OBI

OLIMPÍADA BRASILEIRA
DE INFORMÁTICA

OBI2001 - FASE 2

CADERNO DE TAREFAS

16/6/2001 • 9:00 às 14:00

LEIA ATENTAMENTE ESTAS INSTRUÇÕES ANTES DE INICIAR A PROVA

1. É proibido consultar livros, anotações ou qualquer outro material durante a prova. É permitido a consulta ao *help* do ambiente de programação se este estiver disponível.
2. A correção é automatizada, portanto siga atentamente as exigências da tarefa quanto ao formato da entrada e saída de seu programa.
3. Não implemente nenhum recurso gráfico nas suas soluções (janelas, menus, etc), nem utilize qualquer rotina para limpar a tela ou posicionar o cursor.
4. Todas as tarefas têm o mesmo valor na correção.
5. As tarefas não estão ordenadas, neste caderno, por ordem de dificuldade; procure resolver primeiro as questões mais fáceis.
6. 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.
7. Para problemas diferentes você pode escolher trabalhar com linguagens diferentes, mas apenas uma solução, em uma única linguagem, deve ser submetida para cada problema.
8. 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.
9. 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*.
10. 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
http://www.sbc.org.br Email: sbc@sbc.org.br

Pirâmide

Arquivo fonte: piramide.c, piramide.cc, piramide.cpp ou piramide.pas

Joana quer ser artista plástica, mas enquanto estuda procura trabalhos temporários durante suas férias escolares. Joana conseguiu emprego como auxiliar de almoxarifado em uma grande transportadora. A transportadora recebeu um enorme carregamento de caixas de mesma altura mas com largura e profundidades diferentes. As caixas podem ser empilhadas indefinidamente mas não podem ser deitadas em outra posição (todas têm que ser armazenadas obedecendo à indicação "Este lado para cima"). Joana é a responsável por armazenar o carregamento de caixas, e, seguindo seu senso artístico, quer construir com as caixas a pilha mais alta possível na forma de uma "pirâmide", ou seja, uma pilha construída de tal forma que uma caixa A é empilhada sobre uma outra caixa B somente se as dimensões de A (largura e a profundidade) não são maiores do que as dimensões de B (as caixas podem ser viradas de forma a trocar a profundidade com a largura). Você pode ajudá-la?

1. Tarefa

É dado um conjuto de caixas de mesma altura mas com largura e profundidades diferentes. Sua tarefa é escrever um programa que determine qual a pilha de caixas mais alta que Joana pode construir com as restrições acima.

2. Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de testes contém um número inteiro N que indica a quantidade de caixas do conjunto. As N linhas seguintes contêm, cada uma, a descrição de uma caixa. Uma caixa é descrita por dois inteiros X e Y ($1 \leq X \leq 15000$ e $1 \leq Y \leq 15000$) que representam os valores de cada lado da peça. O final da entrada é indicado por $N = 0$.

Exemplo de Entrada

```
3
100 100
1000 2000
2000 500
6
3 4
5 7
7 5
1 5
4 4
10 2
0
```

3. 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 o número máximo de caixas que podem ser empilhadas na forma de uma pirâmide, 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

Teste 2
4
```

(esta saída corresponde ao exemplo de entrada acima)

4. Restrições

$0 \leq N \leq 5000$ (N = 0 apenas para indicar o final da entrada)
$1 \leq X \leq 15000$
$1 \leq Y \leq 15000$

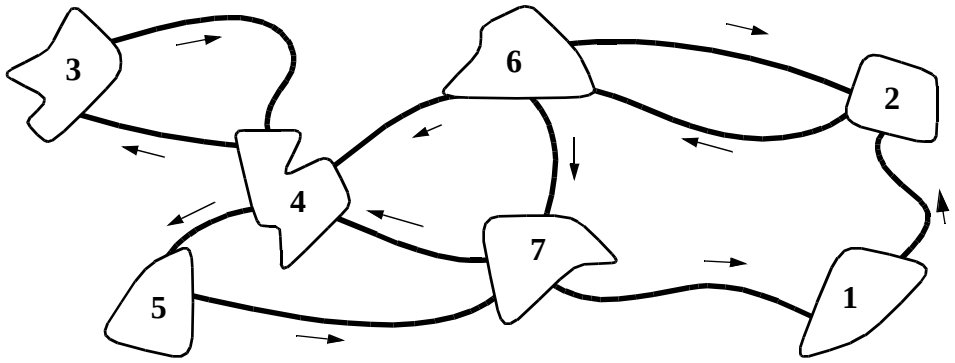
Ambulância

Arquivo fonte: ambula.c, ambula.cc, ambula.cpp ou ambula.pas

O município de Águas Molhadas é formado por várias vilas interligadas por igarapés. Os igarapés funcionam como sistema viário, já que a maioria dos habitantes usa barcos como meio de transporte. Os igarapés têm uma forte corrente, o que obriga os barcos a transitarem todos no mesmo sentido em cada igarapé (na verdade, é proibido o trânsito de barcos na contra-mão). A prefeitura do município mantém um único Posto de Saúde e um barco-ambulância utilizado para transporte de doentes.

O timoneiro do barco-ambulância está sempre com pressa, e tem medo de confundir-se no emaranhado de igarapés quando atende um chamado para buscar um doente em alguma vila. Por isso, ele deseja uma lista de todos os possíveis caminhos entre a vila onde se encontra o Posto de Saúde e cada outra vila do município.

.



Sete vilas interligadas por onze igarapés com mão única

1. Tarefa

Sua tarefa é escrever um programa que liste todos os caminhos que partem da vila onde se encontra o Posto de Saúde até cada outra vila do município, de forma que em cada caminho nenhuma vila é repetida. As vilas são numeradas de 1 a *N*, sendo que a vila onde se encontra o Posto de Saúde será sempre a de número 1.

2. Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um número inteiro *N* que indica a quantidade de vilas do município. As linhas seguintes contêm, cada uma, dois inteiros positivos *X* e *Y* que indicam que a vila *X* tem um igarapé que a liga diretamente com a vila *Y*, sem passar por outras vilas, com o trânsito fluindo na direção de *X* para *Y*. O final da lista de igarapés é indicado por $X = 0$ e $Y = 0$. O final da entrada é indicado por $N = 0$.

Exemplo de Entrada

```
2
1 2
2 1
0 0
7
1 2
2 6
7 4
7 1
4 5
4 3
3 4
6 4
6 7
6 2
5 7
0 0
0
```

3. Saída

Para cada conjunto de teste da entrada seu programa deve produzir um conjunto de saída com uma lista de caminhos. A primeira linha de um conjunto de saída deve conter o identificador do conjunto, no formato "Teste *n*", onde *n* é numerado a partir de 1. As linhas seguintes devem conter os caminhos, ordenados lexicograficamente. Cada caminho é composto de uma seqüência de identificadores de vilas separados por um ou mais espaços em branco. A última linha de um conjunto de saída deve ser deixada em branco.

Exemplo de Saída

```
Teste 1
1 2

Teste 2
1 2
1 2 6
1 2 6 4
1 2 6 4 3
1 2 6 4 5
1 2 6 4 5 7
1 2 6 7
1 2 6 7 4
1 2 6 7 4 3
1 2 6 7 4 5
```

(esta saída corresponde ao exemplo de entrada acima)

4. Restrições

$0 \leq N \leq 100$ (N = 0 apenas para indicar o final da entrada)
$1 \leq Y \leq N$
$1 \leq X \leq N$

Batuíra

Arquivo fonte: batuira.c, batuira.cc, batuira.cpp ou batuira.pas

As batuíras são aves migrantes de pequeno porte que fazem seus ninhos no hemisfério norte, na tundra ártica (no Canadá e Groenlândia). Elas permanecem naquela região durante o verão (de maio a junho no hemisfério norte), que é muito curto, e depois migram para o outro lado da Terra. As batuíras-de-peito-vermelho fazem seus ninhos em um raio de até mil quilômetros do Pólo Norte. Quando chega a época de migrarem, fazem longos vôos, em bandos, até o Sudeste dos Estados Unidos. Em seguida, partem para a Patagônia, onde passam o período quente do Hemisfério Sul (de outubro a março). Em março começam sua longa viagem de volta.

Como essas aves se orientam para realizar trajetos tão longos? Não existe uma resposta definitiva, mas apenas hipóteses. Uma das mais prováveis que é elas têm a capacidade de realizar a navegação celestial, isto é, de se guiarem pelos astros. Outra hipótese é que poderiam sentir os campos magnéticos da Terra, utilizando-os como referência.

As batuíras fazem longos vôos, mas precisam parar de vez em quando para recuperar suas forças, em *pontos de repouso*, normalmente praias ou beiras de lagos. Dependendo do trajeto escolhido e da localização dos *pontos de repouso*, a viagem migratória das batuíras pode variar em cerca de 10% de ano a ano -- o que, no caso, significa uma variação de mil quilômetros!

1. Tarefa

Você deve escrever um programa que, dadas as distâncias entre possíveis *pontos de repouso* das batuíras determine o comprimento total do menor trajeto que as aves poderiam seguir em sua viagem do hemisfério norte para o hemisfério sul.

2. Entrada

A entrada é composta de vários conjuntos de teste. A primeira linha de um conjunto de teste contém um número inteiro N que indica a quantidade de pontos de repouso . Os pontos de repouso são numerados de 1 a N, sendo 1 o ponto de partida e N o ponto de chegada do vôo migratório. As linhas seguintes contêm, cada uma, três inteiros não negativos X e Y que indicam que a distância do ponto de repouso X ao ponto de repouso Y é Z. O final do conjunto de teste é indicado por $X = 0$,$Y = 0$ e $Z = 0$. O final da entrada é indicado por $N = 0$.

Exemplo de Entrada

```
3
1 2 300
2 3 400
0 0 0
6
1 2 10
1 3 20
5 2 4
3 4 7
```

```
6 5 30
2 6 50
4 5 7
4 6 9
0 0 0
0
```

3. 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 o comprimento do menor trajeto 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
700

Teste 2
30
```

(esta saída corresponde ao exemplo de entrada acima)

4. Restrições

$0 \leq N \leq 100$ ($N = 0$ apenas para indicar o final da entrada)
$0 \leq X \leq N$ ($X = 0$ apenas para indicar o final do conjunto de teste)
$0 \leq Y \leq N$ ($Y = 0$ apenas para indicar o final do conjunto de teste)
$0 \leq Z \leq 1000$ ($Z = 0$ apenas para indicar o final do conjunto de teste)

Clipe

Arquivo fonte: clipe.c, clipe.cc, clipe.cpp ou clipe.pas

Arnaldinho é fanático por música, e um de seus passatempos preferidos é assistir clipes de música na televisão. No próximo domingo vai haver um festival de clipes, e várias emissoras de televisão programaram clipes a partir das 9:00 da manhã. Arnaldinho gosta de assistir a um clipe do começo ao fim (detesta ver um clipe já iniciado, ou interromper um clipe que esteja assistindo), e quer programar-se para assistir à maior quantidade possível de clipes (inteiros!).

1. Tarefa

Sua tarefa é escrever um programa que leia a programação de clipes de várias emissoras e determine qual a máxima quantidade de clipes inteiros que Arnaldinho poderá assistir. A programação dos clipes é dada por uma lista com a especificação de início e duração de cada clipe a ser transmitido.

2. Entrada

A entrada é composta de vários conjuntos de testes. A primeira linha de um conjunto de testes contém um número inteiro N ($0 \leq N \leq 10000$), indicando o número de clipes. As N linhas seguintes de um conjunto de testes contêm cada uma a programação de um clipe, indicada por dois inteiros X e Y, representando respectivamente o início do clipe (em segundos a partir das 9:00 de domingo) e a duração do clipe (em segundos). Se o término de um clipe ocorre no mesmo segundo que o início de outro clipe, considere que há conflito (Arnaldinho não gosta de perder nem um segundo de clipe), ou seja, apenas um dos clipes pode ser escolhido (os dois clipes não podem ser escolhidos juntos). Por exemplo, se um clipe A inicia no instante 1000 e dura 250 segundos, e um clipe B inicia no instante 1249, então B não pode ser escolhido juntamente com A. Por outro lado, se um clipe C inicia no instante 1250, então A e C podem ser escolhidos juntos.

Exemplo de Entrada

```
3
0 320
200 340
700 380
5
1800 280
0 340
2080 300
339 310
800 430
0
```

3. Saída

Para cada conjunto de testes da entrada seu programa deve produzir três linhas. A primeira linha deve conter um identificador do conjunto de teste, no formato “Teste *n*”, onde *n* é numerado a par-

tir de 1. Na segunda linha deve aparecer o número máximo de clipes encontrado 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
2

Teste 2
4
```

(esta saída corresponde ao exemplo de entrada acima)

4. Restrições

$1 \leq N \leq 10000$
$0 \leq X \leq 10000$
$1 \leq Y \leq 10000$