INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO

VI MARATONA DE PROGRAMAÇÃO INTERIF 2023

PRIMEIRA FASE

Caderno de Problemas

Informações Gerais

A) Sobre a entrada

1) A entrada de seu programa deve ser lida da entrada padrão.
2) A entrada é composta de um ou mais casos de teste, depende do problema.
3) 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.
4) Cada linha, incluindo a última, contém o caractere final-de-linha.
5) O final da entrada pode coincidir com o final do arquivo ou com uma entrada determinada

B) Sobre a saída

1) A saída de seu programa deve ser escrita na saída padrão.
2) Espaços em branco só devem ser colocados quando solicitado.
3) Cada linha, incluindo a última, deve conter o caractere final-de-linha.

C) Regras

1) Só é permitida a comunicação entre os membros de um mesmo grupo.
2) Não é permitida a comunicação com o técnico (coach) do time.
3) Eventuais dúvidas sobre a prova utilizar o menu “clarification” do sistema de submissão.

D) Ambiente computacional

O sistema de correção das submissões será executado utilizando a distribuição Ubuntu GNU/Linux 20.04.2 LTS amd64, tendo os seguintes compiladores/interpretadores configurados:

- C - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- C++ - gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04)
- Python 3 - Python 3.8.10
- Java - openjdk-11.0.11
- C# - mono JIT 6.12

Problema A
O Bruxo e os Frascos

Por Márcio Kassouf Crocomo (IFSP – Campus Piracicaba)

Arquivo: main.[c|cpp|java\cs\py]

***Timelimit:** 1*

Em um mundo fictício, bruxos são conhecidos por estudarem a vida toda sobre criaturas monstruosas e os métodos mais adequados para derrota-las em combate. Uma das estratégias usadas por estes bruxos é a de passar um óleo nas lâminas de suas espadas, deixando-as mais eficazes ao serem utilizadas contra determinadas criaturas. O tipo de óleo que deve ser utilizado varia de acordo com a criatura enfrentada.

O bruxo Geraldo, muito estudioso, percebeu que existe um padrão no nome das criaturas que permite identificar o óleo correto a ser utilizado contra cada uma. Assim, Geraldo numerou 3 frascos de óleo como 0, 1 e 2. Para saber qual óleo é o mais adequado a ser usado, o bruxo conta as vogais existentes no nome da criatura e, então, olha para o resto da divisão deste número por 3. O resultado (o resto da divisão por 3) é o número do frasco que contém o óleo mais eficaz para se utilizar!

Cansado de fazer estas contas, Geraldo contrata um programador para criar um programa que o ajude em suas caçadas (Geraldo sempre leva um notebook em sua mochila). Crie um programa que ajude Geraldo a identificar o óleo correto a ser utilizado.

Entrada

A entrada possui uma única linha contendo o nome da criatura, que possui no mínimo 1 e no máximo 20 caracteres. Os caracteres podem ser letras maiúsculas, minúsculas ou caracteres de espaço. Nenhuma entrada possui acentuação.

Saída

A palavra “frasco” (com todos os caracteres minúsculos) seguida de um espaço em branco e o número do frasco que deve ser utilizado, dado pelo resto da divisão do número de vogais no nome por 3.

Exemplos de Entradas	Exemplos de Saídas
Manticora	frasco 1
Grifo	frasco 2
Dragao Fofao	frasco 0
Liche bondoso bacanA	frasco 2

Problema B
Classificação para corrida de kart

Por Tiago Alexandre Docusse (IFSP – Campus Barretos)

Arquivo: classificacao.[c|cpp|java\cs\py]

***Timelimit:** 1*

O Instituto Federal de São Paulo irá organizar uma nova corrida de kart, e desta vez, as posições de largada da corrida serão definidas através de um treino classificatório, sendo que os pilotos mais velozes no treino classificatório largarão na frente na corrida. O treino classificatório ocorre logo antes da corrida, então, para otimizar as contas de quem foi mais rápido no treino e definir o *grid* de largada, a organização da corrida pediu a ajuda do seu grupo para desenvolver um programa que receba os dados do treino classificatório, em tempo real e, ao término da classificação, exiba uma lista com as posições de largada para a corrida.

Entrada

A primeira linha da entrada é formada por três números, a, b e c, que representam, respectivamente, a quantidade de pilotos participantes do treino classificatório, a quantidade de voltas a serem analisadas, e a quantidade de voltas inválidas registradas, tais que $0 < a < 40$, $0 < b < 1000$, e $0 \leq c < b$. Em seguida, são informadas *a* linhas, contendo o nome dos pilotos participantes do treino classificatório, todos únicos e formados por apenas uma palavra. Logo após, são informadas *b* linhas representando voltas registradas no treino classificatório, contendo a sigla de um piloto e, separado por um espaço em branco, o tempo de volta registrado por aquele piloto. A sigla do piloto é formada pelas três letras iniciais do seu nome. Garante-se que não há siglas repetidas e que todos os nomes de pilotos possuem ao menos três letras. O tempo de volta a ser lido está sempre no formato: $m{:}\,ss.fff$, sendo *m* a quantidade inteira de minutos, com ao menos um dígito, em que $0 \leq m < 100$; *ss* a quantidade inteira de segundos, sempre com dois dígitos, sendo $00 \leq ss < 60$; e *fff* a quantidade inteira de milissegundos, sempre com três dígitos, em que $000 \leq fff \leq 999$. As voltas registradas são feitas de acordo com o tempo em que elas foram registradas, ou seja, a volta registrada na linha *i* sempre terá sido realizada antes da volta registrada na linha *i+1*. Após a informação das voltas registradas, seguem-se *c* linhas, contendo informações sobre as voltas invalidadas por motivos quaisquer, contendo a sigla do piloto, um espaço em branco, e o tempo de volta, formatados como no caso anterior. Garante-se que todo piloto tenha registrado ao menos uma volta válida.

Saída

A saída do programa deve conter *a* linhas, exibindo a posição (iniciando em 1 e incrementada em uma unidade), seguida de um espaço, seguida do nome do piloto com a primeira letra em caixa alta e as demais em caixa baixa, seguido de um espaço, seguido do tempo de volta mais rápida do piloto, no formato $m{:}\,ss.fff$, sendo *m* exibido com ao menos 1 dígito, *ss* sempre exibido com dois dígitos e *fff* sempre exibido com três dígitos. A lista de pilotos deve ser ordenada pelo menor tempo de volta válida para o maior tempo de volta válida. Caso pilotos tenham registrado o mesmo tempo de volta, deverá ser classificado à frente o piloto que marcou o tempo de volta primeiro.

Exemplos de Entradas	Exemplos de Saídas
4 12 4 Bryan Luiggy Marina Rafael BRY 1:52.438 LUI 1:52.520 RAF 1:52.640 MAR 1:51.956 BRY 1:52.350 LUI 1:52.389 MAR 1:52.393 LUI 1:52.680 RAF 1:53.420 MAR 1:52.389 RAF 1:52.399 BRY 1:51.930 BRY 1:51.930 LUI 1:52.680 MAR 1:51.956 RAF 1:53.420	1 Bryan 1:52.350 2 Luiggy 1:52.389 3 Marina 1:52.389 4 Rafael 1:52.399

Problema C
O Tesouro dos Números Primos

Por Cássio Agnaldo Onodera (IFSP – Campus Birigui)

Arquivo: primos.[c|cpp|java\cs\py]

***Timelimit:** 2*

Havia uma vez um reino mágico chamado Numerolândia, onde os números eram mais do que meros símbolos matemáticos; eles eram seres vivos com personalidades e propósitos únicos. No coração de Numerolândia, estava a Torre dos Números, uma estrutura majestosa que abrigava os números primos mais poderosos e especiais de todos.

Os números primos eram conhecidos por sua singularidade e raridade. Eles eram os guardiões dos segredos matemáticos do reino e eram considerados verdadeiros tesouros. No entanto, a bruxa malvada Divisória lançou um feitiço sombrio sobre Numerolândia, causando a dispersão dos números primos em todo o reino.

O Rei Matemático, desesperado para restaurar a ordem em Numerolândia, convocou você, um habilidoso programador, para ajudar a recuperar os números primos perdidos. Para fazer isso, você precisa criar um programa mágico que liste todos os números primos positivos até um valor inteiro N, a fim de trazer de volta a magia dos números primos à Numerolândia.

A jornada começa aqui. Você aceita o desafio do Rei Matemático e partirá em busca dos números primos perdidos para devolver a Numerolândia a sua antiga glória?

Sua missão: Escrever um programa que liste todos os números primos positivos até o valor inteiro N e, assim, restaurar a magia dos números primos em Numerolândia.

Este texto foi gerado com a assistência da inteligência artificial da OpenAI (https://chat.openai.com).

Entrada

Um único número inteiro N (1 <= N < 10^5).

Saída

Todos os números primos positivos menores ou iguais a N, em ordem crescente, separado por espaços.

Exemplos de Entradas	Exemplos de Saídas
10	2 3 5 7
29	2 3 5 7 11 13 17 19 23 29
100	2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97

Problema D
Jujubas

Por Giovana Yuko Nakashima (IFSP – Campus Salto)

Arquivo: jujubas.[c|cpp|java\cs\py]

***Timelimit:** 1*

Jujubas ou balas de goma são snacks doces confeitadas em diversas cores e sabores. A empresa do Sr. João fabrica os sabores abacaxi, morango e uva, e as comercializa em pacotes de diversos tamanhos: 1kg, 500g, 200g; mas, o lucro maior vem das embalagens em tubo, com 8 unidades sortidas. Um estudo sobre os tubos mais vendidos identificou que, além dos sabores (e cores), há padrões de disposição das gomas que influenciam a escolha dos clientes. Por esse motivo, a partir de agora, para todos os lotes de fabricação, Sr. João quer saber quantos tubos apresentam uma semelhança mínima com os padrões identificados.

Entrada

A entrada é composta por várias linhas com números inteiros. Na primeira linha, é fornecido o padrão de tubo **P** com oito números inteiros (p_1, p_2, p_3, ..., p_8) separados por espaço. Cada bala é representada por um número inteiro: 1 – sabor abacaxi, 2 – sabor morango, 3 – sabor uva. Na segunda linha, é fornecido um inteiro **N** ($1 \leq N \leq 100$) que corresponde à quantidade de tubos do lote a analisar. As **N** linhas seguintes apresentam a disposição das balas em cada um dos tubos $\mathbf{T}_i$ ($1 \leq i \leq N$) do lote. Na próxima linha é fornecido um inteiro **M** ($1 \leq M \leq 100$) que indica quantas análises devem ser realizadas. Nas M linhas seguintes, são informados os percentuais mínimos a considerar.

Saída

A saída consiste de uma linha para cada análise, com o percentual de tubos do lote que apresentam semelhança maior ou igual à semelhança mínima solicitada. O percentual deve ser exibido com duas casas decimais.

Exemplos de Entradas	Exemplos de Saídas
1 2 3 2 1 3 3 1 3 1 3 3 2 2 2 3 3 1 2 1 3 3 1 2 2 3 3 2 1 1 3 3 1 1 40	66.67
1 3 3 2 1 3 3 1 10 3 3 2 1 2 2 3 1 2 2 1 1 3 1 3 2 3 3 1 2 2 1 3 1 1 3 3 2 3 3 3 2 1 2 2 1 2 3 3 3 2 1 2 1 3 2 1 1 1 2 1 2 2 1 1 2 2 1 2 2 2 1 1 3 1 3 3 1 3 3 2 3	40.00 10.00 0.00

3 2 3 2 3 3 3 2 3 50 60 80	

Problema E
Alguém ainda sabe o que é um cheque?

Por Daniel Corrêa Lobato (IFSP – Campus São José do Rio Preto)
Arquivo: cheque.[c|cpp|java\cs\py]
***Timelimit:** 1*

Num passado muito, muito antigo, se usava um pedaço de papel chamado cheque para pagar as pessoas. Era uma demonstração de confiança muito grande, porque você ia até uma loja, pegava um produto, e entregava um pedaço de papel, com a sua assinatura, que a loja depois trocava por dinheiro no banco. Se você tivesse, na sua conta, o valor expresso no cheque, ótimo: o cheque era descontado (o banco tirava o dinheiro da sua conta e passava para a conta de quem apresentou o cheque), e todos ficavam felizes; por outro lado, se você não tivesse dinheiro suficiente na conta para honrar o valor de face do cheque, era um problema para o vendedor (que ficava sem o dinheiro e sem o produto que o emissor do cheque já levou para casa) e para o emissor (que cometia o crime de estelionato, tipificado no código penal). Em alguns casos, quem recebeu o cheque preferia sair do banco com o dinheiro ao invés de depositar o dinheiro na conta. Nesses casos, normalmente, era desejável sair com a menor quantidade possível de cédulas, para que o volume não chamasse a atenção de terceiros. A sua função é calcular qual a menor quantidade possível de cédulas que devem ser entregues pelo banco que totalizam o valor de face do cheque. Sempre será possível trocar um cheque por uma quantidade inteira de cédulas

Entrada

Uma linha contendo um número inteiro N ($1 \leq N \leq 10$) indicando a quantidade de valores diferentes de cédulas existentes no banco; N linhas contendo valores inteiros com o valor, F, de face de cada variedade de cédula ($1 \leq F \leq 1000$, sem repetição); uma linha contendo um valor inteiro V ($0 \leq V \leq 10^{10}$) indicando o valor de face do cheque a ser descontado

Saída

Um valor inteiro, indicando a menor quantidade de cédulas que são necessárias para trocar o valor de face do cheque por cédulas

Exemplos de Entradas	Exemplos de Saídas
3 5 1 10 28	6
4 5 10 1 20 456	25

Problema F
Cavalinhos do Recantão

Por Carlos "Carlão" José de Almeida Pereira (IFSP – Campus São Carlos) e Jorge Francisco Cutigi (IFSP – Campus São Carlos)

Arquivo: recantao.[c|cpp|java\cs\py]

***Timelimit:** 1*

Otto adora futebol e acompanha de perto o Campeonato Brasileiro (Brasileirão) e o campeonato da sua vizinhança (Recantão). Ele gosta muito do quadro dos "Cavalinhos" do Fantástico e decidiu fazer o mesmo para o Recantão. Para isso, Otto precisa saber a posição do time no campeonato e a quantidade de pontos que ele fez, além de outros critérios de desempate. No Recantão a quantidade de pontos é similar ao Brasileirão, ou seja:

- Em caso de empate: as duas equipes somam 1 ponto
- Em caso de vitória: a equipe vitoriosa soma 3 pontos, enquanto a equipe derrotada não soma pontos

A classificação dos times é ordenada de maneira decrescente, ou seja, os times com mais pontos estão acima na tabela. Caso existam equipes com o mesmo número de pontos, o primeiro critério de desempate é o número de vitórias. Caso ainda permaneça empate, o segundo critério é o saldo de gols da equipe (diferença entre o número de gols que a equipe marcou e o número de gols que a equipe sofreu).

Assim como futebol, Otto também gosta muito de programação. Com isso ele decidiu fazer um programa que o ajude nessa tarefa.

Entrada

- A primeira linha contém um número inteiro **E**, que representa o número de equipes que estão disputando o Recantão.
- A segunda linha contém um número inteiro **N** que representa o número de jogos realizados
- As **N** linhas seguintes representam o resultado de cada jogo, no seguinte formato: NomeEquipe1 GolsEquipe1 X NomeEquipe2 GolsEquipe2. Por exemplo o placar **CARIACICA 2 X 3 CASCADURA** indica que a equipe CASCADURA venceu a equipe CARIACICA por três gols a dois. Considere que os nomes da equipe não possuem caracteres de espaço e que os gols são números inteiros não negativos.

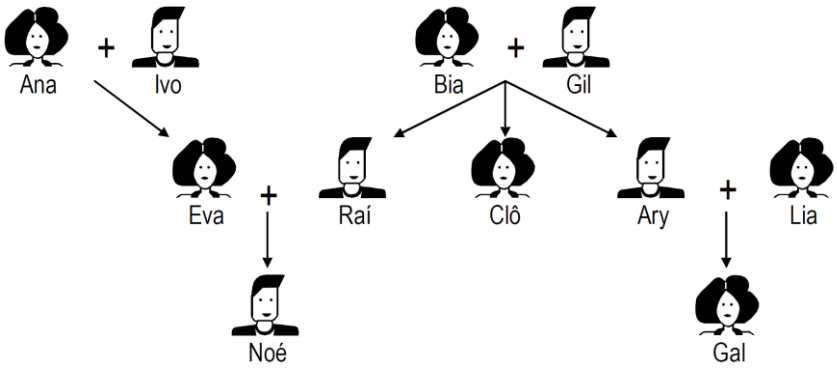
Saída

A saída deve conter **E** linhas com o nome da equipe, acompanhada da quantidade de pontos, vitórias e saldo de gols, todos separados por espaço. A saída deve ser ordenada conforme o enunciado, ou seja, seguindo os seguintes critérios nessa ordem: ordenados de forma decrescente pela quantidade de pontos da equipe seguido pelo número de vitórias e por último o saldo de gols. Assuma que não há casos de teste em que há empate em todos os critérios.

Exemplos de Entradas	Exemplos de Saídas
3 6 JAGUATIRICAFC 2 X 0 TAMANDUAFC TAMANDUAFC 1 X 0 TATUBOLAFC TATUBOLAFC 0 X 1 JAGUATIRICAFC JAGUATIRICAFC 0 X 0 TATUBOLAFC TAMANDUAFC 3 X 0 JAGUATIRICAFC TATUBOLAFC 2 X 1 TAMANDUAFC	JAGUATIRICAFC 7 2 0 TAMANDUAFC 6 2 1 TATUBOLAFC 4 1 -1
2 2 CASCADURA 2 X 0 CARIACICA CASCADURA 0 X 1 CARIACICA	CASCADURA 3 1 1 CARIACICA 3 1 -1

Problema G
Reprodução Nlogônia Island

Por Murilo Varges da Silva (IFSP – Campus Birigui)
Arquivo: nlogonia.[c|cpp|java\cs\py]
***Timelimit:** 1*



Bem-vindo a Nlogônia, uma ilha de beleza mística e segredos profundos, perdida em meio ao vasto oceano. Com praias de areias douradas que se estendem até encontrar as águas cristalinas, Nlogônia é um refúgio paradisíaco que esconde uma intrigante e delicada questão em seu cerne.

A comunidade de Nlogônia é constituída por um povo afetuoso e dedicado, cuja história remonta a gerações passadas. No entanto, um desafio persistente assola essa ilha idílica. Um fenômeno incomum e controverso assombra a população, uma situação que fez com que os laços de parentesco se entrelaçassem de maneira inesperada e problemática.

Os habitantes de Nlogônia, por razões ainda desconhecidas, enfrentam dificuldades em suas tentativas de reprodução. A ilha tem sido abalada por uma série de relacionamentos que transcendem os limites familiares tradicionais, resultando em laços de parentesco cruzados e entrelaçados. Essa complexa teia de conexões familiares emaranhadas tornou-se a raiz dos problemas de reprodução que afligem a comunidade.

Embora essa questão desafiadora tenha causado turbulência e questionamentos dentro da sociedade de Nlogônia, os habitantes permanecem unidos em sua busca por soluções. Os anciãos e líderes da ilha estão empenhados em desvendar os mistérios por trás desse fenômeno, buscando conhecimento e orientação em fontes antigas e nos sábios locais.

Apesar das adversidades, Nlogônia ainda mantém sua aura de encanto e fascínio. A ilha é um espetáculo de maravilhas naturais, com jardins exuberantes que exalam aromas exóticos, montanhas majestosas que parecem tocar o céu e uma fauna marinha diversificada que atrai exploradores e amantes da natureza de todo o mundo. Nlogônia não é apenas um local geográfico, mas também uma encruzilhada entre a tradição e o mistério, onde a busca pelo entendimento e a resolução dos problemas de reprodução são partes fundamentais da jornada. Enquanto os habitantes lutam para desvendar os enigmas que afetam sua continuidade, a ilha permanece como um lembrete de que mesmo em um paraíso aparentemente intocado, desafios complexos podem surgir, exigindo resiliência, cooperação e compreensão para serem superados.

Diante dessa situação um tanto quanto curiosa e para evitar problemas maiores os anciãos e líderes de Nlogônia pretendem desenvolver um software para avaliar se duas pessoas possuem parentesco antes de iniciar um relacionamento, você foi contratado para desenvolver um programa que dada uma árvore de parentescos e um par de pessoas, decida se eles têm parentesco ou não.

Entrada

A primeira linha da entrada possui três números inteiros **N** ($3 \leq \mathbf{N} \leq 15$), **C** ($0 < \mathbf{C} \leq N$) e **T** ($0 < \mathbf{T} \leq \mathbf{N}*(\mathbf{N}-1)$), que representam, respectivamente, o número de pessoas, o número de relações de parentesco conhecidas e o número de casos de teste. Em seguida, são fornecidas as C relações de parentesco com os dois nomes dos pais e o respectivo filho gerado. Depois das relações de parentesco são fornecidos os T casos de teste, cada um composto pelos nomes de duas pessoas que pretendem se relacionar. Cada nome de pessoal é composto por até 10 letras do alfabeto português (26 letras) e todos os nomes são separados por um único espaço entre si.

Saída

Seu programa deve imprimir, para cada caso de teste, "verdadeiro" se existir parentesco entre as pessoas e "falso" se não existir.

Exemplos de Entradas	Exemplos de Saídas
11 6 5 Ana Ivo Eva Bia Gil Rai Bia Gil Clo Bia Gil Ary Eva Rai Noe Ary Lia Gal Eva Ary Noe Gal Lia Rai Lia Noe Gal Rai	falso verdadeiro falso falso verdadeiro

Problema H
Clubinho da Isa

Por Jorge Francisco Cutigi (IFSP – Campus São Carlos)
Arquivo: clubinho.[c|cpp|java\cs\py]
***Timelimit:** 1*

O avô de Isa, o sr Décio, sabe fazer de tudo. Sabendo disso, Isa pediu para que ele construísse uma casa na árvore para que ela pudesse montar seu clubinho. Se avô gostou muito da ideia e construiu a casa do jeito que Isa sonhava. Agora, Isa fundou o clubinho, onde ela passa horas e horas brincando com seus amigos.
Isa e seus amigos não gostariam que nenhuma pessoa que não fizesse parte do clubinho pudesse entrar na casinha. Para isso, Isa inventou um código para ter certeza se ela poderia abrir a porta quando alguma pessoa batesse à porta. O código da Isa era composto por quatro números. Para que o código estivesse correto, os números deveriam ser informados em ordem crescente sendo que o último número deveria ser a soma dos três primeiros. Por exemplo, a entrada da pessoa seria liberada se ela chegasse na casinha e falasse a seguinte sequência de números: ***1 2 3 6***. Por outro lado, a entrada seria negada se a sequência falada fosse ***1 2 3 7*** ou ***1 3 2 6***.

Entrada

A entrada contém quatro linhas, em que cada linha é um número inteiro que representa um número do código da Isa.

Saída

A saída deve conter apenas uma linha. Se os números da entrada corresponderem ao código da Isa, o programa deve imprimir a string **LIBERADO**, e **NEGADO** caso contrário

Exemplos de Entradas	Exemplos de Saídas
1 2 3 6	LIBERADO

1 2 3 7	NEGADO

1 3 2 6	NEGADO