Programação Estruturada

Exercícios sobre recursividade

1999/2000

Pretende-se implementar um programa que liste todos os subconjuntos de $\{1, 2, \cdots, n\}$ em que n é um valor dado. Por exemplo, para $n = 3$, o resultado deve ser (a ordem não é importante)

```
{ }
{ 1 }
{ 2 }
{ 3 }
{ 1 2 }
{ 1 3 }
{ 2 3 }
{ 1 2 3 }
```

Pretendem-se fazer 2 implementações diferentes, utilizando ambas um "vector de bits", `boolean bit[ ]` cujo significado é o seguinte: `bit[i]` tem o valor 1 se o elemento `i` pertence ao subconjunto que se está a construir e 0 caso contrário.

1. Utilize uma contagem binária com o vector `bit[ ]`, começando em 0 e terminando num valor apropriado; por exemplo, o conjunto $\{2, 4, 5\}$ é representado por

   ```
     i    0   1   2   3   4   5   6
        ----------------------------...
   v[i] |   | 0 | 1 | 0 | 1 | 1 | 0 ...    {2,4,5}    (inteiro 26)
        ----------------------------...

    Conjunto seguinte:
     i    0   1   2   3   4   5   6
        ----------------------------...
   v[i] |   | 1 | 1 | 0 | 1 | 1 | 0 ...    {1,2,4,5}  (inteiro 27)
        ----------------------------...
   ```

   Implemente em particular a função de protótipo `void soma1(void);` que incrementa de uma unidade o inteiro representado no vector global `bit[ ]`.

2. Utilize uma função recursiva com o protótipo "`void cria(int i, int n);`". A chamada inicial é "`void cria(0,n);`". A função gera todos os subconjuntos com a parte "`bit[1..i]`" já fixada. Quando `i` $\geq$ `n` temos um subconjunto gerado. Senão, fazendo primeiro `bit[i+1]=0` e chamamos `cria(i+1,n)` e depois `bit[i+1]=1` e chamamos `cria(i+1,n);`