OBI2005

Caderno de Tarefas

Modalidade Programação • Seletiva 2 • IOI

A PROVA TEM DURAÇÃO DE QUATRO HORAS

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

- Este caderno de tarefas é composto por 6 páginas (não contando esta folha de rosto), numeradas de 1 a 6. Verifique se o caderno está completo.
- A prova deve ser feita individualmente.
- É proibido consultar 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.
- 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.

Sociedade Brasileira de Computação
www.sbc.org.br

Tarefa A
Mapas do Gugo

A internet oferece uma variedade de facilidades interativas para mapas, de tal forma que os usuários podem ter uma visão geral de uma região geográfica ou podem fazer *zoom* para uma rua específica, ou mesmo um prédio específico, em um mapa mais detalhado. Por exemplo, a Unicamp pode aparecer em um mapa de Campinas com um certo nível de detalhes, ou em um mapa de Barão Geraldo com mais detalhes.

Gugo comprou uma grande coleção de mapas retangulares e deseja criar um serviço para processar seqüências de consultas em mapas com vários níveis de detalhes. As localidades são representadas por um par de coordenadas inteiras (x, y). Os mapas têm identificadores únicos e são representados por dois pares de coordenadas, descrevendo cantos opostos do mapa. Os lados de todos os mapas são paralelos aos eixos cartesianos padrão x e y. Um mapa abrange todas as localidades contidas no retângulo, incluindo as que estão sobre a borda.

O nível de detalhe de um mapa pode ser estimado pelo tamanho da área retangular representada pelo mapa: um mapa que cobre uma área menor contém informações mais detalhadas e portanto maior nível de detalhe.

Pode haver sobreposição das áreas cobertas pelos mapas. Se uma consulta é feita para uma localidade presente em dois ou mais mapas, o mapa preferido é o de maior nível de detalhe. Em caso de empate, o preferido é o mapa em que a localidade é mais próxima do centro do mapa. Finalmente, se ainda houver empate, o mapa preferido é o de menor identificador.

Tarefa

Dados o conjunto de mapas e uma lista de consultas de localidades você deve escrever um programa que determine, se existir, o mapa com o maior nível de detalhe para cada uma das consultas.

Entrada

A entrada é composta de vários conjuntos de testes. A primeira linha de um conjunto de testes contém dois inteiros M, e R, separados por um espaço em branco, que indicam respectivamente o número de mapas ($1 \leq M \leq 3000$) e o número de consultas ($1 \leq R \leq 5000$). Cada uma das M linhas seguintes contém cinco inteiros I, X_1, Y_1, X_2, Y_2, separados por um espaço em branco; I representa o identificador do mapa ($1 \leq I \leq M$), (X_1, Y_1) representa a coordenada do canto inferior esquerdo do mapa, e (X_2, Y_2) representa a coordenada do canto superior direito do mapa ($-10000 \leq X_1 < X_2 \leq 10000$ e $-10000 \leq Y_1 < Y_2 \leq 10000$). Muitos dos mapas (mesmo de áreas disjuntas) têm cantos com coordenadas x ou y coincidentes. O número máximo de cantos de mapas com coordenadas x não coincidentes é 500, e o número máximo de cantos de mapas com coordenadas y não coincidentes também é 500. As R linhas seguintes contêm as consultas. Cada consulta é dada em uma linha contendo dois inteiros X_R e Y_R, representando uma localidade ($-10000 \leq X_R \leq 10000$ e $-10000 \leq Y_R \leq 10000$). O final da entrada é indicado por $M = R = 0$.

A entrada deve ser lida da entrada padrão.

Saída

Para cada conjunto de teste da entrada seu programa deve produzir a saída da seguinte forma. A primeira linha deve conter um identificador do conjunto de teste, no formato '`Teste n`', onde '`n`' é numerado seqüencialmente a partir de 1. Para cada consulta do conjunto de testes da entrada seu programa deve imprimir uma linha, contendo um inteiro N, representando o mapa preferido para a localidade, se existir. Se não existir mapa para a localidade desejada imprima o valor zero. A última linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser serguida rigorosamente.

A saída deve ser escrita na saída padrão.

Restrições

$1 \leq M \leq 3000$ ($M = 0$ apenas para indicar final de entrada)
$1 \leq R \leq 5000$ ($R = 0$ apenas para indicar final de entrada)
$-10000 \leq X_1 < X_2 \leq 10000$
$-10000 \leq Y_1 < Y_2 \leq 10000$

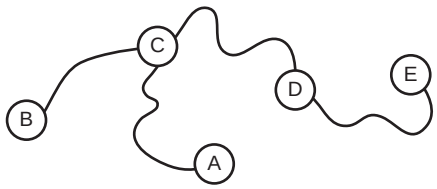
Exemplo de Entrada	**Saída para o Exemplo de Entrada**

`2 4`
`1 1 1 4 5`
`2 3 2 7 8`
`4 2`
`5 3`
`6 9`
`2 2`
`4 4`
`4 1 -3 4 2`
`2 -3 -4 2 -1`
`3 -4 -2 -1 3`
`1 -2 1 3 4`
`0 0`
`2 -3`
`1 3`
`-2 1`
`0 0`
	`Teste 1`
	`1`
	`2`
	`0`
	`1`
	
	`Teste 2`
	`0`
	`2`
	`1`
	`3`

Tarefa B
O Problema do Caixeiro Viajante

Asdrúbal é um caixeiro viajante (essa profissão é antiga, e designa uma pessoa que trabalha com vendas e percorre as cidades atendendo clientes).

Asdrúbal viaja somente de trem; a malha ferroviária é composta por um conjunto de *trechos* de linhas de trem. Um trecho de linha liga uma diretamente uma cidade A a uma cidade B, sem passar por outra cidade, conforme a figura abaixo.



Um *trajeto* é formado por uma seqüência de trechos consecutivos ligando duas cidades. A malha ferroviária é construída de tal forma que entre qualquer par de cidades existe apenas um trajeto possível, e a partir de uma cidade é possível viajar para qualquer outra cidade.

A empresa fornece a Asdrúbal tíquetes de transporte que lhe permitem viajar por toda a malha ferroviária. Asdrúbal gasta um tíquete a cada trecho do trajeto viagem, independente do comprimento do trecho. Assim, considerando o mapa acima, para ir da cidade A para a cidade E Asdrúbal gasta três tíquetes.

A empresa entregou a Asdrúbal a lista com as cidades em que ela possui clientes que ele deve visitar. Asdrúbal deve partir da cidade onde mora e retornar, ao final da viagem, para a mesma cidade. Ele pode visitar as cidades determinadas pela empresa em qualquer ordem, mas com uma restrição importante: ele deve gastar o menor número possível de tíquetes de viagem.

Tarefa

Dada a descrição da malha ferroviária (cidades e trechos de linhas de trem) e uma lista de cidades que devem ser visitadas, você deve escrever um programa para determinar qual o menor número de tíquetes necessários para Asdrúbal efetuar a viagem.

Entrada

A entrada é composta de vários conjuntos de testes. A primeira linha de um conjunto de testes contém dois inteiros C e V, separados por um espaço em branco, que indicam respectivamente o número de cidades na malha ferroviária ($1 \leq C \leq 300$) e o número de cidades que devem ser visitadas ($1 \leq V \leq C$). As cidades são identificadas por inteiros entre 1 e C; a cidade em que Asdrúbal reside é sempre identificada pelo número 1. Cada uma das $C - 1$ linhas seguintes contém dois inteiros X e Y, separados por um espaço em branco, indicando que há um trecho de linha férrea inteligando as cidades X e Y, sem passar por outra cidade ($1 \leq X \leq C$, $1 \leq Y \leq C$ e $X \neq Y$). Note que uma ligação entre X e Y permite o trajeto de X para Y e de Y para X. A última linha de um conjunto de testes contém uma seqüência de V inteiros, separados por espaço em branco, indicando as cidades que possuem clientes que Asdrúbal deve visitar. O final da entrada é indicado por $C = V = 0$.

A entrada deve ser lida da entrada padrão.

Saída

Para cada conjunto de teste da entrada seu programa deve produzir três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato '`Teste n`', onde `n` é numerado seqüencialmente a partir de 1. A segunda linha deve conter um número inteiro indicando o número mínimo de tíquetes que Asdrúbal gasta na viagem. A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser serguida rigorosamente.

A saída deve ser escrita na saída padrão.

Restrições

$1 \leq C \leq 300$ ($C = 0$ apenas para indicar final de entrada)
$1 \leq V \leq C$ ($V = 0$ apenas para indicar final de entrada)
$1 \leq X \leq C$
$1 \leq Y \leq C$
$X \neq Y$

Exemplo de Entrada	**Saída para o Exemplo de Entrada**

`4 3`
`2 3`
`4 2`
`2 1`
`4 3 1`
`5 5`
`1 2`
`1 3`
`1 4`
`1 5`
`1 2 3 4 5`
`0 0`
	`Teste 1`
	`6`
	
	`Teste 2`
	`8`

Tarefa C
Criptologia

Um dos métodos mais antigos métodos de cifragem de mensagens é a *cifragem por subsituição*. Variantes desse método têm sido usadas através dos tempos, por usuários como Julius Cæsar (100–44 a.C.) e a Rainha Mary da Escócia (1542–1587).

O método é bem simples: cada letra da mensagem é substituída por outra letra. Por exemplo, todas as letras 'A' do texto são substituídas por 'X', todas as letras 'B' por 'E', e assim por diante.

A deficiência do método de cifragem por substituição é que ele é vulnerável à análise de freqüências. Ela se baseia no fato de que, para trechos grandes de mensagens, certas letras ocorrem com freqüências que são aproximadamente as mesmas para a maioria dos trechos escritos na mesma língua. Como a mensagem foi cifrada de forma que cada letra seja substituída por uma outra, a nova letra herdará todos os atributos da letra antiga, incluindo sua freqüência. Em português, a letra mais freqüente é 'A'. Assim, se a letra mais freqüente em uma mensagem cifrada é 'T', muito provavelmente 'T' representa a letra 'A' no texto original.

Embora não se saiba quem descobriu que a variação de freqüência das letras pode ser explorada para quebrar cifras, a descrição mais antiga conhecida é do cientista Abu Yusuf Ya 'qub ibn Is-haq ibn as-Sabbah ibn ómran ibn Ismail al-Kindi, que viveu no século IX. Abu Yusuf escreveu 290 livros em assuntos tão variados como medicina, astronomia, matemática, linguística e música. Mas a sua maior obra, redescoberta em 1987 nos Arquivos Otomanos Sulaimaniyyah, em Istambul, intitula-se "*Um Manuscrito Sobre Como Decifrar Mensagens Criptografadas*".

A quebra da cifragem por substituição marcou o nascimento da ciência da criptologia, que tem múltiplas aplicações nos dias de hoje.

Tarefa

Sua tarefa é decifrar uma mensagem cifrada utilizando o método de substituição, dadas a mensagem cifrada e a lista de freqüências das letras em uma determinada língua.

Entrada

A entrada é composta de vários conjuntos de testes. A primeira linha de um conjunto de testes contém dois inteiros T e F, separados por um espaço em branco, que indicam respectivamente o número de caracteres do texto cifrado ($1 \leq T \leq 10000$), e o número de letras da lista de freqüências ($1 \leq F \leq 26$). A segunda linha de um caso de testes contém uma lista de F letras $c_1 c_2 c_3 \ldots c_F$ utilizadas na mensagem original, em ordem decrescente da freqüência em que elas aparecem na língua considerada (ou seja, c_1 é a letra mais freqüente, c_F a menos freqüente, '`a`' $\leq c_i \leq$ '`z`'). A terceira linha de um caso de testes contém os T caracteres do texto cifrado. O texto cifrado contém caracteres de '`a`' a '`z`' e caracteres ' ' (espaço em branco). O final da entrada é indicado por $T = F = 0$.

A entrada deve ser lida da entrada padrão.

Saída

Para cada conjunto de teste da entrada seu programa deve produzir três linhas na saída. A primeira linha deve conter um identificador do conjunto de teste, no formato '`Teste n`', onde '`n`' é numerado seqüencialmente a partir de 1. A segunda linha deve conter o texto original (decifrado). A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser serguida rigorosamente.

A saída deve ser escrita na saída padrão.

Restrições

$1 \leq T \leq 10000$ ($T = 0$ apenas para indicar final de entrada)
$1 \leq F \leq 26$ ($F = 0$ apenas para indicar final de entrada)

Exemplo de Entrada	**Saída para o Exemplo de Entrada**

`15 9`
`aegilmqtu`
`ahagibf a caeda`
`33 13`
`saeontcbrpdum`
`kfqdt dt twdt ldtgt hft pgmkghagz`
`0 0`
	`Teste 1`
	`ataquem a galia`
	
	`Teste 2`
	`todas as suas bases nos pertencem`