ToPAS'12
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

11 de maio de 2012

Este conjunto de problemas deverá conter sete (7) problemas e dezasseis (16) 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.

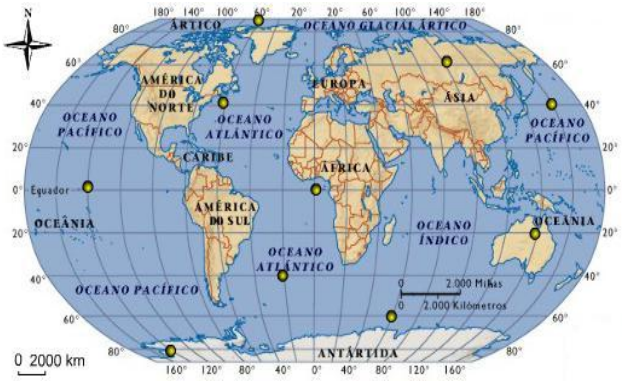
ToPAS'12
Torneio de Programação para Alunos do Secundário

Dep. Ciência de Computadores – FCUP
11 de Maio de 2012

Conteúdo

Problema A

Rock in Rio



A tripulação dos aviões da companhia AirGlobe recebe um bónus sempre que atravessa o meridiano de Greenwich ou o equador. O bónus pode ser convertido em dias de descanso e é expresso em pontos, atribuídos do seguinte modo: 1 ponto se o voo atravessa o meridiano de Greenwich; 2 pontos se o voo atravessa o equador; 4 pontos se o voo atravessa o meridiano de Greenwich e o equador.

O comandante Passarinho quer ir ao Rock in Rio de Lisboa e precisa de saber se já tem os pontos necessários para poder descansar nesses dias. O semimeridiano de Greenwich, usado como referência para a determinação da longitude, é por vezes designado por "meridiano de Greenwich", sendo normalmente representado nas cartas geográficas (marca de 0 graus de longitude). Contudo, o comandante Passarinho sabe também que o meridiano de Greenwich é um círculo máximo que passa pelos polos e que divide a Terra em dois hemisférios: o hemisfério oriental e o hemisfério ocidental. Não dará a volta ao mundo para ganhar mais pontos mas quer os pontos a que tem direito, pois meridiano é meridiano, não é semimeridiano!

Tarefa

Para efeito de cálculo do bónus, cada voo é descrito pelas coordenadas geográficas (latitude e longitude) do aeroporto de partida e do aeroporto de chegada. A latitude, que mede a distância ao equador, é expressa em graus e varia entre 0 e 90 graus, para norte ou para sul. A longitude, que mede a distância ao "meridiano de Greenwich", é também expressa em graus e varia entre 0 e 180 graus, para este ou para oeste.

Com esta informação disponível para cada um dos voos pilotados pelo comandante Passarinho, será fácil calcular quantos pontos ele tem.

Os dados para o programa são introduzidos da seguinte maneira: primeiro, o número de voos do comandante Passarinho. Depois seguem-se tantas linhas quantas o número de voos. Em cada linha

surgem oito elementos de informação, em dois grupos de quatro. O primeiro grupo descreve as coordenadas do aeroporto de partida e o segundo grupo descreve as coordenadas do aeroporto de chegada. Em cada um dos grupos, primeiro vem um número inteiro, com valor entre 0 e 90 exclusive, depois uma letra maiúscula `N` ou `S`, depois um número com valor entre 0 e 180 exclusive e depois uma letra maiúscula `E` ou `W`. Nesta descrição, o primeiro par de valores, número e letra, representa a latitude do aeroporto, com `N` significando hemisfério norte e `S` o hemisfério sul; o segundo par representa a longitude do aeroporto, com `E` significando que o aeroporto fica no hemisfério oriental e `W` no ocidental.

O programa escreverá uma linha, com um número inteiro representando o número de pontos do comandante Passarinho.

Note que não há nenhum aeroporto situado sobre o meridiano de Greenwich nem sobre o equador.

Exemplo 1

Input

```
4
41 N 8 W 38 N 9 W
33 N 7 W 68 N 95 E
1 S 69 W 3 N 25 W
48 N 175 E 48 N 175 W
```

Output

```
4
```

Exemplo 2

Input

```
5
81 N 120 W 12 S 11 W
9 S 5 E 33 N 155 W
34 N 56 E 68 N 95 E
2 S 2 W 9 N 42 W
48 N 6 E 38 N 9 W
```

Output

```
9
```

Problema B

Quando é o ToPAS?

É frequente perguntarem-nos: "Quando é este ano o ToPAS?". É simples. O ToPAS é habitualmente na sexta feira da semana académica da Universidade do Porto, que por sua vez se realiza na primeira semana completa do mês de maio. Ou seja, é a sexta feira da primeira semana cujo domingo calha em maio. Até se pode fazer um programa para determinar esta data!

Só é necessário saber como se calcula o dia da semana de uma data do calendário representada por *ano*, *mês* e *dia*. Uma forma simples de o fazer é usar o algoritmo de Tondering e calcular a seguinte expressão.

```
(ano + ano/4 - ano/100 + ano/400 + Tabela[mes] + dia) % 7
```

O valor da expressão é um inteiro entre 0 e 6, inclusive, sendo 0 interpretado como domingo, 1 como segunda e assim sucessivamente. Antes de a avaliar é necessário começar por subtrair 1 ao ano se o mês for anterior a março (janeiro ou fevereiro).

Nesta expressão o operador `%` corresponde à operação módulo (resto da divisão inteira), sendo as divisões indicadas também inteiras. A expressão também depende do mês, sendo essa função definida pela tabela que se segue.

Mês	**Valor**	**Mês**	**Valor**
janeiro	0	fevereiro	3
março	2	abril	5
maio	0	junho	3
julho	5	agosto	1
setembro	4	outubro	6
novembro	2	dezembro	4

Para a data do ToPAS, que é afinal o que se pretende, esta fórmula simplifica-se porque maio é posterior a março e o respetivo valor da tabela é zero.

Tarefa

Escreva um programa que funcione como um calendário perpétuo do ToPAS, e que, dado um ano, determina o dia do mês de maio em que este concurso tem lugar.

Os dados são constituídos por uma linha com apenas o inteiro superior a 2010 e inferior a 3000, que identifica o ano.

O resultado é também constituído por uma única linha com um inteiro entre 1 e 31 (inclusive), e que é o dia em que se realiza o concurso no ano dado.

Exemplo 1

Input

```
2012
```

Output

```
11
```

Exemplo 2

Input

```
2022
```

Output

```
6
```

Problema C

Patinagem

Existem desportos, como a patinagem artística, em que não há um valor facilmente observável e mensurável para aferir o mérito dos atletas. Não há bolas a entrar na baliza, tempos de corrida ou alturas de fasquia. A avaliação é feita por um júri e cada elemento atribui uma classificação individual. A classificação final é a média das classificações dos vários elementos.

Mesmo assim, por vezes há elementos que atribuem classificações bastante diferentes dos restantes, o que pode distorcer significativamente a classificação final. Por isso é habitual retirar os valores extremos antes de calcular a média. No entanto, se os valores extremos ocorrerem mais de uma vez, ou seja, se mais do que um elemento do júri atribuir essa classificação (máxima ou mínima), o valor correspondente é tomado como significativo e é considerado para o cálculo da média.

Tarefa

Pretende-se um programa que, dadas as n classificações individuais atribuídas pelos n elementos do júri, calcule a classificação final. Estas classificações são números inteiros entre 0 e 100.

A classificação final deve ser calculada como um valor inteiro. Isto é, deve ser usada a divisão inteira para dividir a soma dos valores considerados pelo número de valores considerados.

A primeira linha dos dados contem o valor de n, que é sempre maior do que 2 e menor ou igual a 10. Seguem-se n inteiros não negativos e menores ou iguais a 100, que são as classificações individuais atribuídas.

O resultado é constituído por uma única linha com um inteiro não negativo e menor ou igual a 100.

Exemplo

Input

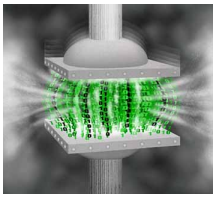
```
5
45
55
43
55
44
```

Output

```
49
```

Problema D

Codificação *run-length*



A codificação *run-length* é uma técnica para comprimir textos onde existem sequências longas de carateres repetidos. Por exemplo, a sequência `AAAAAAA` podia ser representada por `7A` (sete vezes o carater `A`), facto em que se inspira a representação que vamos usar.

Para evitar ambiguidade e não aumentar muito o comprimento do textos nos casos de sequências curtas, não comprimimos as sequências com menos de três carateres repetidos (i.e. ficam inalteradas). Se um carater for repetido três ou mais vezes, então deve ser seguido de um dígito (`0` a `9`) que indica o número de carateres repetidos (para além dos três iniciais). Se um carater ocorrer repetido mais do `12` vezes `(3+9)` então as restantes devem ser tratadas como uma nova sequência; por exemplo, `AAA9AAA0` seria a codificação de `AAAAAAAAAAAAAAA`, isto é, da sequência de `15 (=3+9+3+0)` carateres `A`.

Tarefa

Escreva um programa que lê da entrada-padrão um texto e produz na saída-padrão a sua codificação *run-length*. O texto é dado numa única linha, tem apenas letras ou dígitos, tendo pelo menos uma letra ou dígito e não mais do que 1000. Além desses carateres, terá apenas o terminador de linha.

Exemplo 1

Input

```
ABBA
```

Output

```
ABBA
```

Exemplo 2

Input

```
AAABBBBAAAAAAAAAAAAAA
```

Output

```
AAA0BBB1AAA9AA
```

Exemplo 3

Input

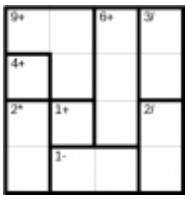
```
000bbbaaaaaa1110AAA000000000000000BaaaaaBbbb000000
```

Output

```
0000bbb0aaa311100AAA000090000Baaa2Bbbb00003
```

Problema E

Ken-Ken



O **ken-ken** é um desafio lógico-aritmético semelhante ao sudoku. Baseia-se numa grelha quadrada de tamanho n (i.e., com n **linhas** e n **colunas**) que deve ser preenchida com números e obedecer a um conjunto de regras estruturais e restrições aritméticas.

A grelha deve ser preenchida com números inteiros de 1 a n e de modo que em cada linha e em cada coluna não haja números repetidos.

Está dividida em **regiões**. As regiões são um conjunto de células contíguas, com pelo menos uma e menos de n células, que podem ter diferentes formas. As células duma região podem estender-se por uma mesma linha, uma mesma coluna, ou simultaneamente por várias linhas e colunas, formando um quadrado ou um L (ele).

Cada região tem associada uma **restrição**, definida por um valor e uma operação aritmética (`+`, `*`, `-` ou `/`). A operação agrega o conteúdo de todas as células da região e o resultado deve ser o valor dado. Por exemplo, se a restrição for `9 +` então os valores da região somados devem totalizar `9`.

As operações soma e multiplicação podem ter um número arbitrário de operandos e, sendo comutativas, a ordem pelo qual são operados é irrelevante. As operações de subtração e divisão não são comutativas nem associativas e, por isso, assume-se que têm sempre apenas 2 operandos e que o primeiro tem valor igual ou superior ao segundo. Por exemplo, se a restrição for `2 /` e a região contiver os valores `2` e `4` então a operação em questão é `4/2`. No caso de a região ter apenas uma célula pode assumir que a operação é uma soma (sendo o resultado o valor dessa célula).

Tarefa

Pretende-se um programa que leia a definição de um problema de ken-ken e um conjunto de preenchimentos da grelha e indique quais os corretos ("`OK`") e os incorretos ("`KO`"). O programa deve verificar quer as regras estruturais (não existência de números repetidos em cada linha e em cada coluna) quer as restrições aritméticas.

A primeira linha dos dados contem um par de inteiros n r, sendo n a dimensão da grelha (maior do que 3 e menor do que 10) e r o número de regiões. Seguem-se r definições de regiões e respetivas restrições. A linha inicial de cada uma dessas definições contem um terno b v o, constituído por dois inteiros e um caracter: b é o número de células na região, v o valor agregado (i.e., o resultado da operação de agregação) e o a operação de agregação (`+`, `*`, `-` ou `/`). As b linhas seguintes têm as

coordenadas das células que formam a região: em cada linha terá um par de inteiros l c sendo l o índice de linha e c o de coluna (são ambos não negativos e inferiores a n; a célula com coordenadas `(0,0)` encontra-se no canto inferior esquerdo da grelha).

Depois da descrição do ken-ken, tem uma linha com um inteiro p que indica o número de resoluções a verificar. Segue-se a descrição dessas p resoluções: para cada resolução, tem um identificador (sequência de letras ou dígitos, de comprimento inferior a 20) numa primeira linha e nas n linhas seguintes (cada uma com n inteiros), tem a definição dos valores da grelha após essa resolução.

O resultado é constituído por p linhas iniciadas com o identificador da resolução seguido de um carater de dois-pontos e um espaço seguido da sequência de caracteres `OK` se a resolução estiver correcta, ou `KO` no caso contrário.

Exemplo

Input - (desdobrado em duas colunas)

```
4 8
2 2 *
0 0
1 0
1 1 +
1 1
2 1 -
0 1
0 2
2 2 /
0 3
1 3
1 4 +
2 0
3 9 +
3 0
3 1
2 1
3 6 +
3 2
2 2
1 2
2 3 /
3 3
2 3
3
A
3 4 2 1
4 2 1 3
2 1 3 2
1 3 4 1
B
1 2 3 4
2 3 4 1
3 4 1 2
4 1 2 3
C
3 4 2 1
4 2 1 3
2 1 3 4
1 3 4 2
```

Output

```
A: KO
B: KO
C: OK
```

Problema F

Afinal ainda há metro

A administração do metro decidiu substituir as máquinas automáticas de venda de bilhetes nas estações, porque, sendo muito sofisticadas, a verdade é que são muito caras e avariam muito. As novas máquinas são mais baratas e mais robustas, mas infelizmente são mais básicas: exigem que o comprador introduza a quantia exata (não dão troco, portanto) e além disso, não aceitam mais de quatro moedas de cada vez, numa mesma compra. Por exemplo, se o preço do bilhete fosse €1.30, o comprador poderia introduzir uma moeda de um euro e uma de 20 cêntimos e uma de 10 cêntimos, ou então uma de um euro e três de 10 cêntimos, mas a máquina recusaria se o pagamento fosse feito com uma moeda de um euro, duas de 10 cêntimos e duas de 5 cêntimos.

Assim sendo, os preços das viagens têm de ser escolhidos de forma a poderem ser pagos com quatro moedas. Por exemplo, não pode haver viagens que custem €3.35, pois precisaríamos de cinco moedas, pelo menos, para pagar esse valor.

Portanto, no momento de aumentar o preço do bilhete, o novo preço tem de ser escolhido de maneira a poder ser pago com quatro moedas, quando muito. A administração não quer perder dinheiro e, por isso, se o novo valor pensado pela administração não for aceitável pelas máquinas, esse valor terá de ser aumentado, o menos que for possível, de maneira a atingir-se um valor aceitável. Retomando o exemplo acima, se a administração quisesse inicialmente que o novo preço fosse €3.35, teria de o aumentar para €3.40, que já pode ser pago com quatro moedas.

O problema é, dado o preço que a administração pretende e os valores da moedas que a máquina aceita, calcular o menor preço maior ou igual ao preço pretendido que pode ser pago com quatro moedas dessas. Note que a máquina está configurada de maneira a não aceitar todas as moedas em circulação (por exemplo, pode não aceitar moedas de um cêntimo ou de dois cêntimos, ou de dois euros). Além disso, não é impossível que outras moedas entrem em circulação em breve (por exemplo, moedas de 3 cêntimos ou de 25 cêntimos, como há em alguns países). Por isso é que é preciso indicar também os valores das moedas que a máquina está preparada para aceitar.

Tarefa

Escrever um programa para calcular o preço do bilhete, dado o valor que administração pretende e os valores das moedas que a máquina aceita. O preço deve ser o menor valor maior ou igual ao

preço pretendido pela administração que pode ser formado usando no máximo quatro moedas das que a máquina aceita.

Os dados para o programa são introduzidos pela seguinte ordem: primeiro, o preço pretendido pela administração, em cêntimos; depois, na linha seguinte, um número, que indica quantos tipos de moedas a máquina aceita; finalmente, na linha seguinte, uma lista de números, com tantos números quantos os tipos de moedas, com o valor correspondente a cada um desses tipos de moedas, expressos em cêntimos, por ordem crescente.

O programa escreverá o preço calculado para o bilhete, na forma de um número inteiro de cêntimos.

Sabemos que preço pretendido pela administração não é maior do que o quádruplo do valor moeda de maior valor e que o número de moedas diferentes nunca será maior do que 10.

Exemplo 1

Input

```
335
6
5 10 20 50 100 200
```

Output

```
340
```

Exemplo 2

Input

```
44
5
1 3 10 20 50
```

Output

```
44
```

Exemplo 3

Input

```
90
3
5 10 25
```

Output

```
100
```

Problema G

Balão à vista

Quem bloqueou as portas? Não foi um ET em estado de choque, mas `TAKIOR`, robot inteligente especialmente programado para pôr termo a feriados, festas e balões. Como podia o décimo aniversário do ToPAS lhe escapar? Rapidamente se pôs a caminho do Departamento em festa, sem perder um minuto a estudar a planta do amplo edíficio, tendo apenas as coordenadas da porta de entrada. À entrada, pareceu-lhe um labirinto. Por isso, decidiu adoptar um algoritmo simples, mas arriscado. Percorreria laboratórios, anfiteatros, gabinetes, salas e afins, sem se afastar das paredes. Bloquearia as portas por onde tivesse de passar, evitando andar as voltas e parecer uma cigarra tonta. Confiscaria os balões que encontrasse, que utilizaria para levantar voo quando saisse.

Recapitulando, `TAKIOR` deve percorrer todos os compartimentos que conseguir. Quando entra num compartimento, deve bloquear a porta por onde entrou. Depois, sem se afastar da parede, percorre o compartimento e recolhe os balões que encontra junto da parede (supomos que as portas não bloqueadas estão encostadas e funcionam como paredes nesta pesquisa). A seguir procura a porta não bloqueada. Se existir, abre-a para passar ao compartimento vizinho ou abandonar o edíficio se essa porta for a de saída. Se o compartimento só tiver uma porta (a que bloqueou quando entrou), aguarda pacientemente que o encontrem agarrado aos balões que pretendia confiscar. Se sair do edíficio, levanta imediatamente voo. Terá sempre confiscado pelo menos um balão.

`TAKIOR` parece um boneco de corda, mas, de facto, esse dispositivo lateral tem os sensores que lhe permitem manter o contacto com as paredes e voltar orientar-se quando entra num compartimento novo. Assim, quando entra numa sala, **vira sempre à sua direita**, ficando o sensor direccionado para a parede que tem à sua direira. Manterá esta condição durante o movimento no compartimento.

Tarefa

Escrever um programa para analisar um cenário virtual, e determinar o número de balões que `TAKIOR` confiscou ou pretendia confiscar, e detectar ainda se ficou preso no edíficio ou levantou voo. Cada cenário é composto por uma planta do edíficio, representada por uma grelha rectangular, com identificação das posições das paredes e das portas, bem como das posições que contêm balões.

Não há balões atrás de portas. **Todos os compartimentos são rectangulares e têm ou uma ou duas portas. Não há portas junto aos cantos**. O compartimento de entrada tem sempre duas portas e só uma delas dá acesso ao exterior. Existe pelo menos uma porta de saída para o exterior num outro compartimento.

Na primeira linha de dados tem dois inteiros m e n, menores do que 100, e que representam o número de linhas e de colunas da grelha. Seguem-se m linhas. Cada uma contém uma sequência de n carateres, formada por `'E'` (porta de entrada), `'S'` (porta de saída), `'P'` (porta interior), `'X'` (parede), `'B'` (balão), ou espaços `' '`. No ínicio e fim de cada linha pode ter `'X'`, `'S'` ou `'E'`, existindo apenas um `'S'` e `'E'` no cenário. Pode supor que a porta de entrada se encontra sempre na **célula (0,2)**, junto ao canto inferior esquerdo da grelha.

O resultado é composto por duas linhas: na primeira tem o número de balões recolhidos; a segunda tem a frase "`Levantou voo`" ou "`Veio para ficar`", conforme `TAKIOR` tenha ou não conseguido sair do edíficio.

Exemplo 1

Input

```
18 20
XXXXXXXXXXXXXXXXXXXX
X      X           X
X   B  X           X
X      XXPXXXXXXXXXX
X      X    X      X
X     BX    X    B S
X      P    X      X
X      X    XXPXXXXX
X      XXXXXX      X
X      X    X      X
X      X    P      X
X      X    X      X
XXXXPXXXXXXXXXXXXXX
X                 BX
X       B           X
X       B           X
X             B     X
XXEXXXXXXXXXXXXXXXX
```

Output

```
3
Veio para ficar
```

Exemplo 2

Input

```
14 23
XXXXXXXXXXXXXXXXXXXXXXX
X          X           X
X   B      X    B      X
X          X        B  S
X       B  XB          X
X  B       XXXXXXXPXXXXX
X          X           X
X     B    X   B     B X
X          XXXXXXXXXPXXX
X          X     X     X
X          P     P     X
X    B     X B   X    BX
X          X     X     X
XXEXXXXXXXXXXXXXXXXXXX
```

Output

```
4
Levantou voo
```