

## ep04 (25min)

I (10 val.) Considere o algoritmo de ordenação `sort` descrito abaixo:

```
1 typedef int Item;
2 #define key(A) (A)
3 #define less(A, B) (key(A) < key(B))
4
5 void sort(Item a[], int left, int right) {
6     int i, j;
7
8     for (i = right-1; i >= left; i--) {
9         Item v = a[i];
10        j = i+1;
11        while (j <= right && less(a[j], v)) {
12            a[j-1] = a[j];
13            j++;
14        }
15        a[j-1] = v;
16    }
17 }
```

Supondo a chamada da função `sort(a, 1, 8)` e que o vector `a` tem o conteúdo indicado abaixo, indique qual o conteúdo do vector `a` após as cinco primeiras iterações do ciclo `for`.

`a = { 9, 3, 27, 4, 8, 16, 13, 18, 14, 15 }`

Nota: A chamada `sort(a, 1, 8)` não irá ordenar todos os elementos do vector `a`.

II (10 val.) Considere a seguinte função em linguagem C, onde  $n$  denota o número de elementos no vector `a`.

```
1 int myfun(int a[], int n, int m) {
2     int i, j, count = 0;
3
4     j = 1;
5     while (j < n*n) {
6         for (i = 0; i < n; i++) {
7             if (a[i] == j+m)
8                 count++;
9         }
10        j = j + 2;
11    }
12
13    return count;
14 }
```

Indique e **justifique**, em função de  $n$  e/ou  $m$ , a complexidade assintótica de uma análise de pior caso da função `myfun`.