

Competidor(a): _____

Número de inscrição: _____ – _____ (opcional)



OBI2020

Caderno de Tarefas

Modalidade **Programação** • **Nível 1** • Fase **Estadual**

24 outubro de 2020

A PROVA TEM DURAÇÃO DE **2 HORAS**

Promoção:



Sociedade Brasileira de Computação

Apoio:



Instruções

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 6 páginas (não contando a folha de rosto), numeradas de 1 a 6. 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; em particular, seu programa não deve escrever frases como “Digite o dado de entrada:” ou similares.
- 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 necessariamente 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*; soluções na linguagem Java devem ser arquivos com sufixo *.java* e a classe principal deve ter o mesmo nome do arquivo fonte; soluções na linguagem Python 2 devem ser arquivos com sufixo *.py2*; soluções na linguagem Python 3 devem ser arquivos com sufixo *.py3*; e soluções na linguagem Javascript devem ter arquivos com sufixo *.js*.
- Na linguagem Java, **não** use o comando *package*, e note que o nome de sua classe principal deve usar somente letras minúsculas (o mesmo nome do arquivo indicado nas tarefas).
- Para tarefas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada tarefa.
- Ao final da prova, para cada solução que você queira submeter para correção, copie o arquivo fonte para o seu diretório de trabalho ou pen-drive, 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*.
 - em Java: qualquer classe ou função padrão, como por exemplo *Scanner, BufferedReader, BufferedWriter* e *System.out.println*
 - em Python: *read, readline, readlines, input, print, write*
 - em Javascript: *scanf, printf*
- Procure resolver a tarefa 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.

Dona Formiga

Nome do arquivo: “formiga.x”, onde x deve ser `c`, `cpp`, `pas`, `java`, `js`, `py2` ou `py3`

Dona Formiga é uma ótima trabalhadora e todos os dias coleta muitas folhas para seu formigueiro. Mas no final de semana, quando todas as outras formigas estão descansando, ela gosta de se divertir escorregando pelos túneis do formigueiro.

O formigueiro de Dona Formiga tem muitos túneis e salões. Cada túnel conecta exatamente dois salões diferentes. Cada salão está a uma altura no formigueiro. Se existe um túnel ligando um salão I a um salão J e o salão I está a uma altura maior do que o salão J , então Dona Formiga pode escorregar do salão I para o salão J usando esse túnel.

Dados o mapa dos túneis do formigueiro, as alturas em que estão os salões e o salão de onde Dona Formiga quer partir, escreva um programa para determinar o maior número de salões que ela pode visitar (não contando o salão do qual ela parte), usando túneis exclusivamente para escorregar entre os salões.

Entrada

A primeira linha da entrada contém três inteiros S , T e P , respectivamente o número de salões, o número de túneis e o salão do formigueiro do qual Dona Formiga quer partir. Os salões são numerados de 1 a S . A segunda linha contém S números inteiros A_i , a altura em que o salão i está no formigueiro. Cada uma das T linhas seguintes contém dois inteiros I e J , indicando que há um túnel entre o salão I e o salão J .

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o maior número de salões que Dona Formiga pode visitar (não contando o salão do qual ela parte), usando os túneis exclusivamente para escorregar entre os salões do formigueiro.

Restrições

- $1 \leq S \leq 200$
- $1 \leq T \leq S \times (S - 1)/2$
- $1 \leq P \leq S$
- $-1000 \leq A_i \leq 1000$ para $1 \leq i \leq S$
- $1 \leq I < J \leq S$

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 20 pontos, $1 \leq S \leq 10$.
- Para um conjunto de casos de testes valendo 80 pontos adicionais, nenhuma restrição adicional.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
<pre>4 5 2 100 150 -50 200 1 2 2 4 1 4 3 4 1 3</pre>	<pre>2</pre>

Exemplo de entrada 2 4 5 3 100 150 -50 200 1 2 2 4 1 4 3 4 1 3	Exemplo de saída 2 0
Exemplo de entrada 3 4 5 4 100 150 -50 200 1 2 2 4 1 4 3 4 1 3	Exemplo de saída 3 3

Estrada

Nome do arquivo: “`estrada.x`”, onde x deve ser `c`, `cpp`, `pas`, `java`, `js`, `py2` ou `py3`

Para melhorar a integração com os países vizinhos, o Rei da Nlogônia decidiu que uma nova estrada será construída cruzando o país, da fronteira oeste à fronteira leste. O formato da estrada é uma única reta, que passará pelo centro de algumas cidades.

O Rei também decidiu que a construção será paga pelo Tesouro Real, mas cada cidade pela qual a estrada passar será responsável pela manutenção do trecho da estrada que constitui a *vizinhança da estrada* para aquela cidade. A *vizinhança da estrada* de uma cidade A é definida como todos os pontos da estrada que são mais próximos do centro da cidade A do que do centro de qualquer outra cidade.

Dados o comprimento total da estrada, de fronteira a fronteira, e as distâncias da fronteira oeste até os centros de cada cidade ao longo da nova estrada, escreva um programa para determinar qual a menor vizinhança de estrada entre as cidades pelas quais a estrada vai passar.

Entrada

A primeira linha da entrada contém um inteiro T , o comprimento total da estrada. A segunda linha contém um inteiro N , o número de cidades pelas quais a estrada vai passar. Cada uma das N linhas seguintes contém um inteiro X_i , indicando a distância da fronteira oeste até o centro da cidade i . Não há cidades nas fronteiras e cada centro de cidade tem uma localização distinta.

Saída

Seu programa deve produzir uma única linha, contendo um número real com duas casas após o ponto decimal, a menor vizinhança de estrada entre as cidades pelas quais a estrada vai passar.

Restrições

- $3 \leq T \leq 10^6$
- $2 \leq N \leq 10^4$
- $0 < X_i < T$, para $1 \leq i \leq N$
- $X_i \neq X_j$, para todo par $1 \leq i, j \leq N$.

Informações sobre a pontuação

- Para um conjunto de casos de testes valendo 10 pontos, $N = 2$.
- Para um conjunto de casos de testes valendo 90 pontos adicionais, nenhuma outra restrição.

Exemplos

Exemplo de entrada 1	Exemplo de saída 1
10 2 8 5	3.50

Exemplo de entrada 2	Exemplo de saída 2
10 3 7 6 8	1.00

Dona Lesma

Nome do arquivo: “lesma.x”, onde x deve ser c, cpp, pas, java, js, py2 ou py3

Dona Lesma é esportista e aventureira e definiu como objetivo deste verão alcançar o topo do muro do jardim em que vive. A cada dia, valente e metodicamente ela sobe exatamente uma certa distância (sempre a mesma a cada dia). Mas a cada noite enquanto dorme Dona Lesma escorrega para baixo uma outra distância (sempre a mesma a cada noite)...

Dadas a altura do muro, a distância que ela sobe a cada dia e a distância que ela desce a cada noite, ajude Dona Lesma a calcular quantos dias ela levará para chegar ao topo do muro.

Entrada

A primeira linha contém um inteiro A , a altura do muro. A segunda linha contém um inteiro S , distância que Dona Lesma sobe a cada dia. A terceira linha contém um inteiro D , a distância que Dona Lesma escorrega para baixo a cada noite.

Saída

Seu programa deve produzir uma única linha, contendo um único inteiro, o número de dias que Dona Lesma demorará para chegar ao topo do muro.

Restrições

- $1 \leq A \leq 10000$
- $1 \leq D < S \leq 10000$

Exemplos

Exemplo de entrada 1 4 2 1	Exemplo de saída 1 3
Exemplo de entrada 2 12 5 2	Exemplo de saída 2 4
Exemplo de entrada 3 10000 100 50	Exemplo de saída 3 199