Olimpíadas Nacionais de Informática
16 de Abril de 1999
Lisboa, Escola Secundária Professor Herculano de Carvalho

Primeiro Problema: Números aparentes.

Considere, como exemplo, o número 118, que habitualmente se lê “cento e dezoito”. Imagine que, para variar, se opta por ler algarismo a algarismo. Diremos então: “um, um, oito”. Alternativamente, se ao ler agruparmos os algarismos consecutivos iguais, diremos “dois uns, um oito”, ou, mais simplesmente, “dois um um oito”, o que finalmente corresponde ao número 2118.

Podemos fazer esse exercício para qualquer número, e assim implicitamente definir uma função natural de variável natural. Chamemos “aparência” a esta função, e denotemo-la por Ap. Então teremos, por exemplo, Ap(1) = 11 (a aparência de 1 é 11), Ap(11) = 21, Ap(21) = 1211, Ap(1211) = 111221, e ainda, Ap(10) = 1110, Ap(1110) = 3110, Ap(22) = 22, etc.

Percebidos estes casos, qual é a aparência de 1111111111? Como este número seria lido “dez uns”, decretamos que Ap(1111111111) vale 101, e fazemos da mesma maneira em casos análogos.

A função Ap tem a ver com a representação dos números na base 10, é claro. Também é interessante ver o que se passa na base 2. O número 14, por exemplo, escreve-se “1110”, que se leria “três uns, um zero” ou “três um um zero”. Como 3 se escreve “11” na base 2, a aparência binária, ou natural, de 14 corresponderia à cadeia binária “11110”, que representa o número 30. Ou seja, designando Ap2 esta nova função, acabámos de estabelecer que Ap2(14) = 30.

Vejamos mais um caso. Qual é a aparência natural de 4? 4 escreve-se “100”, que se lê “um um dois zero”, ou seja “11100”, o que corresponde a 28. Logo, Ap2(4) = 28.

Um caso muito interessante é 7. 7 escreve-se “111”, o que se lê “três um”, ou seja “111”. Logo, Ap2(7) = 7.

Finalmente, dados dois números x e y diz-se que x é menos aparente que y se e só se Ap2(x) <= Ap2(y). Note que a aparência natural não é monótona: já vimos que 7 é menos aparente que 4.

Tarefa:

A sua tarefa neste problema é escrever um programa para calcular a aparência binária de cada um dos números de uma lista de números, e ordenar essa lista por ordem de aparência crescente.

Dados:

Os dados vêm no ficheiro INPUT1.TXT, o qual tem n+1 linhas. Em cada linha existe um número. O número na primeira linha é n, e os números nas restantes n linhas constituem a lista de números a processar. Todos os números presentes no ficheiro estão entre 0 (zero) inclusive e 1024, exclusive.

Resultado:

O resultado vem no ficheiro OUTPUT1.TXT, que contém 2n+1 linhas. Nas primeiras n linhas estão as aparências (binárias) dos números da lista a processar, a linha seguinte fica em branco, e as restantes n linhas contêm os números da lista inicial ordenados por ordem crescente de aparência.

Exemplo:

INPUT1.TXT
```
5
4
6
8
10
12
```

OUTPUT1.TXT
```
28
22
30
238
44

6
4
8
12
10
```

Note bem:
Se não tiver tempo para resolver a parte da ordenação, faça o seu programa escrever apenas as primeiras n linhas com as aparências calculadas.

Segundo problema: *Clipping*.

Um problema clássico de computação gráfica é o *clipping*: dados um rectângulo de base horizontal, e um segmento de recta definido por dois pontos, determinar a porção desse segmento que fica dentro do rectângulo, se houver, bem entendido, pois o segmento pode ficar todo fora do rectângulo.

Tarefa:
A sua tarefa neste problema é escrever um programa fazer o *clipping* de um segmento.

Dados:
Os dados estão no ficheiro INPUT2.TXT, em duas linhas. Na primeira linha vêm quatro números inteiros representando as coordenadas x e y do canto inferior esquerdo e do canto superior direito do rectângulo. Na segunda linha vêm outros quatro números inteiros representando as coordenadas x e y dos dois extremos do segmento. Os dados representam um rectângulo de área não nula e um segmento de comprimento não nulo.

Resultado:
O resultado vem na única linha do ficheiro OUTPUT2.TXT. Essa linha contém ou as coordenadas dos extremos do sub-segmento interior ao rectângulo, ou a cadeira "invisível" no caso de o segmento dado ser totalmente exterior ao rectângulo. As coordenadas são quatro números reais, (dois para um dos extremos e mais dois para o outro) e cada um deve ser escrito com duas casas decimais.

Exemplo:

INPUT2.TXT
```
0 0 4 4
-1 0 4 5
```

OUTPUT2.TXT
```
0.00 1.00 3.00 4.00
```