



# CAMPEONATO DE PROGRAMAÇÃO UNIVERSITÁRIO

**QUESTÃO 01: A VIAGEM DE TEOBALDO**( <http://br.spoj.pl/problems/TEOBALDO/>)

Teobaldo trabalha para o governo brasileiro. No seu trabalho, ele costuma viajar muito. Quando Teobaldo viaja de uma cidade S para uma cidade E ele pode gastar até D dias nesta viagem. Como Teobaldo não gosta de trabalhar, ele sempre gasta o número máximo de dias nas suas viagens. Em cada dia da viagem, Teobaldo dorme em uma cidade diferente da cidade do dia anterior, pois ele acha chato ficar na mesma cidade dois dias consecutivos. Neste problema, você deve ajudar Teobaldo a planejar suas viagens.

**Entrada**

O arquivo de entrada contém vários conjuntos de teste. A descrição de cada conjunto é dada a seguir:

Cada conjunto começa com dois inteiros C ( $0 < C \leq 100$ ), o número de cidades, e L ( $0 \leq L \leq 500$ ), o número de estradas entre as cidades. Seguem L linhas, onde cada linha possui dois números: A e B ( $1 \leq A, B \leq C$ ), indicando que há uma estrada entre estas duas cidades. Você pode assumir que A e B são números diferentes. Depois dessas L linhas, há três inteiros: S, E e D, onde S é a cidade onde a viagem deve começar, E é a cidade onde a viagem deve terminar, e D ( $0 \leq D \leq 200$ ) é o número máximo de dias para Teobaldo ir de S para E.

A entrada é terminada por um conjunto onde  $C = L = 0$ . Este conjunto não deve ser processado. Há uma linha em branco entre dois conjuntos de entrada.

**Saída**

Para cada conjunto de entrada produza uma linha de saída indicando se Teobaldo pode viajar do jeito que ele deseja. Veja os exemplos a seguir para o formato exato de entrada/saída.

Exemplo

Entrada:

```
3 2
1 2
2 3
3 1 2
```

```
0 0
```

**Saída:**

Yes, Teobaldo can travel.

**QUESTÃO 02: SALDO DE GOLS** (<http://br.spoj.pl/problems/SALDO/>)

Hipólito é um torcedor fanático. Coleciona flâmulas, bandeiras, recortes de jornal, figurinhas de jogadores, camisetas e tudo o mais que se refira a seu time preferido. Quando ganhou um computador de presente em uma festa, resolveu montar um banco de dados com os resultados de todos os jogos de seu time ocorridos desde a sua fundação, em 1911. Depois de inseridos os dados, Hipólito começou a ficar curioso sobre estatísticas de desempenho do time. Por exemplo, ele deseja saber qual foi o período em que o seu time acumulou o maior saldo de gols. Como Hipólito tem o computador há muito pouco tempo, não sabe programar muito bem, e precisa de sua ajuda.

**Tarefa**

É dada uma lista, numerada seqüencialmente a partir de 1, com os resultados de todos os jogos do time (primeira partida: 3 x 0, segunda partida: 1 x 2, terceira partida: 0 x 5 ...). Sua tarefa é escrever um programa que determine em qual período o time conseguiu acumular o maior saldo de gols. Um período é definido pelos números de seqüência de duas partidas, A e B, onde  $A \leq B$ . O saldo de gols acumulado entre A e B é dado pela soma dos gols marcados pelo time em todas as partidas realizadas entre A e B (incluindo as mesmas) menos a soma dos gols marcados pelos times adversários no período. Se houver mais de um período com o mesmo saldo de gols, escolha o maior período (ou seja, o período em que  $B - A$  é maior). Se

ainda assim houver mais de uma solução possível, escolha qualquer uma delas como resposta.

#### Entrada

Seu programa deve ler vários conjuntos de teste. A primeira linha de um conjunto de teste contém um inteiro não negativo,  $N$ , que indica o número de partidas realizadas pelo time (o valor  $N = 0$  indica o final da entrada). Seguem-se  $N$  linhas, cada uma contendo um par de números inteiros não negativos  $X$  e  $Y$  que representam o resultado da partida:  $X$  são os gols a favor e  $Y$  os gols contra o time de Hipólito. As partidas são numeradas seqüencialmente a partir de 1, na ordem em que aparecem na entrada.

#### Exemplo de Entrada

```
2
2 3
7 1
3
0 2
0 3
0 4
0
```

#### 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 a partir de 1. A segunda linha deve conter um par de inteiros  $I$  e  $J$  que indicam respectivamente a primeira e última partidas do melhor período, conforme determinado pelo seu programa, exceto quando o saldo de gols do melhor período for menor ou igual a zero; neste caso a segunda linha deve conter a expressão "nenhum". A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

#### Exemplo de Saída

```
Teste 1
2 2
```

```
Teste 2
nenhum
```

#### Restrições

```
0 ≤ N ≤ 10000 (N = 0 apenas para indicar o fim da entrada)
1 ≤ A ≤ N
A ≤ B ≤ N
0 ≤ X ≤ 50
0 ≤ Y ≤ 50
```

#### QUESTAO 3: QUERMESSE

(<http://olimpiada.ic.unicamp.br/pratique/programacao/nivel2/quermesse>)

Os alunos do último ano resolveram organizar uma quermesse para arrecadar fundos para a festa de formatura. A festa prometia ser um sucesso, pois o pai de um dos formandos, Teófilo, dono de uma loja de informática, decidiu doar um computador para ser sorteado entre os que comparecessem. Os alunos prepararam barracas de quentão, pipoca, doces, ensaiaram a quadrilha e colocaram à venda ingressos numerados seqüencialmente a partir de 1. O número do ingresso serviria para o sorteio do computador. Ficou acertado que Teófilo decidiria o método de sorteio; em princípio o sorteio seria, claro, computadorizado.

O local escolhido para a festa foi o ginásio da escola. A entrada dos participantes foi pela porta principal, que possui uma roleta, onde passa uma pessoa por vez. Na entrada, um funcionário inseriu, em uma lista no computador da escola, o número do ingresso, na ordem de chegada dos participantes. Depois da entrada de todos os participantes, Teófilo começou a trabalhar no computador para preparar o sorteio. Verificando a lista de presentes, notou uma característica notável: havia apenas um caso, em toda a lista, em que o participante que possuía o ingresso numerado com  $i$ , havia sido a  $i$ -ésima pessoa a entrar no ginásio. Teófilo ficou tão encantado com a coincidência que decidiu que o sorteio não seria necessário: esta pessoa seria o ganhador do computador.

### Tarefa

Conhecendo a lista de participantes, por ordem de chegada, sua tarefa é determinar o número do ingresso premiado, sabendo que o ganhador é o único participante que tem o número do ingresso igual à sua posição de entrada na festa.

### Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um número inteiro positivo  $N$  que indica o número de participantes da festa. A linha seguinte contém a sequência, em ordem de entrada, dos  $N$  ingressos das pessoas que participaram da festa. O final da entrada é indicado quando  $N = 0$ . Para cada conjunto de teste da entrada haverá um único ganhador.

### Exemplo de Entrada

```
4
4 5 3 1
0
```

### Saída

Para cada conjunto de teste da entrada seu programa deve produzir três linhas. A primeira linha identifica o conjunto de teste, no formato "Teste  $n$ ", onde  $n$  é numerado a partir de 1. A segunda linha deve conter o número do ingresso do ganhador, conforme determinado pelo seu programa. A terceira linha deve ser deixada em branco. A grafia mostrada no Exemplo de Saída, abaixo, deve ser seguida rigorosamente.

### Exemplo de Saída

```
Teste 1
3
```

### Restrições

$0 \leq N \leq 10000$  ( $N = 0$  apenas para indicar o fim da entrada)

## QUESTAO 4 : ORKUT (<http://br.spoj.pl/problems/ORKUT/>)

Larissa acaba de entrar para o Orkut, um site na internet que permite que as pessoas se reúnam em comunidades e grupos de amigos. Como ela acabou de se registrar, ela ainda não possui muitos amigos na sua lista de contatos. Após fazer uma pesquisa, ela descobriu que os seus antigos amigos de escola (que adoravam mexer com computadores) também fazem parte do Orkut. Larissa então decidiu chamá-los para serem seus amigos virtuais. Porém, eles resolveram brincar com a Larissa, e cada um deles só vai aceitar o pedido de Larissa quando ela já for amiga virtual de alguns dos outros amigos do grupo. Assim, para conseguir ter todos os seus antigos amigos de escola na sua lista de amigos do Orkut, ela deve cumprir as exigências de cada um deles.

### Tarefa

Larissa acha que pode encontrar uma seqüência de nomes dos amigos, de modo que se ela pedir a cada um deles para ser sua amiga no Orkut, obedecendo a seqüência, todas as exigências serão cumpridas e todos eles irão aceitar o seu pedido. Larissa precisa da sua ajuda para resolver esse problema de forma rápida. A sua tarefa é escrever um programa para encontrar uma seqüência de nomes que resolva o problema, ou dizer que não é possível encontrar tal seqüência.

### Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um inteiro  $N$  que indica o número de antigos amigos da Larissa ( $1 \leq N \leq 30$ ). A linha seguinte irá conter  $N$  nomes de amigos, separados por espaço em branco. Cada nome não terá mais de 15 letras, e serão todos distintos. Nas próximas  $N$  linhas serão indicadas as exigências que a Larissa deve cumprir. Cada linha descreve a exigência de um amigo e começará com o nome desse amigo, seguido de um número  $M$  ( $0 \leq M \leq N - 1$ ), que indica o número de pessoas que aquele amigo quer que a Larissa seja amiga antes, e seguido pelos  $M$  nomes de amigos (cada item na linha separado por espaço em branco). O final da entrada é indicado por  $N = 0$ .

### Exemplo de Entrada

```
5
Joao Maria Tadeu Jose Ricardo
Joao 2 Maria Ricardo
Maria 1 Tadeu
Jose 1 Joao
Tadeu 0
Ricardo 0
0
```

### Saída

Para cada conjunto de teste seu programa deve produzir três linhas na saída. A primeira linha deverá 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 a seqüência de nomes de amigos (cada nome seguido de um espaço em branco) que resolve o problema da Larissa, ou a palavra "impossivel", quando não houver uma seqüência possível (note a ausência de acentuação). Se existir mais de uma seqüência de amigos que resolve o problema, imprima qualquer uma delas (mas apenas uma). A terceira linha deverá ser deixada em branco. A grafia mostrada no Exemplo de Saída abaixo deverá ser seguida rigorosamente.

### Exemplo de Saída

```
Teste 1
Ricardo Tadeu Maria Joao Jose
```

### Restrições

$0 \leq N \leq 30$  ( $N = 0$  apenas para indicar o fim da entrada)

$0 \leq M \leq N - 1$

Cada nome de amigo terá no máximo 15 letras

### QUESTÃO 05: EMOTICONS

(<http://br.spoj.pl/problems/EMOTIC/>)

Emoticons são usados em conversas pela internet e e-mails para tentar expressar as emoções que palavras impressas não conseguem. Isso pode parecer uma coisa boa para muitos, mas muitas pessoas acham eles realmente irritantes e querem se livrar deles.

George é uma dessas pessoas. Ele odeia emoticons tanto, que está preparando um plano para remover todos emoticons de todos os e-mails no mundo. Já que você divide seus planos visionários, você está preparando um programa especial para ajudá-lo.

Seu programa receberá uma lista de emoticons para remover. Cada emoticon será uma string

de caracteres não incluindo espaços em branco. Você também receberá diversas linhas de texto. O que você precisa fazer é mudar alguns caracteres do texto para espaços para garantir que nenhum emoticon foi deixado no texto. Para um emoticon aparecer no texto ele tem que aparecer em uma única linha e ser constituído de caracteres consecutivos.

Para que o plano de George permaneça em segredo pelo máximo de tempo possível, você precisa fazer seu trabalho com o mínimo possível de mudanças de caracteres.

### Entrada

O arquivo de entrada contém diversos casos de teste. Cada caso de teste consiste de diversas linhas. A primeira linha de cada caso de teste conterà dois inteiros separados por um espaço: N, o número de emoticons para remover, e M, o número de linhas que o texto tem. As próximas N linhas contém um emoticon cada, uma string não-vazia de no máximo 15 caracteres. Cada uma das últimas M linhas do caso de teste contém uma linha de texto com no máximo 80 caracteres. Você pode assumir  $1 \leq N, M \leq 100$ . Caracteres válidos para emoticons são letras maiúsculas e minúsculas, dígitos e os símbolos “!?,.,;,-\_’#\$\$%&/=\*+(){}[]”. Cada linha de texto pode conter os mesmos caracteres com a adição do caractere de espaço. A entrada acaba com  $N = M = 0$ .

### Saída

Para cada caso de teste, imprima exatamente uma linha contendo um único inteiro que indica o número mínimo de mudanças que você precisa fazer no texto inteiro para garantir que nenhum emoticon na lista apareça nele.

Exemplo

Entrada

4 6

:-)

:-(

(-:

):-

Hello uncle John! :-) :-D

I am sad or happy? (-:-(?

I feel so happy, my head spins

(-:)(-:)(-:)(-:)-) (-: :-)

but then sadness comes :-)

Loves you, Joanna :-))))))

0 0

Saída

11

### QUESTÃO 06: MÁRIO (<http://br.spoj.pl/problems/MARIO/>)

Mário é dono de uma empresa de guarda-volumes, a Armários a Custos Moderados (ACM). Mário conquistou sua clientela graças à rapidez no processo de armazenar os volumes. Para isso, ele tem duas técnicas:

- Todos os armários estão dispostos numa fila e são numerados com inteiros positivos a partir de 1. Isso permite a Mário economizar tempo na hora de procurar um armário;
- Todos os armários têm rodinhas, o que lhe dá grande flexibilidade na hora de reorganizar seus armários (naturalmente, quando Mário troca dois armários de posição, ele também troca suas numerações, para que eles continuem numerados sequencialmente a partir de 1).

Para alugar armários para um novo cliente, Mário gosta de utilizar armários contíguos, pois no início da locação um novo cliente em geral faz muitas requisições para acessar o conteúdo armazenado, e o fato de os armários estarem contíguos facilita o acesso para o cliente e para Mário.

Desde que Mário tenha armários livres em quantidade suficiente, ele sempre pode conseguir

isso. Por exemplo, se a requisição de um novo cliente necessita de quatro armários, mas apenas os armários de número 1, 3, 5, 6, 8 estiverem disponíveis, Mário pode trocar os armários 5 e 2 e os armários 6 e 4 de posição: assim, ele pode alugar o intervalo de armários de 1 até 4. No entanto, para minimizar o tempo de atendimento a um novo cliente, Mário quer fazer o menor número de trocas possível para armazenar cada volume. No exemplo acima, ele poderia simplesmente trocar os armários 1 e 4 de posição, e alugar o intervalo de 3 até 6. Mário está muito ocupado com seus clientes e pediu que você fizesse um programa para determinar o número mínimo de trocas necessário para satisfazer o pedido de locação de um novo cliente.

### Entrada

A entrada contém vários casos de teste. A primeira linha de cada caso de teste contém dois números inteiros  $N$  e  $L$  ( $1 \leq N \leq L \leq 100.000$ ), indicando quantos armários são necessários para acomodar o pedido de locação do novo cliente e quantos armários estão disponíveis, respectivamente. A linha seguinte contém  $L$  números inteiros positivos separados por espaços em branco, nenhum deles maior do que 1.000.000.000, indicando as posições dos armários disponíveis. Os números dos armários livres são dados em ordem crescente. O final da entrada é indicado por um caso onde  $N = L = 0$ .

A entrada deve ser lida da entrada padrão.

### Saída

Para cada caso de teste, imprima uma linha contendo um único número inteiro, indicando o número mínimo de trocas que Mário precisa efetuar para satisfazer o pedido do novo cliente (ou seja, ter  $N$  armários consecutivos livres).

Exemplo de entrada

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

Saída para o exemplo de entrada

```
1
2
```

### QUESTÃO 07: MAPAS (<http://br.spoj.pl/problems/MAPAS/>)

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 ( $1 \leq M \leq 30000$ ) e o número de consultas ( $1 \leq R \leq 50000$ ). Cada uma das  $M$  linhas seguintes contém cinco inteiros  $I, X1, Y1, X2, Y2$ , separados por um espaço em branco;  $I$  representa o identificador do mapa ( $1 \leq I \leq M$ ),  $(X1, Y1)$  representa a coordenada do canto inferior esquerdo do mapa, e  $(X2, Y2)$  representa a coordenada do canto superior direito do mapa ( $-10000 \leq X1 < X2 \leq 10000$  e  $-10000 \leq Y1 < Y2 \leq 10000$ ). 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  $XR$  e  $YR$ , representando uma localidade ( $-10000 \leq XR \leq 10000$  e  $-10000 \leq YR \leq 10000$ ). O final da entrada é indicado por  $M = R = 0$ .

### 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 seguida rigorosamente.

Exemplo

Entrada:

```
2 4
1 1 1 4 5
2 3 2 7 8
4 2
5 3
6 9
2 2
0 0 (Fim da entrada )
```

Saída:

```
Teste 1
1
2
0
1
```

Restrições

$1 \leq M \leq 30000$  ( $M = 0$  para indicar final da entrada)  
 $1 \leq N \leq 50000$  ( $N = 0$  para indicar o final da entrada)  
 $-10000 \leq X1 < X2 \leq 10000$   
 $-10000 \leq Y1 < Y2 \leq 10000$

### QUESTÃO 08: RAMANUJAN (Professor Dany Sanchez)

Números de Ramanujan. Dado um número inteiro  $n$ , os Números de Ramanujan denotam o número de partições de  $n$ , isto é, o número de formas diferentes de escrever o número  $n$  como uma soma de inteiros positivos sem considerar a ordem.

Exemplos:

Para  $n = 5$   
 $1 + 1 + 1 + 1 + 1$   
 $2 + 1 + 1 + 1$   
 $2 + 2 + 1$   
 $3 + 1 + 1$   
 $3 + 2$   
 $4 + 1$   
 $5$

Para  $n = 3$   
 $1 + 1 + 1$   
 $2 + 1$   
 $3$

Escreva um programa que receba um arquivo com  $k$  números inteiros positivos. Seu programa deve criar um novo arquivo com as partições de Ramanujam para cada valor inteiro do arquivo.

Arquivo de entrada

3  
1 2 5

Arquivo de saída

1  
1+1  
2

$1 + 1 + 1 + 1 + 1$   
 $2 + 1 + 1 + 1$   
 $2 + 2 + 1$   
 $3 + 1 + 1$   
 $3 + 2$   
 $4 + 1$   
 $5$

### QUESTÃO 9: LINGUAGEM MESD (Professor Carlos José Pereira)

A linguagem de programação MESD (muito-extremamente-simples-demais) é composta pelos seguintes comandos:

- 1 x - desvia a execução do programa (“goto”) para a linha “x”
- 2 - retorna para a linha seguinte daquela onde foi feito o desvio
- 3 - incrementa uma posição de memória chamada de “acumulador”
- 4 - decrementa o acumulador
- 5 - comando sem ação
- 0 - finaliza a execução do programa, exibindo o conteúdo final do acumulador

Um programa escrito em MESD tem suas linhas numeradas sequencialmente, a partir do número 1. Quando um programa em MESD inicia, o acumulador armazena o valor 0 (zero).

Construa um interpretador de programas em MESD que simule sua execução e ao final mostre o valor armazenado no acumulador.

#### Exemplo de entrada:

1 5 2 1 7 3 4 4 3 5 5 6 0 7 3 8 3 9 2

Esta entrada representa o seguinte programa em MESD:

- 1 5 - linha 1, comando 5 ( "sem ação" )
- 2 1 7 - linha 2, comando 2 ( "desvio para a linha 7" )
- 3 4 - linha 3, comando 4 ( "decrementa o acumulador" )
- 4 3 - linha 4, comando 3 ( "incrementa o acumulador" )
- 5 5 - linha 5, comando 5 ( "sem ação" )
- 6 0 - linha 6, comando 0 ( "finaliza a execução do programa" )
- 7 3 - linha 7, comando 3 ( "incrementa o acumulador" )
- 8 3 - linha 8, comando 3 ( "incrementa o acumulador" )
- 9 2 - linha 9, comando 2 ( "retorna para a linha seguinte a do desvio, isto é, linha 3" )

Valor do acumulador ao final da execução: 2

#### Saída:

Valor do Acumulador = 2

### QUESTÃO 10: ROTA CRÍTICA

Uma tragédia aconteceu recentemente em sua cidade. Um paciente em condição crítica, que necessitava tratamento urgente, morreu enquanto era transportado para um grande hospital da capital do estado. O que ocorreu foi que a ambulância ficou presa no trânsito, devido a uma rocha que deslizou na estrada. A população reclamou com o governador, que agora deseja evitar acontecimentos similares no futuro. Infelizmente, deslizamentos de rochas são muito comuns nesse estado, com muitas montanhas e serras. Assim, para minimizar o número de tragédia devidas a deslizamentos de rochas e outros imprevistos, o governador decidiu criar rotas alternativas entre cada cidade do estado e a capital. Para isso, é necessário inicialmente identificar quais segmentos de estradas são atualmente críticos, isto é, se bloqueados causam que não haja caminho possível entre alguma cidade e a capital. Um segmento de estrada é um trecho de estrada que liga duas cidades distintas.

Sua tarefa é escrever um programa para identificar esses segmentos críticos de estradas.

#### Entrada

A entrada é composta de vários casos de testes. A primeira linha de um caso de teste contém dois inteiros  $N$  e  $M$  que indicam respectivamente o número de cidades ( $2 \leq N \leq 100$ ) e o número de segmentos de estrada ( $1 \leq M \leq 10000$ ). Cada uma das  $N$  linhas seguintes contém o nome de uma cidade (apenas letras minúsculas e maiúsculas, comprimento máximo de 20 caracteres). A primeira dessas cidades é a capital do estado. Cada uma das  $M$  linhas seguintes descreve um segmento de estrada, contendo um par de nomes de cidades separados por um espaço em branco. Note que, como as montanhas causam dificuldade na construção de estradas, muitos segmentos de estrada são de mão única. Um segmento com duas mãos é representado por dois trechos de mão única. Você deve supor que existe ao menos um caminho de cada cidade para a capital. O final da entrada é indicado por  $N = M = 0$ . A entrada deve ser lida da entrada padrão. a

#### Saída

Para cada caso de teste seu programa deve listar os segmentos críticos, com um segmento crítico por linha. Cada segmento crítico deve ser representado por dois nomes de cidades separados por um espaço em branco. Os segmentos críticos de estrada devem ser listados na mesma ordem em que aparecem na entrada; para cada segmento, as cidades devem ser listadas na mesma ordem em que aparecem na entrada. Se não existir nenhum segmento crítico seu programa deve imprimir uma linha contendo apenas a palavra "Nenhuma". Imprima uma linha em branco após cada caso de teste.

A saída deve ser escrita na saída padrão.

Exemplo de entrada

Saída para o exemplo de entrada

6 10	Gramado NovoHamburgo
PortoAlegre	NovoHamburgo PortoAlegre
Gramado	RioGrande Pelotas
Canela	Pelotas PortoAlegre
NovoHamburgo	
Pelotas	
RioGrande	
Canela Gramado	
Canela NovoHamburgo	
Gramado NovoHamburgo	
NovoHamburgo PortoAlegre	
PortoAlegre NovoHamburgo	
RioGrande Pelotas	
Pelotas PortoAlegre	
PortoAlegre Pelotas	
Pelotas RioGrande	
NovoHamburgo Canela	
3 4	Nenhuma
Recife	
Olinda	
Paulista	
Olinda Recife	
Paulista Recife	
Olinda Paulista	
Paulista Olinda	
0 0	