Maratona de Programação

IME – USP

Caderno de problemas

18 de Agosto de 2002

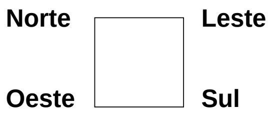
Problema A: Formigando

`Arquivo: formiga.[c|cc|pas|java]`
`Entrada: formiga.in`

Um grupo de cientistas malucos da ACGOP (Academia de Ciência e Ginástica Olímpica Pindamonhangabense) está estudando o comportamento de uma estranha raça de formigas cegas. Após anos de estudo, eles concluíram que:

1. Essas formigas só andam em linha reta, e seu comportamento é o mesmo da luz. Ou seja, quando encontram uma parede pela frente, sua rota é "refletida" (como se fosse um raio de luz incidindo num espelho plano) e ela segue andando de modo que sua rota forma com o obstáculo um ângulo igual ao de incidência.
2. Essas formigas só param quando passam por algum alimento, e voltam a andar no mesmo sentido em que vinham logo após devorar o alimento;
3. O máximo que uma formiga consegue viajar sem morrer de fome é 2 metros.

Para comprovar essas conclusões, os cientistas construíram uma série de cenários, onde eles colocariam as formigas para andar. Cada cenário consiste num retângulo de dimensões $m \times n$ metros com 4 entradas, situadas nos vértices, que de agora em diante chamaremos pelos nomes dos pontos cardeais, conforme a figura:



Dentro do retângulo foram distribuídas mn comidas, colocadas no centro de cada quadrado de 1 metro de lado interior ao retângulo. As formigas são introduzidas nos retângulos por uma das quatro portas, numa rota que forma 45 graus com os lados do retângulo de modo que, após percorrer aproximadamente $0,707$ metros, alcançam o primeiro alimento e, em seguida, continuam suas viagens pelo mesmo caminho, até que uma das três situações ocorra:

1. A formiga bate na parede: neste caso sua rota é refletida e ela segue andando de modo que sua rota forma com a parede um ângulo igual ao de incidência (ou seja, permanece 45 graus).
2. A formiga anda mais de 2 metros e não encontra comida: nesse caso ela morre (tadinha);
3. A formiga encontra uma porta: nesse caso ela sai e o experimento está concluído.

Sua missão é escrever um programa que preveja o que vai acontecer com a formiga colocada em cada um dos cenários da pesquisa.

Entrada

A entrada consistirá de diversos cenários. Para cada cenário serão fornecidas, numa única linha, as medidas do retângulo, primeiro a largura m (Norte – Oeste), depois a profundidade n (Norte – Leste) $(0 < m, n \leq$

100.000), e a porta por onde entrou a formiguinha. Uma linha contendo o valor 0 para uma das dimensões do retângulo marca o final da entrada.

Saída

Para cada cenário, você deverá escrever uma linha no arquivo de saída, com um dos seguintes conteúdos:

1. No caso da formiga morrer de fome, a expressão "Morreu..."
2. Caso contrário, você deverá escrever a porcentagem das comidas que foram devoradas na peregrinação da formiga rumo à saída (com três dígitos, espaços à esquerda se for o caso, arredondado para o inteiro mais próximo, ou para baixo, no empate, seguido do sinal %); a quantidade de trombadas que a formiga deu nas paredes do retângulo (10 casas, alinhado à direita, espaços à esquerda), e a porta pela qual ela saiu.

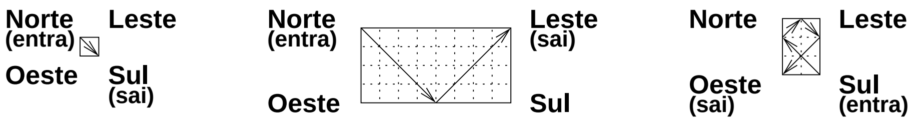
Exemplo

Entrada

```
1 1 Norte
4 8 Norte
3 2 Sul
100000 100000 Oeste
100000 0 Qualquercoisaaquitavalendo
```

Saída

```
100%         0 Sul
 25%         1 Leste
100%         3 Oeste
  0%         0 Leste
```



Problema B: California Über Alles

`Arquivo: alles.[c|cc|pas|java]`
`Entrada: alles.in`

Depois que a imprensa californiana noticiou, com grande entusiasmo, a realização das finais mundiais do XXVII International Collegiate Programing Contest da ACM nos arredores de Hollywood, uma onda de otimismo atingiu os comerciantes locais. Com a concentração de estudantes de computação das mais diferentes partes do mundo, eles esperam aumentar significativamente as vendas de seus produtos durante o evento.

De olho neste possível aquecimento do mercado, também encontra-se uma das maiores empresas de transporte de mercadorias do estado: a "California Über Alles". Seu atual diretor, K. Flouride, encomendou algumas pesquisas de campo que identificaram pares de localidades com demandas para escoamento de produtos. Com estes dados, Flouride considera a instalação de terminais de compra e venda em alguns destes pares. Mais especificamente, sendo i e j duas localidades, para satisfazer a demanda existente entre i e j e, obter um faturamento f_{ij}, a companhia deve instalar terminais em ambas as localidades. A instalação de um terminal em uma localidade i acarreta, no entanto, um custo c_i.

Como programador em treinamento dos laboratórios computacionais da California Über Alles, sua tarefa é construir um programa que ajude Flouride a determinar onde instalar os terminais de forma que o lucro líquido, isto é, o faturamento obtido por satisfazer as demandas menos o custo de instalação dos terminais, seja máximo.

Entrada

A entrada é constituída por várias pesquisas, armazenadas em seqüência ao longo do arquivo de entrada. Cada pesquisa possui a estrutura que segue. Na primeira linha é especificado um valor inteiro $0 \leq n \leq 15$ que indica o número de localidades em questão. Cada localidade deverá ser identificada por um número inteiro distinto entre 1 e n. O caso em que $n = 0$ representa o final do arquivo e não deve ser processado.

Na segunda linha, são fornecidos n valores inteiros correspondentes aos custos de instalação de um terminal em cada localidade (na ordem de identificação das mesmas). Na terceira linha é especificado um valor inteiro $0 \leq m < n^2$ que indica o número de pares com demandas identificados na pesquisa. Em cada uma das próximas m linhas, são fornecidos dois inteiros $1 \leq i, j \leq n$ indicando as localidades i e j e um inteiro f_{ij} que é o faturamento obtido pela satisfação da demanda existente entre tal par.

Saída

Para cada pesquisa processada, seu programa deve imprimir na primeira linha o texto `Pesquisa #x`, onde x é o número da pesquisa no arquivo de entrada. (As pesquisas são numeradas sequencialmente a partir de um.) Na segunda linha deve ser impresso o lucro líquido calculado e, na terceira linha, os números das localidades escolhidas, em ordem crescente. No caso de existência de mais de uma lista de localidades, escolha uma delas. Uma linha em branco deve ser deixada entre cada pesquisa.

Exemplo

Entrada

```
4
8 6 2 4
5
1 2 4
1 3 10
1 4 5
2 3 6
3 4 3
3
1 1 1
3
1 2 0
1 3 0
2 3 1
0
```

Saída

```
Pesquisa #1
8
1 2 3 4

Pesquisa #2
0
```

Problema C: The Oscar goes to...

```
Arquivo:  oscar.[c|cc|pas|java]
Entrada:  oscar.in
```

Não há momento de maior angústia entre as estrelas do cinema americano que a cerimônia de entrega das estatuetas no Kodak Theatre. Não são raros os atores e atrizes que vêem suas pulsações atingirem níveis alarmantes. Os riscos de um enfarto crescem muito quando os batimentos se tornam mais e mais rápidos em períodos consecutivos, caracterizando um período de grande ansiedade.

Preocupada com este fato, a Academia de Artes e Ciências de Hollywood resolveu monitorar em intervalos regulares os batimentos cardíacos dos vários candidatos a receber a estatueta. Com estes dados pretende-se fazer um programa que alerte os médicos de plantão caso um dos famosos esteja à beira de um ataque cardíaco. Sua tarefa neste problema é auxiliar a Academia nesta nobre missão.

Entrada

São dadas várias instâncias. A primeira linha de cada instância contém o número $0 \leq n \leq 1000$ de atores/atrizes monitorados (que serão identificados pelos números $1, 2, \ldots, n$) e o número $0 \leq m \leq 100$ de batimentos observados nestes atores. A seguir, em cada uma das próximas n linhas são dadas as m medições (o batimento cardíaco é um inteiro entre 0 e 200[1]). O caso em que $n = 0$ representa o final do arquivo e não deve ser processado.

Saída

Você deverá imprimir um cabeçalho indicando o número da instância que está tratando (`Instancia #i`) e na linha seguinte o número do ator/atriz que está com maior risco de sofrer um infarto. O risco para o infarto cresce com o número de observações consecutivas em que o número de batimentos cresceu. Caso haja empate neste critério, devolva o ator que apresenta a maior diferença entre a primeira e última medição do intervalo em que os batimentos foram aumentando. Se persistir o empate, o risco será maior para aquele com o maior batimento no fim do intervalo. Se ainda não houver decisão, retorne o de menor índice.

[1] Claro que com 0 ou 200 o cara já morreu :-))

Exemplo

Entrada

```
4 10
67 78 87 66 78 87 89 66 67 66
77 79 99 98 98 98 97 78 78 89
66 67 68 69 70 71 72 87 88 66
75 77 90 95 94 97 99 66 88 99
4 7
120 135 167 165 188 170 150
98 76 60 78 108 100 110
90 95 138 135 133 130 100
70 68 74 67 75 67 100
0
```

Saída

```
Instancia #1
3

Instancia #2
3
```

Problema D: Cruzamentos no ar

```
Arquivo:  cruza.[c|cc|pas|java]
Entrada:  cruza.in
```

Atores e atrizes de Hollywood são famosos por suas constantes viagens. Assim, a região de Los Angeles tem diversas pistas de decolagem espalhadas por muitas das propriedades dos atores, diretores e executivos da indústria americana do cinema. Como a maior parte destes atores e atrizes têm jatinhos que voam basicamente na mesma altitude os riscos de colisão no ar são muito grandes. Como a preocupação com a segurança das estrelas é uma de suas prioridades, a Academia de Artes e Ciências de Hollywood resolveu fazer um programa que permite que se antecipem possíveis cruzamentos entre as rotas dos artistas para que seus assessores acertem as agendas dos mesmos evitando assim qualquer risco de colisão. Sua tarefa neste problema será implementar o programa para a Academia.

Para que fique claro o que é um cruzamento, as rotas poderão partir e chegar na mesma pista de vôos, pois as torres de controle das pistas controlam esta parte. O problema é quando o cruzamento ocorre em um ponto que não é extremo da rota de nenhum dos dois aviões.

Entrada

São dadas várias instâncias. A primeira linha de cada instância contém o número $0 \leq n \leq 1000$ de vôos da instância. A seguir, em cada uma das próximas n linhas são dadas as coordenadas (cartesianas) da pista de decolagem e aterrisagem do vôo. As coordenadas são dois inteiros entre 0 e 1000. O caso em que $n = 0$ representa o final do arquivo e não deve ser processado.

Saída

Você deverá imprimir um cabeçalho indicando o número da instância que está tratando (`Instancia #i`) e na linha seguinte a mensagem se há ou não cruzamentos entre as rotas.

Exemplo

Entrada

```
4
1 2 1 4
4 5 4 13
1 4 4 1
4 14 8 14
3
1 1 2 2
1 2 2 1
1 2 2 2
0
```

Saída

```
Instancia #1
Nao ha cruzamentos.

Instancia #2
Ha cruzamentos.
```

Problema E: Agentes Secretos

`Arquivo:  agentes.[c|cc|pas|java]`
`Entrada:  agentes.in`

Todas as pessoas que já assistiram a filmes ou seriados de espionagem como *007*, *Missão Impossível* ou *Hawai 5-0*, sabem que alguns países do mundo mantém grupos de agentes secretos infiltrados em governos e organizações do Oriente Médio, América do Sul e Leste Europeu.

Um dado serviço de inteligência possui n agentes espalhados em um país não muito amigável. Cada agente conhece alguns outros agentes e tem procedimentos específicos para arranjar um encontro secreto com cada um deles. Normalmente, são trocadas mensagens codificadas para marcar tais encontros. Dados dois agentes que se conhecem i e j, existe uma certa probabilidade p_{ij} de que uma mensagem trocada entre eles seja interceptada por pessoas hostis.

De tempos em tempos, o líder do serviço de inteligência precisa difundir informações confidenciais a todos seus agentes em campo. Para tanto, ele utiliza-se do mecanismo de troca de mensagens dos agentes, isto é, ele contacta alguns dos agentes que conhece e estes se encarregam de propagar as informações de modo que a probabilidade de interceptação $\mathcal{P}$ seja mínima. Como você pode perceber, o serviço é tão secreto que nem o líder conhece todos os agentes subordinados a ele. Sua tarefa neste problema é construir um programa que calcule $\mathcal{P}$.

Entrada

Seu programa deverá estar preparado para trabalhar sobre diversos cenários, isto é, diversas difusões de informações confidenciais em diversos países. Cada cenário é descrito da forma que segue. Na primeira linha são especificados o número de agentes no país, $0 < n \leq 100$, incluindo o líder do serviço de inteligência, e o número de pares de agentes que estão no país e que se conhecem, $0 \leq m \leq 4950$. Nas m linhas seguintes, existem dois inteiros i, j e um racional p_{ij}, com $1 \leq i, j \leq n$ e $0 \leq p_{ij} \leq 1$. Cada linha significa que os agentes i e j se conhecem e que uma mensagem trocada entre eles é interceptada com probabilidade p_{ij}. Uma valor igual a zero para n indica o fim dos cenários. Você pode supor que sempre será possível difundir as informações confidenciais entre todos os agentes.

Saída

Para cada cenário da entrada, seu programa deve imprimir o texto `Cenario x, probabilidade de interceptacao = P`, onde `x` é a posição do respectivo cenário no arquivo de entrada (numerado a partir de 1) e `P` a probabilidade da informação a ser difundida ser interceptada. Tal probabilidade deve ser impressa com três casas decimais. Você deve deixar uma linha em branco entre cada cenário.

Exemplo

Entrada

```
4 6
1 2 0.3
1 3 0.1
1 4 0.5
2 3 0.9
2 4 0.1
3 4 0.5
3 2
1 2 1.0
2 3 1.0
0 0
```

Saída

```
Cenario 1, probabilidade de interceptacao = 0.433

Cenario 2, probabilidade de interceptacao = 1.000
```

Problema F: Contando os Segundos

```
Arquivo:  secs.[c|cc|pas|java]
Entrada:  secs.in
```

Através da pesquisa e do desenvolvimento espetacular de técnicas revolucionárias de computação, estatística e intuição, um grupo de mulheres da Universidade de Torrinha desenvolveram um software capaz de prever o futuro. Na verdade, o programa não consegue descrever exatamente o que vai acontecer, mas através da análise da resposta a uma série de perguntas sobre a pessoa, ele consegue identificar quando acontecerão os próximos cinco eventos mais importantes na vida dela. O programa apresenta dois pequenos problemas. O primeiro é que a pessoa sobre a qual se quer adivinhar o futuro precisa passar centenas de horas ininterruptas na frente do computador, respondendo às perguntas feitas pelo software, para que ele consiga fazer a previsão. O segundo é que as previsões são feitas em número de segundos, a contar do término do processamento da previsão. As moças de Torrinha estão empenhadas na solução do primeiro problema, e pediram aos concorrentes da Maratona de Programação uma ajuda para resolver o segundo.

Entrada

A entrada consiste de várias previsões, referentes a diferentes pessoas. A primeira linha de cada previsão contém o momento exato em que os 5 números (que correspondem aos 5 momentos importantes) foram impressos pelo programa. Este momento segue o formato:

<dia da semana>,<dia><mês><ano>:<hora>:<minuto>:<segundo>

onde <dia da semana> é a abreviação do dia da semana correspondente, composto pela seqüência de 3 caracteres maiúsculos correspondentes aos dias da semana de domingo a sábado, respectivamente, tais sejam: DOM, SEG, TER, QUA, QUI, SEX ou SAB. <dia> é o dia do mês, escrito com 2 dígitos, zero à esquerda se for o caso. <mês> é a abreviação do mês correspondente, composto pela seqüência de 3 caracteres maiúsculos correspondentes aos meses de Janeiro a Dezembro, respectivamente, tais sejam: JAN, FEV, MAR, ABR, MAI, JUN, JUL, AGO, SET, OUT, NOV ou DEZ. <ano> é o ano, escrito com 4 dígitos. <hora>, <minuto> e <segundo> escrito com 2 dígitos, zero à esquerda se for o caso.

Nas linhas seguintes vêm os 5 momentos previstos, um por linha, 5 linhas por previsão. Cada momento consiste no número d de segundos ($0 < d < 2.000.000.000$) a contar do momento do processamento. Após o último caso de teste, uma linha iniciada por '`FIM`' indica o final do arquivo de entrada.

Você pode assumir que o programa foi desenvolvido em 01 de Março de 2002, de modo que nenhuma data de previsão será anterior a esta. Outro fato importante é que as moças de Torrinha só vão usar o programa até às 23:59:59 de 31/12/2099 quando uma delas acha que vai morrer, pois, isso foi previsto por outro programa elaborado pelo departamento de intuição feminina da Unitor (a Universidade de Torrinha).

Saída

Seu programa deverá identificar na saída cada previsão por um número seqüencial na primeira linha, e nas cinco seguintes deverá escrever os cinco horários completos referentes à previsão efetuada, no mesmo formato utilizado na entrada para o horário de processamento da previsão. Você deve também pular uma linha ao final de cada previsão.

Exemplo

Entrada

```
DOM,18AGO2002:12:00:00
1
2
60
3600
86400
QUI,28FEV2002:23:59:59
1
2
3
4
5
FIM
```

Saída

```
Previsao #1
DOM,18AGO2002:12:00:01
DOM,18AGO2002:12:00:02
DOM,18AGO2002:12:01:00
DOM,18AGO2002:13:00:00
SEG,19AGO2002:12:00:00

Previsao #2
SEX,01MAR2002:00:00:00
SEX,01MAR2002:00:00:01
SEX,01MAR2002:00:00:02
SEX,01MAR2002:00:00:03
SEX,01MAR2002:00:00:04
```

Problema G: Beverly Hills, Century City

`Arquivo: hills.[c|cc|pas|java]`
`Entrada: hills.in`

Beverly Hills e Century City figuram entre os condados mais abastados e sofisticados da grande Los Angeles. Seus habitantes, pessoas de gostos pouco convencionais e atitudes demasiadamente excêntricas, estão sempre procurando novas diversões, novas quinquilharias e novos hobbies para empregar seus dividendos. A grande mania da região, no momento, é a criogenia. Inspirados pelo enredo do filme *Vanilla Sky*, vários cidadãos e cidadãs têm recorrido à empresa local A.C.M. – Agência Criogênica keith-Morris – com o intuito de "prolongar" sua existência.

Tal movimentação deixou a A.C.M. com um problema a ser resolvido: construir um plano de produção de suas células criogênicas num dado período de tempo. De maneira mais precisa a empresa deseja encontrar um plano de produção de custo mínimo num período de n dias, sujeito a:

- Custos de produção fixos não negativos $\{f_t\}_{t=1}^n$;
- Custos unitários de produção $\{p_t\}_{t=1}^n$;
- Custos unitários de estocagem $\{h_t\}_{t=1}^n$;
- Demandas não negativas $\{d_t\}_{t=1}^n$.

Sua tarefa é construir um programa para ajudar a A.C.M. a resolver seu problema.

Entrada

A entrada é constituída por várias instâncias, armazenadas em seqüência ao longo do arquivo de entrada. Cada instância possui a estrutura que segue. Na primeira linha é especificado um valor inteiro $0 \leq n \leq 180$ que indica o número de dias do planejamento. O caso em que $n = 0$ representa o final do arquivo e não deve ser processado.

Nas próximas n linhas são especificados, na ordem que segue, os inteiros $f_t \geq 0$, p_t, h_t e $d_t \geq 0$, em que $1 \leq t \leq n$.

Saída

Para cada pesquisa processada, seu programa deve imprimir na primeira linha o texto `Instancia #x`, onde x é o número da instância no arquivo de entrada. (As instâncias são numeradas seqüencialmente a partir de um.) Na segunda linha deve ser impresso o custo mínimo do período de n dias. Nesta fase de seu trabalho, não é necessário imprimir o plano de produção. Uma linha em branco deve ser deixada entre cada pesquisa.

Exemplo

Entrada

```
4
12 3 1 2
20 3 2 4
16 3 1 5
 8 3 1 1
0
```

Saída

```
Instancia #1
69
```