Aquecimento para

Maratona de Programação de 2008

Universidade de São Paulo

Caderno de Problemas

Departamento de Ciência da Computação IME-USP

Sábado, 14 de junho de 2008.

Problema A: Estacionamento maluco

`Arquivo: estacionamento.[c|cpp|java]`

O mais novo shopping center de São Paulo resolveu inovar. Ao invés de ter preços fixos para cada hora em que os carros ficam parados em seu estacionamento (ou, no máximo um preço diferente para a primeira hora e outro para as demais), eles resolveram fazer preços quase que completamente aleatórios para as horas em que os carros ficam estacionados. Inclusive, os preços podem ser negativos (o sortudo que ficar estacionado naquela hora, ao invés de pagar ganhará dinheiro)!!

O estacionamento funciona da seguinte forma. Ao entrar no estacionamento o cliente recebe um cartão com um número que corresponde ao número de horas cheias que passaram desde a inauguração do shopping. Ao sair, a máquina calcula novamente o número de horas, bem como o preço a ser pago. Por exemplo, se o carro ficou estacionado do período 1345 até o período 1346, o cliente pagará pelo preço dessas duas horas.

O preço das horas é uma tabela gerada anteriormente, com base em estudos estatísticos complicadíssimos. A tabela será dada a partir da primeira hora desde a inauguração do shopping. Sua tarefa é, dada a tabela de preços das horas e as diversas consultas (hora da entrada e hora da saída dos carros), gerar o preço total a ser pago (ou recebido) por cada um dos clientes do shopping.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro n ($2 \leq n \leq 100$) indicando a quantidade de horas da tabela. Na linha seguinte temos n inteiros, com valores entre -100 e 100, indicando o preço do estacionamento em cada hora. A próxima linha contém um inteiro m ($1 \leq m \leq 100$) indicando o número de consultas, e as m linhas seguintes possuem dois inteiros a e b ($1 \leq a, b \leq n$ e $a \leq b$) que indicam respectivamente a hora de entrada e hora de saída.

Saída

Para cada instância imprima m linhas com os resultados das respectivas consultas.

Após cada instância imprima uma linha em branco.

Exemplo de entrada

```
2
4
2 3 -1 2
2
1 1
1 3
3
1 -1 -1
2
1 2
1 3
```

Exemplo de saída

```
2
4

0
-1
```

Problema B: Sistema criptográfico

`Arquivo: cripto.[c|cpp|java]`

O RSA é um sistema criptográfico desenvolvido por três professores do MIT: Ron Rivest, Adi Shamir e Len Adleman. Das iniciais dos nomes dos três autores vem o nome do sistema criptográfico. Este é um dos sistemas de chave pública mais seguros que existem, e possibilitou assinatura digital. O funcionamento do sistema é baseado em operações básicas de teoria dos números, como multiplicação e potenciação.

Sua tarefa nesse exercício será realizar de forma eficiente a seguinte conta com números inteiros: dados inteiros positivos n, a, b e c, determinar o valor de $(a + b)^n \bmod c$.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

Cada instância é composta por uma linha contendo os inteiros n ($0 \leq n \leq 1000000000$), a, b ($0 \leq a, b \leq 1000$) e c ($1 \leq c \leq 1000$).

Saída

Para cada instância imprima uma linha contendo o resultado da conta.

Exemplo de entrada

```
3
3 8 9 4
5 3 3 4
0 2 7 5
```

Exemplo de saída

```
1
0
1
```

Problema C: Amigo secreto

`Arquivo: amigo.[c|cpp|java]`

Todo fim de ano é a mesma coisa. Amigos, colegas de trabalho, companheiros de pelada, todos se juntam e fazem a mesma brincadeira do amigo secreto. Nessa brincadeira cada um dos participantes sorteia um amigo e deve dar a ele um presente. Em alguns casos, o sorteio pode ser repetido até três vezes se você não gostar do seu amigo secreto. A brincadeira fica um pouco sem graça quando ocorrem pequenos grupos de pessoas (pares, no pior caso) que formam um pequeno ciclo de presentes: *chegado* presenteia *grilex* que presenteia *mau-mau* que presenteia *chegado*. O ideal é que quando esses ciclos ocorrerem (e eles devem ocorrer!), que eles sejam o mais longo possível.

Sua tarefa neste problema é, dada uma lista de sorteios de uma brincadeira de amigo secreto, determinar o tamanho do menor ciclo.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro n ($2 \leq n \leq 1000$) indicando a quantidade de participantes que são identificados por inteiros de 1 a n. Cada uma das próximas n linhas possuem dois inteiros a e b ($1 \leq a, b \leq n$) indicando que a pessoa a sorteou a pessoa b. Cada participante é sorteado exatamente uma vez e sorteia exatamente uma vez. A pessoa pode se sortear.

Saída

Para cada instância imprima uma linha contendo o tamanho do menor ciclo na lista de sorteios.

Exemplo de entrada

```
3
2
1 2
2 1
4
1 2
2 3
3 1
4 4
3
1 2
2 3
3 1
```

Exemplo de saída

```
2
1
3
```

Problema D: Chuva de dinheiro

`Arquivo: chuva.[c|cpp|java]`

O famoso apresentador de televisão Wilvio Wantos criou no seu eterno programa dominical o famoso quadro "Quem quer dinheiro!!!". Inspirado nessa obra-prima da televisão nacional, o técnico Mau-Mau (fanático por dinheiro) criou uma brincadeira entre seus comandados, a que ele deu o nome de "chuva de dinheiro". Inicialmente, Mau-Mau percorre as esquinas da cidade, pendurando no cruzamento uma pequena lata, dentro da qual ele esconde ou não uma nota de 50 mangos. Mau-Mau distribuiu uma certa quantia na cidade, conhecida por dois atletas (seus comandados).

Depois disso, dois de seus atletas, munidos com uma lista de pontos onde Mau-Mau colocou dinheiro, são levados a pontos diferentes da cidade de onde partem tentando encontrar a maior quantidade possível de dinheiro (brincadeirinha bem capitalista...). Cada atleta se move para o cruzamento mais próximo que contém uma nota (eles têm walkie-talkies para informar os pontos onde cada um retirou uma nota, assim o outro atleta não precisa ir até a mesma esquina) e se os atletas chegarem à mesma esquina no mesmo instante, o mais velho tem prioridade. Se um atleta estiver caminhando para pegar uma nota e o outro atleta consegue pegar a nota antes dele, então o segundo atleta avisa pelo walkie-talkie e o primeiro atleta pára imediatamente onde está e escolhe outra nota para pegar. Se um atleta tiver duas ou mais notas para escolher, ele opta pela mais próxima. Em caso de empate seleciona a com menor coordenada x, e se ainda tiver empate a nota escolhida é com menor y. A cidade é planejada e forma uma grade.

Além disso, o tirano técnico Mau-Mau, determinou que os atletas, ao escolherem a próxima esquina, andam primeiro na horizontal e depois na vertical. O tempo gasto para percorrer do ponto (x_a, y_a) até o ponto (x_b, y_b) é $|x_a - x_b| + |y_a - y_b|$. A cada unidade de tempo os atletas percorrem uma unidade ou na horizontal, ou na vertical. No final da brincadeira, após acabar o dinheiro, vence aquele que conseguiu arrecadar mais durante o percurso. O dinheiro é obviamente devolvido a Mau-Mau (o cara adora dinheiro...) e o perdedor é obrigado a fazer flexões como castigo.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro n ($1 \leq n \leq 1000$) indicando as notas distribuídas. A linha seguinte contém os inteiros x_1, y_1, x_2 e y_2 ($0 \leq x_1, y_1, x_2, y_2 \leq 1000000$) onde (x_1, y_1) indica a localização do atleta mais velho, e (x_2, y_2) indica a localização do atleta mais novo. Cada uma das n linhas seguintes possui dois inteiros x, y ($0 \leq x, y \leq 1000000$) que indicam a localização de uma nota.

Saída

Para cada instância imprima uma linha contendo o número de notas que o atleta mais velho pegou e o número de notas que o atleta mais novo pegou.

Exemplo de entrada

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

Exemplo de saída

```
2 3
3 0
```

Problema E: Primos distantes

`Arquivo: primosdist.[c|cpp|java]`

Juquinha é tido em sua turma na escola como um perfeito *nerd*. Ele diz que não é, mas mesmo assim não escapa das brincadeiras dos colegas. No último domingo, no meio da macarronada, a avó de Juquinha, Nona Lola, contou uma história de um primo distante na Itália. Bastou a referência a primos distantes para que a mente de Juquinha disparasse, e ele imediatamente parasse de comer o macarrão. Ele começou a imaginar quais eram os primos mais distantes que ele conhecia, mas ele estava pensando em **números primos**, é claro. Depois de algum tempo Juquinha começou a conjecturar que para qualquer inteiro positivo n existem n números não primos consecutivos, ou seja, os dois primos "vizinhos" estão a uma distância de pelo menos n. Incapaz de provar a conjectura, Juquinha está recorrendo à ajuda de vocês. É importante salientar que a Nona Lola está desesperada com o menino, que não comeu nada desde domingo, e só fica repetindo "primos distantes..."

A sua tarefa é dado um inteiro positivo n, imprimir uma seqüência de inteiros consecutivos não primos.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

Cada instância é composta por uma linha contendo um inteiro n ($1 \leq n \leq 153$).

Saída

Para cada instância imprima uma linha contendo a seqüência de inteiros consecutivos não primos cujo primeiro elemento é o menor possível.

Exemplo de entrada

```
2
1
10
```

Exemplo de saída

```
4
114 115 116 117 118 119 120 121 122 123
```

Problema F: Não pára, não pára, não pára ...

`Arquivo: vaipracimatimao.[c|cpp|java]`

Na véspera da decisão com o Sport o técnico do Timão, o Mano, resolveu que precisava se precaver contra a espionagem dos adversários da Ilha do Retiro. Ele resolveu que até mesmo as instruções por escrito que ele dava deveriam ser codificadas antes da transmissão, de forma que, mesmo que um espião do Leão tivesse acesso ao texto, dificilmente conseguiria entender o esquema tático do Timão.

O esquema escolhido pelo Mano foi, num primeiro passo, completar a mensagem a ser enviada até que o tamanho da mesma fosse um quadrado perfeito. Para completar a mensagem ele escrevia no final: "NAO PARA, NAO PARA, NAO PARA...". Feito isso, a mensagem era escrita como em uma matriz $n \times n$, e passava por uma série de transformações. Essas transformações tinham o objetivo de tornar a mensagem ilegível.

Considere a seguinte matriz A com dimensões 5×5:

$$\begin{array}{|c|c|c|c|c|} \hline N & A & O & & P \\ \hline A & R & A & & N \\ \hline A & O & & P & A \\ \hline R & A & & N & A \\ \hline O & & P & A & R \\ \hline \end{array}$$

As transformações por que passava a matriz eram de três tipos:

- **RD**: a matriz é rotacionada para a direita. Assim, por exemplo, após a operação **RD** na matriz A temos a seguinte matriz:

$$\begin{array}{|c|c|c|c|c|} \hline O & R & A & A & N \\ \hline & A & O & R & A \\ \hline P & & & A & O \\ \hline A & N & P & & \\ \hline R & A & A & N & P \\ \hline \end{array}$$

- **EH**: a matriz é espelhada na horizontal. Por exemplo, se aplicarmos a operação **EH** na matriz A temos a seguinte matriz:

$$\begin{array}{|c|c|c|c|c|} \hline P & & O & A & N \\ \hline N & & A & R & A \\ \hline A & P & & O & A \\ \hline A & N & & A & R \\ \hline R & A & P & & O \\ \hline \end{array}$$

- **EV**: a matriz é espelhada na vertical. Assim, por exemplo, após a operação **EV** na matriz A temos a seguinte matriz:

$$\begin{array}{|c|c|c|c|c|} \hline O & & P & A & R \\ \hline R & A & & N & A \\ \hline A & O & & P & A \\ \hline A & R & A & & N \\ \hline N & A & O & & P \\ \hline \end{array}$$

Um bloco é uma seqüência de operações iguais de tamanho máximo. O técnico do Timão determinou que não haverá mais do que 10 blocos de transformações.

Sua tarefa neste problema é ajudar o técnico do Corinthians. Você deve fazer um programa que lê uma mensagem e uma seqüência de operações descritas acima e imprime a mensagem codificada.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém um inteiro n ($1 \leq n \leq 1000$) indicando o tamanho da matriz. Cada uma das n linhas seguintes possui n caracteres diferentes de espaço. A próxima linha contém um inteiro m ($1 \leq m \leq 10000$) indicando o número de operações que serão executadas. As m operações são dadas nas m linhas seguintes.

Saída

Para cada instância imprima a matriz após aplicar todas as operações.

Após cada instância imprima uma linha em branco.

Exemplo de entrada

```
2
3
AB+
D_F
*HI
4
RD
EV
RD
EH
8
NAO.PARA
NAO.PARA
NAO.PARA
VAI.....
PRA.....
CIMA....
TIMAO!!!
NAO.PARA
41
RD
EV
EV
EV
RD
RD
RD
RD
RD
EH
EH
EH
RD
EV
EV
EV
RD
RD
```

```
RD
RD
RD
EH
EH
EH
EV
EV
EV
EH
EH
EH
EH
EH
EH
EH
EH
EH
EH
EH
EH
EH
EH
```

Exemplo de saída

```
IH*
F_D
+BA

NAO.PARA
NAO.PARA
NAO.PARA
VAI.....
PRA.....
CIMA....
TIMAO!!!
NAO.PARA
```

Problema G: Trânsito

`Arquivo: transito.[c|cpp|java]`

O candidato a prefeito de São Paulo, Quessab Suplício Aidemim teve uma idéia fantástica para resolver os problemas de trânsito que enfrentamos todos os dias. A idéia se dá em duas fases. Na primeira fase, todas as ruas da cidade são transformadas em mão única. A fim de evitar problemas com pessoas que se perdem na cidade e ficam circulando, ele contratou uma equipe de engenheiros de tráfico que eliminaram, numa segunda fase, todas as possibilidades de alguém circular nas ruas. Assim, será impossível partir de um ponto qualquer da cidade e retornar a ele respeitando as mãos das ruas.

O candidato Aidemim achou a idéia fantástica, pois permitiria que o trânsito fluísse livremente na rua, afinal todos iam na mesma direção! Ele já estava estourando rojões quando um assessor o lembrou que isso tornaria impossível para as pessoas voltar para casa no fim do dia de trabalho. O Quessab não se deu por vencido. Teve uma nova idéia: todas as ruas teriam suas mãos invertidas exatamente às 17:00 e às 5:00 horas. Ou seja, das 5:00 às 16:59 você poderia andar na rua em uma direção, e das 17:00 às 4:59 você só poderia andar na mesma rua na direção contrária.

O plano do candidato causou um certo tumulto entre seus assessores, mas nosso querido Suplício garantiu que o plano era perfeito, desde que eles tivessem certeza de que as pessoas pudessem descobrir de forma eficiente qual seria o melhor caminho (cuja soma dos comprimentos das ruas é mínimo) entre dois pontos quaisquer da cidade em uma determinada hora. Sua tarefa neste problema é resolver o problema proposto pelo candidato, e manter vivas as esperanças de elegê-lo nas próximas eleições.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém quatro inteiros n, m, p e q ($1 \leq n \leq 20000$, $1 \leq m \leq 2000000$ e $1 \leq p, q \leq n$) que indicam respectivamente o número de pontos na cidade, o número de ruas e dois pontos da cidades. Os pontos são identificados por números de 1 a n. Cada uma das m linhas seguintes possui três inteiros a, b e c ($1 \leq a, b \leq n$ e $1 \leq c \leq 1000000$) que indicam a existência de uma rua que vai de a a b, no horário das 5:00 às 16:59, e possui comprimento c.

Saída

Para cada instância imprima uma linha contendo o comprimento do melhor caminho que vai de p a q no horário das 5:00 às 16:59. Se não existir caminho entre p e q, então imprima -1.

Exemplo de entrada

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

Exemplo de saída

```
2
-1
```

Problema H: Vikings

`Arquivo: vikings.[c|cpp|java]`

Os vikings foram habitantes da Escandinávia entre os séculos VIII e XI. Eram conhecidos como comerciantes, guerreiros e piratas. Eles invadiram o norte da Europa e as ilhas britânicas levando horror às civilizações locais. Além de serem conhecidos como guerreiros e destruidores, hoje sabe-se que os vikings deram importantes contribuições à tecnologia de navegação e de construção de cidades.

A sociedade viking era organizada da seguinte forma: havia o rei no topo da pirâmide. Abaixo dele estavam os proprietários de terra, conhecidos como *jarls*. Abaixo deles os *karls*, o povo livre, sem grandes propriedades, em geral comerciantes ou lavradores. Eles formavam também o grosso do exército. Abaixo dos *karls* havia os *thralls*, os escravos. Os *røodes* eram os ruivos, que eram considerados descendentes diretos dos deuses. Os *berseks* (não ruivos) eram muitas vezes considerados amaldiçoados, e sua presença nas aldeias indesejada.

A língua falada pelos vinkings era a mesma nos vários clãs, e a escrita era feita no alfabeto rúnico. Após a introdução do cristianismo nos domínios vikings houve o declínio do império, que culminou com o desaparecimento quase total da cultura a partir do século XII.

Há registros demográficos bastante precisos, ainda que confusos, a respeito das cidades vikings. Apenas os *jarls* e *karls* eram contados (não há registros de escravos). Tais registros contabilizavam algumas informações, tais como, o número de jarls, o número de røode, o número de karls que eram berseks e o número de jarls que eram røode. Sua tarefa nesse problema é, de posse desses registros, descobrir o número total de habitantes da aldeia (jarls e karls), e também descobrir quantos karls que eram røode e quantos jarls eram berseks.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

Cada instância é composta por uma linha contendo a, b, c e d ($1 \leq a, b, c, d \leq 100, d \leq a$ e $d \leq b$) onde a é o número de jarls, b é o número de røodes, c é o número de karls que são berseks e d o número de jarls que são røodes.

Saída

Para cada instância imprima uma linha contendo o número total de habitantes, o número de karls que eram røode e o número de jarls que eram berseks.

Exemplo de entrada

```
1
42 24 13 9
```

Exemplo de saída

```
70 15 33
```

Problema I: Reunião de Diretoria

`Arquivo: reuniao.[c|cpp|java]`

Uma grande empresa multinacional resolveu fazer sua reunião anual da diretoria no Brasil, e para isso reservou o centro de convenções de um importante Resort no litoral do Nordeste. Esse centro de convenções é dotado de k salas. O presidente, o grandioso Carlos Justos, já decidiu que ficará no salão nobre e ali fará suas reuniões. O salão nobre é central no centro de convenções e tem uma porta para cada sala de reunião, e cada sala de reunião tem apenas uma porta que a liga com o salão nobre. Na empresa de Carlos Justos, cada funcionário possui apenas um chefe, e pode também ser chefe de no máximo um funcionário. O único membro da empresa que poder ter mais do que um subordinado é o grandioso Carlos Justos, é claro! Mesmo assim se Carlos é chefe de um funcionário então este funcionário é subordinado apenas dele.

Uma característica interessante da empresa comandada por Carlos é que existe comunicação entre todos os funcionários. A comunicação é feita passando mensagens de um subordinado a seu chefe, ou de um chefe para seu subordinado. Assim, por exemplo, se o funcionário Grilex é chefe do Chegado que é chefe do Mau-Mau, então para o saudoso Grilex enviar uma mensagem para Mau-Mau ele tem que passar a mensagem para Chegado que depois envia para Mau-Mau. Este sistema de comunicação é eficaz no sentido de evitar fofocas.

Antes da reunião os funcionários visitaram as salas e preencheram uma ficha que lhes perguntava o quanto eles odiaram cada sala. A pontuação para cada sala varia entre 0 (o funcionário adorou a sala) e 1000 (o funcionário odiou a sala). O gigante do mundo dos negócios, Carlos Justos, deseja satisfazer ao máximo seus funcionários, ou seja, ele quer minimizar o ódio que os funcionários vão sentir ao serem designados às salas, e pediu que você determinasse em qual sala cada funcionário vai ficar respeitando uma regra: dois funcionários de uma mesma sala devem poder se comunicar passando mensagens apenas por funcionários daquela sala. Além disso, cada funcionário deve ser capaz de mandar uma mensagem para o presidente sem ferir o esquema de comunicação. Afinal de contas a comunicação é tudo! O custo de uma configuração é a soma da pontuação que cada funcionário deu para sala onde ele está na configuração.

Não esqueça que Carlos Justos sempre fica no salão nobre.

Entrada

A entrada é composta de diversas instâncias. A primeira linha da entrada contém um inteiro T indicando o número de instâncias.

A primeira linha de cada instância contém dois inteiros n e m ($1 \leq n, m \leq 100$) que indicam respectivamente o número de funcionários e o número de salas. Os funcionários são identificados por números de 1 a n, e as salas são identificadas por números de 1 a m. Carlos Justos é sempre identificado por 1 e o salão nobre é sempre identificado por 1. Cada uma das $n-1$ linhas seguintes possui dois inteiros a e b ($1 \leq a, b \leq n$ e $b \neq 1$) indicando respectivamente que o funcionário a é chefe do funcionário b. Cada uma das próximas $n-1$ linhas possui $m+1$ inteiros $a, c_1, c_2, .., c_m$ ($1 \leq a \leq n, a \neq 1$ e $0 \leq c_1, c_2, .., c_m \leq 1000$), onde a é um funcionário e c_i, para $1 \leq i \leq m$, é pontuação que o funcionário a deu para sala i.

Saída

Para cada instância imprima o custo da configuração de custo mínimo que respeita as regras descritas no problema.

Exemplo de entrada

```
1
7 3
1 2
1 4
1 5
2 3
5 6
6 7
2 100 50 0
3 100 50 0
4 30 0 0
5 30 0 50
6 200 200 50
7 200 200 50
```

Exemplo de saída

```
260
```