de atingir o primeiro alienígena, o raio deve continuar seu trajeto atingindo apenas alienígenas adjacentes ao último alienígena atingido. Dessa forma, o disparo deve atingir uma sequência contínua de alienígenas, sem poder pular nenhum alienígena entre dois alvos consecutivos. O disparo pode seguir em qualquer uma das duas direções ao redor do círculo.

Dado o posicionamento dos alienígenas e os tipos de seus escudos, determine a maior quantidade de alienígenas que Kurt pode pulverizar com um único disparo sem destruir sua própria nave. Formalmente, ajude Kurt a encontrar o tamanho da maior sequência contínua de alienígenas adjacentes que possui no máximo K tipos distintos de escudo.

Entrada

A primeira linha contém dois inteiros N e K ($1 \leq N, K \leq 10^5$). A segunda linha contém N inteiros $A_1, A_2, \dots, A_N$ ($1 \leq A_i \leq 10^5$). O valor A_i representa o tipo de escudo do i -ésimo alienígena. Os alienígenas estão dispostos em círculo, de modo que os alienígenas 1 e N são adjacentes.

Saída

Imprima um único inteiro representando a maior quantidade de alienígenas que Kurt pode pulverizar em um único disparo do Raio Pulverizador.

Exemplos

Exemplo de entrada 1

11 3 5 9 5 4 5 6 7 14 10 9 5

Exemplo de saída 1

7

Explicação do exemplo 1

Neste exemplo, ilustrado anteriormente no enunciado, observamos que a sequência de alienígenas com tipos de escudos [9, 5, 5, 9, 5, 4, 5] possui apenas 3 tipos distintos de escudos (4, 5 e 9), sendo essa a maior sequência de alienígenas que Kurt pode pulverizar.

Exemplo de entrada 2

8 6 4 8 1 9 4 15 12 1

Exemplo de saída 2

8

Explicação do exemplo 2

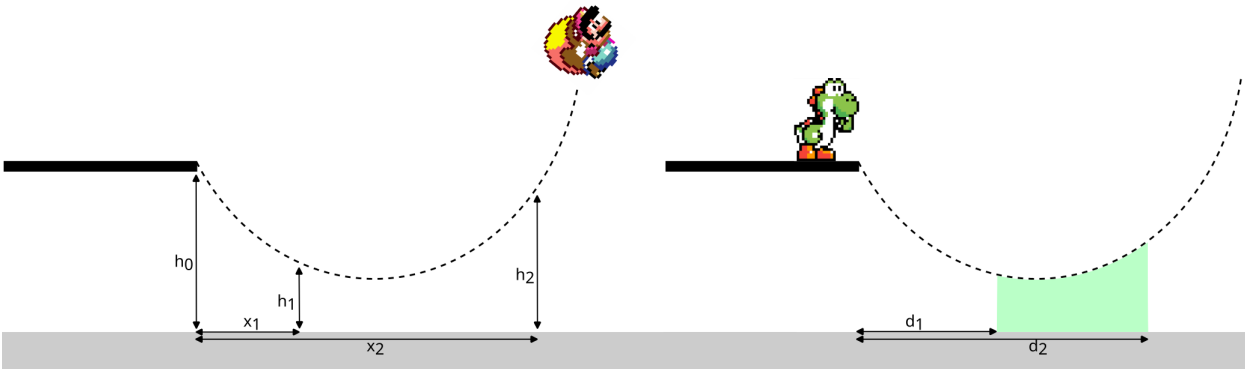
Neste exemplo, existem entre os 8 alienígenas apenas 6 tipos de escudos (1, 4, 8, 9, 12 e 15), logo Kurt pode pulverizar todos os alienígenas.

Problema J. Jornada pelo Reino

Tempo limite: 1000 ms
Memória limite: 256 MiB

Mário está em sua jornada pelo reino dos cogumelos para salvar a princesa. Em determinada fase, Mário encontrou uma capa mágica que o permite voar. Mário então subiu em uma plataforma e pulou dela com a capa. Você percebeu que:

- Mário pulou da plataforma a uma altura de h_0 metros;
- a trajetória do pulo teve o formato de uma parábola, com a concavidade para cima;
- a x_1 metros de distância (horizontal) da plataforma, Mário estava a uma altura de h_1 m;
- a x_2 metros de distância (horizontal) da plataforma, Mário estava a uma altura de h_2 m;
- Mário não tocou o chão em nenhum momento.



Yoshi, o dinossauro ajudante do Mário, está com medo de que Mário perca uma vida e, na próxima tentativa, faça o pulo em com ele junto (podendo, tragicamente, deixar Yoshi para trás durante o pulo!). Yoshi precisa calcular suas chances de sobrevivência; para isso, dadas duas distâncias d_1 e d_2 metros (horizontais) da plataforma, ajude Yoshi a calcular a área abaixo da trajetória que Mário fez entre d_1 e d_2 .

Entrada

A primeira linha contém um inteiro T , o número de casos de teste ($1 \leq T \leq 10^5$). Cada caso de teste é dado um uma linha com sete inteiros $h_0, x_1, y_1, x_2, y_2, d_1$ e d_2 ($1 \leq h_0, x_1, y_1, x_2, y_2, d_1, d_2 \leq 200, x_1 \neq x_2, d_1 < d_2$). É garantido que a entrada é tal que a concavidade da trajetória é para cima e que ela não toca o chão.

Saída

Para cada caso teste, imprima a área pedida, em metros quadrados. A resposta será considerada correta se seu erro absoluto (diferença entre sua resposta e a resposta exata) for menor ou igual a 10^{-6} .

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
2 6 2 2 6 6 1 5 2 2 2 3 7 8 9	8.6666666667 94.2222222222

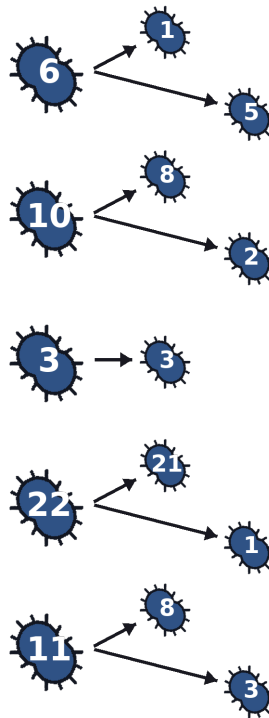
Problema K. Kurt Manipulando Células

Tempo limite: 1000 ms
Memória limite: 256 MiB

Kurt é um camaleão azul que gosta muito de matemática, computação e biologia celular. Atualmente ele está pesquisando um tipo peculiar de célula, que possui um nível de energia K e é capaz de se multiplicar através de um processo que ele batizou de Mitose Fibonáccica: a célula se divide em um conjunto de novas células cuja soma das energias é exatamente K , e cada uma dessas novas células possui energia igual a um número de Fibonacci.

Usando sua expertise em computação, Kurt conseguiu programar a Mitose Fibonáccica de forma manipulada, fazendo com que o conjunto resultante tenha a menor quantidade possível de células. Formalmente, ele obtém a decomposição de K como soma da menor quantidade de números de Fibonacci distintos que somam K .

Kurt possui N células, cada uma com seu próprio valor energético K_i . Ele aplica a Mitose Fibonáccica manipulada em todas elas, de forma que cada célula i se divide em um novo conjunto de subcélulas como demonstrado na imagem a seguir:



Como o resultado gerou muitos conjuntos pequenos, Kurt decidiu:

1. Manter todas as subcélulas derivadas de uma mesma célula original i em um único conjunto;
2. Unir dois conjuntos sempre que eles possuírem pelo menos uma subcélula com o mesmo valor energético em comum.

Depois dessas operações, Kurt quer saber qual é o conjunto com a maior quantidade de células com valores energéticos distintos.

Entrada

A primeira linha contém um único inteiro N ($1 \leq N \leq 10^5$). A quantidade de células que Kurt possui.

A segunda linha contém N inteiros $K_1, K_2, \dots, K_N$ ($1 \leq K_i \leq 10^{18}$), os valores energéticos de cada célula. É garantido que todo K_i dado na entrada pode ser escrito como a soma de um conjunto de números de Fibonacci.

Saída

Imprima duas linhas:

Na primeira linha, a maior quantidade de valores energéticos distintos encontrado em um único conjunto.

Na segunda linha, os valores energéticos distintos desse conjunto, em ordem crescente, separados por espaço. Caso haja empate entre dois ou mais conjuntos com a maior quantidade de valores distintos, deve-se escolher o conjunto que contiver a célula de menor valor energético.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
6 6 10 3 22 1 11	3 1 5 21

Explicação do exemplo 1

Neste exemplo, ilustrado na imagem anterior, os conjuntos de células com valor K inicial 6 e 22 tem o uma célula com valor 1 em comum, logo passam a formar um conjunto de tamanho $3 = [1, 5, 21]$.

E os conjuntos de células com valores energéticos iniciais 3 e 11 possuem uma célula com valor 3 em comum e os conjuntos das células com valores energéticos iniciais iguais a 10 e 11 possuem uma célula com valor 8 em comum, logo passam a formar um conjunto de tamanho $3 = [2, 3, 8]$.

Como ambos conjuntos formados tem tamanho 3, o critério de desempate pelo conjunto com menor valor é o $[1, 5, 21]$.

Exemplo de entrada 2	Exemplo de saída 2
4 21 2 8 13	1 2

Explicação do exemplo 2

Neste exemplo todas as N células já possuem seu valor enegético um número de fibonacci, logo o menor conjunto de números de fibonacci cuja a soma seja K_i são eles mesmos, então obtemos 4 conjuntos de tamanho 1, e pelo critério de desempate, o 2 possui o menor valor energético.

Exemplo de entrada 3	Exemplo de saída 3
5 3178 44 243 6 423	12 1 2 3 5 8 13 34 55 144 233 377 2584

Problema L. Lobo Faminto

Tempo limite: 1000 ms
Memória limite: 384 MiB



Na floresta vive um lobo selvagem que adora melancias. Sempre que está com fome, ele parte na floresta em busca da sua fruta favorita.

A floresta tem N pontos e M trilhas, que ligam um par de pontos cada. As trilhas são as únicas formas de se mover pela floresta e, para ir um ponto a outro, o lobo pode passar por várias trilhas. São dadas Q consultas. Em cada consulta, são dadas as posições atuais do lobo e de uma melancia. Determine se o lobo consegue ir até essa melancia e saciar sua fome.

Entrada

A primeira linha contém dois inteiros N e M ($1 \leq N \leq 10^5$, $0 \leq M \leq 10^5$), o número de pontos e de trilhas. Considere que pontos são numerados de 1 até N .

As próximas M linhas descrevem uma trilha cada. Cada trilha é dada por dois inteiros A e B ($1 \leq A, B \leq N$), indicando os pontos os quais a trilha conecta.

A próxima linha contém um inteiro Q ($1 \leq Q \leq 10^5$), o número de consultas. As próximas Q linhas descrevem uma consulta cada. Cada consulta é dada por dois inteiros l e m ($1 \leq l, m \leq N$), indicando o ponto onde está o lobo e a melancia, respectivamente.

Saída

Para cada consulta, imprima **sim** se o lobo pode ir até a melancia, ou **nao** caso contrário.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
7 5 1 3 6 2 7 3 5 1 5 7 5 1 7 6 2 5 4 2 3 4 4	sim sim nao nao sim

Explicação do exemplo 1

Na primeira consulta, o lobo consegue chegar até a melancia percorrendo, por exemplo, a trilha $1 \leftrightarrow 5$ e depois a trilha $5 \leftrightarrow 7$.

Problema M. Material de Consulta

Tempo limite: 1000 ms
Memória limite: 384 MiB



Pensando em estar sempre preparada para as Maratonas de Programação que ocorrem durante o ano, sua equipe está montando seu material de consulta. O material contém a implementação pronta de N algoritmos diferentes, que sua equipe sempre leva impressa para as provas para consultar.

O material tem N páginas. Tanto os algoritmos quanto as páginas são numerados de 1 a N . Inicialmente, o algoritmo i estava na página i ($1 \leq i \leq N$).

Entretanto, sua equipe decidiu fazer M operações no material. Em cada operação, a equipe inverteu a ordem dos algoritmos entre as páginas L_i e R_i , inclusive. Sua equipe acredita que, fazendo isso, o material fica mais organizado.

Temos outra Maratona amanhã, na LATINOWARE em Foz do Iguaçu, e sua equipe precisa do material de consulta atualizado para levar para a competição. Por isso, determine a ordem na qual os algoritmos ficaram no material, após todas as operações feitas pela equipe, ainda hoje!

Entrada

A primeira linha contém dois inteiros N e M ($1 \leq N \leq 2 \times 10^5$, $0 \leq M \leq 10^5$), o número de algoritmos e páginas, e o número de operações feitas. As próximas M linhas descrevem uma operação cada, na ordem em que foram feitas. Cada linha contém dois inteiros L_i e R_i ($1 \leq L_i \leq R_i \leq N$).

Saída

Imprima uma linha com N inteiros, indicando a ordem final dos algoritmos no material de consulta.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
<pre>6 3 2 3 3 6 2 4</pre>	<pre>1 5 6 3 4 2</pre>

Explicação do exemplo 1

No exemplo de entrada, as seguintes operações serão feitas:

- $[1, \underline{2}, 3, 4, 5, 6] \rightarrow [1, 3, 2, 4, 5, 6]$
- $[1, 3, \underline{2}, 4, 5, 6] \rightarrow [1, 3, 6, 5, 4, 2]$
- $[1, 3, \underline{6}, 5, 4, 2] \rightarrow [1, 5, 6, 3, 4, 2]$