unesp UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO" Câmpus de Presidente Prudente

DEPARTAMENTO DE MATEMÁTICA E COMPUTAÇÃO

Maratona de Programação do curso de Bacharelado em Ciência da Computação da FCT/Unesp

Competição Principal

Caderno de Problemas

Informações Gerais

Este caderno contém 10 problemas. Verifique se o caderno está completo.

A) Sobre a entrada

1) A entrada de seu programa deve ser lida da entrada padrão.

2) A entrada é composta de um único caso de teste, descrito em um número de linhas que 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 coincide com o final do arquivo, quando não é indicado outro modo para definir o fim da entrada.

B) Sobre a saída

1) A saída de seu programa deve ser escrita na saída padrão.

2) Quando uma linha da saída contém vários valores, estes devem ser separados por um único espaço em branco; a saída não deve conter nenhum outro espaço em branco.

3) Cada linha, incluindo a última, deve conter o caractere final-de-linha.

17 de Agosto de 2016

Problema A

Zerinho ou Um

Arquivo: zerinho.[c|cpp|java]

Todos devem conhecer o jogo Zerinho ou Um (em algumas regiões também conhecido como Dois ou Um), utilizado para determinar um ganhador entre três ou mais jogadores. Para quem não conhece, o jogo funciona da seguinte maneira. Cada jogador escolhe um valor entre zero ou um; a um comando (geralmente um dos competidores anuncia em voz alta "Zerinho ou... Um!"), todos os participantes mostram o valor escolhido, utilizando uma das mãos: se o valor escolhido foi um, o competidor mostra o dedo indicador estendido; se o valor escolhido foi zero, mostra a mão com todos os dedos fechados. O ganhador é aquele que tiver escolhido um valor diferente de todos os outros; se não há um jogador com valor diferente de todos os outros (por exemplo todos os jogadores escolhem zero, ou um grupo de jogadores escolhe zero e outro grupo escolhe um), não há ganhador.

Alice, Beto e Clara são grandes amigos e jogam Zerinho a toda hora: para determinar quem vai comprar a pipoca durante a sessão de cinema, quem vai entrar na piscina primeiro, etc. Jogam tanto que resolveram fazem um plugin no Facebook para jogar Zerinho. Como não sabem programar, dividiram as tarefas entre amigos que sabem, inclusive você. Dados os três valores escolhidos por Alice, Beto e Clara, cada valor zero ou um, escreva um programa que determina se há um ganhador, e nesse caso determina quem é o ganhador.

Entrada

A entrada de cada caso de teste é composta de uma única linha, que contém três inteiros A, B e C, indicando respectivamente os valores escolhidos por Alice, Beto e Clara.

Saída

Seu programa deve produzir uma única linha, contendo um único caractere. Se o vencedor é Alice o caractere deve ser `A', se o vencedor _e Beto o caractere deve ser `B', se o vencedor é Clara o caractere deve ser `C' e se não há vencedor o caractere deve ser `*' (asterisco).

Restrições

- $A, B, C \in \{0, 1\}$

Exemplos

Entrada	Saída
1 1 0	C

Entrada	Saída
0 0 0	*

Entrada	Saída
1 0 0	A

Problema B

Construindo Ferrovias

Arquivo: ferrovias.[c|cpp|java]

A Nlogônia é um país que possui todo seu território conectado por meio de ferrovias, que foram construídas há muito tempo e necessitam ser reformadas. Entretanto, o governo não fará uma reforma em todas as ferrovias, pois elas não foram projetadas de forma eficiente. Portanto, o governo decidiu que as ferrovias que ficarem ativas deverão estar totalmente conectadas, ou seja, dado uma estação, deve ser possível chegar a qualquer outra estação.

A sua tarefa é ajudar o governo da Nlogônia a saber qual é a quantidade máxima de quilômetros que pode ser economizada com a reforma das ferrovias, seguindo a restrição apresentada.

Entrada

Cada caso de teste inicia com dois números inteiros m ($1 \leq m \leq 200000$) e n ($m-1 \leq n \leq 200000$), que são, respectivamente, a quantidade de estações e a quantidade de ferrovias que precisam ser reformadas. Em seguida, são especificados n conjuntos de três valores inteiros x, y e z, especificando um caminho entre as estações x e y (bidirecional) com z quilômetros ($0 \leq x, y < m$ e $x \neq y$).

A soma total de todas as ferrovias é menor do que 2^31 para cada caso de teste.

Saída

Para cada caso de teste imprima uma linha contendo a quantidade máxima de quilômetros que pode ser economizada com a reforma das ferrovias.

Exemplos

Exemplo de Entrada	Exemplo de Saída
7 11 0 1 7 0 3 5 1 2 8 1 3 9 1 4 7 2 4 5 3 4 15 3 5 6 4 5 8 4 6 9 5 6 11	51

Problema C

Inversão

Arquivo: inversao.[c|cpp|java]

Pedro é um garoto curioso que gostava de eletrônica. Certo dia, o menino estava mexendo no laboratório de sua escola e encontrou uma caixa cheia de pequenos aparelhos eletrônicos feitos por outros alunos em anos anteriores.

Dentro dessa caixa havia um aparelho que possuía apenas um visor e dois botões. Esse visor apresentava um número inteiro. Mexendo nos botões, Pedro descobriu para que servia cada um deles. O primeiro botão adicionava uma unidade ao número no visor. O segundo botão invertia os dígitos do número, por exemplo, 123 invertido resulta em 321 e 150 invertido resulta em 51 (ignora-se os zeros a esquerda).

Inicialmente, o visor apresentava o número A. Após a descoberta da função dos botões, Pedro quer saber como fazer o número do visor mudar de A para um número maior igual a B. O seu trabalho nesse problema é ajudar Pedro a descobrir qual é o número mínimo de apertos de botão para que o número no visor passe a ser igual a B.

Entrada

A entrada de cada caso de teste contém dois inteiros A e B, $0 < A < B < 10000$, indicando respectivamente o número inicial no visor e o número que deve ser mostrado no visor depois de apertar os botões.

Saída

Para cada caso de teste, o programa deve imprimir um inteiro indicando o número mínimo de apertos de botão para que o número do visor passe de A para B.

Exemplos

Exemplo de Entrada	Exemplo de Saída
1 9	8

Exemplo de Entrada	Exemplo de Saída
100 301	4

Exemplo de Entrada	Exemplo de Saída
808 909	3

Exemplo de Entrada	Exemplo de Saída
133 233	3

Problema D

Analisador de Código

Arquivo: analisador.[c|cpp|java]

Um problema muito comum enfrentado nos cursos de Ciência da Computação é a cópia dos trabalhos de programação. Para resolver esse problema, o professor FCT (Fabrício César Teixeira) decidiu utilizar um analisador de código para comparar os trabalhos desenvolvidos por seus alunos com um banco de dados de código que possui.

O professor decidiu que o analisador será feito pelos próprios alunos, todavia, para comparar apenas linhas de código (um passo de cada vez). Dessa maneira, sua tarefa é: dada duas sequências de código, comparar a similaridade entre essas partes de código, identificando cópia ou não cópia. A similaridade entre duas sequências será calculada pela maior quantidade de correspondências entre caracteres, não sendo necessário haver sequência nas correspondências, ou seja, a correspondência pode ser "quebrada". Por exemplo, dadas as duas sequências a seguir:

Sequência 1: if (x < 1 && 3+variavel <= q) {
Sequência 2: if(x<= 1 && 3 < z) {

dizemos que a similaridade entre as duas, no contexto de nosso problema é 12, já que a maior correspondência é equivalente a "if(x<1&&3<){" (note que os espaços são ignorados).

Entrada

A entrada de cada caso de teste é composta por um inteiro N (1 < N <= 100), que especifica o limiar se uma sequência de código é cópia da outra. As duas linhas a seguir especificam as duas sequências de código de devem ser analisadas, ambas com tamanho 5 < M <= 150.

Saída

Para cada caso de teste, o programa deve imprimir uma string "nao copia" caso a similaridade entre as duas porções de código seja menor que o limiar, ou "copia" caso a similaridade entre as duas seja maior ou igual. Deve-se usar uma quebra de linha no fim de cada saída.

Exemplos

Exemplo de Entrada	Exemplo de Saída

10
if (x < 1 && 3+variavel <= q) {
if(x<= 1 && 3 < z) {
	copia

Exemplo de Entrada	Exemplo de Saída
20 i=0; while(i++ < 10) { printf(i); } i=0; do{ printf(i++); } while(i < 10);	nao copia

Problema E

Mergulho

Arquivo: mergulho.[c|cpp|java]

O recente terremoto em Nlogônia não chegou a afetar muito as edificações da capital, principal epicentro do abalo. Mas os cientistas detectaram que o principal dique de contenção teve um dano significativo na sua parte subterrânea que, se não for consertado rapidamente, pode causar o seu desmoronamento, com a consequente inundação de toda a capital.

O conserto deve ser feito por mergulhadores, a uma grande profundidade, em condições extremamente
difíceis e perigosas. Mas como é a sobrevivência da própria cidade que está em jogo, seus moradores acudiram em grande número como voluntários para essa perigosa missão.

Como é tradicional em missões perigosas, cada mergulhador recebeu no início do mergulho uma pequena placa com um número de identificação. Ao terminar o mergulho, os voluntários devolviam a placa de identificação, colocando-a em um repositório.

O dique voltou a ser seguro, mas aparentemente alguns voluntários não voltaram do mergulho. Você foi contratado para a penosa tarefa de, dadas as placas colocadas no repositório, determinar quais voluntários perderam a vida salvando a cidade.

Entrada

A entrada de cada caso de teste é composta de duas linhas. A primeira linha contém dois inteiros N e R, indicando respectivamente o número de voluntários que mergulhou e o número de voluntários que retornou do mergulho. Os voluntários são identificados por números de 1 a N. A segunda linha da entrada contém R inteiros, indicando os voluntários que retornaram do mergulho (ao menos um voluntário retorna do mergulho).

Saída

Seu programa deve produzir uma única linha para cada caso de teste, contendo os identificadores dos voluntários que não retornaram do mergulho, na ordem crescente de suas identificações. Deixe um espaço em branco após cada identificador (note que isto significa que deve haver um espaço em branco também após o último identificador). Se todos os voluntários retornaram do mergulho, imprima apenas o caractere `*' (asterisco).

Restrições

- $1 \leq R \leq N \leq 10^4$

Exemplos

Entrada	Saída
5 3 3 1 5	2 4

Entrada	Saída
6 6 6 1 3 2 5 4	*

Problema F

Futebol

Arquivo: futebol.[c|cpp|java]

Seu time favorito de futebol está disputando um campeonato beneficente, para auxiliar uma fundação a conseguir dinheiro para ajudar crianças com necessidades especiais. Como em um torneio normal, três pontos são dados ao time que vence uma partida e nenhum ponto para o perdedor. Se uma partida termina empatada, cada time recebe um ponto.

Seu time jogou N partidas durante a primeira fase do torneio, que já terminou. Somente alguns times, aqueles que acumularam mais pontos, irão avançar para a segunda fase do campeonato. Entretanto, como o principal objetivo do torneio é conseguir dinheiro, ante de definir os times que passarão para a segunda fase, é determinado que cada time pode comprar gols adicionais. Esses novos gols contam como gols na pontuação normal das partidas disputadas e podem ser usados para alterar o resultado de qualquer uma delas.

Seu time tem dinheiro suficiente para comprar G gols. Você poderia dizer qual o número máximo de pontos que seu time obteria após comprar novos gols, supondo que outros times não poderão comprar gols?

Entrada

Para cada caso de teste, a primeira linha contém dois inteiros N ($1 \leq N \leq 10^5$) e G ($0 \leq G \leq 10^6$) representando respectivamente o número de partidas que seu time disputou e o número de gols que seu time é capaz de comprar. Cada uma das próximas N linhas descrevem os resultados das partidas com dois inteiros S e R ($0 \leq S, R \leq 100$), indicando respectivamente os gols que seu time marcou e os gols que ele recebeu em cada partida antes da compra dos novos gols.

Saída

Para cada caso de teste, deve ser exibida uma linha com um inteiro representando o máximo de pontos que seu time pode obter após comprar novos gols.

Sample input 1	Sample output 1
2 1 1 1 1 1	4

Sample input 2	Sample output 2
3 2 1 3 3 1 2 2	6

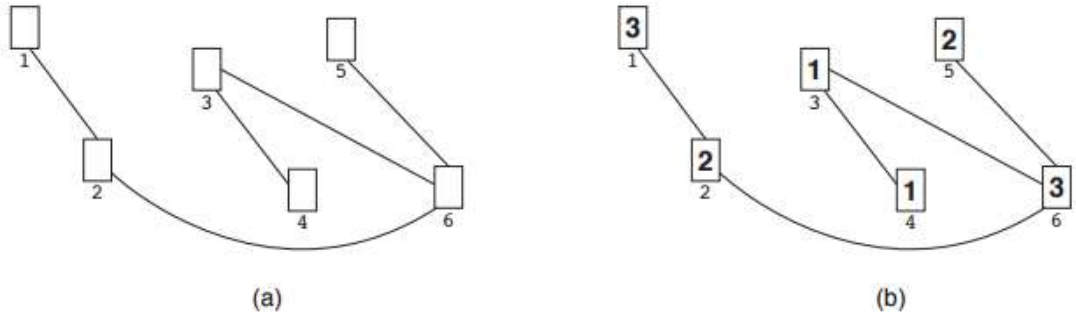
Sample input 3	Sample output 3
4 10 1 1 2 2 1 3 0 4	12

Problema G

Jogo da Memória

Arquivo: memoria.[c|cpp|java]

Pedro e Paulo resolveram complicar um pouco o tradicional Jogo da Memória, em que os jogadores precisam virar duas cartas iguais. Eles colocam N cartas no chão, com as faces viradas para baixo. A face de cada carta tem a figura de um número de 1 até N/2, sendo que exatamente duas cartas possuem a figura de cada número entre 1 e N/2. Como as cartas têm as faces viradas para baixo, elas podem também ser identificadas por suas posições, que são inteiros de 1 a N. Pedro e Paulo então desenham no chão, usando giz, algumas linhas ligando pares de cartas, de modo que para qualquer par de cartas (A, B) existe uma e apenas uma sequência de cartas e linhas desenhadas que leva de A até B. A figura abaixo mostra um exemplo de jogo, (a) com todas as cartas com as faces viradas para baixo, e (b) com todas as cartas com as faces viradas para cima.



O jogo é jogado com todas as cartas com as faces viradas para baixo. A cada jogada, o jogador deve escolher um par de cartas A e B. Se as faces das duas cartas escolhidas têm a mesma figura, o jogador acumula um número de pontos igual ao número de linhas desenhadas que existem no caminho entre as cartas A e B. Pedro e Paulo, agora, estão estudando qual é a melhor estratégia para esse jogo e precisam da sua ajuda para resolver uma tarefa específica: dadas as cartas existentes em cada posição, e as ligações desenhadas com giz, calcular o maior valor total de pontos que é possível acumular.

Entrada

Para cada caso de teste, primeira linha da entrada contém o número de cartas N ($2 \leq N \leq 50000$, N é par). A segunda linha da entrada contém N inteiros Ci, indicando qual número está anotado na carta na posição i ($1 \leq Ci \leq N/2$, para $1 \leq i \leq N$). As cartas são dadas na ordem crescente das posições: a primeira carta ocupa a posição 1, a segunda a posição 2, e assim por diante até a última carta, que ocupa a posição N. Cada uma das $N - 1$ linhas seguintes contém dois números A e B, indicando que existe uma linha desenhada entre as cartas nas posições A e B ($1 \leq A \leq N$ e $1 \leq B \leq N$).

Saída

Seu programa deve produzir uma linha contendo um inteiro, o maior valor total de pontos que é possível acumular.

(VER OS EXEMPLOS NA PRÓXIMA PÁGINA)

Exemplos

Exemplo de Entrada	Exemplo de Saída
6 3 2 1 1 2 3 1 2 3 4 6 5 2 6 3 6	5

Exemplo de Entrada	Exemplo de Saída
8 1 2 3 3 2 4 1 4 1 2 2 3 2 6 5 6 6 8 7 8 4 7	12

Problema H

Ajude o Cupido

Arquivo: cupido.[c|cpp|java]

Como o trabalho do Cupido está ficando cada vez mais difícil, ele está adotando novas tecnologias para ajudar com a difícil tarefa de encontrar casais e torná-los felizes. Ele escolheu os melhores programadores para ajudá-lo em um novo projeto chamado *Advanced Couples Matching* (ACM). Para seu projeto, os programadores precisam produzir um algoritmo para encontrar N/2 casais a partir de N pessoas que estão sozinhas, tal que uma pessoa não esteja em diferentes casais.

Infelizmente, os dados disponíveis sobre cada pessoa são limitados. Por isso, foi decidido focar nos fusos horários em que as pessoas vivem. Pessoas morando perto, de acordo com o fuso horário, são mais propensas a encontrar tempo para interagir uma com a outra. Assim, os programadores decidiram criar casais de modo a minimizar a diferença total entre os fusos horários que elas vivem.

Cada fuso horário é definido como um inteiro entre -11 e 12, inclusive, representando sua diferença em horas a partir de um fuso horário particular conhecido como *Coordinated Universal Time* (ou UTC). A diferença em horas entre duas pessoas morando nos fusos horários dados pelos inteiros *i* e *j* é o valor mínimo entre |i - j| e 24 - |i - j|. A diferença total de uma partição de um conjunto de N candidatos unidos em N/2 casais é dado pela soma total das diferenças entre os fusos horários de cada casal.

Você foi selecionado para escrever um programa que receba como entrada os fusos horários onde as N pessoas vivem. A saída do programa deve ser a menor diferença total entre todas as possíveis partições de casais que podem ser formados com as N pessoas.

Entrada

Em cada caso de teste, a primeira linha contém um número inteiro par N ($2 \leq N \leq 1000$) representando o número de pessoas candidatas a formar um casal. A segunda linha contém os N inteiros $T_1, T_2, \ldots, T_N$ ($-11 \leq T_i \leq 12$ para i = 1,2,...,N) indicando os fusos horários onde os candidatos vivem.

Saída

O programa deve exibir uma linha com um inteiro representando a menor diferença total entre todas as possíveis combinações de partições para unir os candidatos em casais.

Exemplos

Sample input 1	Sample output 1
6 -3 -10 -5 11 4 4	5

Sample input 2	Sample output 2
2 -6 6	12

Sample input 3	Sample output 3
8 0 0 0 0 0 0 0 0	0

Problema I

Flowers Flourish from France

Arquivo: flowers.[c|cpp|java]

Fiona sempre amou poesias e recentemente ela descobriu uma forma fascinante de criar poemas. Tautogramas são casos especiais de aliteração, que é a ocorrência de uma mesma letra no início de palavras adjacentes. Em particular, uma sentença é um tautograma se todas as suas palavras começam com a mesma letra. Por exemplo, as seguintes sentenças são tautogramas:

- Flowers Flourish from France
- Sam simmonds speaks softly
- Fazendo fiado fico freguês
- Peter picked peppers
- abcd abacade aliteração
- Amor
- Namorado

Fiona deseja presentear seu namorado com uma romântica carta cheia desse tipo de sentenças. Por favor, ajude Fiona a checar se cada sentença que ela escreveu é um tautograma ou não.

Entrada

Cada caso de teste contém uma sequência de linhas compostas por palavras separadas por espaços simples. Uma palavra é uma sequência de no máximo 20 letras (maiúsculas ou minúsculas) do alfabeto. Uma palavra contém pelo menos uma letra e uma sentença contém pelo menos uma palavra. O caso de teste termina com um único caractere '*' (asterisco).

Saída

Para cada linha do caso de teste a saída deve conter uma linha com um 'Y' maiúsculo se a sentença é um tautograma, ou um 'N' maiúsculo caso contrário. Note que uma única palavra também é considerada um tautograma.

Exemplos

Sample input	Output for the sample input
Flowers Flourish from France	Y
Sam Simmonds speaks softly	Y
Peter pIckEd pePPers	Y
truly tautograms triumph	Y
this is NOT a tautogram	N
*	

Problema J

Ácido Ribonucleico Alienígena

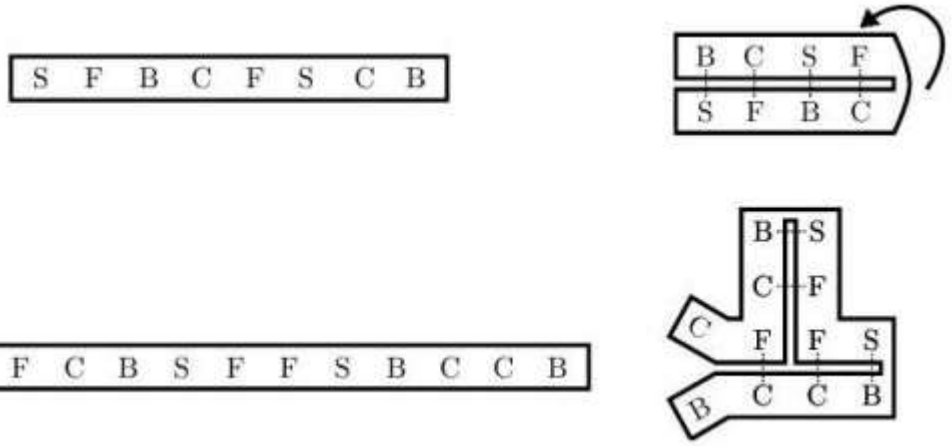
Arquivo: acido.[c|cpp|java]

Foi descoberta uma espécie alienígena de ácido ribonucleico (popularmente conhecido como RNA). Os cientistas, por falta de criatividade, batizaram a descoberta de ácido ribonucleico alienígena (RNAA). Similar ao RNA que conhecemos, o RNAA é uma fita composta de várias bases. As bases são B C F S e podem ligar-se em pares. Os únicos pares possíveis são entre as bases B e S e as bases C e F.

Enquanto está ativo, o RNAA dobra vários intervalos da fita sobre si mesma, realizando ligações entre suas bases. Os cientistas perceberam que:

- Quando um intervalo da fita de RNAA se dobra, todas as bases neste intervalo se ligam com suas bases correspondentes;
- Cada base pode se ligar a apenas uma outra base;
- As dobras ocorrem de forma a maximizar o número de ligações feitas sobre fitas;

As figuras abaixo ilustram dobras e ligações feitas sobre fitas.



Sua tarefa será: dada a descrição de uma tira de RNAA, determinar quantas ligações serão realizadas entre suas bases se a tira ficar ativa.

Entrada

Para cada caso de teste, a entrada é composta por diversos casos de teste e termina com EOF. Cada caso de teste possui uma linha descrevendo a sequência de bases da fita de RNAA. Uma fita de RNAA na entrada contém pelo menos 1 e no máximo 300 bases. Não existem espaços entre bases de uma fita da entrada. As bases são 'B', 'C', 'F' e 'S'.

Saída

Para cada instância imprima uma linha contendo o número total de ligações que ocorre quando a fita descrita é ativada.

(VER EXEMPLOS NA PRÓXIMA PÁGINA)

Exemplos

Exemplo de Entrada	Exemplo de Saída
SBC	1

Exemplo de Entrada	Exemplo de Saída
CFCBSFFSBCCB	5

Exemplo de Entrada	Exemplo de Saída
FCC	1

Exemplo de Entrada	Exemplo de Saída
SFBC	0

Exemplo de Entrada	Exemplo de Saída
SFBCFSCB	4