ToPAS'14
Torneio de Programação para Alunos do Secundário

Departamento de Ciência de Computadores

http://www.dcc.fc.up.pt/topas/

Conjunto de Problemas

U.PORTO

Faculdade de Ciências da Universidade do Porto

9 de maio de 2014

Este conjunto de problemas deverá conter sete (7) problemas e dezassete (17) páginas.
Se faltar algum problema, por favor avise a organização.

Edição realizada em colaboração com o Departamento de Engenharia Electrónica e Informática, Faculdade de Ciências e Tecnologia, Universidade do Algarve.

ToPAS'14
Torneio de Programação para Alunos do Secundário

Dep. Ciência de Computadores – FCUP
9 de Maio de 2014

Conteúdo

Problema A

Quantos dias fazes?

Todos sabemos quantos anos temos mas raramente nos damos ao trabalho de saber quantos dias temos exatamente. E isso é importante porque há anos que não só parecem custar mais a passar como realmente têm mesmo mais dias, como os anos bissextos.

Na verdade há o conceito de data juliana, ou dia juliano, que conta o número de dias seguidos a partir de uma data arbitraria. A data juliana é usada sobretudo para referenciar observações astronómicas, mas simplifica também o cálculo do número de dias entre datas. Por exemplo, para calcular o número de dias passados desde o nosso nascimento até hoje basta subtrair as respetivas datas julianas.

Tarefa

Calcular a data juliana para datas da nossa era. São dados o ano, mês e dia como inteiros (janeiro = 1). A data juliana é calculada usando o algoritmo seguinte.

Se o mês for anterior a março (mês 3) então

ano := ano -1
mes := mes + 12

A partir destes valores calcula-se

A := ano / 100 (divisão inteira)
B := A / 4 (divisão inteira)

Se a data for posterior a 15/10/1582, quando teve início o Calendário Gregoriano que usamos atualmente, então

C := 2 - A + B

No calendário Gregoriano foram introduzidos os anos bissextos para compensar o facto de um ano não ter um número inteiro de dias, evitando o desajuste progressivo entre o calendário e as observações astronómicas.

Se for uma data igual ou anterior a 4/10/1582, o fim do calendário juliano, então

C := 0

No calendário juliano, anteriormente em vigor, os anos tinham sempre 365 dias. De notar que em Portugal ao dia 4/10/1582 seguiu-se o dia 15/10/1582, para reajustar o calendário.

A partir destes valores calcula-se

D := parte inteira de [365,25 x (ano + 4716)]
E := parte inteira de [30,6001 x (mes + 1)]

sendo que a data juliana (DJ) é dada por

DJ := D + E + dia + C - 1524

Input

Uma linha com três inteiros (ano, mês e dia) que definem uma data válida. Podemos assumir que não são fornecidas datas inválidas, como as datas entre 4/10/1582 e 15/10/1582. Também não serão fornecidas datas anteriores à nossa era.

Output

A data juliana referente à data lida.

Exemplo

Input

```
2014 5 9
```

Output

```
2456787
```

Problema B

Caixa Negra

Nas buscas pelo avião da Malaysia Airlines que desapareceu no oceano índico, alguns navios oceanográficos detetaram sinais que parecem estar a ser emitidos pela caixa negra do avião. Esses sinais são muito fracos, trazem muito ruído e por isso têm de ser analisados com muito cuidado. Os sinais chegam-nos na forma de uma sequência de números, cada um deles representando a intensidade medida, dez vezes por segundo, ao longo do tempo. Muitos dessas intensidades são muito baixas, o que quer dizer que se trata de ruído oceânico, sem interesse, mas periodicamente há valores que se destacam e que só podem ter tido origem num dispositivo eletrónico. Eis um exemplo de um troço de observações:

2 2 2 3 4 5 15 4 3 4 5 2 2 10 4 2 2 6 13 6 7 3 4 18 6

Os números assinalados destacam-se claramente dos seus vizinhos e, por hipótese, são provenientes da caixa negra do avião. As equipas de busca precisam de um programa para contar os sinais que vêm da caixa negra do avião, na sequência de números em análise. Consideramos que um sinal foi emitido pela caixa negra quando a sua intensidade é maior que o dobro da intensidade de cada um dos dois sinais vizinhos, antes e depois.

Tarefa

Escrever um programa que, dada uma sequência de números inteiros positivos conte o número de números que são simultaneamente maiores que o dobro do número imediatamente anterior na sequência e maiores que o dobro do número imediatamente a seguir na sequência.

Input

A primeira linha contém o número de números da sequência (sempre maior ou igual a 3 e menor ou igual a 1000). Seguem-se os números da sequência, um em cada linha.

Output

O output terá uma única linha, com o número de sinais detetados.

Exemplo 1

Input

```
22
2
2
3
4
5
15
4
3
4
5
2
10
4
2
6
13
6
7
3
4
18
6
```

Output

```
4
```

Exemplo 2

Input

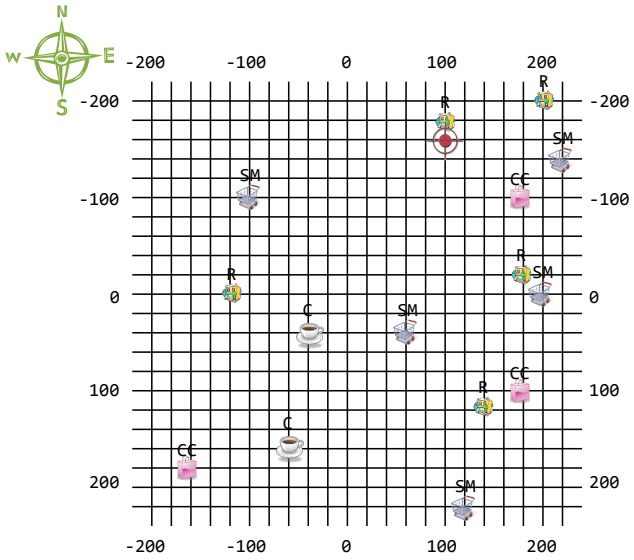
```
5
3
6
4
8
4
```

Output

```
0
```

Problema C

iFind – a App para encontrares o que queres



Para se encontrar a loja de um certo tipo que se encontra o mais perto possível de onde estamos podemos recorrer a uma *app* para o nosso telemóvel. Foi isso que pensaram o Manuel e os seus colegas do curso. Assim resolverem desenvolver uma aplicação dessas para telemóvel esperto (vulgo *smart-phone*). Como são organizados decidiram dividir o programa em diversos módulos, ficando cada um responsável por uma parte. No fim integrá-los-iam numa aplicação.

O Manuel ficou de implementar o módulo para determinar o local do tipo pedido que está a menor distância do utilizador. São dadas as coordenadas do local onde o utilizador está e o tipo de loja que pretende, bem como a descrição dos locais existentes. Para cada local, são dadas as coordenadas e tipo (restaurante, centro comercial, bar, café, etc.). Se houver vários locais do tipo pretendido à mesma distância, deve indicar o que ocorreu primeiramente na lista de dados.

Tarefa

Escrever o programa para processar os dados e dar resposta ao pedido. As coordenadas de cada local, relativas a um mesmo referencial (ortonormado), são definidas por um par de inteiros. O tipo de local é identificado por uma sequência de maiúsculas (no máximo três).

Input

A primeira linha contém a descrição da localização atual do utilizador e o tipo de loja pretendido, sendo as coordenadas e o tipo separados por um espaço (por exemplo, `100 -160 SM`). Na segunda linha, tem o número de locais existentes (nunca terá mais de 200). Segue-se uma linha por cada local com os dados respetivos: coordenadas do ponto (inteiros) e tipo de local separados por um

espaço (por exemplo, `140 120 R`, que indica um Restaurante). O tipo é sempre uma sequência de letras maiúsculas (não mais do que três). Podem aparecer outras lojas que não as indicadas no exemplo. As coordenadas pertencem ao intervalo $[-300, 300]$.

Output

Uma linha com as coordenadas do local pedido, seguido da indicação do tipo e de um inteiro que é o *quadrado da distância* a que está (por exemplo, `220 -140 SM 14800`). **Não será assim necessário (nem deverá) calcular raízes quadradas**. Como se referiu, se houver dois ou mais locais à mesma distância, dará como resposta o que ocorrer primeiro nos dados. Se não houver nenhuma loja do tipo pretendido deve imprimir `Nenhum` (note a maiúscula na primeira letra).

Exemplo

Input

```
100 -160 SM
15
140 120 R
-60 160 C
200 -200 R
-160 180 CC
-100 -100 SM
220 -140 SM
-40 160 C
180 100 CC
60 40 SM
180 -20 R
120 220 SM
100 -180 R
180 -100 CC
200 0 SM
-120 0 R
```

Output

```
220 -140 SM 14800
```

Exemplo 2

Input

```
100 -160 SM
0
```

Output

```
Nenhum
```

Problema D

Indicador mais-ou-menos

A empresa ComputerWave decidiu implementar um novo sistema para a gestão mensal do seu stock de computadores, pedindo ao engenheiro informático Bernardo para tratar do assunto. O Bernardo, que é um engenheiro inovador, implementou um indicador, a que deu o nome de indicador mais-ou-menos, que calcula, mensalmente, o número de computadores que deverá haver em stock. O indicador baseia-se nas vendas dos últimos três meses e nas vendas do mesmo mês nos dois anos anteriores. O cálculo do indicador não é mais do que a média entre esses dois valores: a média das vendas nos últimos três meses e a média das vendas do mesmo mês nos dois anos anteriores. Estas médias devem ser truncadas, isto é, devem ser arredondadas para o maior número inteiro menor ou igual ao valor exato. Para construir o indicador para o mês corrente, o Bernardo extrai da base de dados de vendas os valores das vendas nos últimos 24 meses.

Tarefa

Escrever um programa para ajudar o engenheiro Bernardo a calcular o indicador mais-ou-menos.

Input

O input é constituído por uma sequência de 24 linhas. Em cada linha há dois números: a data e o valor das vendas. A data aparece no formato AAAAMM, por exemplo 201401. O input está por ordem cronológica e não há meses, do mesmo ano, repetidos. Sabemos ainda que o número de vendas em cada mês não ultrapassa o valor 1000.

Output

Uma linha com apenas um número inteiro que representa o número de computadores que é necessário ter em stock no próximo mês, isto é o valor do indicador mais-ou-menos.

Exemplo

Input

```
201205 28
201206 33
201207 28
201208 28
201209 28
201210 28
201211 28
201212 28
201301 28
201302 28
201303 28
201304 28
201305 26
201306 17
201307 28
201308 28
201309 28
201310 28
201311 28
201312 28
201401 28
201402 70
201403 43
201404 28
```

Output

```
37
```

Problema E

Código d'Avintes

A julgar pelos filmes e séries de mistério, o melhor para enviar mensagens cifradas é ter um bom livro de códigos. Na verdade qualquer livro serve. Se emissor e recetor tiverem previamente convencionado um livro, podem apenas transmitir o número da página, o número da ordem da linha na página a contar do topo, e o número de ordem da palavra na linha. São mesmo espertos estes tipos que escrevem ficção...

O nosso amigo Zé, natural de Avintes, matutava nisto tudo enquanto saboreava um naco da afamada broa da sua região. Subitamente teve uma ideia. Porque não usar um dicionário como livro de código? Bastava enviar um número por palavra, o da ordem da palavra no dicionário. Funcionário público em horário diurno e agente secreto nas horas vagas, o nosso amigo Zé ficou esfuziante. Assim já podia enviar mensagens secretas pelas declarações de IRS!

Tarefa

Criar um programa de codifica e descodifica mensagens a partir de um dicionário. O programa começa por ler um dicionário e seguidamente lê uma sequência de mensagens a codificar ou descodificar. No processo de codificação cada palavra é convertida no seu número de ordem no dicionário, sendo a primeira palavra associada ao número 1. O processo de descodificação é o inverso, cada número é convertido na palavra do dicionário na ordem correspondente.

Input

A linha inicial contém um inteiro n, com $1 \leq n \leq 1000$, o número de palavras do dicionário, seguida de n linhas contendo as palavras do dicionário.

As linhas seguintes contêm as ações a efetuar. Estas linhas iniciam-se com um número inteiro a, com $0 \leq a \leq 2$,com o seguinte significado:

0 - fim das mensagens;
1 - linha com mensagem a codificar;
2 - linha com mensagem a descodificar.

No caso das mensagens a cifrar ou decifrar, o valor inteiro seguinte c indica o número de palavras da mensagem. Se a for 1, ou seja cifrar a mensagem, o resto da linha é constituído por c palavras, com $2 \leq c \leq 100$. Se a for 2, ou seja decifrar a mensagem, o resto da linha é constituída por c inteiros. No máximo haverá 100 mensagens a transmitir. Cada palavra não tem mais do que 20 carateres.

Output

Uma linha por cada mensagem a cifrar ou a decifrar, na mesma ordem em que apareceu no *input*. No primeiro caso contendo uma sequência de inteiros, no segundo uma sequência de palavras.

Exemplo

Input

```
3
AMANHECER
AO
ATACAR
1 3 ATACAR AO AMANHECER
2 3 3 2 1
0
```

Output

```
3 2 1
ATACAR AO AMANHECER
```

Problema F

Lançado às feras

"Limpinho, limpinho..." – Futebol ou triste fado? No ToPAS 2013, Calimero procurava uma saída de um túnel. Está agora festejar ou em apuros? Vamos seguir os seus passos.

Sabemos que está algures num domínio retangular, discreto, à beira mar plantado. Tenta gerir as suas economias. Segue instruções rigorosas: movimentos para Sul (**S**), Norte (**N**), Este (**E**), ou Oeste (**W**), de um certo número de posições. Pelo caminho, ganha algum dinheiro, que vai dando para as despesas apenas. O pior são as taxas (**T**) que o surpreendem no caminho e que, instantaneamente, lhe retiram 10% das economias correntes. Na verdade, pode não ser tanto pois essas perdas são sempre aproximadas às unidades e nunca excedem 10% dessas economias. Com sorte, encontra algum investimento (**I**) que lhe permitirá instantaneamente repor 5% das economias correntes (com aproximação às unidades e sem exceder 5% dessas economias).

Entre sortes e azares, vai andando, às vezes desnorteado (trocando o Norte com o Sul) por ter apanhado uma pancada forte (**X**). Após tal pancada, prossegue imediatamente, desorientado, só recuperando com uma nova pancada forte (**X**) ou um grande susto (por quase se afogar). Se recuperar durante um movimento para Sul/Norte, inverte imediatamente o sentido, e prossegue.

Por vezes, recebe indicações para emigrar, saindo do domínio retangular por uma fronteira (**A**). Mas não sai para Oeste nem para Sul, porque não se quer afogar. Antes de entrar no oceano, interrompe sempre o movimento (e, com o susto, até recupera o norte, se for caso disso!), aguardando nova instrução. Fora ou dentro segue instruções precisas. Quando está fora, consegue reforçar as poupanças (10 unidades monetárias por cada posição que visita, mesmo de passagem) e não tem taxas nem investimentos. Quando emigra, espera um dia poder voltar. Mas, na fronteira (Norte e Este), existem feras (**F**) que o podem aniquilar se tiver a infelicidade de esbarrar nalguma, num movimento de saída ou entrada, ou no exterior (sobre a fronteira).

Tarefa

Escrever um programa para seguir os passos de Calimero, na execução das diretrizes recebidas e, se não for aniquilado, determinar a sua posição final e o valor final das suas economias.

Input

A primeira linha tem três inteiros, s, m e n, com $0 \leq s < 1000$ e $m, n < 20$, que representam as economias iniciais e o número de linhas e de colunas da tabela que define o *domínio retangular*. Segue-se a tabela, formada por m sequências de n letras maiúsculas, **T**, **I**, **X**, **A**, **F**, com o significado acima, e **V** (vazia). A Sul e Oeste tem apenas **A**'s. A Norte e Este tem **A**'s e **F**'s. No interior tem **T**, **I**, **X** ou **V**'s. Depois da tabela, tem o número de movimentos a efetuar e, finalmente, a descrição de cada movimento, dada por um carater (S,N,E,W) seguido de um dígito (0, ..., 9) que é o deslocamento. No início, Calimero está dentro do domínio, no canto inferior esquerdo. Essa posição tem coordenadas (1,1) e contém **V**. Sobre a fronteira, já está fora. Pontos com abcissa ou ordenada zero estão no oceano. As coordenadas das posições visitadas no exterior são sempre positivas. **O referencial é o canónico**. O número de movimentos a efetuar não excede 100.

Output

Uma linha com a palavra `Aniquilado`, se tal acontecer. Caso contrário, uma linha com o formato `Dentro(x,y):s:z` ou `Fora(x,y):s:z`, sem espaços, em que `x` e `y` definem as coordenadas da posição final e `s` as economias, e `z` o estado (`D` ou `ND` para desnorteado ou não desnorteado).

Exemplo 1

Input 1

```
108 9 9
AFFFAFFFF
AVXXXTVVF
AIXVVTVVA
AVXVVVVVA
AVXVVVXVF
AVVVVVXVA
AVVVIVXVA
AVVVVVTVF
AAAAAAAAA
4
E6
N2
W3
N6
```

Output 1

```
Dentro(4,1):102:ND
```

Exemplo 2

Input 2

```
108 5 9
AFFFAFFFF
AVVVVVXVA
AVVVIVXVA
AVVVVVTVF
AAAAAAAAA
8
E6
N2
W3
N6
S3
W2
N1
E8
```

Output 2

```
Fora(10,2):137:D
```

Exemplo 3

Input 3

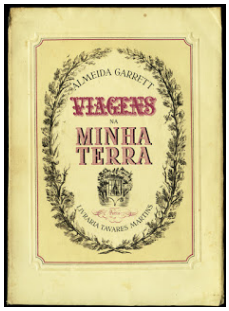
```
510 5 9
AFFFAFFFF
AVVVVVXVA
AVVVIVXVA
AVVVVVTVF
AAAAAAAAA
8
E6
N2
W3
N6
N1
W9
S1
E7
```

Output 3

```
Aniquilado
```

Problema G

Viagens na minha terra



Não vamos falar de Carlos, Frei Dinis e Joaninha, personagens desse clássico de Almeida Garrett, editado em 1846, mas de um grupo de pessoas que está a tentar reservar uma viagem no portal da empresa *Viagens na minha terra – Vá para fora cá dentro* (VMT@pt). O grupo pretende viajar até um certo local, devendo a viagem decorrer num certo período de tempo. A VMT@pt efetua várias rotas, algumas com passagem na origem e no destino pretendidos, no período previsto, e outras não. Nem todas as rotas têm já lugares suficientes para o grupo. A VMT@pt tenta cumprir horários, mas o estado de alguns troços dos trajetos tem provocado atrasos. Por isso, tem um aviso no seu portal com informação sobre os troços em que há problemas. O grupo gostaria de evitar problemas que atrasem a sua viagem, o que pode acontecer se os problemas ocorrerem entre a origem da rota e o destino do grupo na rota selecionada para a viagem. Por isso, estão a tentar saber quantas alternativas teriam com número mínimo de problemas (que pode não ser zero).

Tarefa

Escrever um programa para indicar quantas alternativas o grupo teria com número mínimo de problemas (só interessa os que afetem a sua viagem). As rotas adequadas para o grupo devem respeitar as restrições de horário, de lugares e passar na origem e destino do grupo, claro!

Input

Na primeira linha tem cinco inteiros: número de elementos do grupo, origem do grupo, destino do grupo, e dois inteiros (entre 7 e 22) que representam o intervalo em que a viagem deverá decorrer (as horas de partida e chegada do grupo terão de pertencer a esse intervalo). Na segunda linha tem um inteiro n (inferior a 200) que é o número de locais existentes (os locais são identificados por inteiros de 1 a n). Segue-se uma tabela quadrada, com n linhas e n colunas: o elemento que está na linha i e na coluna j é 1 se existir algum problema no troço de i para j, é 0 se não existir problema, e é 2 se o troço não existir.

Depois tem a descrição das rotas, seus horários, e lugares disponíveis atualmente. Para cada rota, tem duas linhas com inteiros

$$k \;\; h$$
$$v_1 \;\; n_1 \;\; d_1 \;\; v_2 \;\; n_2 \;\; d_2 \;\; v_3 \;\; n_3 \;\; d_3 \;\; \ldots v_{k-1} \;\; n_{k-1} \;\; d_{k-1} \;\; v_k$$

em que k (maior ou igual a 2) é o número de locais por onde a rota passa e h é a hora de partida da rota (sempre a horas certas), cada v_i é um local, cada troço (v_i, v_{i+1}) é sempre válido, o valor n_i é o número de lugares disponíveis nesse troço e d_i é a duração prevista para esse troço (em minutos), nessa rota. As rotas são unidirecionais e nenhuma rota passa duas vezes pelo mesmo local. O input termina com `0 0` (dois zeros, separados por espaço).

Output

Se não for possível efetuar a reserva da viagem, terá uma linha com a palavra `Impossivel`. Se for possível, terá uma linha com dois inteiros `c p`, separados por um espaço, em que `c` é o número de alternativas encontradas e `p` é o número de problemas entre a origem da rota e o destino do grupo.

Exemplo 1

Input

```
10 5 2 10 15
7
2 1 2 1 1 2 2
0 2 0 2 2 2 1
1 0 2 0 1 2 2
0 2 0 2 0 1 2
0 2 0 0 2 2 2
2 2 2 2 0 2 0
0 0 2 2 2 1 2
7 8
1 20 30 2 13 50 7 15 15 6 25 12 5 17 15 3 12 23 4
4 11
5 30 60 3 2 10 2 15 23 7
7 9
4 20 70 6 13 10 5 18 30 3 10 90 2 25 90 7 2 120 1
5 7
7 13 17 6 12 50 5 20 23 4 17 35 3
7 12
6 2 10 5 10 23 4 12 30 3 15 35 2 3 60 7 18 35 1
4 14
6 20 30 5 11 20 4 2 5 3
7 9
3 7 45 1 18 20 5 15 30 4 10 40 6 13 30 7 12 75 2
4 9
3 12 40 4 10 5 1 16 60 2
0 0
```

Output

```
1 0
```

Exemplo 2

Input

```
11 5 2 7 20
7
2 1 2 1 1 2 2
0 2 0 2 2 2 1
1 0 2 0 1 2 2
0 2 0 2 0 1 2
0 2 0 0 2 2 2
2 2 2 2 0 2 0
0 0 2 2 2 1 2
7 8
6 2 90 5 8 30 4 12 35 3 15 45 2 3 25 7 18 35 1
4 16
6 20 30 5 11 90 4 2 25 3
7 18
6 2 10 5 15 20 4 2 25 3 5 30 2 3 30 7 18 95 1
6 19
6 2 30 5 15 30 4 20 25 3 15 30 2 3 30 7
0 0
```

Output

```
Impossivel
```