II Maratona de Programação do Inatel

Caderno de Problemas

15 de maio de 2019

A PROVA TEM DURAÇÃO DE **4:00** HORAS

Realização:

Semana da Computação

CODETROOPERS MARATONA DE PROGRAMAÇÃO

Inatel

Informações gerais

Este caderno de tarefas é composto por 13 páginas (não contando a folha de rosto), numeradas de 1 a 13. Verifique se o caderno está completo.

Regras

- Os problemas serão avaliados automaticamente pelo BOCA.
- Só é permitido usar para consulta material impresso ou *help* do sistema BOCA (apenas para dúvidas quanto enunciado ou possíveis erros do problema). Não é permitido uso de Internet ou aparelhos eletrônicos tais como celulares, pendrives, etc. Sujeito a desclassificação.
- Quando um participante julgar que tem um programa que resolve um problema, deverá submetê-lo à correção do juiz eletrônico, que compilará e executará o mesmo para uma bateria de testes desconhecida dos participantes. Um problema será considerado resolvido se, para todos os testes da bateria, devolver o resultado esperado pelo juiz. Para cada submissão o participante receberá uma resposta, que poderá ser satisfatória (e o problema está resolvido) ou indicará algum erro: resposta errada, tempo de execução excedido, erro de compilação, erro de apresentação, etc.
- Será considerado vencedor aquele que resolver a maior quantidade de problemas nas 4 horas de competição. Empates no número de problemas resolvidos serão classificados pelo tempo acumulado. Ganhará aquele que tiver o menor tempo acumulado. O tempo acumulado será dado pela soma dos tempos corrigidos somente dos problemas corretamente resolvidos pelo participante. O tempo corrigido de um problema será dado pelo número de minutos decorridos desde o início da competição até o momento da submissão correta somado a uma penalidade de 20 minutos por submissão incorreta feita anteriormente neste problema. Em caso de empate, será considerado vencedor o participante que tiver a primeira submissão correta. Persistindo o empate, a organização fará um sorteio entre os participantes envolvidos.
- A organização da competição será responsável pela decisão de qualquer caso não previsto.

Nome do programa

As soluções para os problemas deveram ser feitas na linguagem C++ com o nome do arquivo com extensão *.cpp*, como indicado abaixo do título de cada problema.

Entrada

- A entrada deve ser lida da entrada padrão.
- Não é necessário realizar crítica de entrada de dados. Todos os valores de entrada estarão nos limites descritos de cada problema.
- A entrada consiste em exatamente um caso de teste, que é descrito usando uma quantidade de linhas que depende do problema. O formato da entrada é como descrito em cada problema. A entrada não contém nenhum conteúdo extra.
- Todas as linhas da entrada, incluindo a última, terminam com o caractere de fim de linha (\n).
- A entrada não contém linhas vazias.
- Quando a entrada contém múltiplos valores separados por espaços, existe exatamente um espaço em branco entre dois valores consecutivos na mesma linha.

Saída

- A saída deve respeitar o formato especificado no enunciado. A saída não deve conter nenhum dado extra.
- Todas as linhas da saída, incluindo a última, devem terminar com o caractere de fim de linha (\n).
- Quando uma linha da saída apresentar múltiplos valores separados por espaços, deve haver exatamente um espaço em branco entre dois valores consecutivos.

Problema A. Apenas uma pizza

Nome do arquivo fonte: `pizza.cpp`

Você está treinando para a maratona de programação do Inatel com os irmãos Matheus e Vinicius. Depois de algumas horas de treino intenso, vocês resolvem dar uma pausa e pedir uma pizza. Vocês decidem pedir uma única pizza grande porém estão com problemas em escolher qual pizza pedir. Você sabe o seu sabor favorito, mas Matheus e Vinicius tem outras preferências: Como Matheus está de dieta, então ele quer uma pizza com menos calorias o possível. Já Vinicius só quer contrarir seu irmão, então ele irá escolher a pizza com mais calorias possível.

Como votar na pizza que cada um quer não irá dar em nada, um sistema de veto será utilizado. Então vocês irão vetar as pizza que não querem comer, até que reste só uma. Primeiro Matheus veta uma pizza que ela não quer, então Vinicius veta outra e depois você veta outra, então Matheus denovo, e depois Vinicius, e assim segue até que reste só uma.

Lembre-se, Matheus sempre irá vetar a pizza mais calórica ainda disponível. Vinicius irá vetar a pizza com menor quantidade de calorias ainda não vetada. Você irá tentar vetar de tal forma que sua pizza seja a última.

Entrada

A entrada começa com a quantidade de pizzas N ($1 \leq N \leq 10^5$) e o índice da sua pizza favorita P ($1 \leq P \leq N$) (indexado em 1). Então segue a descrição das pizzas, cada uma dado em uma linha. A descrição consiste de um inteiro C ($1 \leq C \leq 10^6$) , indicando a quantidade de calorias da pizza seguido de seu nome (até 100 caracteres sem espaços ou caracteres especiais). As pizzas serão dadas em ordem crescente, da menor para maior em relação as calorias. Não terá duas pizzas distintas com a mesma quantidade de calorias na entrada.

Saída

Seu programa deverá exibir em uma linha a palavra `SIM` caso você consiga influenciar os outros de forma que sua pizza favorita seja selecionada. E a palavra `NAO` caso não seja possível.

Exemplos

Entrada	**Saída**
5 2 500 Brocolis 600 Portuguesa 700 Bacon 800 Marguerita 900 Catupiresa	SIM

Entrada	**Saída**
5 4 500 Brocolis 600 Portuguesa 700 Bacon 800 Marguerita 900 Catupiresa	NAO

Problema B. Bibika International

Nome do arquivo fonte: `bibika.cpp`

Bibika International é uma agência de turismo de baixo custo que tem como objetivo dar oportunidade às pessoas de baixa renda a conhecerem outros países. O plano básico para uma família viajar para os Estados Unidos, por exemplo, tem como custo 300 reais fixos (despesas contratuais) e mais 100 reis por pessoa.

Bibika tem recebido muitos pedidos de viagem e está se perdendo nos cálculos. Ajude-a descobrir o valor total para uma família dado a quantidade de membros da mesma.

Entrada

A entrada é composta de uma linha contendo um único inteiro Q ($1 \leq Q \leq 10$) representando a quantidade de pessoas da família que irão viajar.

Saída

Escreva na saída uma linha contendo o valor total da viagem para os Estados Unidos utilizando a agência de turismo Bibika International.

Exemplos

Entrada	**Saída**
1	400

Entrada	**Saída**
3	600

Problema C. Coleção de Mariana

Nome do arquivo fonte: `colecao.cpp`

Marianazinha possui uma coleção de figurinhas e, como na maioria das coleções, sempre existem algumas figurinhas repetidas. Cada figurinha possui um número inteiro que a identifica (duas figurinhas iguais possuem o mesmo número de identificação).

Marianazinha tem o costume de pegar N figurinhas da sua coleção e colocá-las lado a lado em cima da mesa de forma que formem uma sequência perfeita. Para Marianazinha, uma sequência perfeita é quando nenhuma figurinha vizinha se repete.

Entrada

A entrada é composta por duas linhas. Na primeira delas temos um inteiro N ($1 \leq N \leq 20$), indicando a quantidade de figurinhas que Mariazinha usou para montar a sequência. A segunda linha possui N inteiros F_i ($1 \leq F_i \leq 100$) separados por um espaço, representando a sequência de Mariazinha.

Saída

Escreva na saída uma linha com a palavra `perfeita` caso as figurinhas de Marianazinha formem uma sequência perfeita ou `imperfeita` caso contrário.

Exemplos

Entrada	**Saída**
2 1 2	perfeita

Entrada	**Saída**
10 8 4 2 3 20 20 2 4 1 8	imperfeita

Problema D. Diferente é bom

Nome do arquivo fonte: `diferente.cpp`

Um sábio uma vez disse a Trooper que *"Diferente é bom"* e desse dia em diante Trooper quer que todas as coisas em sua vida sejam diferentes.

Trooper recentemente recebeu uma string S (uma sequência de caracteres) consistindo de letras minúsculas do alfabeto. Já que Trooper quer que tudo seja diferente, ele quer que todas as *substrings* de sua string sejam distintas. Uma substring é formada por uma sequência de caracteres consecutivos da string original. Por exemplo, a string **"aba"** tem as seguintes substrings: **""** (susbtring vazia), **"a"**, **"b"**, **"a"**, **"ab"**, **"ba"** e **"aba"**.

Se a string S tem pelo menos duas substrings iguais, então Trooper irá substituir algumas letras por outras do alfabeto, de forma que todas as substrings sejam distintas. Trocar letras é um trabalho muito cansativo e entediante, então Trooper quer realizar a menor quantidade de trocas possível.

Como Trooper descobriu que você está participando de uma maratona de programação, ele resolveu pedir sua ajuda nesta tarefa.

Sua tarefa é encontrar o menor número de letras que devem ser substituídos de forma que a string resultante contenha apenas letras minúsculas do alfabeto e que tenha apenas substrings distintas, ou determinar que tal tarefa é impossível.

Entrada

A entrada é composta de duas linhas. Na primeira delas temos um inteiro N, $(1 \leq N \leq 100000)$, o tamanho da string S. A segunda linha contêm a string S consistindo de N letras minúsculas do alfabeto.

Saída

Exiba em uma linha um inteiro representando a menor quantidade de trocas necessárias para que a string S possua apenas substrings distingas ou o inteiro -1 caso tal tarefa seja impossível.

Exemplos

Entrada	**Saída**
2 aa	1

Entrada	**Saída**
4 haha	2

Entrada	**Saída**
8 abcdefgh	0

Problema E. Estimando o prazer

Nome do arquivo fonte: `docinhos.cpp`

Drica sempre foi fascinada por quase tudo que é doce e decidiu ganhar a vida vendendo o que mais ama fazer (ou comer): doces. Como ela está muito ocupada na criação de novas receitas, ela te pediu para que você fizesse um programa para lhe informar qual a estimativa de prazer máxima que um cliente consegue adquirir com as atuais receitas que ela sabe fazer.

Inicialmente Drica não sabe fazer nenhum docinho, apenas comê-los. Por isso, ela irá te passar Q operações de dois tipos:

- 1 C P: adicionar uma nova receita de um doce que custa C reais e proporciona P de prazer;
- 2 R: informar o prazer máximo que se consegue gastando até R reais com as receitas que ela sabe fazer até o momento.

Será que você consegue ajudá-la? Em troca, quem sabe, ela possa te dar um docinho :D

Entrada

A entrada é composta de diversas linhas. Na primeira delas será fornecido um inteiro Q representando a quantidade de operações a serem feitas. Aas próximas Q linhas são compostas de 2 ou 3 inteiros descrevendo as operações conforme explicado acima. Considere $(1 \leq Q, C, P, R \leq 5000)$

Saída

Para cada operação do tipo 2 seu programa deve exibir em uma linha um inteiro representando a quantidade máxima de prazer que se consegue com as receitas que Drica já aprendeu. É garantido que haverá pelo menos uma operação do tipo 2.

Exemplos

Entrada	**Saída**
7 1 3 2 2 5 1 1 1 2 5 1 2 3 2 5 2 6	2 5 7 9

Problema F. Farrinhos

Nome do arquivo fonte: `farrinhos.cpp`

Geraldo durante a sua infância era muito apaixonado por carrinhos de brinquedo, tanto que hoje ele ainda tem uma vasta coleção com diferentes tipos e, obviamente, valores de carrinho. Alves, um cara também aficionado por carrinhos, ao se encantar com a coleção de Geraldo lhe disse:

Alves:	Olha, o que tu acha de me vender uma parte da sua coleção?
Geraldo:	Hmm, pode ser. Quais você vai querer?
Alves:	Eu quero qualquer conjunto **contínuo** de sua coleção de tal maneira que eu consiga **ao menos um** carrinho de cada tipo.
Geraldo:	Deixa eu ver... Perfeito, mas você terá que me pagar o **maior** valor de todos os carrinhos desta coleção que eu te separar.
Alves:	Combinado, quanto eu te devo então?

Incerto a respeito do valor que teria de responder, já que ele quer ganhar o máximo possível, Geraldo pediu sua ajuda para que ele consiga responder Alves o quanto antes. Aliás, ele não ve a hora de pegar um dinheirinho e ir bater um rango na Franboi.

Entrada

A entrada é composta de três linhas. Na primeira delas serão fornecidos dois inteiros N e T ($1 \leq T \leq N \leq 10^5$) indicando respectivamente a quantidade e os tipos de carrinhos na coleção de Geraldo. A próxima linha contêm N inteiros t_i ($1 \leq t_i \leq T$) representando o tipo do i-ésimo carrinho. Por fim, a última linha contêm N inteiros c_i ($1 \leq c_i \leq 10^9$) representando o valor do i-ésimo carrinho. Lembre-se, como a coleção de Geraldo é bastante grande, é garantido que haja pelo menos um carrinho de cada tipo.

Saída

Seu programa deve informar qual o maior ganho que Geraldo pode ter ao vender uma parte de sua coleção respeitando as condições impostas por Alves.

Exemplos

Entrada	**Saída**
5 3 1 2 1 3 2 5 7 9 6 5	9

Entrada	**Saída**
3 3 1 2 3 1 2 3	3

Problema G. Garrinhos

Nome do arquivo fonte: `garrinhos.cpp`

Geraldo durante a sua infância era muito apaixonado por carrinhos de brinquedo, tanto que hoje ele ainda tem uma vasta coleção com diferentes tipos e, obviamente, valores de carrinho. Alves, um cara também aficionado por carrinhos, ao se encantar com a coleção de Geraldo lhe disse:

Alves:	Olha, o que tu acha de me vender uma parte da sua coleção?
Geraldo:	Hmm, pode ser. Quais você vai querer?
Alves:	Eu quero qualquer conjunto **contínuo** de sua coleção de tal maneira que eu consiga **ao menos um** carrinho de cada tipo.
Geraldo:	Deixa eu ver... Perfeito, mas você terá que me pagar o **menor** valor de todos os carrinhos desta coleção que eu te separar.
Alves:	Combinado, quanto eu te devo então?

Incerto a respeito do valor que teria de responder, já que ele quer ganhar o máximo possível, Geraldo pediu sua ajuda para que ele consiga responder Alves o quanto antes. Aliás, ele não ve a hora de pegar um dinheirinho e ir bater um rango na Franboi.

Entrada

A entrada é composta de três linhas. Na primeira delas serão fornecidos dois inteiros N e T ($1 \leq T \leq N \leq 10^6$) indicando respectivamente a quantidade e os tipos de carrinhos na coleção de Geraldo. A próxima linha contêm N inteiros t_i ($1 \leq t_i \leq T$) representando o tipo do i-ésimo carrinho. Por fim, a última linha contêm N inteiros c_i ($1 \leq c_i \leq 10^9$) representando o valor do i-ésimo carrinho. Lembre-se, como a coleção de Geraldo é bastante grande, é garantido que haja pelo menos um carrinho de cada tipo.

Saída

Seu programa deve informar qual o maior ganho que Geraldo pode ter ao vender uma parte de sua coleção respeitando as condições impostas por Alves.

Exemplos

Entrada	**Saída**
5 3 1 2 1 3 2 5 7 9 6 5	6

Entrada	**Saída**
3 3 1 2 3 1 2 3	1

Problema H. Habilitando a senha

Nome do arquivo fonte: `senha.cpp`

Silveira está tentando abriar uma fechadura eletrônica bastante complexa. A fechadura contêm N botões e para abri-la, você deve pressionar os botões na ordem correta. Quando você pressiona um botão, ele ficará pressionado caso você tenha adivinhado corretamente ou todos os botões já pressionados retornarão para a posição inicial. Quando todos os botões estiverem pressionados, a fechadura irá abrir.

Considere um exemplo com três botões. Digamos que o segredo da fechadura é: $\{2, 3, 1\}$. Se você pressionar primeiro 1 ou 3, os botões serão soltos imediatamente. Se você pressionar primeiro o 2, ele continuará pressionado. Se você pressionar 1 depois do 2, todos botões serão soltos. Se você pressionar 3 depois do 2, ambos continuarão pressionados, agora só falta pressionar o botão 1 para abrir a fechadura.

Silveira não sabe o segredo da fechadura. Mas ele é um menino esperto e irá agir de forma ótima. Calcule o número de vezes que Silveira irá pressionar botões para abrir a fechadura no pior caso.

Entrada

A entrada consiste de uma única linha contendo um inteiro N, $(1 \leq N \leq 2000)$, o número de botões que a fechadura contêm.

Saída

Seu programa deverá exibir em uma única linha a quantidade de vezes que Silveira irá pressionar teclas no pior caso até que a fechadura destrave.

Exemplos

Entrada	**Saída**
2	3

Entrada	**Saída**
3	7

Problema I. Igualizando as bolachas

Nome do arquivo fonte: `bolachas.cpp`

Ajlune acabou de chegar de viagem e trouxe um pacotão com N bolachas e decidiu dividí-lo com a sua grande amiga Stefhany.

E assim foi feito:

(1) Um para você; um para mim.
(2) Um para você; um, dois para mim.
(3) Um para você; um, dois, três para mim.
$\vdots$ $\vdots$ $\vdots$ $\ddots$
(X) Um para você; um, dois, três, ..., X para mim.

Percebendo que dividir desta maneira ia levar muito tempo, Ajlune decidiu apenas entregar à Stefhany a quantidade que ela ficaria ao final respeitando obrigatoriamente esta sequência. Incerto se ele consegue calcular tal valor, Ajlune pediu a sua ajuda em troca de um balão e uma bolacha que inclusive ele já até separou para você `:D`

Entrada

Será fornecido o inteiro N ($1 \leq N \leq 10^{18}$) representando a quantidade de bolachas que Ajlune trouxe de viagem. Eram muitas bolachas.

Saída

Seu programa deve exibir dois inteiros A e S representando a quantidade de bolachas que Ajlune e Stefhany ficarão ao final da divisão.

Exemplos

Entrada	**Saída**
`9`	`3 5`

Entrada	**Saída**
`10`	`3 6`

Entrada	**Saída**
`11`	`4 6`

Problema J. Joãozinho tá atrasado

Nome do arquivo fonte: `joazinho.cpp`

Joãozinho está morando em São Paulo e todos os dias depende de transporte público para ir de sua casa até seu trabalho. Alguns dias ele vai de ônibus e outros de metrô, sua escolha depende basicamente do tempo total que o ônibus ou o metrô gastam nas várias paradas durante o trajeto.

Como Joãozinho já acorda atrasado para o trabalho e a quantidade de paradas não são iguais todos os dias, ele gasta um bom tempo fazendo cálculos até decidir qual transporte pegar. Como ele descobriu que acontecerá uma Maratona de Progamação no Inatel, ele decidiu pedir sua ajuda!

Joãozinho irá informar a quantidade de paradas dos dois tipos de transportes, quanto tempo (em segundos) cada parada gasta e o valor dos tickets de cada meio de transporte, visto que se o tempo gasto em ambos é o mesmo, ele com certeza irá escolher o transporte mais barato. É garantido que os valores dos tickets dos transportes nunca serão iguais.

Entrada

A entrada possui duas linhas, cada uma com três inteiros Q ($1 \leq Q \leq 25$), T ($1 \leq T \leq 300$), V ($1 \leq V \leq 20$), representando a quantidade de paradas, o tempo de cada parada e o valor do ticket, respectivamente. A primeira linha se refere ao ônibus e a segunda ao metrô.

Saída

Seu programa deverá imprimir em uma linha o meio de transporte que Joãozinho irá utilizar, "`onibus`"ou "`metro`", sem aspas, como mostra os exemplos abaixo.

Exemplos

Entrada	**Saída**
`3 50 4` `4 30 6`	`metro`

Entrada	**Saída**
`12 80 5` `10 96 8`	`onibus`

Problema K. Keparticionando a sequência

Nome do arquivo fonte: `keparticionando.cpp`

Ferreirinha ao treinar para maratona de programação adora desafiar seus colegas de equipe. Hoje ele apareceu com uma sequência a que contém N inteiros a qual deve ser particionada em outras duas sequências b e c. Considere B como sendo a soma de todos elementos pertencentes em b e C como sendo as somas dos elementos de c. Caso uma sequência seja vazia, então a sua soma é 0. Dado a sequência proposta por Ferreirinha, será que você consegue informar qual o máximo possível para $B - C$?

Entrada

A primeira linha contém um inteiro N ($1 \leq N \leq 100$) como o número de elementos da sequência a. A segunda linha contém N inteiros a_i ($-100 \leq a_i \leq 100$) como sendo a sequência.

Saída

Seu programa deve informar qual o maior valor $B-C$, onde B é a soma dos elementos da sequência b e C é a soma dos elementos em c.

Exemplos

Entrada	**Saída**
3 1 -2 0	3

Entrada	**Saída**
6 16 23 16 15 42 8	120

Problema L. Legado

Nome do arquivo fonte: `legado.cpp`

Por muitos e muito anos, Damião foi quem coordenou a maratona de programação no Inatel. Após tantos anos de programação na veia, ele desenvolveu um poder um tanto quanto especial: ser capaz de alterar o estado de amizade entre um conjunto de pessoas!

Basicamente seu poder funciona da seguinte maneira: Sempre que Damião imagina um grupo de pessoas, e estala seus dedos e instantaneamente todas as relações de amizades dentro deste grupo são invertidas.

Imagine que temos 3 pessoas: André, Igor e Pedro, e a princípio ninguém é amigo de ninguém. Se Damião pensar em André e Pedro, por exemplo, e estalar seus dedos, André passará a ser amigo de Pedro mas ambos não serão amigo de Igor. Em seguida se Damião pensar nos 3 e estalar novamente seus dedos, teremos que André será amigo de Igor e Igor será amigo de Pedro, porém André não será mais amigo de Pedro.

Como Damião é um cara muito gente boa, ele decidiu largar a maratona para usufruir de seu dom para deixar o mundo mais feliz. Ele então está visitando alguns lugares onde a tristeza prevalece, ou seja, a princípio ninguém é amigo de ninguém, e alterando as relações de amizades dos habitantes locais com o objetivo que ao final cada cidadão tenha ao menos um amiguinho.

Não se sabe ao certo o por que Damião não usa seu poder de forma ótima para realizar tal objetivo, mas sabendo que você está participando de mais uma maratona de programação no Inatel, ele te pede ajuda! Ele irá te passar todo o histórico dos grupos de pessoas que ele alterou o estado de amizade e você deve informar ao final as pessoas que ficaram sem nenhum amiguinho.

Entrada

A primeira linha da entrada constitui de dois inteiros N ($1 \leq N \leq 10^5$) e Q ($1 \leq Q \leq 10^5$), representando, respectivamente, o número de pessoas e a quantidade de vezes que Damião utilizou seu poder. As próximas Q linhas iniciam com um inteiro G seguido de G inteiros distintos P ($1 \leq P \leq N$) representando os índices das pessoas que foram afetadas pelo poder desta vez. Considere que a quantidade de vezes que alguma pessoa será afetada pelo poder de Damião não será maior que 10^6.

Saída

Caso Damião tenha concluído seu objetivo, seu programa deverá exibir `"Todo mundo feliz!"`, sem aspas, caso contrário deverá imprimir uma linha com todos os índices, em ordem crescente, das pessoas que terminaram sem nenhum amiguinho, assim Damião poderá dar um tratamento especial para elas.

Exemplos

Entrada	**Saída**
3 2 2 1 3 3 1 2 3	Todo mundo feliz!

Entrada	**Saída**
4 3 3 2 3 4 2 2 3 2 3 4	1 3