Programação Estruturada

Exercício – "bin packing"

1999/2000

Pretende implementar-se um programa que indique um modo de gravar em cassetes de vídeo um conjunto de filmes com durações dadas, por forma a minimizar o número de cassetes gastas.
Dados Tempo de gravação de uma cassete de vídeo (em minutos), número de filmes e duração de cada um. Exemplo

```
180         tempo de uma cassete
7           número de filmes
100         duração do filme 1 (em minutos)
120         duração do filme 2 (em minutos)
50          ...
160         ...
30          ...
20          ...
40          duração do filme 7 (em minutos)
```

Exemplo de resultado

```
Video 1:   1 (t=100)   3 (t= 50)   5 (t= 30)
Video 2:   2 (t=120)   6 (t= 20)   7 (t= 40)
Video 3:   4 (t=160)
```

A informação sobre os filmes é guardada num vector de estruturas ("`struct`") indexado pelo número do filme, tendo cada elemento do vector dois campos: a duração do filme (dada, em minutos) e o número da cassete em que, segundo o programa, esse filme deverá ser gravado.
Nota Este problema foi formulado em termos de "filmes" e "cassetes" mas poderia obviamente ter sido formulado de outros modos equivalentes, por exemplo em termos de "objectos" que devem ser empacotados em "malas". O nome técnico deste problema é "bin packing".

1. Defina estruturas de informação apropriadas para resolver este problema, incluindo as variáveis simples que vai utilizar.

2. Defina uma função com o protótipo "`void ler_dados(void);`" que lê os dados (no formato descrito no exemplo dado) da entrada padrão.

3. Defina uma função com o protótipo "`void resultado(void);`" que imprime o número de cassetes utilizadas bem como os filmes gravados em cada cassete (segundo o formato do exemplo apresentado).

4. Uma primeira estratégia muito simples ("first-fit") consiste em considerar os filmes um a um pela ordem que são dados, e gravar cada filme na primeira cassete em que cabe, ou numa

nova cassete se não couber em nenhuma. Suponhamos por exemplo, que num certo momento já foram utilizadas 5 cassetes e que os seus tempos livres são de respectivamente 60, 40, 100, 10 e 80 minutos. Se o filme seguinte tem a duração de 70 minutos, é gravado na cassete 3 (tempo livre 100); se tivesse a duração de 120 minutos, seria gravado numa nova cassete (e passaria a haver 6 cassetes utilizadas). Para a implementação desta estratégia

(a) Defina estruturas de informação apropriadas.

(b) Escreva uma função com o protótipo `void colocaff(int nfilme);` que "grava" o filme número "`nfilme`" na cassete correspondente ao "first-fit".

(c) Complete o seu programa, definindo estruturas de informação, escrevendo o "`main`", etc..

5. Outra estratégia ("best-fit") consiste em considerar os filmes um a um pela ordem que são dados e gravar cada filme na cassete com o menor tempo livre onde o filme cabe, ou numa nova cassete se não couber em nenhuma. Para o exemplo da alínea anterior, um filme de 35 minutos seria gravado na cassete 2 (a qual ficaria com 5 minutos livres). Escreva uma função com o protótipo "`void colocabf(int nfilme);`" que "grava" o filme número "`nfilme`" na cassete correspondente ao "best-fit".

6. Mostre que nem "first-fit" nem o "best-fit" são estratégias óptimas, através de um exemplo em que existe uma gravação em menos cassetes do que a que resulta da aplicação dessas estratégias.

7. Implemente um programa que determina uma solução óptima. Para isso, escreva uma função (recursiva) com o protótipo "`void optimo(int filme, int usadas);`" onde "`nfilme`" é o filme a colocar numa cassete, e "`usadas`" é o número de cassetes correntemente utilizadas. A chamada de "`main()`" deve pois ser "`optimo(1,0);`". Pode utilizar a seguinte ideia:

```
void optimo(int nfilme, int usadas){
1)if usadas>=melhor ==> return
    /* Na solução que estamos a construir já gastamos mais cassetes
       do que na melhor solução já encontrada                     */
2)if nfilme>nf ==> Guarda a solução  /* É uma solução melhor!     */
                   return
3)Para cada possível cassete c, 1 <=c <= usadas
   colocamos "nfilme" em c:
      grava nfilme na cassete c
      chama (recursivamente) optimo(nfilme+1,usadas)
      retira nfilme da cassete c
  Tenta a colocação numa cassete vazia:
      grava nfilme na cassete usadas+1
      chama (recursivamente) optimo(nfilme+1,usadas+1)
      retira nfilme da cassete usadas+1
}
```