XI Maratona de Programação do IFB

Caderno de Problemas

20 de junho de 2026

(Este caderno contém 13 problemas)

Comissão Organizadora:
Alberto Tavares Duarte Neto
Daniel Porto
Daniel Saad Nogueira Nunes
Edson Alves da Costa Júnior
Felipe Alves da Louza
Guilherme Novaes Ramos
Jeremias Moreira Gomes
João Carlos Gonçalves de Oliveira Filho
Leonardo Moreira Leódido

Instituto Federal de Brasília, Campus Taguatinga

Orientações

- É permitido consultar livros, anotações ou qualquer outro material impresso durante a prova, entretanto, o mesmo não vale para materiais dispostos eletronicamente.
- A correção é automatizada, portanto, siga atentamente as exigências da tarefa quanto ao formato da entrada e saída conforme as amostras dos exemplos. Deve-se considerar entradas e saídas padrão;
- Para cada problema, além dos testes públicos, o juiz executará a sua submissão contra uma série de testes secretos para fornecer um parecer sobre a correção do programa.
- Procure resolver o problema de maneira eficiente. Se o tempo superar o limite pré-definido, a solução não é aceita. Lembre-se que as soluções são testadas com outras entradas além das apresentadas como exemplo dos problemas;
- Utilize a aba *clarification* para dúvidas da prova. Os juízes podem opcionalmente atendê-lo com respostas acessíveis a todos;

C/C++

- Seu programa deve retornar zero, executando, como último comando, `return 0` ou `exit(0)`.
- É sabido que em alguns casos de problemas com entradas muito grandes, os objetos `cin` podem ser lentos, por conta da sincronização de buffers da biblioteca `stdio`.
- Caso se deseje utilizar `cin` e `cout`, um jeito de se contornar este problema é executando-se o comando `ios_base::sync_with_stdio(0)`, no início de sua função `main`.
- Note que, neste caso, o uso de `scanf` e `printf` no mesmo programa é contra-indicado, uma vez que, com buffers separados, comportamentos inadequados podem ocorrer.

Java

- Não declare "package" no seu programa Java.
- Note que a convenção para o nome do arquivo fonte deve ser obedecida, o que significa que o nome de sua classe pública deve ser uma letra maiúscula (A, B, C, etc).
- Comando para executar uma solução Java:
  `java -Xms1024m -Xmx1024m -Xss100m codigo_do_problema`

Kotlin

- Não declare "package" no seu programa Kotlin.
- Note que a convenção para o nome do arquivo de solução deve ser obedecida, o que significa que seu arquivo-fonte deve ser nomeado (A.kt, B.kt, C.kt, etc).
- Comando para executar uma solução Kotlin:
  `java -Xms1024m -Xmx1024m -Xss100m codigo_do_problemaKt`

Python

- Note que apenas Python 3 é suportado.
- As submissões em Python terão sua sintaxe verificada; programas que não passarem nessa verificação receberão o veredito "Compilation Error".

Problema A – Alfabeto da Gamma

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Guilherme Ramos

Gamma é uma criança feliz e adora brincar em seu tapete de letras. O que ela mais gosta é acompanhar o padrão de letras no tapete, que segue repetidamente a sequência do alfabeto latino em um formato retangular.

Ajude-a a criar seus próprios tapetes! Faça um programa que, dadas as quantidades de letras verticais e horizontais, crie um tapete alfabético de Gamma.

Entrada

A entrada consiste em dois números M e N ($1 \leq M, N \leq 2500$) indicando, respectivamente, a largura e a altura do tapete.

Saída

Imprima um tapete alfabético de Gamma com as dimensões solicitadas, usando apenas letras maiúsculas.

Exemplo

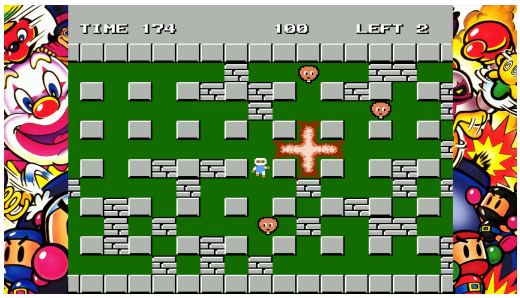
Entrada	Saída
3 6	ABC DEF GHI JKL MNO PQR
6 3	ABCDEF GHIJKL MNOPQR
30 3	ABCDEFGHIJKLMNOPQRSTUVWXYZABCD EFGHIJKLMNOPQRSTUVWXYZABCDEFGH IJKLMNOPQRSTUVWXYZABCDEFGHIJKL

Problema B – Bomberman

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Saad Nogueira Nunes

Bomberman é um jogo clássico em que os personagens precisam colocar bombas para destruir obstáculos, inimigos ou até mesmo outros jogadores. Os jogadores posicionam bombas que explodem após um certo tempo, em formato de cruz, com um determinado alcance. O tempo e o alcance da explosão são variáveis e dependem dos *power-ups* que os jogadores coletam durante o jogo.



Crie um programa que, ao receber o tempo para explosão de uma bomba e o seu alcance, imprima uma contagem regressiva até a explosão e, em seguida, a cruz resultante da explosão, conforme o alcance da bomba. A cruz é composta por dois segmentos de reta, um horizontal e outro vertical, ambos com o mesmo comprimento, que se cruzam no ponto onde a bomba foi colocada. O comprimento de cada segmento é determinado por $2x + 1$, em que x é o alcance da bomba.

Entrada

A entrada possui dois inteiros n e x ($1 \leq n, x \leq 100$), separados por um espaço em branco, representando o tempo para explosão da bomba e o seu alcance, respectivamente.

Saída

Imprima uma contagem regressiva, de n até 1. Em seguida, exiba a cruz resultante da explosão da bomba, considerando o seu alcance. A cruz deve estar contida em um quadrado de tamanho $(2x+1) \times (2x+1)$, onde x é o alcance da bomba. A cruz deve ser composta por caracteres '*' para representar as partes da explosão e espaços para representar as áreas não afetadas. Os números da contagem regressiva devem ser impressos um por linha, e a cruz deve ser impressa logo após a contagem regressiva, sem linhas em branco entre ela e os números da contagem regressiva.

Exemplo

Entrada	Saída
1 1	1 * *** *

2 2	2
	1
	  *
	  *
	*****
	  *
	  *

3 3	3
	2
	1
	   *
	   *
	   *
	*******
	   *
	   *
	   *

Problema C – Cores

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Alberto Neto

Edson ganhou de presente de aniversário uma prateleira com n bolas de futebol de copas passadas. Cada bola possui uma cor, e podem haver cores repetidas. Edson percebe que é visualmente desagradável quando duas bolas da mesma cor estão adjacentes na prateleira, e quer reordenar as bolas de modo que isso não aconteça.

Formalmente, as cores são representadas como números de 1 até n e as bolas inicialmente estão ordenadas com cores $c_1, \ldots, c_n$. Você deve reordenar as bolas em uma ordenação $b_1, \ldots, b_n$ tal que $b_i != b_{i+1}$ para todo $i < n$.

Se um tal ordenamento não for possível, apenas imprima o valor -1.

Entrada

A primeira linha contém um inteiro n ($2 \leq n \leq 2 \cdot 10^5$) — o número de bolas na prateleira.

A segunda linha contém n inteiros $c_1, \ldots, c_n$ ($1 \leq c_i \leq n$) — as cores das bolas na ordenação inicial.

Saída

Se não for possível obter um ordenamento agradável das cores, imprima uma única linha contendo o valor -1. Caso contrário, imprima uma linha contendo n inteiros $b_1, \ldots, b_n$ — uma ordenação agradável de cores. Caso exista mais de uma resposta, imprima qualquer uma delas.

Exemplo

Entrada	Saída
7 1 1 1 1 2 2 2	1 2 1 2 1 2 1
6 1 1 1 1 2 3	-1

Problema D – Dia de Trote

Limite de tempo: 1s
Limite de memória: 256MB

Autor: João Carlos Gonçalves de Oliveira

Élvis, Leitão e Medeiros são excelentes veteranos do curso de Computação. Uma das partes mais legais de ser veterano, além de ajudar com monitorias, distribuir dicas de sobrevivência e compartilhar materiais de estudo, é o tão temido dia de aplicação do Trote. Os recém-chegados calouros estão reunidos para sofrerem as mais absurdas e saudáveis atividades possíveis: desde levar farinhada e ovada até beber refrigerante em recipientes não convencionais.

Em uma das brincadeiras, Élvis organiza todos os calouros em uma fila de N pessoas. Nessa organização, o calouro mais à esquerda ocupa a posição 1 da fila, o imediatamente à sua direita ocupa a posição 2, e assim por diante, até o último calouro mais à direita, que ocupa a posição N. Enquanto a fila é formada, Leitão passa distribuindo para cada pessoa um papel com um número inteiro impresso. Como essa brincadeira envolve uma quantidade absurda de sujeira, você e mais alguns veteranos decidiram ficar de fora, apenas assistindo a situação.

Agora, Medeiros entra em ação. Ele escolhe um intervalo contíguo de pessoas na fila, indo da posição L até a posição R, e faz uma pergunta para que os calouros respondam em consenso: "Neste pedaço da fila de L até R, quantas pessoas possuem um número estritamente maior do que o número do colega que está imediatamente à sua esquerda?".

Formalmente, dado o intervalo $[L, R]$, eles devem contar quantas posições K ($L < K \leq R$) satisfazem a condição $A[K] > A[K-1]$, sendo A a fila que representa os números dos calouros. Os calouros só sabem o seu próprio número e não conseguem ver o número dos vizinhos, o que torna o consenso um verdadeiro jogo de adivinhação. Se eles errarem... bem, melhor nem falar o que acontece.

Para piorar a vida dos coitados, a fila é dinâmica: a qualquer momento, Leitão pode ir até a pessoa da posição i e trocar o papel antigo dela por um novo com o número X. Como um bom veterano (ou um calouro infiltrado que sabe programar), você decide criar um programa em segredo para calcular as respostas corretas instantaneamente e soprar para a sua turma, salvando-os dos veteranos. Dado o estado inicial da fila e o histórico de ações de Medeiros e Leitão, processe cada ação.

Entrada

A primeira linha da entrada contém dois inteiros N e Q ($1 \leq N, Q \leq 10^5$), representando o número de calouros na fila e a quantidade de eventos (perguntas de Medeiros ou alterações de Leitão), respectivamente. A segunda linha contém N inteiros $A_1, A_2, \ldots, A_N$ ($0 \leq A_i \leq 10^9$), os números inicialmente distribuídos por Leitão para os calouros da fila, ordenados da posição 1 (mais à esquerda) até a posição N (mais à direita).

As próximas Q linhas descrevem os eventos em ordem cronológica. Cada linha começa com um inteiro T ($T = 1$ ou $T = 2$):

Se $T = 1$: Seguem dois inteiros L e R ($1 \leq L \leq R \leq N$). Isso representa uma pergunta de Medeiros sobre o intervalo de calouros da posição L até a posição R.

Se $T = 2$: Seguem dois inteiros i e X ($1 \leq i \leq N$, $0 \leq X \leq 10^9$). Isso significa que Leitão alterou o número do calouro na posição i da fila para X.

Saída

Para cada evento do tipo 1 ($T = 1$), imprima uma única linha contendo um inteiro: a resposta para a pergunta de Medeiros.

Exemplo

Entrada	Saída
5 4 10 20 15 30 5 1 1 4 2 3 25 1 1 4 1 4 5	2 3 0

Notas

Considere a fila inicial de calouros com seus respectivos números: $[10, 20, 15, 30, 5]$.

Olhando o pedaço da fila da pessoa 1 até a 4 ($[10, 20, 15, 30]$), existem 2 pessoas com um número maior que o do colega da esquerda: a pessoa 2 ($20 > 10$) e a pessoa 4 ($30 > 15$).

Em seguida, Leitão troca o número do calouro na posição 3 de 15 para 25. A fila passa a ser: $[10, 20, \mathbf{25}, 30, 5]$. Olhando o mesmo pedaço da fila ($[10, 20, 25, 30]$), agora temos 3 pessoas com números maiores que os dos seus vizinhos da esquerda ($20 > 10$, $25 > 20$ e $30 > 25$).

Problema E – E.T.

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Jeremias Moreira Gomes

O estado do Paraná está passando por outro rebuliço esse ano. Dessa vez, um de seus residentes avistou e registrou em vídeo um OVNI (Objeto Voador ou Não, Iluminado) nas redondezas de sua moradia.

A gravação mostra que o OVNI possui cinco luzes que ficam acendendo e apagando de maneira intermitente. Além disso, em alguns intervalos de tempo, o OVNI emite sons extremamente altos, que todo mundo consegue ouvir.

Tentando entender se todas essas ações estão de alguma forma relacionadas, o IFB (Instituto de Fenômenos Bizarros) pediu a sua ajuda para analisar os dados registrados. Para isso, você deve contar quantas vezes cada uma das cinco luzes apareceu acesa nos registros observados e, sempre que o OVNI emitir um som, informar a quantidade acumulada de vezes que cada luz apareceu acesa até aquele momento.

Entrada

A entrada contém um único caso de teste. A primeira linha contém um inteiro N ($1 \leq N \leq 1000$), que indica o número de eventos registrados. Cada uma das próximas N linhas contém uma string de um dos dois tipos abaixo:

1. Uma string S_1 ($|S_1| = 5$) composta pelos caracteres '`-`' ou '`*`', indicando quais das cinco luzes estavam acesas naquele registro. O caractere '`-`' indica que a luz correspondente estava apagada, enquanto o caractere '`*`' indica que a luz correspondente estava acesa.
2. Uma string S_2 ($S_2 =$ `PIP`), indicando que o OVNI emitiu um som estranho.

Saída

Para cada vez que o OVNI emitiu um som, imprima uma linha contendo cinco inteiros separados por um espaço em branco, representando a quantidade de vezes que cada luz acendeu até aquele momento, da primeira à quinta luz.

Exemplo

Entrada	Saída
5 ----* PIP ----* ---** PIP	0 0 0 0 1 0 0 0 1 3
4 PIP *-*-* PIP PIP	0 0 0 0 0 1 0 1 0 1 1 0 1 0 1

Problema F – Formatura

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Alberto Neto

Ruan, membro do UnBalloon, está prestes a se formar! Ele deve cursar n disciplinas, que estão indexadas de 1 até n. As disciplinas podem possuir pré-requisitos, de forma que ela só poderá ser cursada após todos seus pré-requisitos serem cursados. Devido ao planejamento de Ruan, cada disciplina é pré-requisito de **no máximo uma** outra disciplina.

Dudu, curioso sobre a formatura de Ruan, se pergunta: de quantas maneiras Ruan pode cursar suas disciplinas restantes? Para isso, considere que as disciplinas são realizadas uma a uma. Como este valor pode ser muito grande, calcule-o módulo 998244353.

Entrada

A primeira linha de entrada contém um único inteiro n ($2 \leq n \leq 2 \cdot 10^5$) — o número de disciplinas.

A segunda linha de entrada contém n inteiros $a_1, \ldots, a_n$ ($0 \leq a_i \leq 2 \cdot 10^5$) — indicando que a disciplina a_i tem i como pré-requisito se $a_i \neq 0$.

É garantido que não há ciclos de pré-requisitos e que cada disciplina é pré-requisito do no máximo uma outra disciplina.

Saída

Imprima um único inteiro — o número de maneiras de cumprir as disciplinas módulo 998244353.

Exemplo

Entrada	Saída
7 2 4 4 6 6 0 0	105
6 0 1 1 0 4 4	80
3 0 0 0	6

Problema G – Gooool

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Jeremias Moreira Gomes

A CaseTV garantiu os direitos de transmissão de todos os jogos da Copa do Mundo de 2026 e pretende competir diretamente com a televisão aberta. No entanto, existe um desafio importante com o qual eles estão preocupados: o atraso da transmissão.

Enquanto a TV aberta utiliza sinal digital e consegue entregar a imagem aos espectadores com poucos segundos de atraso, transmissões via streaming precisam passar por diversas etapas adicionais de processamento, como codificação, distribuição pela rede e decodificação.

Os engenheiros da CaseTV desenvolveram um modelo simplificado para estimar o atraso total da transmissão, baseado na sua infraestrutura e no número de espectadores assistindo simultaneamente. Com a plataforma atendendo simultaneamente a n espectadores, o atraso total, em segundos, é dado pela seguinte função:

$$f(n) = T \cdot \sqrt{n} + G \cdot \log_2(n+1) + D$$

em que:

- n é a quantidade de espectadores assistindo à transmissão;
- T representa o impacto da carga gerada pelos espectadores conectados;
- G representa o custo de gerenciamento da infraestrutura; e
- D representa o atraso fixo de codificação, transmissão e decodificação.

Para competir com a televisão aberta, a CaseTV estabeleceu que o atraso da transmissão não poderia ultrapassar um determinado limite. Assim, ela pediu a sua ajuda para, dadas as configurações da infraestrutura e o limite de atraso, determinar a quantidade máxima de espectadores que a plataforma consegue atender simultaneamente.

Entrada

A entrada contém uma única linha contendo quatro inteiros T, G, D e X ($1 \leq T, G, D, X \leq 10^9$), separados por um espaço em branco, representando o impacto da carga gerada pelos espectadores conectados, o custo de gerenciamento da infraestrutura, o atraso fixo da plataforma e o atraso máximo permitido, respectivamente.

Saída

A saída deve conter um único inteiro, representando a quantidade máxima de espectadores que a plataforma consegue atender simultaneamente sem que o atraso total da transmissão ultrapasse o máximo permitido.

Exemplo

Entrada	Saída
2 7 6 60	50
123 456 789 100000	541666

Notas

No primeiro caso de teste, com 51 espectadores, o atraso é igual a

$$f(51) = 2 \cdot \sqrt{51} + 7 \cdot \log_2(52) + 6 \approx 60.1859,$$

que é um valor maior do que o permitido. Já com 50 espectadores, o atraso é de

$$f(50) = 2 \cdot \sqrt{50} + 7 \cdot \log_2(51) + 6 \approx 59.8491,$$

que está abaixo do máximo permitido e é a resposta para o caso de teste.

Problema H – Herói das Cartas

Limite de tempo: 1s
Limite de memória: 256MB

Autor: João Carlos Gonçalves de Oliveira

Jp e Samuel são jogadores exímios de Magic: um jogo de cartas meio nerd, com personagens mágicos de diferentes universos e muitas habilidades. Os dois estão tentando converter sua colega Adrielly a jogar também, mas com uma versão mais simples e alternativa do jogo. Para isso, ela precisa montar um deck. Nesta versão simplificada, um deck consiste estritamente em um trio de cartas cujos índices no conjunto disponível são distintos. O poder total do deck é determinado pelo produto do poder das três cartas escolhidas. Uma regra peculiar dessa versão alternativa é que o jogador pode, opcionalmente, escolher no máximo uma de suas três cartas para ser o seu "Herói". A carta que for atribuída como Herói terá o seu valor de poder multiplicado por -1.

Formalmente, caso Adrielly escolha três cartas com poderes a, b e c:

- se ela optar por não utilizar a mecânica do Herói, o poder total do deck será dado por: $a \times b \times c$;
- se ela escolher a primeira carta para ser o Herói, o poder total do deck será dado por: $(-a) \times b \times c$;
- a situação é análoga se ela escolher a segunda ou a terceira carta para ser o Herói, de modo que o poder total do deck será igual a $a \times (-b) \times c$ e $a \times b \times (-c)$, respectivamente.

Samuel e Jp ainda não sabem se essa mecânica do Herói é realmente vantajosa para o jogador em todas as situações. No entanto, como Adrielly é extremamente competitiva e gostaria de estrear no jogo com o deck mais forte possível, ela pediu sua ajuda. Dado o conjunto de cartas oferecido por seus amigos, sua tarefa é calcular o maior poder total possível que Adrielly pode obter, assumindo que ela escolha as três cartas e utilize a mecânica do Herói de forma ótima.

Entrada

A primeira linha da entrada contém um único inteiro N ($3 \leq N \leq 10^5$), representando o número total de cartas disponíveis.

A segunda linha contém N inteiros A_i ($-10^5 \leq A_i \leq 10^5$), onde A_i indica o poder da i-ésima carta.

Saída

Imprima uma única linha contendo um inteiro: o maior poder total possível que Adrielly pode obter para o seu deck.

Exemplo

Entrada	Saída
4 -3 2 4 5	60
3 -10 -10 -10	1000

Problema I – Império dos Bits

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Felipe Louza

No distante Império dos Bits, cada cidadão é identificado por um número inteiro positivo.

O imperador decidiu organizar todos os cidadãos em uma hierarquia peculiar. Para isso, ele definiu a seguinte operação:

$$f(x) = \text{popcount}(x),$$

onde popcount(x) representa a quantidade de bits iguais a 1 na representação binária de x.

Sempre que a operação é aplicada a um cidadão de identificação x, ele passa a ser representado pelo número $f(x)$. O processo é repetido sucessivamente até que o cidadão alcance o número 1, considerado o fundador do império.

Por exemplo, o cidadão 13 evolui da seguinte forma:

$$13 \to 3 \to 2 \to 1.$$

Nesse caso, foram necessárias 3 aplicações da operação para alcançar o fundador.

O imperador define o *nível imperial* de um cidadão como o número de aplicações da operação necessárias para chegar ao número 1.

Sua tarefa é determinar o nível imperial de diversos cidadãos.

Entrada

A entrada é composta por vários casos de teste.

Cada linha contém um inteiro positivo x ($1 \leq x \leq 10^{18}$), representando o identificador de um cidadão.

O fim da entrada é indicado por uma linha contendo apenas o valor 0, que não deve ser processado.

Saída

Para cada cidadão informado na entrada, imprima uma linha contendo seu nível imperial.

Exemplo

Entrada	Saída
1	0
2	1
3	2
13	3
255	2
0	
7	3
8	1
31	3
60	2
0	

Problema J – Jogadores Icônicos

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Saad Nogueira Nunes

Cada time de futebol possui alguns jogadores icônicos associados ao número de suas camisas. Por exemplo, no Santos a camisa 10 é associada ao Pelé, já no Flamengo, a camisa 10 é associada ao Zico, enquanto a camisa 2 é associada ao Leandro. A camisa 7 do Botafogo está associada ao Garrincha, enquanto a camisa 8 do Corinthians está associada ao Sócrates. No São Paulo, a camisa 1 é associada ao Rogério Ceni. São vários exemplos que se consolidaram ao longo do tempo e que são facilmente reconhecidos pelos torcedores. É uma grande honra para um jogador atual jogar com essas camisas icônicas pertencentes aos grandes ídolos da história do clube.

Crie um programa que receba a lista de jogadores icônicos de um time de futebol, bem como o número de suas camisas e consiga responder perguntas do tipo "*Qual jogador do time x é associado ao número y?*".

Entrada

A primeira linha da entrada contém um inteiro N ($1 \leq N \leq 100$), o número de jogadores icônicos.

As próximas N linhas contém o nome de um jogador icônico, seguido do número de sua camisa e o time ao qual ele pertenceu. As informações estão separadas por espaço.

A próxima linha contém um inteiro Q ($1 \leq Q \leq 100$), o número de perguntas.

Cada uma das próximas Q linhas contém o nome de um time e um número de camisa, separados por espaço.

Os nomes dos jogadores e dos times são compostos apenas por letras maiúsculas ou minúsculas sobre o alfabeto $\{a, \ldots, z, A, \ldots, Z\}$ e não possuem mais do que 30 caracteres. Os números das camisas são inteiros entre 1 e 99. É garantido que não existem dois jogadores icônicos com o mesmo número de camisa no mesmo time. Também é garantido que todas as perguntas são válidas, ou seja, existe um jogador icônico associado ao número de camisa do time indicado na pergunta.

Saída

Para cada pergunta, imprima uma linha contendo o nome do jogador associado ao número de camisa do time indicado na pergunta.

Exemplo

Entrada	Saída
6 Zico 10 Flamengo Socrates 8 Corinthians Garrincha 7 Botafogo Pele 10 Santos Ceni 1 SaoPaulo Leandro 2 Flamengo 6 8 Corinthians 10 Flamengo 2 Flamengo 1 SaoPaulo 7 Botafogo 10 Santos	Socrates Zico Leandro Ceni Garrincha Pele

Notas

O exemplo reflete os jogadores icônicos descritos no enunciado.

Problema K – Kátia e as constelações

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Edson Alves

Kátia gosta muito de observar o céu e as estrelas. Estudando sobre o assunto, ela obteve um mapa celeste no qual constam N estrelas. Como o mapa não discriminava as constelações, ela decidiu organizar as estrelas em constelações usando o seguinte critério: um conjunto C de estrelas formaria uma constelação se,

1. C contém um único ponto; ou
2. para qualquer ponto $p \in C$, existe pelo menos outro ponto $q \in C$, $p \neq q$, tal que a distância $d(p, q)$ entre p e q é menor ou igual a D.

Kátia utiliza a distância euclidiana, isto é, se p tem coordenadas (x_p, y_p) e q tem coordenadas (x_q, y_q), então a distância entre p e q é dada por

$$d(p, q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$$

Kátia concluiu sua classificação, mas gostaria de saber se ela acertou todos os cálculos. Ajude-a escrevendo um programa que recebe os valores de N, D e as coordenadas das estrelas e que retorna o tamanho da maior constelação possível.

Entrada

A primeira linha da entrada contém dois inteiros N e D ($2 \leq N \leq 2 \times 10^3, 1 \leq D \leq 10^3$), separados por um espaço em branco, indicando o número de estrelas no mapa e a distância utilizada no critério 2, respectivamente.

As próximas N linhas contém, cada uma, um par de coordenadas x_i e y_i ($1 \leq x_i, y_i \leq 10^3$), separadas por um espaço em branco, indicando a posição da i-ésima estrela do mapa ($1 \leq i \leq N$).

É garantido que todas as estrelas ocupam posições distintas no mapa.

Saída

Imprima, em uma linha, o tamanho da maior constelação possível, de acordo com os critérios estabelecidos por Kátia.

Exemplo

Entrada	Saída
5 3 1 1 2 1 5 2 6 3 7 2	3
2 5 2 2 10 1	1
9 1 3 1 3 2 3 3 4 1 4 2 4 3 5 1 5 2 5 3	9

Notas

No primeiro caso, as estrelas 1 e 2 formam uma constelação, pois $d(1, 2) = 1$, e as estrelas 3, 4 e 5 formam uma segunda constelação ($d(3, 4) = d(4, 5) = \sqrt{2} < 3$). Portanto a maior constelação tem 3 estrelas.

No segundo caso, cada estrela forma uma constelação composta por uma única estrela (critério 1).

Problema L – Lógica dos Sinais

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Daniel Saad Nogueira Nunes

Ayla e Luísa resolveram brincar de um jogo de matemática conhecida como "Lógica dos Sinais". Dada uma sequência de números $V = \langle v_1, \ldots, v_n \rangle$, cada número v_i pode ser selecionado por Ayla ou por Luísa. Caso seja selecionado por Ayla, o sinal + é atribuído a ele, e, caso seja selecionado por Luísa, o sinal -. O objetivo do jogo é que a soma de todos os números com seus respectivos sinais seja igual a um valor k previamente definido. Será que Ayla e Luísa conseguem encontrar uma maneira de atribuir os sinais para que a soma seja igual ao objetivo? Note que Ayla e Luísa não podem deixar nenhum número sem um sinal atribuído a ele, e que não é necessário que ambas selecionem a mesma quantidade de números, uma jogadora pode, inclusive, selecionar todos os números da sequência.

Entrada

A primeira linha da entrada contém dois inteiros n ($1 \leq n \leq 100$) e k ($-10000 \leq k \leq 10000$), representando, respectivamente, a quantidade de números na sequência e o valor objetivo da soma. A segunda linha contém n inteiros $v_1, v_2, \ldots, v_n$, ($1 \leq v_i \leq 100$), representando os números da sequência.

Saída

Caso haja solução, imprima uma linha contendo uma palavra S com n caracteres. O i-ésimo caractere de S deve ser "`A`" se o número v_i for selecionado por Ayla, ou "`L`", se for selecionado por Luísa.

Se houver mais de uma solução, imprima qualquer uma delas.

Se não houver solução, imprima a palavra "`impossivel`" (sem as aspas).

Exemplo

Entrada	Saída
3 0 1 2 3	AAL
2 2 2 2	impossivel
5 5 1 1 1 1 1	AAAAA

Notas

No primeiro exemplo, se Ayla selecionar o primeiro e segundo números, e Luísa o terceiro, a expressão $+1 + 2 - 3 = 0$ é obtida e o objetivo é alcançado.

No segundo exemplo, não há como as jogadoras selecionarem os sinais para que a soma seja igual a 2.

No terceiro exemplo, Ayla seleciona todos os números, e a soma é $+1 + 1 + 1 + 1 + 1 = 5$, que é igual ao objetivo. Note que Luísa não seleciona nenhum número, o que é permitido pelas regras do desafio.

Problema M – Mudança

Limite de tempo: 1s
Limite de memória: 256MB

Autor: Edson Alves

Após uma excelente estadia no exterior, o professor Borges começou a encaixotar seus pertences para a viagem de volta ao Brasil. No momento ele dispõe de N caixas, cujos formatos são paralelepípedos, e ele precisa embalar M pratos, os quais têm formato de troncos de cilindros circulares, todos com altura H.

Para que nenhum prato se quebre durante o transporte, ele estipulou as seguintes regras para a organização dos pratos nas caixas:

1. Se a caixa com largura a_k, comprimento b_k e altura c_k estiver vazia, ela pode receber um prato cujo raio da base é r_i, desde que ele fique inteiramente contido (podendo tocar as bordas) na caixa e que $H \leq c_k$.

2. Um prato com raio r_i pode ser empilhado em um prato de raio r_j se $r_i \leq r_j$ e a altura da pilha resultante for menor ou igual à altura c_k da caixa.

3. Cada caixa pode acomodar, no máximo, uma pilha de pratos.

Como há muita coisa para ser resolvida para mudança, o professor Borges pediu a sua ajuda. Escreva um programa que receba as dimensões das caixas e os raios das bases dos pratos e que determine uma alocação dos pratos nas caixas de modo a maximizar o número de pratos embalados.

Entrada

A primeira linha da entrada contém o inteiro N ($1 \leq N \leq 2 \times 10^5$), o qual indica o número de caixas que o professor Borges possui.

As N linhas seguintes contêm, cada uma, as dimensões a_k, b_k e c_k ($10^{-6} \leq a_k, b_k, c_k \leq 1.0$) da caixa k ($1 \leq k \leq N$), separadas por um espaço em branco. Cada dimensão é um número real com no mínimo uma e, no máximo, seis casas decimais.

A próxima linha contém o inteiro M ($1 \leq M \leq 2 \times 10^5$) e o real H ($10^{-6} \leq H \leq 1.0$), separados por um espaço em branco, representando o número de pratos e a altura de cada prato, respectivamente. É garantido que H terá no mínimo uma e, no máximo, seis casas decimais.

A última linha contém M números reais r_i ($10^{-6} \leq r_i \leq 1.0$), separados por um espaço em branco, em que r_i corresponde ao raio da base do i-ésimo prato ($1 \leq i \leq M$). Os raios terão entre uma e seis casas decimais em suas representações.

Saída

Imprima, em uma linha, o número máximo P de pratos que o professor Borges consegue embalar.

Em seguida, imprima P linhas. Cada linha deve conter dois inteiros i e k, separados por um espaço em branco, indicando que o prato i deve ser colocado na caixa k. Estas instruções, ao serem seguidas na ordem impressa, devem respeitar os critérios estabelecidos pelo professor.

Se houver mais de uma sequência de instruções que maximize o número de pratos embalados, imprima qualquer uma delas.

Exemplo

Entrada	Saída
2 0.1 0.25 0.7 1.0 1.0 0.8 5 0.3 0.05 0.1 0.8 0.02 0.75	3 2 2 1 2 4 1
2 0.3 0.3 0.1 0.2 0.5 0.83 2 0.1 0.15 0.15	1 2 1
3 0.1 0.1 0.1 0.2 0.2 0.3 0.5 0.5 0.2 5 0.2 0.5 0.4 0.3 0.2 0.1	2 4 3 5 2

Notas

No primeiro caso, o professor Borges pode colocar o prato 2 na caixa 2 e, em seguida, empilhar o prato 1 sobre ele. Ambos formaram uma pilha de tamanho 0,6, impedindo que um novo prato possa ser colocado nesta caixa. Assim, ele finaliza o processo colocando o prato 4 na caixa 1, a qual não comporta os pratos 3 e 5.

No segundo caso, qualquer um dos dois pratos poderia ser colocado na caixa 1, que tem capacidade para apenas um prato.