ToPAS'11
Torneio de Programação para Alunos do Secundário

Departamento de Ciência de Computadores

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

Conjunto de Problemas

Faculdade de Ciências da Universidade do Porto

6 de Maio de 2011

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'11
Torneio de Programação para Alunos do Secundário

Dep. Ciência de Computadores – FCUP
6 de Maio de 2011

Conteúdo

Problema A

Liga Europa

Tivemos ontem os jogos da segunda mão das meias-finais da Liga Europa. À data em que escrevemos, ainda não sabemos os resultados dos jogos, mas, se tudo tiver corrido bem, teremos duas equipas portuguesas na final, o que é verdadeiramente fantástico. Nesta fase, a Liga Europa é um torneio a eliminar e em cada eliminatória há dois jogos entre cada par de equipas, primeiro no campo de uma delas, depois no campo da outra. Vence a eliminatória e passa à eliminatória seguinte (ou à final) a equipa que tiver marcado mais golos no agregado, isto é, no conjunto dos dois jogos. Em caso de empate no agregado, passa a equipa que tiver marcado mais golos fora, isto é, no jogo no campo do adversário. Este critério de desempate não é suficiente, pois pode acontecer que ambas as equipas tenham marcado o mesmo número de golos fora. No caso de empate no agregado e nos golos fora, já contando com o tempo suplementar que terá havido no jogo da segunda mão, a eliminatória resolve-se recorrendo a pontapés na marca de grande penalidade. Na gíria, dizemos *"vai-se a penáltis"*.

Tarefa

É verdade que as contas são fáceis e se fazem de cabeça, mas o sistema de informação da UEFA (entidade que organiza a Liga Europa) tem de calcular automaticamente quem passa à eliminatória seguinte, em função dos resultados dos dois jogos. Pretendemos aqui programar esse módulo do sistema: dados os resultados dos dois jogos de uma eliminatória, calcular a equipa que venceu a eliminatória.

Input

A primeira linha tem duas cadeias de carateres, cada uma delas formada só por letras e pelo caráter '_', e as duas separadas uma da outra por um espaço. A primeira palavra representa o nome da equipa que joga em casa na primeira mão (e fora na segunda); a segunda palavra representa o nome da equipa que joga fora na primeira mão (e em casa na segunda). A segunda linha tem dois números inteiros não negativos, que representam o resultado do jogo da primeira mão e a terceira linha é análoga, mas para o resultado da segunda mão. Em ambos os casos o primeiro número indica o número de golos marcados pela equipa que joga em casa e o segundo o número de golos marcados pela equipa que joga fora.

Output

Há apenas uma linha onde aparece apenas uma cadeia de carateres com o nome da equipa que vence a eliminatória, se a eliminatória se tiver resolvido sem recurso a penáltis, ou então a palavra `penaltis`, sem as aspas e sem acento, no caso de haver empate no agregado e nos golos marcados fora.

Exemplo 1

Input

```
fcporto spartak
5 1
2 5
```

Output

```
fcporto
```

Exemplo 2

Input

```
dinamo_kiev sporting_braga
1 1
0 0
```

Output

```
sporting_braga
```

Exemplo 3

Input

```
fcporto benfica
0 2
1 3
```

Output

```
fcporto
```

Exemplo 4

Input

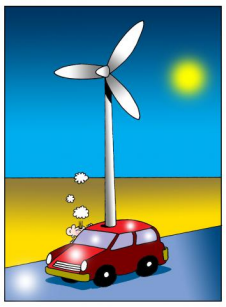
```
vitoria uniao
2 1
2 1
```

Output

```
penaltis
```

Problema B

Gasolina



Estamos em crise (isso todos nós estamos fartos de saber) e para tornar as coisas ainda mais desagradáveis, os preços do petróleo nos mercados internacionais não param de subir. Por causa disso, a gasolina e o gasóleo estão cada vez mais caros. Aliás, até parece que não há semana em que não haja um aumento no preço dos combustíveis. Não será bem assim, mas a memória trai-nos e por isso o melhor mesmo é verificar.

Tarefa

Consultando a página Web da Galp, temos acesso ao preço da gasolina, em cada dia desde o início do ano, ou, em geral, desde alguma data no passado. Pretendemos, usando esses dados, desfazer mitos e calcular exatamente o número de semanas em que o preço da gasolina subiu pelo menos uma vez.

Input

A primeira linha tem o número de semanas para as quais temos dados sobre o preço da gasolina. A segunda linha tem um número representando o preço no domingo anterior ao da primeira semana com registos completos. Para efeitos deste problema, usamos a norma que determina que a semana começa à segunda-feira e termina no domingo seguinte. Seguem-se tantas linhas como o número de semanas indicado na primeira linha, cada uma com sete números, representando o preço da gasolina em cada um dos dias da semana. Os preços da gasolina são expressos por números inteiros em milésimas de euros. Por exemplo, o número 1759 significa que o preço é 1.759 euros.

Output

Há apenas uma linha onde aparece apenas um número: o número de semanas em que o preço subiu pelo menos uma vez, de um dia para o outro. Note que se o preço subir do domingo da semana anterior para a segunda-feira da semana corrente, conta como tendo subido na semana corrente.

Exemplo 1

Input

```
5
1689
1689 1689 1689 1689 1689 1689 1689
1689 1689 1689 1699 1699 1699 1699
1699 1659 1659 1659 1659 1659 1659
1659 1659 1669 1674 1679 1674 1679
1679 1679 1679 1679 1679 1679 1679
```

Output

```
2
```

Exemplo 2

Input

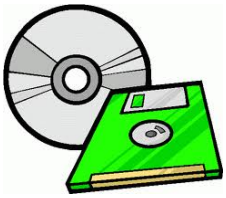
```
12
1499
1524 1524 1524 1524 1524 1524 1524
1524 1524 1524 1524 1524 1524 1524
1529 1529 1529 1529 1529 1529 1529
1529 1529 1529 1529 1529 1529 1529
1529 1529 1529 1529 1529 1529 1529
1529 1529 1529 1529 1529 1529 1529
1529 1529 1529 1539 1539 1539 1539
1514 1514 1514 1514 1514 1514 1514
1514 1514 1514 1514 1514 1514 1514
1514 1544 1544 1544 1544 1544 1544
1544 1544 1544 1544 1544 1544 1544
1544 1554 1554 1554 1544 1544 1544
```

Output

```
5
```

Problema C

Preservação Digital



Acabámos de nos deparar com um grave problema nos nossos arquivos digitais, ainda baseados em CD-ROM e alguns em disquetes. Muitos ficheiros estão corrompidos e não é possível recuperar integralmente a sua informação. Felizmente existem muitas cópias de cada um deles e por isso pensamos que será possível recuperar a maior parte dos ficheiros.

Tarefa

A estratégia de recuperação é simples: analisando bit-a-bit as várias cópias do mesmo ficheiro iremos tomar para cada posição o bit mais frequente. Em caso de frequência igual para um dado bit, é assumido o valor `0`. Por exemplo, se tivermos 4 ficheiros com as seguintes sequências de 6 bits o resultado da recuperação será o indicado na última linha

```
011000
010100
011001
001111

011000
```

Para simplificar o vosso trabalho, os ficheiros foram convertidos numa sequência de caracteres `0` e `1`. A primeira linha de input será constituída por n e m, onde n é o número de "ficheiros" e m é o número de "bits" de cada ficheiro. As n linhas seguintes são sequências de m caracteres contendo exclusivamente os caracteres `0` ou `1`. Sabe-se que n não excede 30 e m não excede 255.

Exemplo 1

Input

```
4 6
011000
010100
011001
001111
```

Output

```
011000
```

Exemplo 2

Input

```
2 4
1100
1010
```

Output

```
1000
```

Problema D

Mastermind

O *Mastermind* é um jogo para duas pessoas que fazem os papeis de *codemaker* e *codebreaker*:

- o *codemaker* escolhe uma chave secreta (uma sequência de quatro pinos escolhidos entre seis cores);
- o *codebreaker* vai deduzir qual é a chave usando o menor número de tentativas.

Por cada tentativa o *codemaker* diz duas pontuações: o *número de cores correctas na posição correcta*; e o *número de cores correctas mas na posição errada*.

Representamos as seis cores pelas letras minúsculas de '`a`' a '`f`'; se a chave for, por exemplo, "`acfb`" e a tentativa "`abcd`" as pontuações são (1, 2), ou seja, uma letra ('`a`') correcta na posição correcta e duas ('`c`', '`b`') correctas mas em posições erradas.

Podem ocorrer cores repetidas. Por exemplo, se a chave for "`acfa`" e a tentativa "`aaac`" as pontuações são (1, 2), ou seja, um '`a`' na posição correcta, um '`a`' em posição errada (o terceiro '`a`' na tentativa não conta), e um '`c`' em posição errada.

Tarefa

Pretende-se um programa que leia a chave e a tentativa e calcule as duas pontuações.

Exemplo 1

Input

```
acfb
abcd
```

Output

```
1
2
```

Exemplo 2

Input

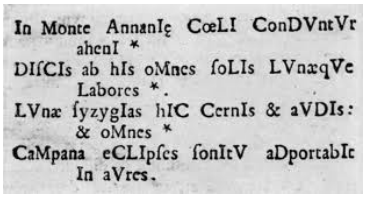
```
acfa
aaac
```

Output

```
1
2
```

Problema E

Cronogramas


In Monte AnnanIę CœLI ConDVntVr
ahenI *
DIſCIs ab hIs oMnes ſoLIs LVnæqVe
Labores *.
LVnæ ſyzygIas hIC CernIs & aVDIs:
& oMnes *
CaMpana eCLIpſes ſonItV aDportabIt
In aVres.

Um cronograma é uma frase que pode ser lida como uma data. A palavra portuguesa deriva das palavras gregas "chronos"(tempo) e "grama"(letra). Embora hoje em dia a palavra "cronograma"tenha adquirido uma nova conotação - a de diagrama representando a evolução temporal - em tempo idos os cronogramas eram simplesmente frases que escondiam uma data. Os cronogramas eram inscritos em monumentos e túmulos como uma forma poética de aludir às datas que estes evocam. Em vez de simplesmente escrever a data em numeração romana, como nesses tempos era norma, as letras da numeração romana eram usadas no meio das palavras de uma frase, misturadas com outras letras que não têm significado como numerais romanos. Por exemplo, da frase

AMOR **MAXI**

podem-se extrair os caracteres da numeração romana MMXI que equivalem ao numero 2011. Como todos sabemos, a numeração romana é de base decimal, como a árabe, mas não é posicional. Cada letra tem um valor e o valor do número baseia-se na soma do valor das letra que o compõem. Para os mais esquecidos a tabela seguinte relembra a correspondência entre letras romanas e o seu valor.

Letra	**Valor**
I	1
V	5
X	10
L	50
C	100
D	500
M	1000

Na numeração romana as letras são geralmente escritas por ordem decrescente do seu valor. Por exemplo 1500 é escrito como MD e nunca como DM. As letras que correspondem a potências de 10 (I, X, C e o M) podem ser repetidas até 3 vezes. Pode também ser usado o princípio subtrativo para simplificar algumas representações. Escreve-se assim IV em vez de IIII, e IX em vez de VIIII, por exemplo. A utilização do princípio subtrativo está limitada a valores da grandeza próxima. Podemos usar um I antes de um V ou um X, um X antes de um L ou um C, ou um C antes de um D ou de um M; mas nunca um I antes de um C ou de um M, ou X antes de um D ou de um M. Por exemplo, não faz sentido escrever MIM para 1999 que deverá ser escrito como MCMXCIX.

Tarefa

Pretende-se um programa que recebe como entrada uma linha com uma sequência de palavras que eventualmente contém um cronograma de valor inferior a 3999, e produza uma linha com o número correspondente em notação árabe. Dado que os romanos não usavam minúsculas, assumimos que as palavras são sequências de uma ou mais maiúsculas e que os cronogramas têm no máximo 100 caracteres. Se nenhuma das letras I, V, X, L, C, D, e M ocorrer, então a linha resultante terá o valor 0. Caso contrário, terá o valor correspondente ao numeral romano, que se supõe estar sempre bem definido.

Exemplo 1

Input

```
AMORE MATVRITAS
```

Output

```
2006
```

Exemplo 2

Input

```
ESTES ROMANOS SAO LOUKOS
```

Output

```
1050
```

Exemplo 3

Input

```
OK
```

Output

```
0
```

Problema F

Plano de Férias

Junho						
Se	Te	Qu	Qu	Se	Sa	Do
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

Julho						
Se	Te	Qu	Qu	Se	Sa	Do
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Agosto						
Se	Te	Qu	Qu	Se	Sa	Do
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

Setembro						
Se	Te	Qu	Qu	Se	Sa	Do
			1	2	3	4
5	6	7	8	9	10	11

Gente do norte que tencionava fazer praia no Algarve e Andaluzia nas férias da Páscoa, foi surpreendida por ventos, aguaceiros e um muito frio. *"Blancos llegaron y más blancos volvieron"*, pelo que já sonham com uma semana ou quinze dias de férias que planeiam gozar em Junho, Julho, Agosto ou Setembro, se o subsídio de férias não for na enxurrada. Parece que não irá... Assim, um grupo de amigos optimistas está já a tentar escolher uma semana de férias, de acordo com as preferências e indisponibilidades de cada um. Essa semana (7 dias de férias) tem de começar numa segunda-feira e terminar no domingo seguinte ou, em alternativa, começar num sábado e terminar na sexta-feira seguinte. Cada membro do grupo indicou uma sequência de períodos que não se sobrepõem e classificou cada período usando inteiros: `-1` para assinalar indisponibilidade e valores de `1` a `5` para manifestar preferências, sendo `5` o nível máximo. Nos períodos que não incluiu, também está disponível mas com nível de preferência zero, se os amigos acharem boa ideia ou se não existir alternativa. A classificação serve para pontuar analogamente cada dia do período referido. A semana escolhida será uma em que todos possam ir, após 1 de Junho (quarta-feira) e com fim antes de 12 de Setembro de 2011, e que tenha a pontuação máxima, considerando a soma das pontuações acumuladas nos sete dias, após processamento das preferências de todos. Se existir mais do que uma semana possível, escolherão a mais próxima, não vá o frio voltar em força.

Tarefa

Pretende-se um programa para processar a informação fornecida e indicar a data em que o grupo iniciará as férias (ou detectar a inconsistência se não existir nenhuma possibilidade de irem juntos).

Input

Uma primeira linha com o número de membros do grupo. Segue-se igual número de blocos, cada um contendo as preferências ou indisponibilidades de um membro do grupo. Um bloco começa por um inteiro positivo que denota o número de períodos que o membro referiu e depois, em cada linha, tem um quinteto no formato *"dia mês dia mês preferência"*, por exemplo, `16 6 30 6 3` (que representaria o período de 16 de Junho a 30 de Junho com preferência 3 em cada um dos dias desse período; se fosse `16 6 30 6 -1`, indicaria que o membro não poderia ir nesse período).

Output

Se existir uma semana em que todos possam, indicará a data de início das férias no formato: número do dia seguido de um espaço seguido de `de Junho`, `de Julho`, `de Agosto` ou `de Setembro`. Caso contrário, a palavra `inconsistente`.

Exemplo 1

Input

```
3
2
10 7 7 9 -1
10 6 23 6 5
2
1 6 15 7 -1
25 7 26 7 4
1
23 6 25 6 -1
```

Output

```
inconsistente
```

Exemplo 2

Input

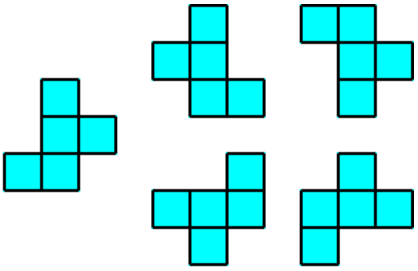
```
5
6
16 6 30 6 3
8 7 8 7 -1
9 7 31 7 5
5 8 12 8 5
23 8 31 8 2
7 6 8 6 -1
1
1 6 31 7 4
1
1 8 5 9 1
2
3 7 20 8 4
1 9 9 9 -1
3
15 6 8 7 5
15 8 31 8 -1
7 9 10 9 5
```

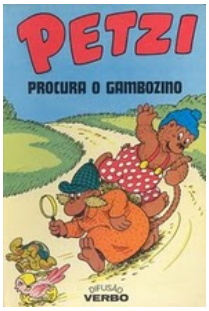
Output

```
9 de Julho
```

Problema G

Caçada aos Gambozinos






```
00000000000000
00101000000010
01111000110010
01001101011010
00000011010010
00010001100110
01110000000000
00100110001000
00000011011100
00111010010000
00000000000110
00000000000000
```

Os gambozinos são descritos nalguns dicionários como "peixes ou pássaros imaginários, com que, por brincadeira, se logram pacóvios mandando-os à caça ou à pesca desses animais", com sacos de serapilheira para que não escapem, e talvez com alguns testos para os chamar ou atordoar.

Tarefa

Pretende-se um programa para caçar gambozinos numa região rectangular dada, dividida em quadrados unitários. Essa região terá sempre um bordo com uma unidade de largura, a toda a volta, e que está vazio. Um gambozino é constituído por um conjunto de cinco quadrados unitários ligados entre si por arestas (ou seja, é um poliominó especial), podendo ser apenas das cinco formas representadas na figura, à esquerda. Nenhuma outra forma que se possa obter destas, por simetria ou rotação, é gambozino. O programa deverá analisar a região e indicar quantos gambozinos encontrou. Durante a procura, os gambozinos não se movem. A região pode ter outras espécies de animais, igualmente imóveis. Sabe-se que os gambozinos são pouco sociáveis: encontram-se isolados uns dos outros, partilham quando muito vértices mas nunca arestas.

Input

Na primeira linha tem dois inteiros M e N que definem as dimensões da região, sendo ambos maiores ou iguais a 5 e inferiores a 50 unidades. Seguem-se M linhas que descrevem o conteúdo da região, supostamente dividida em quadrados unitários. Cada linha tem N inteiros `0` ou `1` separados por espaços, `0` a indicar posição vazia e `1` posição ocupada. Os animais, incluindo os gambozinos, são formados por `1`'s contíguos na horizontal e/ou vertical. O bordo da região está sempre livre como se disse, e, consequentemente, terá sempre `0`'s como nos exemplos apresentados abaixo.

Output

Uma linha com um inteiro (positivo ou zero) que representa o número de gambozinos encontrados.

Exemplo 1

Input

```
12 14
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 0 0 0 1 0
0 1 1 1 1 0 0 0 1 1 0 0 1 0
0 1 0 0 1 1 0 1 0 1 1 0 1 0
0 0 0 0 0 0 1 1 0 1 0 0 1 0
0 0 0 1 0 0 0 1 1 0 0 1 1 0
0 1 1 1 0 0 0 0 0 0 0 0 0 0
0 0 1 0 0 1 1 0 0 0 1 0 0 0
0 0 0 0 0 0 1 1 0 1 1 1 0 0
0 0 1 1 1 0 1 0 0 1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Output

```
5
```

Exemplo 2

Input

```
6 15
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 1 1 0 0 1 0 1 0 0 0 0 1 0
0 1 0 1 1 1 1 0 1 1 0 0 1 0 0
0 1 0 0 0 0 1 1 1 0 0 1 1 1 0
0 0 0 1 0 0 0 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Output

```
1
```