IX Maratona de Programação

Universidade de São Paulo

Caderno de Problemas

Departamento de Ciência da Computação, IME–USP

Domingo, 21 de Agosto de 2005

Problema A: Spurs rocks

`Arquivo: spurs.[c|cpp|java|pas]`

O San Antonio é o time da cidade na NBA. Já foi algumas vezes campeão de sua conferência e revelou vários excelentes jogadores.

Em um campeonato de basquete os times jogam todos entre si em turno único. A vitória vale dois pontos e a derrota vale um ponto (não há empates no basquete). Havendo empates na pontuação do campeonato fica na frente o time com melhor "cesta average" que é dado pela razão entre o número de pontos marcados pelo time dividido pelo número de pontos recebidos (na improvável hipótese de um time vencer todos os jogos do campeonato sem levar cestas seu cesta average é dado pelo número de pontos marcados). Persistindo o empate, leva vantagem quem marcou mais pontos. Ainda havendo empate, o time com menor número de inscrição na liga fica na frente.

Sua tarefa neste problema é fazer um programa que recebe os resultados dos jogos de um campeonato e imprime a classificação final.

Entrada

São dadas várias instâncias. Para cada instância é dada o número $0 \leq n \leq 100$ de times no campeonato. O valor $n = 0$ indica o fim dos dados. A seguir vêm $\frac{n(n-1)}{2}$ linhas indicando os resultados das partidas. Em cada linha são dados quatro inteiros x, y, z e w. Os inteiros x e z pertencem ao conjunto $\{1, 2, \ldots, n\}$ e representam os números de inscrição dos times na liga. Os inteiros y e w são, respectivamente, os números de pontos do time x e do time z na partida descrita.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Na linha seguinte, deve ser impressa a permutação dos inteiros 1 a n da classificação do campeonato. Um espaço em branco deve separar cada um desses inteiros, e uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
5
1 102 2 62
1 128 3 127
1 144 4 80
1 102 4 101
2 62 3 61
2 100 4 80
2 88 5 82
3 79 4 90
3 87 5 100
4 110 5 99
0
```

Exemplo de saída

```
Instancia 1
1 2 4 5 3
```

Problema B: Fly by Night

`Arquivo: fly.[c|cpp|java|pas]`

Bill Poucher anunciou em Shangai (China), em abril último, que a trigésima final mundial do ACM-ICPC será realizada em San Antonio (USA) na segunda semana de 2006.

Ao tomar conhecimento de tal informação (com alguns meses de atraso), uma empresa de transportes aéreos do Texas – conhecida como *Fly by Night Ltd.* – decidiu aproveitar o evento para tentar incrementar seu ganho anual.

O objetivo do CEO da empresa era oferecer transporte aéreo para os times (incluindo competidores e técnicos) e para a equipe de suporte (aqueles que fazem as coisas funcionarem) a partir de suas cidades de origem, em seus países de origem, até o local da competição. Para tentar garantir o sucesso de sua idéia, o mesmo CEO ofereceu tarifas ligeiramente abaixo do preço de mercado para aqueles que seriam transportados. Como eles eram em sua maioria estudantes e professores universitários, toparam na hora.

Como você já deve ter imaginado, a *Fly by Night Ltd.* opera vôos noturnos. No entanto, em vez de possuir seus próprios aviões, a referida empresa apenas vende assentos em vôos de outras companhias. Ela ganha uma boa comissão devido ao fato de tais vôos terem, historicamente, uma baixa ocupação.

No entanto, quando os funcionários da empresa foram verificar os vôos que teriam à disposição para realizarem a operação, tiveram uma bela surpresa. A maior parte dos vôos estava completamente lotada. Os que não estavam lotados, não possuiam muitos assentos livres. Ninguém soube explicar o motivo de tal demanda irregular. Duas hipótese foram levantadas: a proximidade do spring-break americano e a popularidade da competição. :-)

Na tentativa de salvar a empresa (e seu próprio cargo), o CEO percebeu que teria de utilizar escalas e baldeações. O lucro desta forma seria menor, mas nada comparado ao prejuízo que teria se operasse com vôos diurnos ou deixasse de transportar os passageiros (que naquela altura, já tinham pago as passagens...).

Os funcionários da *Fly by Night Ltd.* levantaram então um conjunto de cenários com vôos que poderiam ser utilizados. O que foi percebido pouco depois é que nem todos os cenários eram viáveis, já que nem todos conseguiam transportar o montante de passageiros necessário. Finalmente, o CEO percebeu que não tinha pessoal qualificado para lidar com a situação. Você foi então contratado para desenvolver um programa que, para cada cenário construído, responda se o cenário é viável ou inviável.

Entrada

Um cenário será, daqui em diante, chamado de instância. Seu programa deve estar preparado para lidar com diversas instâncias.

Cada instância começa com um inteiro $0 \leq m \leq 100$ que especifica o número de cidades de origem dos passageiros que devem ser transportados. Um valor $m = 0$ indica o final das instâncias e não deverá ser processado. Em caso contrário, em cada uma das próximas m linhas, são dados o nome de uma cidade de origem e o respectivo número de passageiros

daquela cidade (um inteiro não negativo menor ou igual a 100). O nome de uma cidade possui entre 1 e 20 caracteres tomados do alfabeto $\Sigma = \{a, b, \ldots, z, -\}$.

Na próxima linha são dados um inteiro $0 \leq n \leq 100$, que representa o número de vôos da instância, e o nome da cidade em que ocorrerá o evento (o CEO decidiu que o programa deveria aceitar isso). O nome desta cidade segue as mesmas regras estabelecidas acima.

Em cada uma das próximas n linhas são dados os nomes de duas cidades de um vôo (origem e destino, respectivamente), seguido por um inteiro não negativo menor ou igual a 200 que representa o número de assentos livres naquele vôo. Novamente os nomes das cidades são sobre Σ e de comprimento entre 1 e 20. Você pode supor que não há duas cidades com o mesmo nome, e que as cidades de origem e destino são sempre diferentes. Além disso, a *Fly by Night Ltd.* não trabalha com mais de um vôo entre quaisquer duas cidades.

Em cada linha da entrada, um número qualquer de espaços pode separar os dados fornecidos.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Na linha seguinte, você deve imprimir `viavel` se é possível transportar todos os passageiros de suas origens até o destino especificado, e `inviavel` em caso contrário. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
3
boston    3
sao-paulo 4
waterloo  5
10 san-antonio
atlanta   san-antonio 2
boston    dallas      4
dallas    san-antonio 3
denver    san-antonio 3
houston   san-antonio 6
sao-paulo atlanta     2
sao-paulo houston     3
waterloo  atlanta     1
waterloo  denver      3
waterloo  houston     2
1
san-francisco 11
7 new-york
san-francisco denver   5
san-francisco houston  6
denver        atlanta  4
denver        chicago  2
houston       atlanta  5
atlanta       new-york 7
chicago       new-york 4
0
```

Exemplo de saída

```
Instancia 1
viavel

Instancia 2
inviavel
```

Problema C: Ramsey's Saloon

`Arquivo: ramsey.[c|cpp|java|pas]`

Bill "Snake" Ramsey foi um dos mais famosos donos de saloon em San Antonio. Seu saloon era conhecido até a costa oeste, e suas mesas de pôquer sempre lotadas eram sinônimo de jogos eletrizantes, muito dinheiro e, muitas vezes, muitas disputas sangrentas.

Ramsey tinha uma teoria (e seu 38 intimidava os que dela discordavam ao contestá-la) de que em uma mesa de pôquer com 6 participantes havia sempre ou 3 que eram amigos entre si, ou 3 que eram inimigos entre si (naquela época em San Antonio se você não era amigo de alguém automaticamente se tornava seu inimigo). Hoje sabemos que Ramsey tinha de fato razão.

Sua tarefa neste problema é checar a afirmação de Ramsey para vários exemplos.

Entrada

São dadas várias mesas de pôquer (cada mesa tem sempre 6 jogadores). Para cada mesa é dado o número $-1 \leq m \leq 15$ de pares de amigos seguido, na linha seguinte, dos nomes dos participantes daquele jogo (cada nome é uma string de no mínimo 1 e no máximo 15 caracteres e você pode supor que os nomes dos jogadores são dois a dois distintos). O valor -1 indica o fim dos dados. Em seguida, vêm m linhas, cada uma com os nomes de dois amigos naquela mesa. Considere que um jogador não é amigo de si mesmo.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Nas próximas linhas, você deve imprimir os nomes de três jogadores daquela mesa seguida de `sao amigos` ou `sao inimigos` conforme o caso. Devem haver tantas linhas quantos forem os casos determinados. Estas linhas devem estar listadas em ordem lexicográfica. O mesmo vale para os três nomes em uma mesma linha. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
6
jack john bill jesse james pil
jack john
jack bill
jack jesse
bill james
jesse james
james pil
-1
```

Exemplo de saída

```
Instancia 1
bill jesse john sao inimigos
bill jesse pil sao inimigos
bill john pil sao inimigos
jesse john pil sao inimigos
```

Problema D: Houston, we've got a problem!

`Arquivo: houston.[c|cpp|java|pas]`

No dia primeiro de julho de 1947, um estranho objeto foi detectado por radares da força aérea americana instalados em Roswell, White Sands e Alamogordo. A tremenda velocidade e os movimentos erráticos do objeto indicaram que ele não era um avião ou meteorito. Quatro dias depois um pastor de ovelhas e um grupo de arqueólogos encontram restos de um objeto acidentado ao norte de Roswell. A partir daí, autoridades americanas entram em cena e transportam os restos de tal objeto para Fort Worth no Texas. Elas disseram que os destroços encontrados eram simplesmente restos de um balão meteorológico experimental. Muitas pessoas, no entanto, acharam que se tratavam dos restos de um objeto voador não identificado (UFO). Muitos anos se passaram desde então, e o caso continua atraindo atenção e gerando polêmica.

Um grupo de ufólogos radicado em San Antonio, uma cidade texana situada a sul-sudoeste de Fort Worth, está convencido de que seres extraterrestres têm visitado a região com freqüência desde então. Após muita pesquisa, os ufólogos descobriram que poderiam construir uma rede de comunicação alternativa para tentar contactar os ET's. Tal rede utilizaria resquícios do antigo sistema de telégrafos existentes no deserto do Texas e o fato de sua alternatividade advém da tentativa de evitar, segundo eles, a intromissão das autoridades supra citadas.

Após um minucioso levantamento (que identificou postes, fiações, condensadores, transformadores, etc.), os ufólogos perceberam que informações transmitidas em certos trechos da antiga estrutura dos telégrafos apresentavam qualidade pior do que em outros. Baseados em amostras estatísticas, levantaram, para alguns pares de pontos u e v da antiga rede, uma probabilidade p_{uv} de haver interferência nas informações transmitidas entre u e v. Sabendo que você estaria na região em abril do ano que vem, eles pediram a você que construísse um programa para identificar o menor conjunto de trechos a serem utilizados, de forma que (i) todos os pontos por eles desejados estejam interligados (mesmo que indiretamente), e tal que (ii) a probabilidade total de interferência nas mensagens enviadas nesta rede alternativa seja mínima. Ávido de interesse em descobrir a verdade (que "está lá fora..."), você prontamente atendeu à solucitação.

Entrada

Seu programa deve estar preparado para lidar com diversas instâncias. Cada instância possiu o formato que segue. Na primeira linha, são especificados dois inteiros $0 \leq n \leq 100$ e $0 \leq m \leq \frac{n(n-1)}{2}$ que representam, respectivamente, o número de pontos da rede alternativa e o número de pares desses pontos para os quais as probabilidades de haver interferência foram medidas. Nas m linhas seguintes, são dados (em cada linha) dois inteiros $1 \leq u, v \leq n$ e um racional $0 \leq p_{uv} \leq 1$ representando que entre os pontos u e v, a probabilidade de interferência é p_{uv}. Um valor $n = 0$ indica o término das instâncias e não deve ser processado. Você pode supor que sempre será possível satisfazer a restrição (i).

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Na próxima linha, você deve imprimir (com cinco casas decimais) a probabilidade mínima de interferência calculada para tal instância. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
5 8
1 2 0.4
1 3 0.1
1 4 0.6
2 3 0.9
2 4 0.5
3 4 0.2
3 5 0.7
4 5 0.1
3 3
1 2 1.0
1 3 1.0
2 3 1.0
0 0
```

Exemplo de saída

```
Instancia 1
0.61120

Instancia 2
1.00000
```

Problema E: Defending Alamo

`Arquivo: alamo.[c|cpp|java|pas]`

O forte do Alamo, originalmente chamado de Misión San Antonio de Valero, foi fundado pelos missionários para abrigar os padres e os índios convertidos na região que era disputada pelos colonos americanos e espanhóis. Foi fundado no século XVIII e serviu de palco para a mais sangrenta batalha pela emancipação do Texas. A batalha do Alamo ocorreu em 23 de fevereiro de 1836 quando o exército do Gal. Antonio Lopes de Sant'Anna cercou o forte. Texanos e "tejanos" (chamados "defenders") defenderam a posição por 13 dias.[1]

O forte do Alamo era uma fortificação de formato bastante intrincado, cercado por uma alta cerca. Muitas vezes era difícil dizer se um soldado estava dentro ou fora dos limites do forte.

Sua tarefa neste problema é dada uma instância de um forte, dado pelas coordenadas dos vértices da cerca, as coordenadas da bandeira do Texas, e a posição de vários soldados, determinar quais deles são "defenders" e quais são espanhóis.

Entrada

São dadas várias instâncias. Cada instância começa com um inteiro que é o número $0 \leq n \leq 1000$ de vértices que a cerca do forte tem. O valor 0 indica o fim dos dados. Nas n linhas seguintes vêm as coordenadas dos postes da cerca do forte. Os postes da cerca são dados a partir do primeiro, seguindo a cerca em sentido horário. A seguir vem a posição da bandeira do Texas. Na próxima linha, vem o número $0 \leq m \leq 1000$ de pessoas a serem verificadas. Nas m linhas seguintes vêm as coordenadas das posições de cada uma das pessoas. Todas as coordenadas fornecidas são números inteiros no intervalo $[-100000; +100000]$.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Nas m linhas seguintes, você deve imprimir `soldado k` (para $k = 1, \ldots, m$) seguido de `defender` ou `espanhol` respectivamente se o soldado estiver dentro ou fora do forte. Uma linha em branco deve separar a saída de cada instância.

[1] Tudo isso é sério. A partir do próximo parágrafo, nem tanto...

Exemplo de entrada

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

Exemplo de saída

```
Instancia 1
soldado 1 defender
soldado 2 espanhol
```

Problema F: Entregadores de steaks

`Arquivo: steaks.[c|cpp|java|pas]`

O Texas é famoso pela sua carne de excelente qualidade. "Steaks" com até dois centímetros de espessura assados em churrasqueiras são a especialidade culinária do estado. Em San Antonio é difícil encontrar entregadores de pizza por telefone, mas é muito comum encontrar "disk steaks". Você liga para o número e em poucos minutos chega um suculento bife à sua casa, quente e pronto para comer. É claro que tamanha eficiência depende de um complicado sistema de entregas. Há várias sedes da empresa espalhadas pela cidade, e sempre que uma chamada é feita a sede mais próxima é acionada, o steak é assado e o entregador segue com o suculento jantar.

Sabemos que San Antonio é uma cidade planejada. Podemos imaginar os cruzamentos da cidade como vértices de uma grade. Por algum motivo obscuro, todas as sedes estão instaladas em cruzamentos. Sua tarefa é ajudar a empresa na entrega dos steaks.

Entrada

São dadas várias instâncias. Para cada instância são dadas as dimensões $0 \leq m, n \leq 1000$ da cidade (será uma grade com m linhas e n colunas). Um valor $n = 0$ ou $m = 0$ indica o fim dos dados. A seguir vem o número $0 < k \leq 100000$ de sedes da empresa. Nas k linhas seguintes vêm as coordenadas das sedes. A seguir, vem o número $0 \leq l \leq 100000$ de ligações pedindo steaks. Nas l linhas seguintes vêm as coordenadas da posição de cada chamada (que também são vértices da grade).

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Nas l linhas seguintes, você deve imprimir por qual sede da empresa o pedido correspondente àquela linha foi atendido. Em caso de haver mais de uma sede à mesma distância, dê preferência pela que possuir menor índice de linha. Persistindo o empate, dê preferência pela com menor índice de coluna. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
3 3
2
1 1
1 3
2
2 1
2 3
0 0
```

Exemplo de saída

```
Instancia 1
1 1
1 3
```

Problema G: Números de ahmoc

`Arquivo: ahmoc.[c|cpp|java|pas]`

Antes da colonização hispânica e depois inglesa a região de San Antonio era dominada pelos índios ahmoc-axhozupeck, ancestrais dos sioux e dos apache. A etnia foi completamente destruída pelos colonizadores, no século XVIII, tornando impossível a tarefa de decifrar seus grandes painéis.

O Departamento de Arqueologia da Universidade Baylor dedica boa parte de sua pesquisa aos painéis dos índios ahmoc. Surpreendentemente os índios já conheciam os algarismos hindus, mas não o usavam para cálculos (afinal não existia comércio naquela civilização). Os arqueólogos de Baylor suspeitam que os painéis repletos de números eram apenas decorativos. Também suspeitam que alguns padrões que se repetiam eram assinaturas dos artistas, a fim de garantir a autenticidade do painel.

Sua tarefa neste problema será verificar se os painéis são verdadeiros, ou seja, se, de fato, contêm a assinatura do artista que o arqueólogo suspeita ser o autor.

Entrada

São dadas várias instâncias de teste. Cada instância começa com um número inteiro positivo $0 \leq a \leq 1000000$ que é a assinatura do artista. O inteiro 0 indica o fim dos dados. Na linha seguinte vem a seqüência de números do painel, que poderá ter até 300000 algarismos.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Na linha seguinte, você deverá imprimir `verdadeira` se a seqüência de números contém a assinatura do artista ou `falsa` em caso contrário. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
1234
837384937292379450545045672392303485065402302373543504864694450034302
23034
837384937292379450545045672392303485065402302373543504864694450034302
0
```

Exemplo de saída

```
Instancia 1
falsa

Instancia 2
verdadeira
```

Problema H: Six Flags

`Arquivo: flags.[c|cpp|java|pas]`

O Six Flags Fiesta Texas é um dos maiores parques de diversão do mundo, e fica em San Antonio. Sabendo que as finais do ACM-ICPC de 2006 serão naquela cidade, três colegas começaram a planejar em quais dos famosos brinquedos eles iriam, caso seu time se classificasse para as finais mundiais.

Para isso, estabeleceram notas para cada uma das atrações de acordo com o quanto eles gostariam de brincar lá. Por exemplo, a montanha russa "Superman Krypton Coaster"(que tem $800m$ de giros, loops e quedas com o carrinho indo a mais de $100km/h$) recebeu a maior pontuação possível entre os colegas.

O problema é que é impossível visitar todas as atrações em um mesmo dia. Assim, os colegas pesquisaram, para cada uma delas, quanto tempo durava o brinquedo (e quanto tempo de fila teriam de enfrentar até chegar a ele...). Sua tarefa neste problema é encontrar, dado o tempo disponível pelos colegas no Six Flags, uma coleção (pode haver repetições) de atrações que dá a maior pontuação dentro deste período.

Entrada

Seu programa deve estar preparado para processar diversas instâncias. Na primeira linha são dados dois inteiros $0 \leq n \leq 100$ e $0 \leq t \leq 600$, em que n é o número de atrações nas quais os colegas gostariam de brincar, e t é o tempo (em minutos) que eles terão disponível para isso. Nas próximas n linhas, são dados dois inteiros $0 < d \leq 600$ e $0 \leq p \leq 100$ (em cada linha). O primeiro deles, d, representa a duração do brinquedo (incluído aí o tempo de fila e uma estimativa do tempo de traslado entre os brinquedos). O segundo, p, representa a pontuação atribuída ao brinquedo pelos colegas. Um valor $n = 0$ indica o final das instâncias e não deverá ser processado.

Saída

Para cada instância solucionada, você deverá imprimir um identificador `Instancia h` em que `h` é um número inteiro, seqüencial e crescente a partir de 1. Na linha seguinte, deve ser impressa a pontuação total conseguida com a coleção determinada por seu programa. Com relação a quais são as atrações da coleção determinada, os colegas decidiram que iriam perguntar para você pessoalmente no futuro, já que eles não querem que outras pessoas saibam e venham a utilizá-la. Uma linha em branco deve separar a saída de cada instância.

Exemplo de entrada

```
5 60
10 30
20 32
5  4
50 90
22 45
5 60
10 10
20 32
5  4
50 90
22 45
0 0
```

Exemplo de saída

```
Instancia 1
180

Instancia 2
104
```