OBI2010

Caderno de Tarefas

Modalidade **Programação** • Nível **2**, Fase **1**

20 de março de 2010

A PROVA TEM DURAÇÃO DE **5** HORAS

Promoção:

Sociedade Brasileira de Computação

Patrocínio:

Fundação Carlos Chagas

Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 8 páginas (não contando a folha de rosto), numeradas de 1 a 8. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar a Internet, livros, anotações ou qualquer outro material durante a prova. É permitida a consulta ao *help* do ambiente de programação se este estiver disponível.
- As tarefas têm o mesmo valor na correção.
- A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa.
- Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc.), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
- As tarefas **não** estão ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
- Preste muita atenção no nome dos arquivos fonte indicados nas tarefas. Soluções na linguagem C devem ser arquivos com sufixo *.c*; soluções na linguagem C++ devem ser arquivos com sufixo *.cc* ou *.cpp*; soluções na linguagem Pascal devem ser arquivos com sufixo *.pas*. Para problemas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada problema.
- Ao final da prova, para cada solução que você queira submeter para correção, copie o arquivo fonte para o seu diretório de trabalho ou disquete, conforme especificado pelo seu professor.
- Não utilize arquivos para entrada ou saída. Todos os dados devem ser lidos da entrada padrão (normalmente é o teclado) e escritos na saída padrão (normalmente é a tela). Utilize as funções padrão para entrada e saída de dados:
  - em Pascal: *readln, read, writeln, write*;
  - em C: *scanf, getchar, printf, putchar*;
  - em C++: as mesmas de C ou os objetos *cout* e *cin*.
- Procure resolver o problema de maneira eficiente. Na correção, eficiência também será levada em conta. As soluções serão testadas com outras entradas além das apresentadas como exemplo nas tarefas.

Cometa

Nome do arquivo fonte: `cometa.c`, `cometa.cpp`, *ou* `cometa.pas`

O cometa Halley é um dos cometas de menor período do Sistema Solar, completando uma volta em torno do Sol a cada 76 anos; na última ocasião em que ele tornou-se visível do planeta Terra, em 1986, várias agências espaciais enviaram sondas para coletar amostras de sua cauda e assim confirmar teorias sobre suas composições químicas.

Tarefa

Escreva um programa que, dado o ano atual, determina qual o próximo ano em que o cometa Halley será visível novamente do planeta Terra. Se o ano atual é um ano de passagem do cometa, considere que o cometa já passou nesse ano (ou seja, considere sempre o próximo ano de passagem, não considerando o ano atual).

Entrada

A única linha da entrada contém um único inteiro A ($2010 \leq A \leq 10^4$), indicando o ano atual.

Saída

Seu programa deve imprimir uma única linha, contendo um número inteiro, indicando o próximo ano em que o cometa Halley será visível novamente do planeta Terra.

Informações sobre a pontuação

- Em um conjunto de casos de teste que totaliza 20 pontos, $A \leq 2400$.

Exemplos

Entrada	**Saída**
2010	2062

Entrada	**Saída**
10000	10042

Entrada	**Saída**
2062	2138

Batalha Naval

Nome do arquivo fonte: `batalha.c`, `batalha.cpp`, *ou* `batalha.pas`

Pedro e Paulo gostam muito de jogar batalha naval; apesar de serem grandes amigos, Pedro desconfia que Paulo não esteja jogando honestamente. Para tirar essa dúvida, Pedro decidiu usar um programa de computador para verificar o resultado do jogo, mas Pedro não sabe programar e por isso pediu a sua ajuda.

O jogo de batalha naval é jogado em um tabuleiro retangular com N linhas e M colunas. Cada posição deste tabuleiro é um quadrado que pode conter água ou uma parte de um navio. Dizemos que dois quadrados são *vizinhos* se estes possuem um lado em comum. Se duas partes de navio estão em posições vizinhas, então essas duas partes pertencem ao mesmo navio. A regra do jogo proíbe que os quadrados de duas partes de navios distintos tenham um canto em comum (em outras palavras, que quadrados de duas partes de navios distintos compartilhem um vértice).

Cada *disparo* que um jogador faz deve ser feito em um dos quadrados do tabuleiro do outro jogador. Um jogador informa ao outro a coluna e a linha do quadrado alvo do disparo. Para que um navio seja *destruído*, o jogador deve acertar todas as partes deste navio. O jogador não pode atirar no mesmo lugar mais de uma vez.

Tarefa

Escreva um programa que, dadas a configuração do tabuleiro e uma sequência de disparos feitos por um jogador, determina o número de navios do outro jogador que foram destruídos.

Entrada

A primeira linha da entrada contém números dois inteiros N e M ($1 \leq N \leq 100$ e $M \leq 100$) representando respectivamente o número de linhas e de colunas do tabuleiro. As N seguintes linhas correspondem ao tabuleiro do jogo. Cada uma dessas linhas contém M caracteres. Cada caractere indica o conteúdo da posição correspondente no tabuleiro. Se esse caractere for '`.`', essa posição contém água; se for '`#`', essa posição contém uma parte de um navio. A próxima linha contém um número K que é o número de disparos feitos pelo jogador ($1 \leq K \leq N \times M$). As próximas K linhas indicam os disparos feitos pelo jogador. Cada linha contém dois inteiros L e C, indicando a linha e a coluna do disparo feito pelo outro jogador ($1 \leq L \leq N$ e $1 \leq C \leq M$).

Saída

Seu programa deve imprimir uma única linha contendo um único número inteiro, o número de navios destruídos.

Informações sobre a pontuação

- Em um conjunto de casos de teste que totaliza 30 pontos, os navios são todos compostos por exatamente uma parte (ou seja, um quadrado).
- Em um conjunto de casos de teste que totaliza 50 pontos, cada navio está contido em exatamente uma linha.

Exemplos

Entrada	**Saída**
5 5 ..#.# #.... ...#. #.... ...#. 5 1 3 1 4 1 5 2 1 3 4	4

Entrada	**Saída**
5 5 ..### ..... ##### ..... #.##. 5 5 1 5 2 1 3 1 4 1 5	2

Entrada	**Saída**
7 7 .#....# ###..## .#....# ....#.# .#..#.# .####.# 8 1 1 1 2 2 1 2 2 2 3 3 2 5 2 6 2	1

Elevador

Nome do arquivo fonte: `elevador.c`, `elevador.cpp`, *ou* `elevador.pas`

A Subindo Bem Confortavelmente (SBC) é uma empresa tradicional, com mais de 50 anos de experiência na fabricação de elevadores. Todos os projetos da SBC seguem as mais estritas normas de segurança, mas infelizmente uma série de acidentes com seus elevadores manchou a reputação da empresa.

Ao estudar os acidentes, os engenheiros da companhia concluíram que, em vários casos, o acidente foi causado pelo excesso de passageiros no elevador. Por isso, a SBC decidiu fiscalizar com mais rigor o uso de seus elevadores: foi instalado um sensor em cada porta que detecta a quantidade de pessoas que saem e entram em cada andar do elevador.

A SBC tem os registros do sensor de todo um dia de funcionamento do elevador (que sempre começa vazio). Eles sabem que as pessoas são educadas e sempre deixam todos os passageiros que irão sair em um andar saírem antes de outros passageiros entrarem no elevador, mas ainda assim eles têm tido dificuldade em decidir se a capacidade máxima do elevador foi excedida ou não.

Tarefa

Escreva um programa que, dada uma sequência de leituras do sensor e a capacidade máxima do elevador, determina se a capacidade máxima do elevador foi excedida em algum momento.

Entrada

A primeira linha da entrada contém dois inteiros N e C, indicando o número de leituras realizadas pelo sensor e a capacidade máxima do elevador, respectivamente ($1 \leq N \leq 1000$ e $1 \leq C \leq 1000$). As N linhas seguintes contêm, cada uma, uma leitura do sensor. Cada uma dessas linhas contém dois inteiros S e E, indicando quantas pessoas saíram e quantas pessoas entraram naquele andar, respectivamente ($0 \leq S \leq 1000$ e $0 \leq E \leq 1000$).

Saída

Seu programa deve imprimir uma única linha contendo o caractere '`S`', caso a capacidade do elevador tenha sido excedida em algum momento, ou o caractere '`N`' caso contrário.

Exemplos

Entrada	**Saída**
5 10 0 5 2 7 3 3 5 2 7 0	N

Entrada	**Saída**
5 10 0 3 0 5 0 2 3 4 6 4	S

Entrada	**Saída**
6 4 0 5 3 5 4 5 1 0 1 1 1 1	S

Reunião

Nome do arquivo fonte: `reuniao.c`, `reuniao.cpp`, *ou* `reuniao.pas`

Todos os anos, a SBC (Sociedade Brasileira de Caminhoneiros) reúne seus membros em alguma cidade para discutir sobre a profissão. Nessas reuniões são discutidos os problemas da categoria e são apresentadas sugestões sobre como melhorar as condições de trabalho.

O grande problema desse tipo de encontro é que os membros estão espalhados pelo país, uma vez que a profissão exige que eles viajem para diversos lugares todos os dias. Por isso, a escolha da cidade onde será feita a reunião sempre é feita de modo que não prejudique demais nenhum dos caminhoneiros. O critério para tal é que a maior das distâncias percorridas pelos caminhoneiros para chegar ao local da reunião deve ser a menor possível. Ou seja, a distância percorrida pelo caminhoneiro que vai percorrer a maior distância entre todos os caminhoneiros para chegar à reunião deve ser a menor possível.

Tarefa

Dadas as cidades onde se encontram os caminhoneiros e a descrição das estradas que interligam essas cidades, escreva um programa que determina qual será a menor distância máxima percorrida por um caminhoneiro para chegar até o local da reunião. Os caminhoneiros conhecem bem as estradas, e portando sempre fazem o menor caminho possível até a cidade da reunião. Sempre existe um caminho ligando quaisquer duas cidades.

Entrada

A primeira linha da entrada possui dois números inteiros N ($2 \leq N \leq 100$) e M ($N - 1 \leq M \leq 10000$), que representam, respectivamente, o número de cidades e o número de estradas que as interligam. As cidades são identificadas por números inteiros entre 0 e $N - 1$. As próximas M linhas da entrada possuem, cada uma, a descrição de uma estrada. Cada descrição de entrada é composta por três números inteiros: U, V e W, onde U e V representam cidades ($0 \leq U \leq N - 1$ e $0 \leq V \leq N - 1$) e W representa o comprimento da estrada que une essas duas cidades (todas as estradas são mão dupla, $1 \leq W \leq 100$). É sempre possível viajar entre qualquer duas cidades com as estradas existentes, mas pode haver mais de uma estrada ligando o mesmo par de cidades.

Saída

Seu programa deve imprimir uma única linha contendo um número inteiro, a distância máxima percorrida por um caminhoneiro para ir até a reunião, obedecidas as restrições estabelecidas (ou seja, essa distância máxima deve ser a menor possível).

Informações sobre a pontuação

- Em um conjunto de casos de teste que totaliza 30 pontos, $N \leq 4$.

Exemplos

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

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

Entrada	**Saída**
7 12 0 1 22 0 2 30 0 5 35 1 5 11 1 6 30 1 2 25 2 3 15 2 6 10 3 4 15 3 5 10 4 5 20 5 6 33	30