

Maratona Sydle Levty de Programação

XIV SECOM

25 de Setembro de 2025

Caderno de Problemas

Informações Gerais

Este caderno contém 11 problemas; as páginas estão numeradas de 2 a 18, contando assim com essa página de rosto. Verifique se seu caderno está completo.

Leia atentamente as instruções a seguir. Informações adicionais, incluindo os limites de tempo dos problemas estão disponíveis abaixo do título de cada exercício.

1. Sobre os nomes de programas

- a. Para as soluções em C/C++ e Python, o nome do arquivo-fonte não é significativo, pode ser qualquer nome.
- b. Se a solução é em Java, ela deve ser chamada `codigo.java`, onde `codigo` é o nome do problema sem espaços e em minúsculo, por exemplo, se o problema chama “Amigos da Maratona” o arquivo aqui seria `amigosdamaratona.java`. Além disso, lembre que em Java o nome da classe principal deve ser igual ao nome do arquivo.

2. Sobre a entrada

- a. A entrada de seu programa deve ser lida da entrada padrão.
- b. A entrada é composta de um único caso de teste, descrito em um número de linhas que depende do problema.
- c. Quando uma linha da entrada contém vários valores, estes são separados por um único espaço em branco; a entrada não contém nenhum outro espaço em branco.
- d. Cada linha, incluindo a última, contém exatamente um caractere final-de-linha.
- e. O final da entrada coincide com o final do arquivo.

3. Sobre a saída

- a. A saída de seu programa deve ser escrita na saída padrão.
- b. Quando uma linha de saída contém vários valores, estes devem ser separados por um único espaço em branco; a saída não deve conter nenhum outro espaço em branco.
- c. Cada linha, incluindo a última, deve conter exatamente um caractere final-de-linha.

Problema A
Alimentando Capivaras

Na Universidade Federal de Viçosa (UFV) campus Florestal, as capivaras são as verdadeiras rainhas. Elas têm seu próprio "serviço de alimentação", com pessoas dedicadas a garantir que estejam sempre bem alimentadas.

Essas pessoas, conhecendo os hábitos alimentares das capivaras, sabem que a fome delas pode ser medida por um valor inteiro F , que varia ao longo do dia. Para cada capivara, eles têm uma lista de N alimentos disponíveis, e cada alimento tem um valor de saciedade S_i .

Mas a equipe de pessoas que cuidam da alimentação das capivaras enfrenta um problema: para evitar o desperdício e garantir que as capivaras fiquem satisfeitas, eles querem dar a elas a combinação exata de alimentos cuja soma dos valores de saciedade seja igual a F .

Sua tarefa é descobrir quais são os alimentos que eles precisam usar para que a soma dos valores de saciedade seja exatamente igual à fome F da capivara. Se não for possível atingir a fome exata com nenhuma combinação de alimentos, a equipe de tratadores precisa ser informada. É importante ressaltar que os alimentos listados para cada uma das capivaras não possuem o valor de saciedade igual.

Entrada

A entrada consistirá em duas linhas, a primeira terá dois valores, N ($3 \leq N \leq 1000$), sendo o número de alimentos disponíveis, e F ($1 \leq F \leq 10^6$), indicando a fome da capivara. A segunda linha terá N valores, onde o i -ésimo inteiro representa o valor de saciedade do alimento S_i ($1 \leq S_i \leq 10^6$).

Saída

Sua saída deve apresentar os valores de saciedade dos alimentos que devem ser entregues à capivara, ou em caso de não ser possível saciar sua fome imprimir a mensagem "N".

Exemplos

Entrada	Saída
3 10 2 5 3	2 3 5
5 15 1 2 4 8 10	1 2 4 8
4 7 1 3 5 8	N

Solução

Este problema é uma variação do problema clássico Soma de Subconjuntos. Esse problema tem como objetivo buscar em um conjunto de inteiros um subconjunto que sua soma resulte em um valor alvo determinado. Neste caso, o conjunto de inteiros é a lista de alimentos e o valor alvo a fome da capivara. Dada a complexidade esperada para esse problema, o problema deveria ser resolvido usando a programação dinâmica.

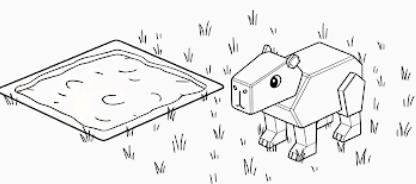
Complexidade: $O(n)$

Pessoa Autora: Estela Miranda

EDITORIAL

Problema B
Capivarinha Robô

Para auxiliar no ensino de pensamento computacional, alunos da UFV campus Florestal desenvolveram um jogo digital. Nele, uma capivarinha robô se move em um plano cartesiano, seguindo uma sequência de instruções definida pelo usuário. O objetivo é que ela não pare na lagoa quadrada, que representa um perigo.



Sua tarefa é simular os movimentos da capivarinha e, para cada instrução, verificar se ela está em uma posição segura. A missão da capivarinha é considerada um fracasso apenas se a posição final de uma instrução estiver estritamente dentro da lagoa. Se a capivarinha parar na borda ou passar pela área da lagoa sem parar estritamente dentro dela, a missão continua, pois ela é uma capivarinha robô que sabe nadar mas que não sabe boiar. No final da sequência, você deve imprimir uma mensagem que resume o resultado da partida.

A capivarinha começa na posição $(0, 0)$ de um plano cartesiano "infinito" e sabe seguir as seguintes instruções:

LEFT X: move X unidades para a esquerda. ($1 \leq X \leq 100$)

RIGHT X: move X unidades para a direita. ($1 \leq X \leq 100$)

UP X: move X unidades para cima. ($1 \leq X \leq 100$)

DOWN X: move X unidades para baixo. ($1 \leq X \leq 100$)

SAME_AS i: repete a i -ésima instrução da sequência. É garantido que i é um inteiro positivo não maior que o número de instruções já executadas. O contador de instruções começa no 1.

A lagoa é uma área quadrada, definida por um ponto no canto inferior esquerdo (x_{lagoa}, y_{lagoa}) e um lado de comprimento L ($1 \leq L \leq 1000$). A posição inicial da capivarinha, $(0,0)$, é sempre segura, ou seja, nunca será uma posição estritamente dentro da lagoa.

Entrada

A entrada consiste em um único caso de teste. A primeira linha contém quatro inteiros, separados por espaço:

- P : o número de instruções. ($1 \leq P \leq 100000$)
- x_{lagoa} : a coordenada x do canto inferior esquerdo da lagoa. ($-1000 \leq x_{lagoa} \leq 1000$)
- y_{lagoa} : a coordenada y do canto inferior esquerdo da lagoa. ($-1000 \leq y_{lagoa} \leq 1000$)
- L : o comprimento do lado da lagoa. ($1 \leq L \leq 1000$)

As próximas P linhas contêm as instruções, uma por linha, no formato descrito acima.

Saída

Para cada instrução executada, imprima uma linha contendo as coordenadas da capivarinha após seguir a instrução, no formato (x, y), seguidas de um espaço e da string NA LAGOA se ela estiver estritamente dentro da área da lagoa, ou FORA DA LAGOA caso contrário.

Após a última instrução, imprima uma linha adicional com o status da missão:

CAPIVARINHA DANIFICADA, se a capivarinha parou dentro da lagoa após alguma instrução

MISSAO CUMPRIDA, se ela nunca parou estritamente dentro da lagoa.

Entrada	Saída
4 1 1 2 RIGHT 2 UP 2 SAME_AS 1 LEFT 1	(2, 0) FORA DA LAGOA (2, 2) NA LAGOA (4, 2) FORA DA LAGOA (3, 2) FORA DA LAGOA CAPIVARINHA DANIFICADA
4 -10 -10 5 RIGHT 15 UP 15 SAME_AS 1 SAME_AS 2	(15, 0) FORA DA LAGOA (15, 15) FORA DA LAGOA (30, 15) FORA DA LAGOA (30, 30) FORA DA LAGOA MISSAO CUMPRIDA
7 0 0 10 RIGHT 5 UP 6 LEFT 10 RIGHT 5 UP 3 SAME_AS 4 DOWN 10	(5, 0) FORA DA LAGOA (5, 6) NA LAGOA (-5, 6) FORA DA LAGOA (0, 6) FORA DA LAGOA (0, 9) FORA DA LAGOA (5, 9) NA LAGOA (5, -1) FORA DA LAGOA CAPIVARINHA DANIFICADA

Solução

A solução exige uma iteração sobre todas as instruções de entrada. Para cada instrução, a posição da capivarinha é atualizada e a condição de estar na lagoa é verificada. O comando SAME_AS é implementado com um acesso por índice a uma lista que armazena o histórico dos movimentos, o que é a estrutura de dados central para este problema.

Complexidade: $O(n)$

Pessoa Autora: Letícia Silva

Problema C

Quem é essa Capivara?

As capivaras da lagoa da UFV são tão famosas que receberam um sistema de cadastro oficial. Cada capivara ganhou um Cadastro de Identificação da Capivara (CIC), uma sequência de 8 dígitos que permite saber exatamente quem é essa capivara.



Porém, após uma falha no sistema, alguns identificadores tiveram seus dígitos corrompidos. Sua missão é verificar se cada identificador é válido e, se não estiverem, descobrir qual seria o número correto para identificar a capivara misteriosa.

Cada CIC possui 8 dígitos, sendo os 6 primeiros a base e os 2 últimos dígitos verificadores, usados para confirmar se o número está correto. Para calcular os dígitos verificadores, siga os passos abaixo:

- Primeiro dígito verificador:
 1. Multiplique os 6 primeiros dígitos pelos pesos 7, 6, 5, 4, 3, 2.
 2. Some os resultados e tire o resto da divisão por 11.
 3. Se o resto for menor que 2, o dígito é 0; caso contrário, é 11 - resto.
- Segundo dígito verificador:
 1. Refaça o cálculo incluindo o primeiro dígito e usando os pesos 8, 7, 6, 5, 4, 3, 2.
 2. Se o resto for menor que 2, o dígito é 0; caso contrário, é 11 - resto.

Sua missão é criar um sistema capaz de verificar se os CICs estão corretos e, quando necessário, apontar qual seria o número certo para revelar a verdadeira identidade da capivara misteriosa.

Entrada

A entrada consiste em vários testes de CICs, um por linha. Cada CIC possui os 8 dígitos e um separador “-” entre os dígitos base e os dígitos verificadores. A entrada termina com o 0 (zero).

Saída

Para cada CIC, imprima uma linha contendo CAPIVARA IDENTIFICADA, se os dois dígitos verificadores estiverem corretos, ou CAPIVARA MISTERIOSA n1n2 se os dígitos verificadores estiverem errados, onde n1n2 são os dígitos verificadores corretos.

Entrada	Saída
123456-01	CAPIVARA IDENTIFICADA
987654-32	CAPIVARA MISTERIOSA 50
123123-99	CAPIVARA MISTERIOSA 55
765432-10	CAPIVARA MISTERIOSA 42
0	

Solução:

O objetivo é verificar se os CICs estão corretos e, quando necessário, apontar qual seria o número certo para os CICs fora do padrão. Para a solução deste problema, é necessário percorrer os dígitos dos CICs presentes na entrada, aplicar as multiplicações com os pesos pré-definidos e calcular o módulo.

Complexidade: $O(1)$

Pessoa Autora: Emily Lopes

EDITORIAL

Problema D
Casamento de Nomes

Quando duas capivarinhas se amam muito, elas adotam uma capivarinha filhote para distribuir um pouco desse amor. Nesse processo, ocorre uma atividade muito complexa: a escolha de um nome!

Pensando nisso, juntam-se os nomes dos dois cônjuges e escolhe-se a maior subsequência entre esses dois nomes, dando, assim, o nome à nova capivarinha. Sua tarefa é ajudar as capivarinhas da UFV a escolher o nome de seus filhotes.

Entrada

A entrada contém duas linhas, cada uma com uma string. Na primeira linha está a string de busca, e na segunda, a do artigo. Ambas as strings contêm apenas caracteres de 'a' a 'z', sem espaços. O tamanho das strings, tanto a de busca quanto a do artigo, é de no máximo 5000 caracteres.

Saída

A saída contém apenas uma linha com um inteiro: o tamanho do maior nome baseado nos dois nomes inseridos.

Exemplos

Entrada	Saída
Estela Estrela	6
Estela estela	5
Estela Marquela	3

Solução

Para resolver este problema era necessário a aplicação do algoritmo Longest Common Subsequence (LCS), tendo como entrada as duas strings dadas como entrada pelo exercício. Este algoritmo era indicado para a solução deste exercício, pois, ele encontra o comprimento e os caracteres da subsequência mais longa que ocorre em duas ou mais sequências de entrada, numa ordem relativa, mas não necessariamente contínua, como dito no enunciado do problema.

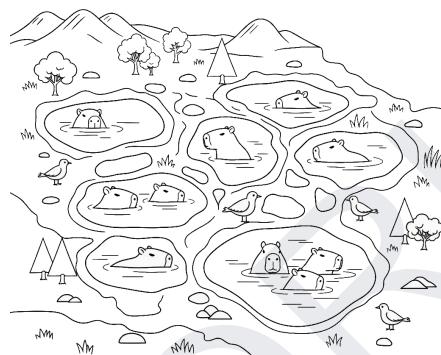
Complexidade $O(m*n)$

Pessoa Autora: João Marcos Alves

Problema E
Juntando Capivaras

Capivaras adoram lagoas! Isso é um fato! E o que elas mais gostam é de estar em casais! No entanto, nem sempre encontrar a capivarinha da sua vida é uma tarefa fácil. Geralmente, é necessário se deslocar entre lagoas através de rios, e este processo pode ser longo e desigual para algumas capivarinhas.

As lagoas são conectadas por rios e, devido à correnteza, só é possível viajar em um sentido do rio. Ou seja, às vezes é possível sair de uma lagoa X e ir até uma lagoa Y, mas pode não ser possível sair da lagoa Y e ir até a lagoa X. Isso dificulta muito que duas capivarinhas se encontrem.



Pensando nisso, ajude-as, encontrando um ponto de encontro ideal para que ambas as capivarinhas realizem o menor número de deslocamentos possível e se encontrem em um ponto ideal para ambas!

Assumindo que duas capivarinhas se encontram, respectivamente, na lagoa A e B, o objetivo é encontrar uma lagoa C, tal que seja acessível tanto de A quanto de B, e que a soma do número de deslocamentos necessários para ir de A para C e do número de deslocamentos necessários para ir de B para C seja minimizada. Note que C pode ser igual a A ou B ou ambos.

Você terá a lista de todos os rios disponíveis e uma lista de consultas consistindo de pares de lagoas onde elas vivem. Para cada consulta, você deve calcular o número mínimo total de deslocamentos que são necessários para que elas se encontrem.

Entrada

Cada caso de teste é descrito usando várias linhas. A primeira linha contém um inteiro N, que representa o número de lagoas ($2 \leq N \leq 10^5$). As lagoas são identificadas por diferentes inteiros de 1 a N. A segunda linha contém N inteiros F_i , onde F_i indica que um rio com sentido da lagoa i é para a lagoa F_i ($1 \leq F_i \leq N$, $F_i \neq i$ para $i = 1, 2, \dots, N$). A terceira linha contém um inteiro Q, que representa o número de consultas ($1 \leq Q \leq 10^5$). Cada uma das próximas Q linhas descreve uma consulta com dois inteiros A e B, indicando a lagoa de origem de cada um dos integrantes do casal ($1 \leq A, B \leq N$). Em cada caso de teste, se for possível se deslocar da lagoa X até a lagoa Y, o número máximo de deslocamentos necessários deverá ser 10^4 .

Saída

Para cada caso de teste, imprima Q linhas. Na enésima (i-ésima) linha, escreva um inteiro com a resposta para a enésima (i-ésima) consulta. Se o casal correspondente puder se encontrar através dos rios, escreva o número mínimo total de deslocamentos que ele deve fazer. Se for impossível para o casal se encontrar, escreva o número "-1".

Exemplos

Entrada	Saída
3	1
2 1 2	2
3	0
1 2	-1
1 3	3
1 1	3
7	-1
2 1 4 5 3 5 6	1
5	
1 3	
4 7	
7 4	
6 2	
2 1	

Solução

Para a solução deste problema era necessário a aplicação de uma variação do algoritmo Depth-First Search (DFS). O algoritmo era indicado pois era necessário realizar cálculos de distância mínima entre dois nós em um grafo e identificar os componentes e os ciclos.

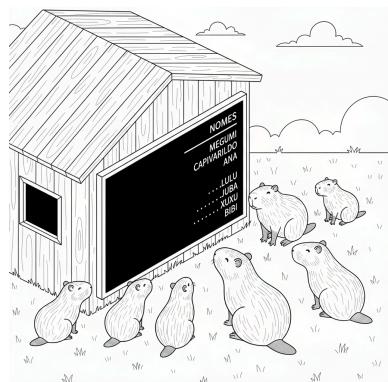
Complexidade $O((N+Q)\log N)$

Pessoa Autora: João Marcos Alves

Problema F
Capivaras Justificadas

A SECOM deste ano será sediada no clube das capivaras da UFV. No entanto, a coordenação do local não foi avisada sobre isso. No clube, há um mural digital que exibe os nomes de todas as capivaras presentes no evento, mas, como a coordenação não teve tempo para organizar essa exibição, sua tarefa é auxiliar nessa formatação.

O mural deve mostrar a lista de nomes de forma organizada: todos alinhados à direita, para transmitir uma boa impressão.



Entrada

A entrada contém diversos casos de teste. A primeira linha de cada caso de teste contém um inteiro N ($1 \leq N \leq 50$), que indica o número de nomes que virão a seguir. Cada um dos N nomes contém no mínimo 1 e no máximo 50 letras (maiúsculas ou minúsculas). O fim da entrada é indicado por N = 0.

Saída

Para cada caso de teste, imprima os nomes inserindo tantos espaços quanto forem necessários à esquerda de cada nome, de modo que todos apareçam alinhados à direita e na mesma ordem da entrada. Deixe duas linhas em branco entre os casos de teste. Não coloque espaços no final das linhas, nem insira espaços desnecessários à esquerda: pelo menos uma das linhas impressas em cada caso deve começar diretamente com uma letra.

Exemplos

Entrada	Saída
3	
MEGUMI	BIBA
CAPIVARILDO	CAPIVARILDO
ANA	ANA
4	
LULU	LULU
JubA	JubA
XUXU	XUXU
BIBI	MIMI
0	

Solução

A ideia é encontrar o tamanho do maior nome em cada caso de teste. Em seguida, para cada nome, calcular quantos espaços devem ser inseridos à esquerda para que o texto ocupe exatamente esse tamanho máximo e então imprimir os nomes nessa ordem, com duas impressões em branco entre cada caso de teste.

Complexidade

A leitura e formatação de cada caso é $O(N \cdot L)$, onde N é o número de nomes ($N \leq 50$) e L é o tamanho máximo de cada nome ($L \leq 50$).

Pessoa Autora: Ingrid Almeida

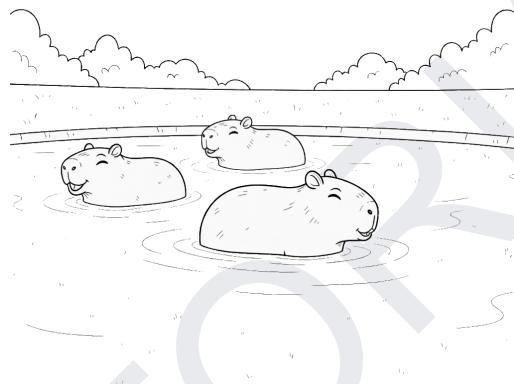
EDITORIAL

Problema G

Capivaras na Lagoa

A Universidade Federal de Viçosa (UFV) campus Florestal, conhecida por sua beleza natural e suas lagoas serenas, está prestes a receber novas e encantadoras moradoras. A partir de agora, as lagoas da universidade não serão apenas cenários para a contemplação, mas lares vibrantes para as capivaras, símbolo da universidade.

No entanto, para garantir que essa estadia seja saudável para as novas moradoras, que apesar de serem animais sociáveis, não gostam de ficar amontoadas, é necessário pensar em seu espaço pessoal. Elas valorizam o seu espaço pessoal, sendo esse espaço necessário para poderem relaxar, se alimentar e aproveitar a vida sem o estresse da superlotação.



É aqui que você entra. Sua missão é crucial: desenvolver um programa que calcule o número máximo de capivaras que podem ser acomodadas confortavelmente em cada lagoa, respeitando essa necessidade de espaço. Para definir isso, você receberá informações sobre a área total da lagoa, a área ocupada por cada capivara, a distância mínima entre elas para que tenham seu espaço pessoal, e o número de capivaras que se pretende alocar em cada lagoa, e assim informar se é possível ou não. Pense nas capivaras como círculos que, para se sentirem confortáveis, não podem se aproximar demaisumas das outras.

Entrada

A entrada consistirá em várias linhas, cada uma descrevendo uma lagoa diferente. Cada linha conterá quatro valores inteiros: L ($100 \leq L \leq 10^6$), representando a área da lagoa, C ($1 \leq C \leq 10^6$), representando a área ocupada por cada capivara, N ($1 \leq N \leq 10^6$), representando o número de capivaras que deseja-se alocar e D ($1 \leq D \leq 10^6$), sendo a distância mínima entre as capivaras. A entrada será finalizada por uma linha contendo quatro zeros, indicando que não há mais lagoas a serem analisadas.

Saída

Para cada conjunto de dados de entrada, seu programa deve imprimir uma única linha com “N” para caso não seja possível alocar todas as capivaras, e “S” caso contrário.

Exemplos

Entrada	Saída
1000 5 20 2	S
100 5 20 5	N
1500 6 30 3	S
0 0 0 0	

Solução

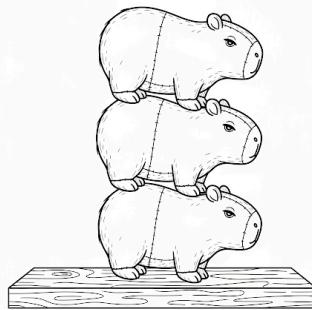
A área total de uma lagoa precisa ser calculada levando em conta que cada capivara exige um espaço extra ao seu redor para manter a distância mínima. Essa área adicional, que podemos chamar de "área de conforto", é crucial para o bem-estar dos animais. Para simplificar o cálculo, considere que as lagoas são retangulares e que a distribuição das capivaras segue um padrão de grade uniforme. A "área de conforto" de cada capivara é um círculo de raio igual à metade da distância mínima ($D/2$). Para calcular a área total de exclusão, você deve somar as áreas de conforto individuais. Ao final, basta multiplicar a "área de conforto" das capivaras, pelo número de capivaras esperado, e verificar se o mesmo ultrapassa a área da lagoa.

Complexidade: $O(1)$

Pessoa Autora: Estela Miranda

Problema H
Empilhando Capivaras

A Universidade Federal de Viçosa (UFV) campus Florestal recebeu recentemente uma novidade: uma loja de lembrancinhas. Nessa loja, têm-se diversos itens: chapéus, camisetas, o famoso doce de leite Viçosa e um item muito querido: pelúcias de capivara, símbolo da universidade! As pelúcias de capivara foram desenvolvidas de maneira anatômica, de forma que podem ser empilhadas, como mostrado abaixo. No entanto, devido à prateleira onde estão, só pode ser adicionado um certo número máximo de capivaras, em razão da sua altura máxima.



Você, então, foi convidado a ajudar as pessoas que trabalham na loja a criar um programa que diga quantas capivaras podem ser empilhadas em cada prateleira, dadas a altura das capivaras e a altura da prateleira.

Entrada

A entrada terá dois números, separados por um espaço em branco, H ($5 \leq H \leq 100$), que representa a altura da prateleira em centímetros, e N ($5 \leq N \leq 15$), que representa a altura de cada capivara em centímetros.

Saída

Sua saída deve apresentar um único número inteiro, representando o número de capivaras que podem ser empilhadas na prateleira.

Exemplos

Entrada	Saída
100 10	10
50 15	3

Solução

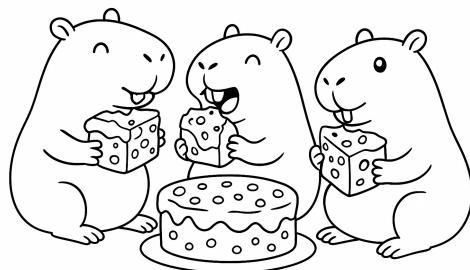
Para este problema, é necessária apenas uma divisão inteira entre os dois valores de entrada, visto que o número de pelúcias de capivara fica exatamente uma em cima da outra.

Complexidade: $O(1)$

Pessoa Autora: Estela Miranda

Problema I
Capicomilança das Capivaras

Carbucci da Silva é uma grande competidora de desafios de quem come mais e ela quer entrar em um campeonato de comer bolos de frutas onde capivaras do mundo inteiro virão para disputar o título de Rainha da Comilança. Como se trata de um evento mundial, os bolos que elas terão que comer terão as mais variadas formas e tamanhos, dignos da grandiosidade do evento, por isso Carbucci quer fazer um treinamento rigoroso para obter o melhor resultado possível e se consagrar a Rainha da Comilança.



Para ajudar a Carbucci da Silva a treinar para esta competição, você deve escrever um programa que fale para ela a melhor forma de comer todos os pedaços de fruta do bolo com o menor número de mordidas. No entanto, existem algumas regras especiais. Pode iniciar a comilança por qualquer fruta. O bolo possui um formato peculiar: nem todas as frutas podem ser alcançadas diretamente, de modo que, para chegar a determinadas frutas, é necessário ter comida outras antes, desbloqueando assim novos caminhos e estes caminhos não são lineares, Carbucci pode escolher qualquer fruta que esteja ligada a uma que já foi comida anteriormente. Além disso, comer a fruta em si não conta como mordida; o que realmente importa é o número de mordidas de deslocamento necessárias para ir de uma fruta até outra.

Entrada:

A entrada terá um número N ($1 \leq N \leq 10^6$) indicando quantos pedaços de fruta o bolo possui, M ($1 \leq M \leq 10^6$) indicando quantos caminhos existem entre pares de pedaços de fruta, seguidas de N linhas que contém três números inteiros P_1 e P_2 ($1 \leq P_1, P_2 \leq 10^8$), que representam pedaços de fruta e Q ($1 \leq Q \leq 10^8$) a quantidade de mordidas para sair de um pedaço e ir para o outro (Q de $P_1 \rightarrow P_2 = Q$ de $P_2 \rightarrow P_1$).

Saída:

Sua saída deve conter um único número inteiro que representa a menor soma dos números de mordidas para consumir todos os pedaços de fruta.

Exemplos

Entrada	Saída
6 9 4 5 7 1 2 1 1 3 4 4 2 2 3 4 4 1 4 3 3 2 4 5 6 5 4 6 4	16
4 6 1 3 2 1 2 20 3 4 1 4 2 5 1 4 8 2 3 6	8

Solução: Este problema se baseia em basicamente em uma construção gulosa de MSP (Minimal Spanning Tree), sendo o objetivo dele passar por todos pedaços de fruta (vértices) percorrendo a menor distância entre eles (arestas) e por fim somar a distância percorrida para chegar na MSP final.

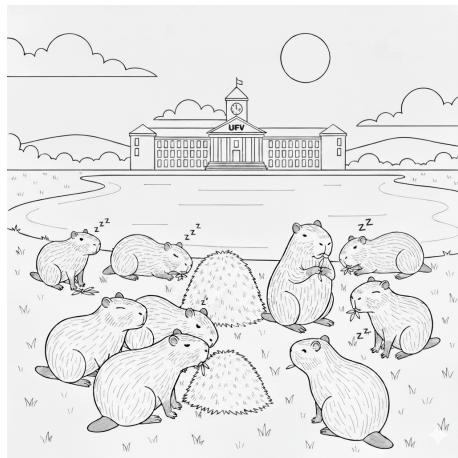
Complexidade: $O(N^2)$

Pessoa Autora: Douglas Diones

Problema J

Sonecas & Mastigações

Na Universidade Federal de Viçosa (UFV) existe um lago famoso, ponto turístico da cidade e cartão-postal do campus. Quem passa por lá sabe bem quem manda no pedaço: as capivaras. Elas passam os dias entre cochilos à sombra, mergulhos estratégicos e, claro, longas sessões de mastigação nos gramados do entorno.



Cada capivara tem um ritmo próprio — e, para não assustar turistas e manter o gramado sempre saudável, elas seguem uma rotina de ciclos: períodos de comer e períodos de descanso. Em dias de sol, aparece um bando; em outros, só meia dúzia. E a dúvida que sempre surge é: “Se temos G unidades de grama e N capivaras revezando entre comer e descansar, em quanto tempo o campo fica totalmente devorado?”

Sua missão é ajudar as capivaras da UFV a calcular o menor tempo inteiro T (em minutos) para que, somando o que todas comem ao longo dos seus ciclos, a quantidade total de grama consumida seja pelo menos G .

Durante o tempo T , a capivara i segue repetidamente o ciclo:

- come por A_i minutos a uma taxa de C_i gramas/minuto;
- depois descansa por B_i minutos (sem comer).

Esse padrão se repete quantas vezes couber no intervalo de T , e no último ciclo parcial ela pode comer apenas até completar os A_i minutos de mastigação.

Entrada

- Um inteiro N ($1 \leq N \leq 10^5$), o número de capivaras.
- Um inteiro G ($1 \leq G \leq 10^{12}$), a quantidade total de grama a ser comida.
- Em seguida, N linhas, cada uma com três inteiros C_i, A_i, B_i ($1 \leq C_i \leq 10^9, 1 \leq A_i, B_i \leq 10^6$).
 - C_i : taxa de consumo (gramas por minuto) da capivara i .
 - A_i : minutos consecutivos que a capivara i consegue comer antes de descansar.
 - B_i : minutos de descanso da capivara i antes de poder voltar a comer.

Saída

Imprima um único inteiro: o menor T (em minutos) tal que a soma do que todas as capivaras conseguem comer até o minuto T seja $\geq G$.

Exemplo

Entrada	Saída
3 2 0 3 1 1 2 2 3 5 1 4	6

Solução

Modelamos “tempo suficiente?” como um predicado monotônico: dado um tempo T, cada capivara contribui com o que consegue mastigar dentro dos seus ciclos (come A_i min a C_i g/min e descansa B_i min), somamos as contribuições das N capivaras e verificamos se o total $\geq G$. Como, ao aumentar T, o total nunca diminui, aplicamos busca binária em T: escolhemos um limite superior seguro (por exemplo, o tempo que a melhor capivara, pela taxa média $C_i \cdot A_i / (A_i + B_i)$ levaria sozinha para comer G) e binarizamos em $[0, h]$ até achar o menor T viável. A checagem para um T é apenas percorrer as N capivaras e acumular suas contribuições.

Complexidade

Cada checagem custa $O(N)$. A busca binária realiza $O(\log h)$ checagens, onde h é um limite superior para a resposta. Logo, o tempo total é $O(N \log h)$.

Pessoa Autora: Paulo Henrique Carneiro

Problema K

Organizando Capivaras

A Universidade Federal de Viçosa (UFV), campus Florestal, não é apenas um centro de excelência acadêmica, mas também o lar de uma das mais queridas populações de capivaras do país. Esses animais, que vivem livremente pelos vastos gramados, interagem constantemente com as diversas lagoas do campus. Para um censo ecológico, os biólogos do departamento de biologia da universidade precisam alocar as capivaras às lagoas de forma estável.

Uma alocação é considerada estável se não houver um par de capivara e lagoa que, embora não estejam alocados entre si, se preferem mutuamente em vez de seus parceiros atuais. Em outras palavras, um par de capivara e lagoa instável acontece quando a capivara prefere a lagoa a sua parceira atual, e a lagoa também prefere essa capivara a sua parceira atual. Para garantir o bem-estar e a tranquilidade das capivaras, os biólogos buscam uma alocação que seja mutuamente satisfatória.

- Chico prefere a Lagoa dos Pescadores à Lagoa do Bosque.
- Belinha prefere a Lagoa do Bosque à Lagoa dos Pescadores.
- A Lagoa dos Pescadores prefere Chico à Belinha.
- A Lagoa do Bosque prefere Chico à Belinha.

Com base nessas preferências, a única alocação estável possível é que Chico fique na Lagoa dos Pescadores e Belinha na Lagoa do Bosque. Qualquer outra combinação geraria um par instável.

Sua tarefa é criar um programa que, dadas as listas de preferências de todas as lagoas e capivaras, encontre uma alocação estável.

Entrada

A entrada consiste em várias linhas, seguindo este formato: a primeira linha contém um inteiro N ($2 \leq N \leq 1000$), indicando o número de capivaras e lagoas. As próximas N linhas contêm as preferências das capivaras. Cada linha terá o nome da capivara, seguido pelos nomes das N lagoas, ordenados por sua preferência (da mais preferida para a menos preferida). As N linhas seguintes contêm as preferências das lagoas. Cada linha terá o nome da lagoa, seguido pelos nomes das N capivaras, ordenados por sua preferência.

Saída

A saída deve apresentar o resultado da alocação encontrada. Cada linha deve conter um par, onde o primeiro elemento é o nome da capivara e o segundo é o nome da lagoa à qual ela foi alocada.

Exemplos

Entrada	Saída
2 Belinha Bosque Pescadores Chico Pescadores Bosque Bosque Chico Belinha Pescadores Chico Belinha	Belinha Bosque Chico Pescadores

4 Alfredo Palmeiras Bosque Corrego Floresta Belinha Bosque Floresta Corrego Palmeiras Cacau Corrego Bosque Palmeiras Floresta Chico Palmeiras Floresta Bosque Corrego Bosque Belinha Chico Alfredo Cacau Corrego Cacau Alfredo Chico Belinha Floresta Chico Belinha Alfredo Cacau Palmeiras Alfredo Cacau Belinha Chico	Alfredo Palmeiras Belinha Bosque Cacau Corrego Chico Floresta
---	--

Solução

Este problema é uma variação do problema do Emparelhamento Estável (ou Stable Matching). Ele é um modelo matemático para encontrar um emparelhamento estável entre dois conjuntos de elementos com base em suas preferências. Uma correspondência é considerada estável quando não há um par de elementos que prefeririam um ao outro em vez de seus parceiros atuais, o que poderia levar à instabilidade no sistema. O algoritmo de Gale-Shapley é um método para encontrar uma correspondência estável, com aplicações que vão desde a alocação de residentes em hospitais até a distribuição de vagas em universidades.

Complexidade: $O(n^2)$

Pessoa Autora: Estela Miranda