Algoritmos de Ordenação

Profa. Sheila Morais de Almeida

DAINF-UTFPR-PG

junho - 2018

Este material é preparado usando como referências os textos dos seguintes livros.

Análise de Algoritmos

Vamos analisar o Algoritmo Ordenação por Inserção.

Quantas operações de comparação são executadas?

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

```
Para i de 2 a n faça:
    j ← i − 1
    Enquanto j > 0 && v[i] < v[j] faça:
        j ← j − 1
    t ← v[i]
    Para k de i − 1 a j + 1 faça:
        v[k + 1] ← v[k]
    v[j + 1] ← t
```

Análise de Algoritmos

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso;
- de pior caso;
- e de caso médio.

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

Para i de 2 a n **faça:**
$j \leftarrow i - 1$
Enquanto $j > 0$ && $v[i] < v[j]$ **faça:**
$j \leftarrow j - 1$
$t \leftarrow v[i]$
Para k de $i - 1$ a $j + 1$ **faça:**
$v[k + 1] \leftarrow v[k]$
$v[j + 1] \leftarrow t$

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso: vetor em ordem crescente.
- de pior caso: vetor em ordem decrescente.
- e de caso médio: quando $v[i]$ é considerado, cada número de $v[0]$ a $v[i-1]$ tem 50% de chance de ser maior que $v[i]$ e 50% de chance de ser menor que $v[i]$. Para um exemplo, podemos supor:
  - os números em posições de 0 a $\lfloor \frac{i}{2} \rfloor$ são menores que $v[i]$ e
  - os números em posições de $\lfloor \frac{i}{2} \rfloor + 1$ a $i-1$ são maiores que $v[i]$.

Para o Algoritmo de Ordenação por Inserção, apresente uma instância

- de melhor caso: vetor em ordem crescente.
- de pior caso: vetor em ordem decrescente.
- e de caso médio: quando $v[i]$ é considerado, cada número de $v[0]$ a $v[i-1]$ tem 50% de chance de ser maior que $v[i]$ e 50% de chance de ser menor que $v[i]$. Para um exemplo, podemos supor:
  - os números em posições de 0 a $\lfloor \frac{i}{2} \rfloor$ são menores que $v[i]$ e
  - os números em posições de $\lfloor \frac{i}{2} \rfloor + 1$ a $i-1$ são maiores que $v[i]$.

  Ex.: 90 12 18 83 24 71 67 35 58 40.

Análise de complexidade de tempo no pior caso

Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

```
Para i de 2 a n faça:
    j ← i − 1
    Enquanto  j > 0 && v[i] < v[j] faça:
        j ← j − 1
    t ← v[i]
    Para k de i − 1 a j + 1 faça:
        v[k + 1] ← v[k]
    v[j + 1] ← t
```

Análise de complexidade de tempo no pior caso

Pergunta: Quantas instruções básicas do modelo computacional RAM (operações aritméticas básicas, atribuições e comparações) são executadas pelo Algoritmo de Ordenação por Inserção, considerando uma entrada de tamanho n de pior caso?

Análise de complexidade de tempo no pior caso

Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

Para i de 2 a n **faça:** $2 + 3(n-1) = 3n - 1$
  $j \leftarrow i - 1$ $2(n-1) = 2n - 2$
  Enquanto $j > 0$ && $v[i] < v[j]$ **faça:** $2\sum_{c=1}^{n-1} c + (n-1)^*$
    $j \leftarrow j - 1$ $2\sum_{c=1}^{n-1} c = n^2 - n$
  $t \leftarrow v[i]$ $n - 1$
  Para k de $i - 1$ a $j + 1$ **faça:** $4\frac{n(n-1)}{2} + 4(n-1)^\dagger$
    $v[k+1] \leftarrow v[k]$ $n(n-1) = n^2 - n$
  $v[j+1] \leftarrow t$ $2(n-1) = 2n - 2$

$^*2\sum_{c=1}^{n-1} c + (n-1) = n(n-1) + (n-1) = (n+1)(n-1) = n^2 - 1$

$^\dagger 4\frac{n(n-1)}{2} + 4(n-1) = 2n^2 + 2n - 4$

Análise de complexidade de tempo no pior caso

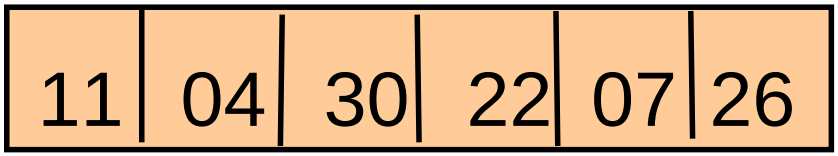
Considere o Algoritmo de Ordenação por Inserção:

Algoritmo Ordenação por Inserção
Entrada: vetor $v[1..n]$ de inteiros; n número de elementos no vetor

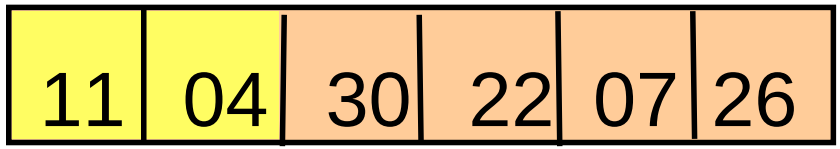
Para i de 2 a n **faça:** $2 + 3(n-1) = 3n - 1$
  $j \leftarrow i - 1$ $2(n-1) = 2n - 2$
  Enquanto $j > 0$ && $v[i] < v[j]$ **faça:** $n^2 - 1$
    $j \leftarrow j - 1$ $2\sum_{c=1}^{n-1} c = n^2 - n$
  $t \leftarrow v[i]$ $n - 1$
  Para k de $i - 1$ a $j + 1$ **faça:** $2n^2 + 2n - 4$
    $v[k+1] \leftarrow v[k]$ $n(n-1) = n^2 - n$
  $v[j+1] \leftarrow t$ $2(n-1) = 2n - 2$

Total: $5n^2 + 8n - 11$

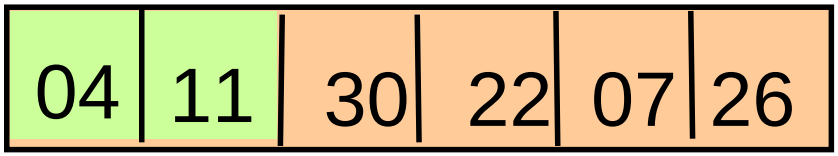
Bubble Sort



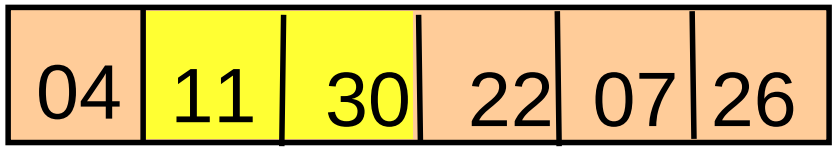
Bubble Sort



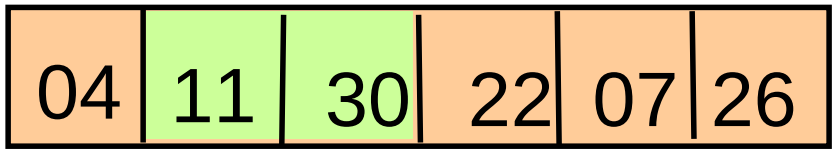
Bubble Sort



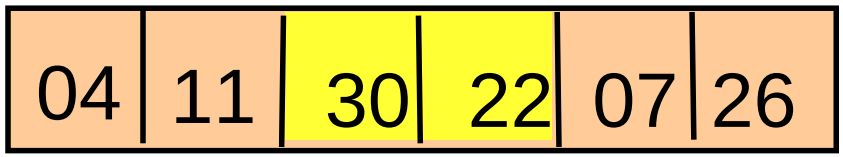
Bubble Sort



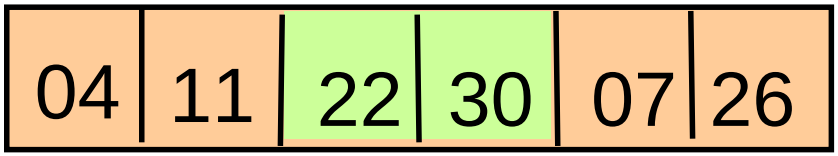
Bubble Sort



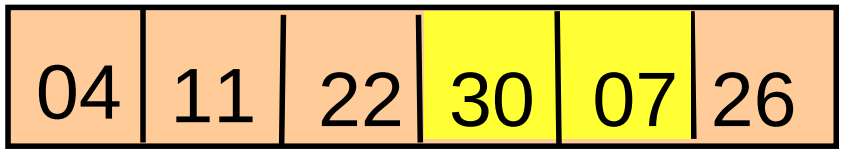
Bubble Sort



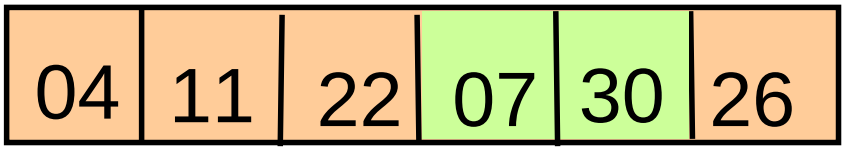
Bubble Sort



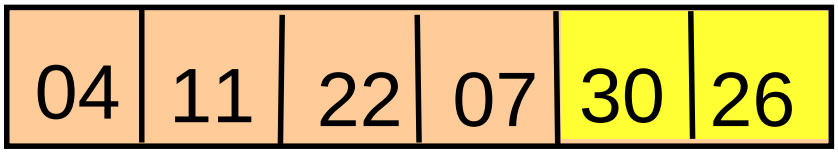
Bubble Sort



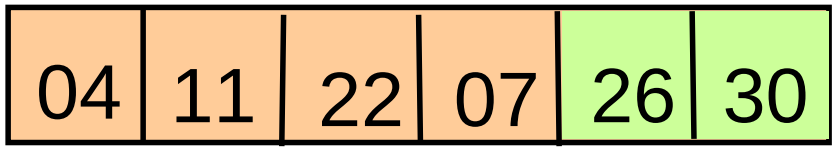
Bubble Sort



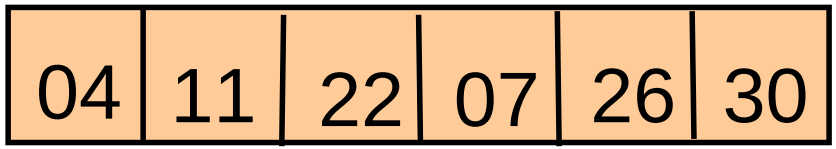
Bubble Sort



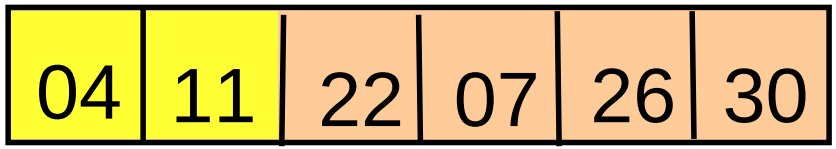
Bubble Sort



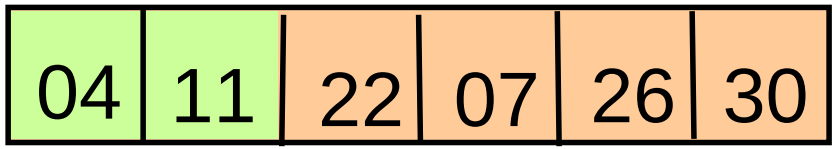
Bubble Sort



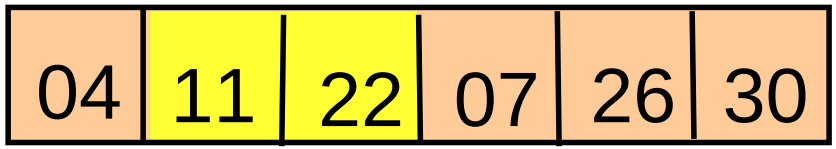
Bubble Sort



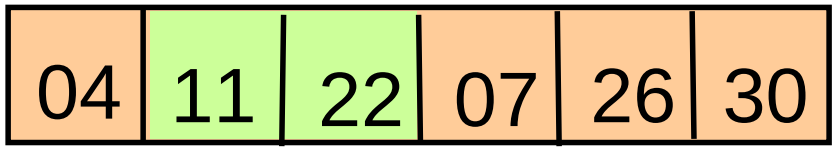
Bubble Sort



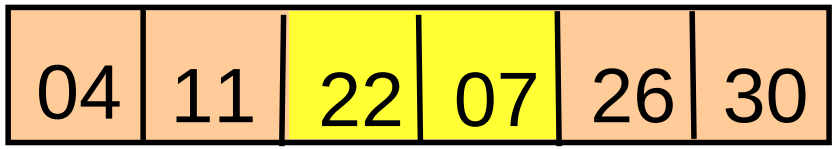
Bubble Sort



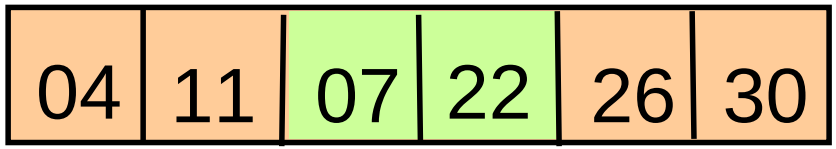
Bubble Sort



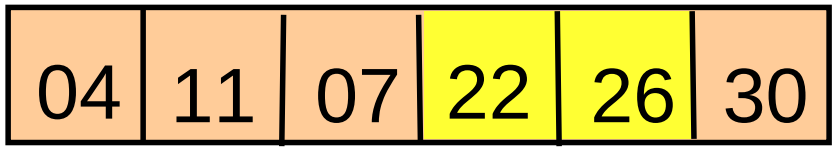
Bubble Sort



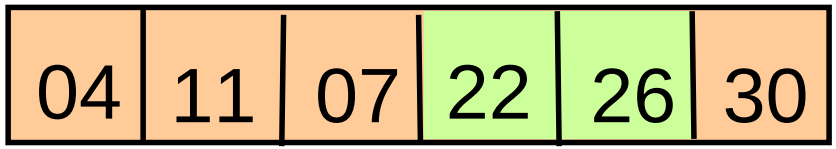
Bubble Sort



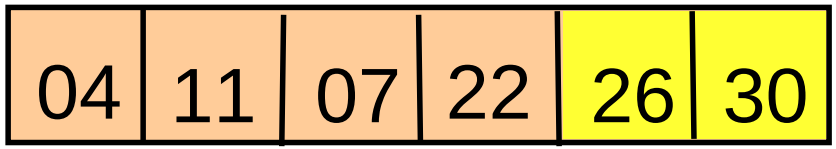
Bubble Sort



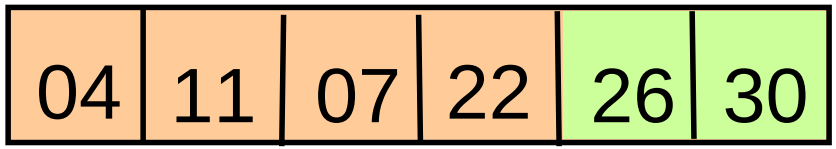
Bubble Sort



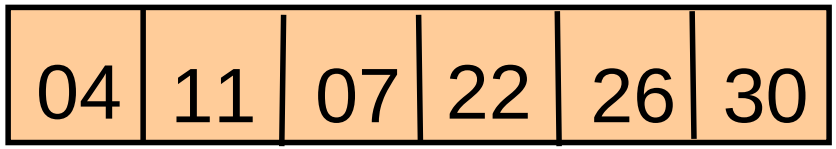
Bubble Sort



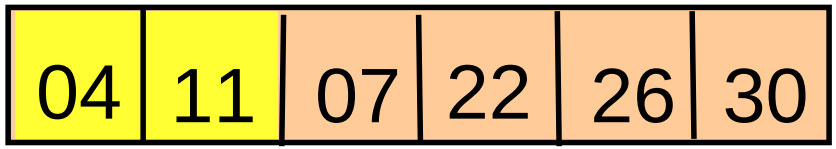
Bubble Sort



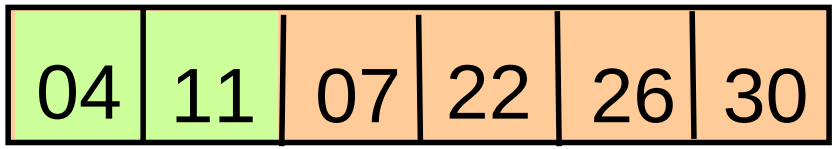
Bubble Sort



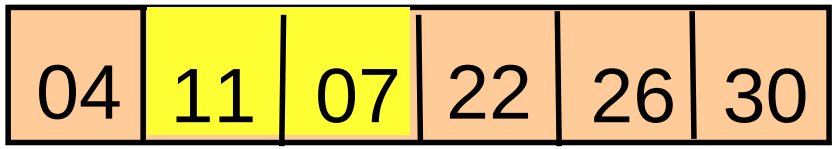
Bubble Sort



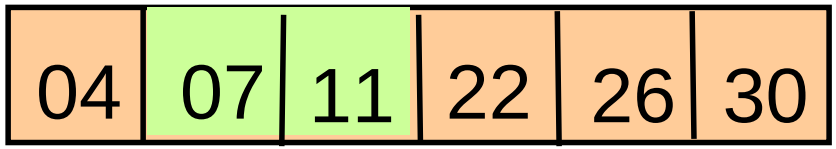
Bubble Sort



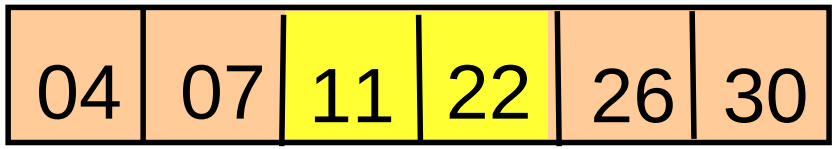
Bubble Sort



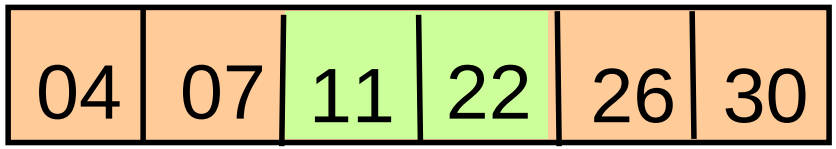
Bubble Sort



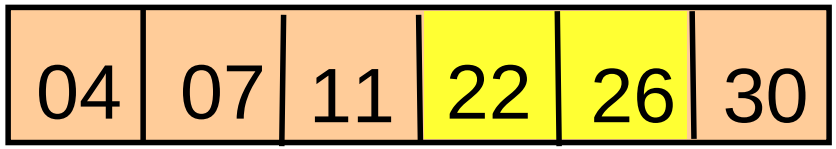
Bubble Sort



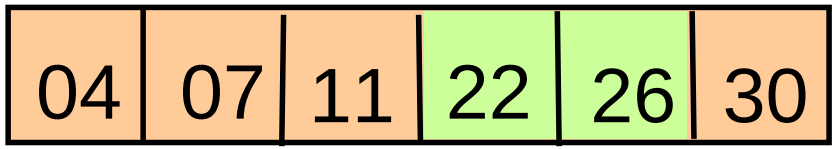
Bubble Sort



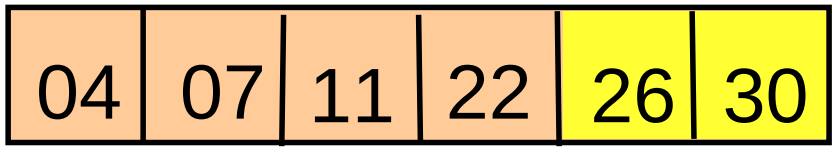
Bubble Sort



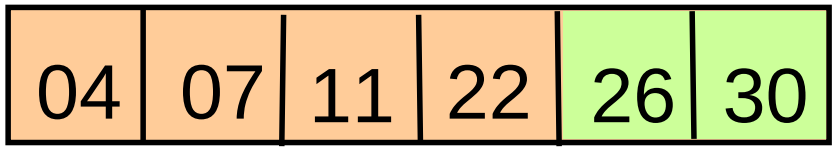
Bubble Sort



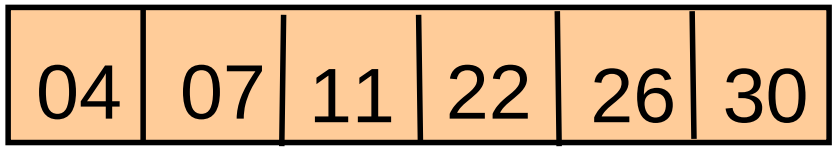
Bubble Sort



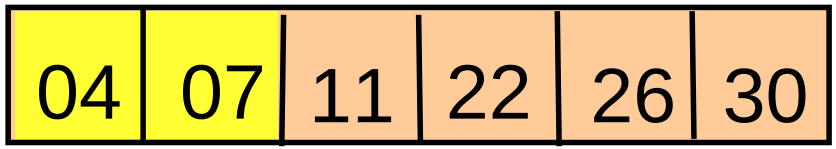
Bubble Sort



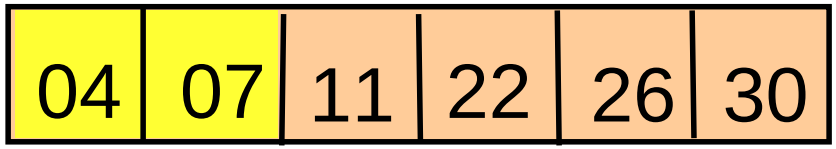
Bubble Sort



Bubble Sort



Bubble Sort

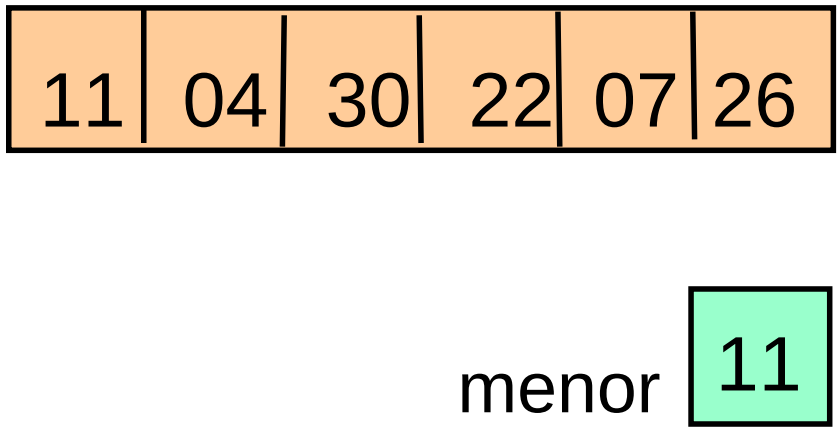


A cada iteração do laço mais externo do Bubble Sort, um número é colocado na posição correta, do maior para o menor.

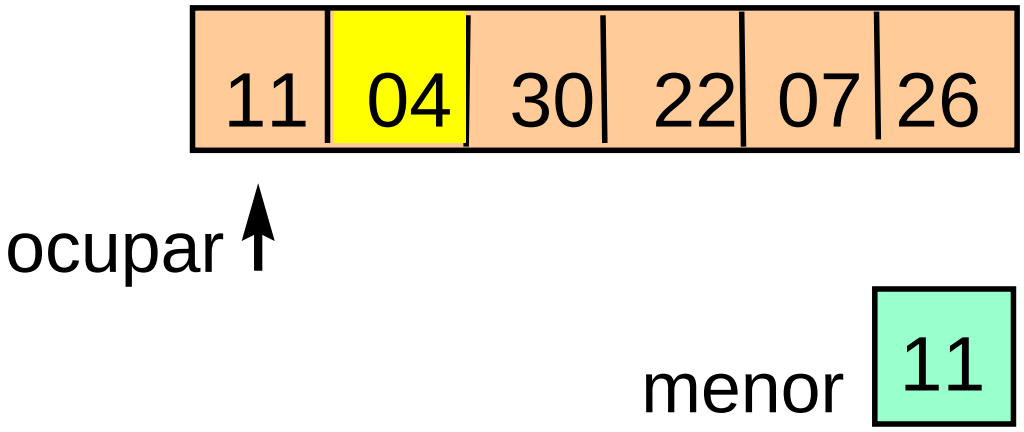
Então, seu laço externo executa $n - 1$ iterações, para colocar todos os números em suas devidas posições.

Se não houver nenhum tratamento, o Buble Sort continua executando até que seu laço de repetição externo complete $n - 1$ iterações.

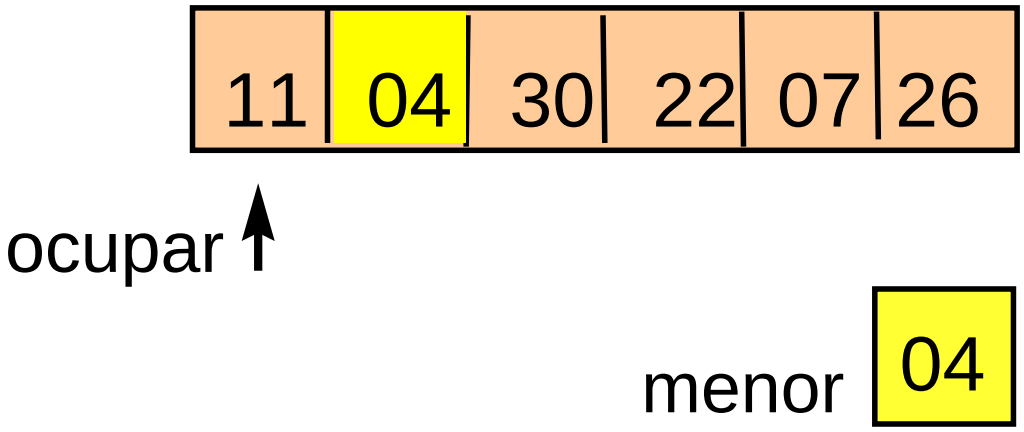
Selection Sort



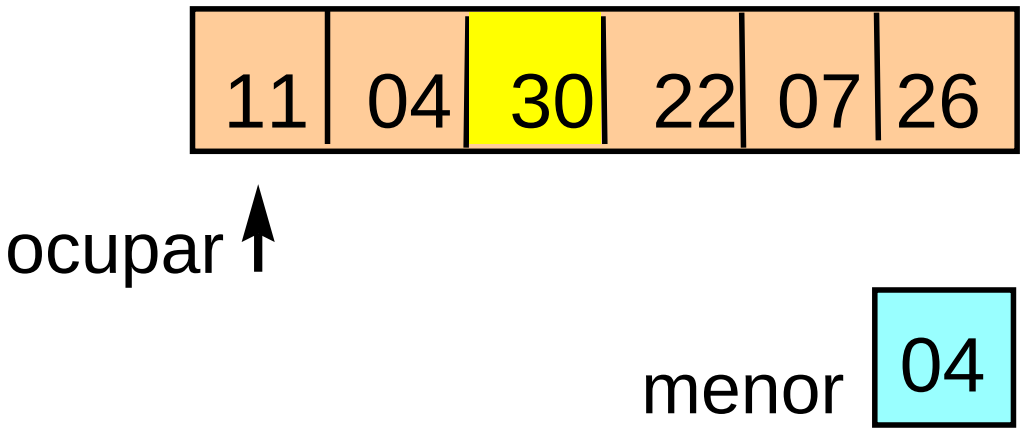
Selection Sort



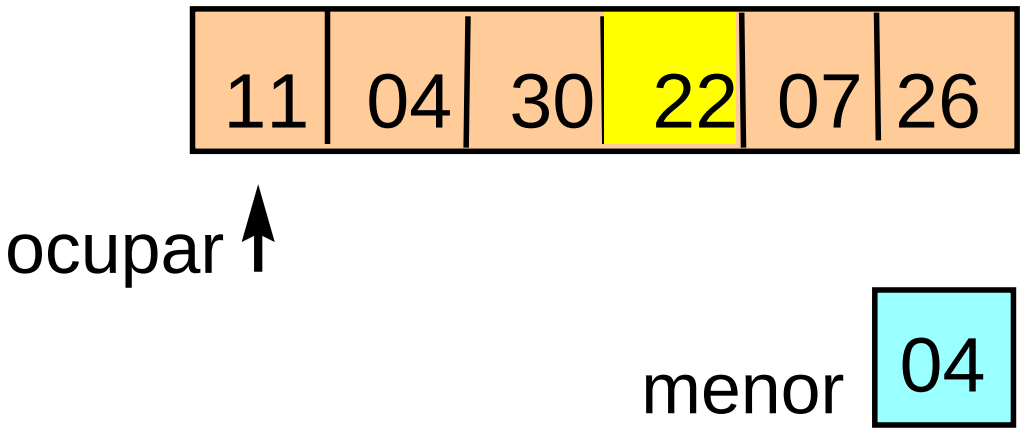
Selection Sort



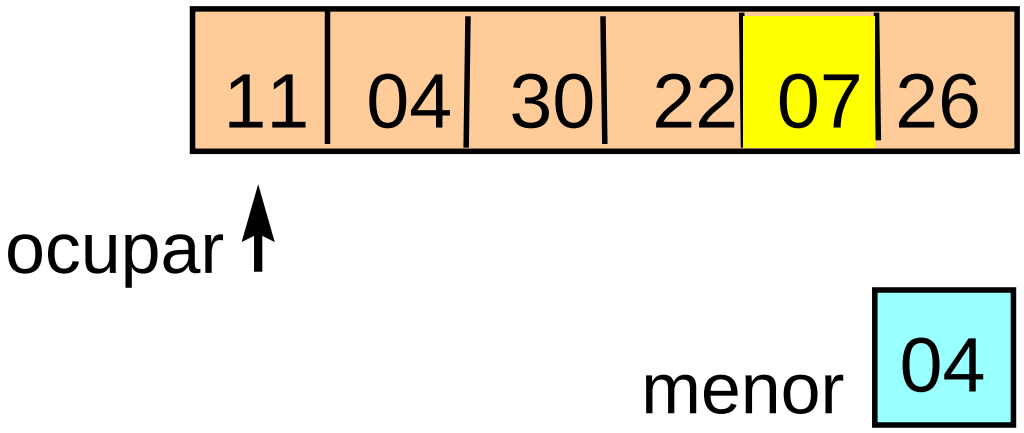
Selection Sort



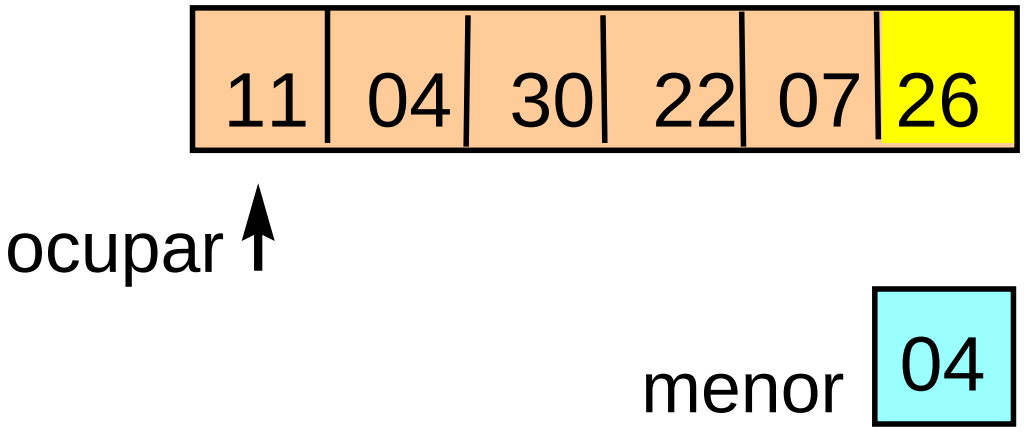
Selection Sort



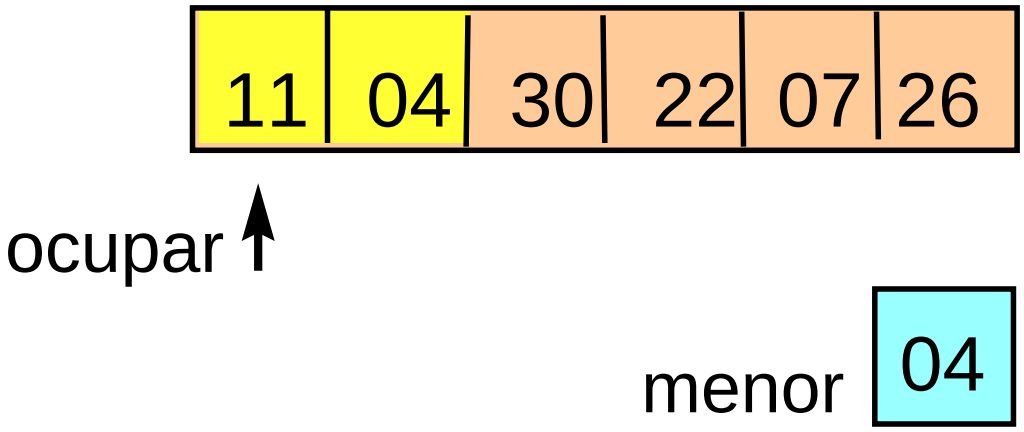
Selection Sort



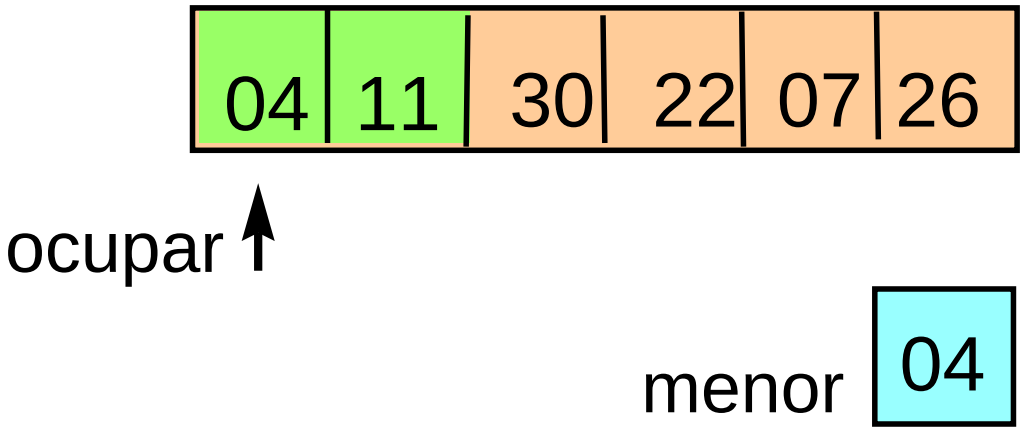
Selection Sort



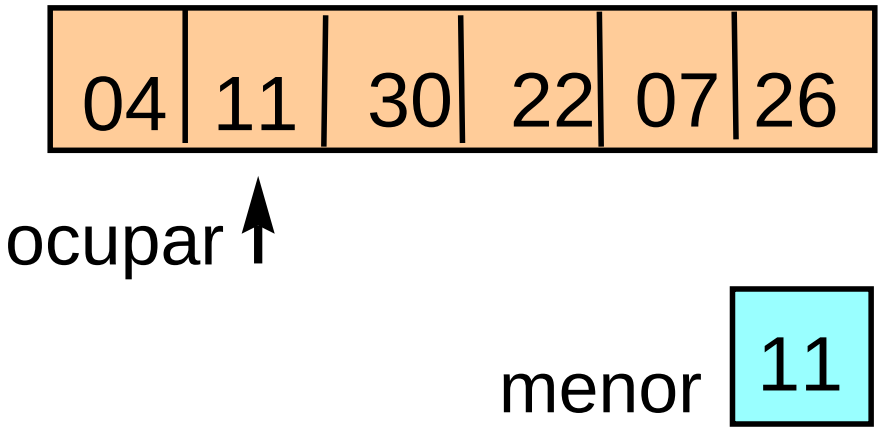
Selection Sort



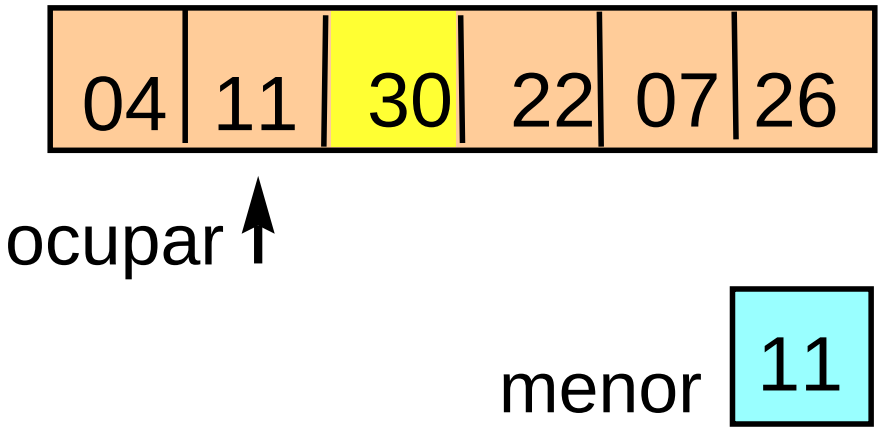
Selection Sort



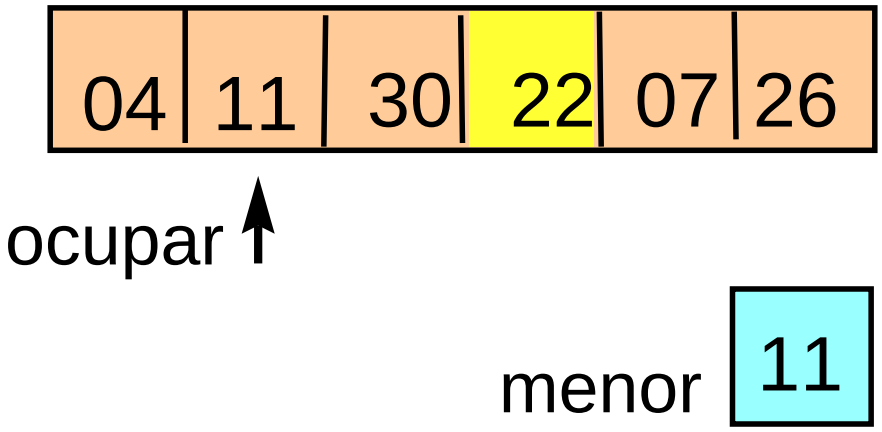
Selection Sort



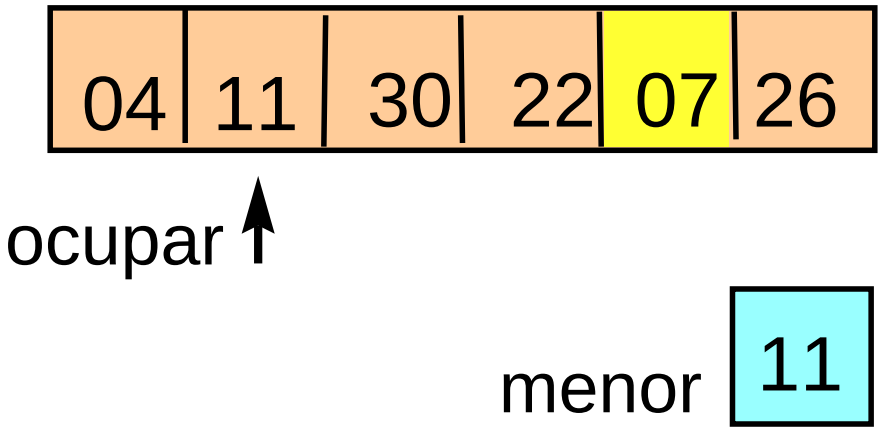
Selection Sort



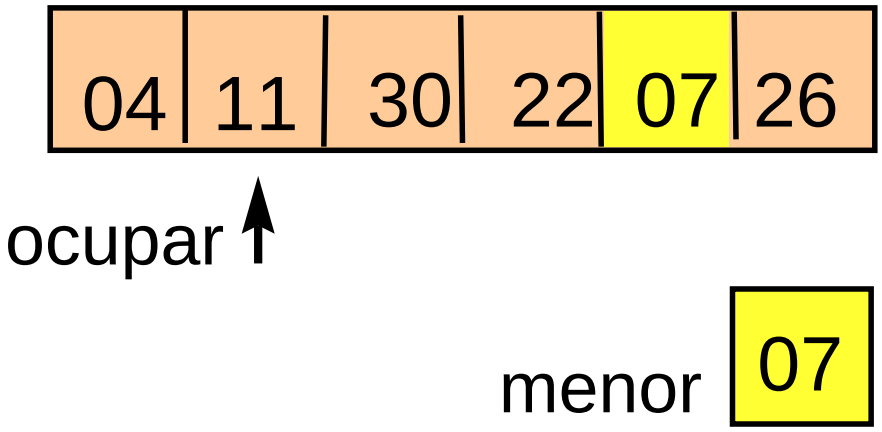
Selection Sort



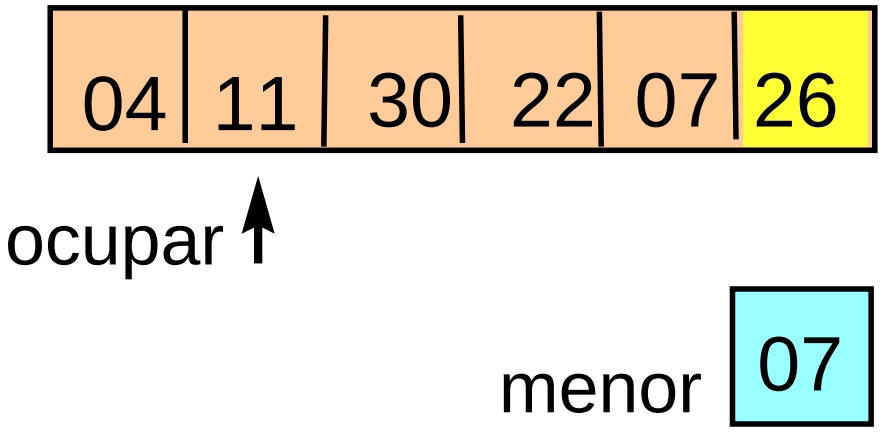
Selection Sort



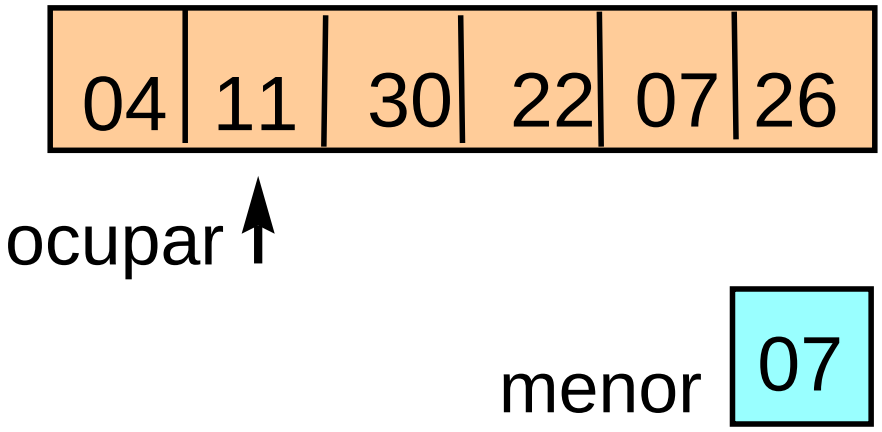
Selection Sort



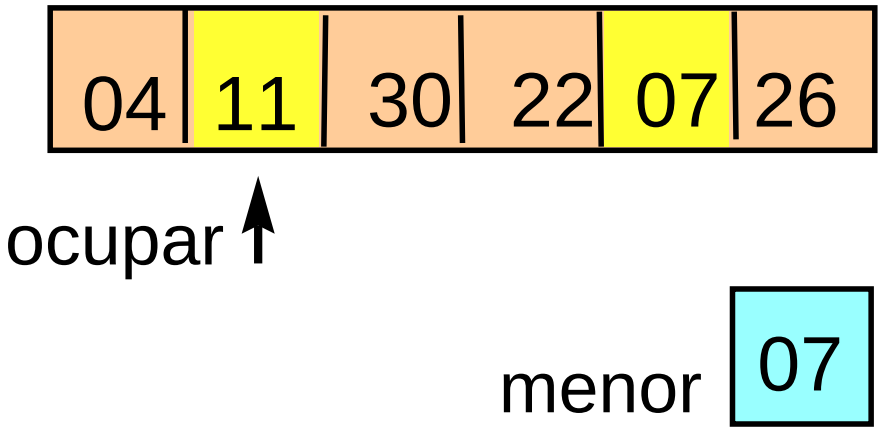
Selection Sort



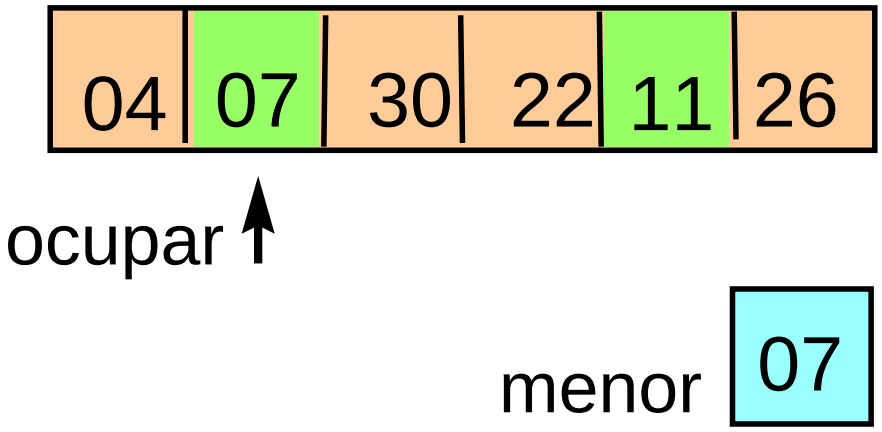
Selection Sort



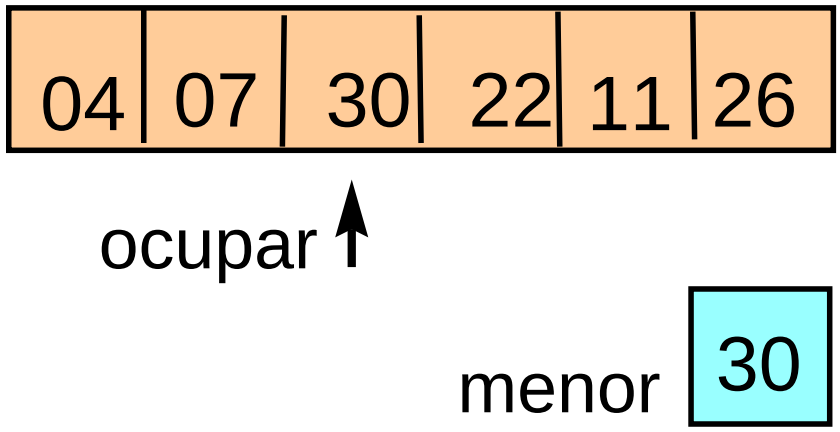
Selection Sort



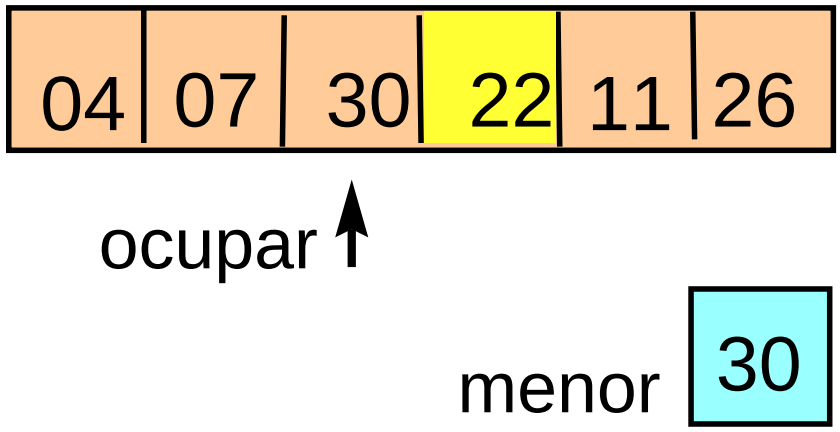
Selection Sort



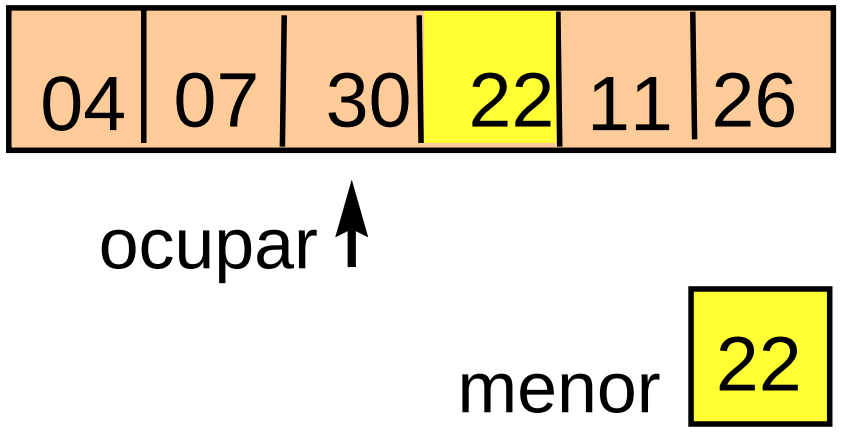
Selection Sort



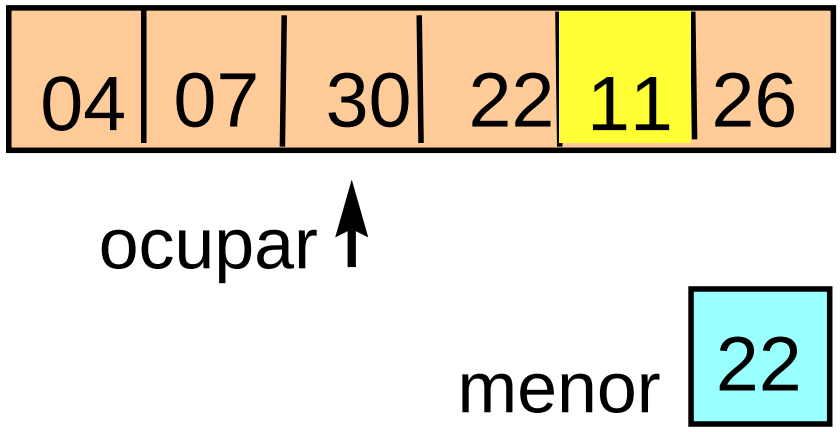
Selection Sort



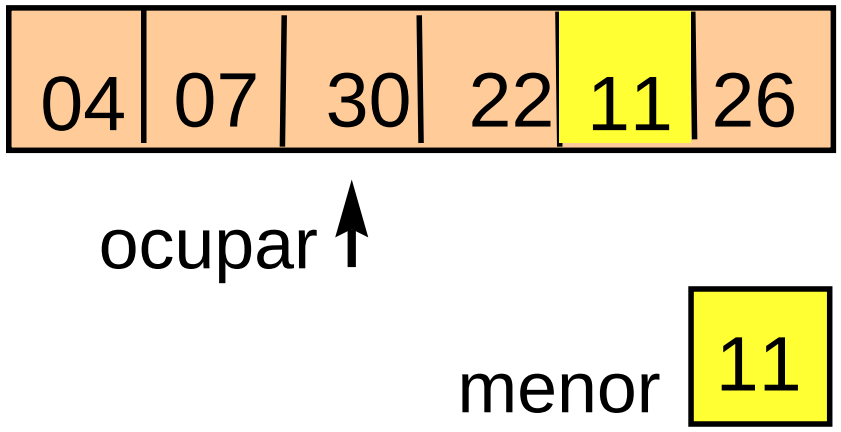
Selection Sort



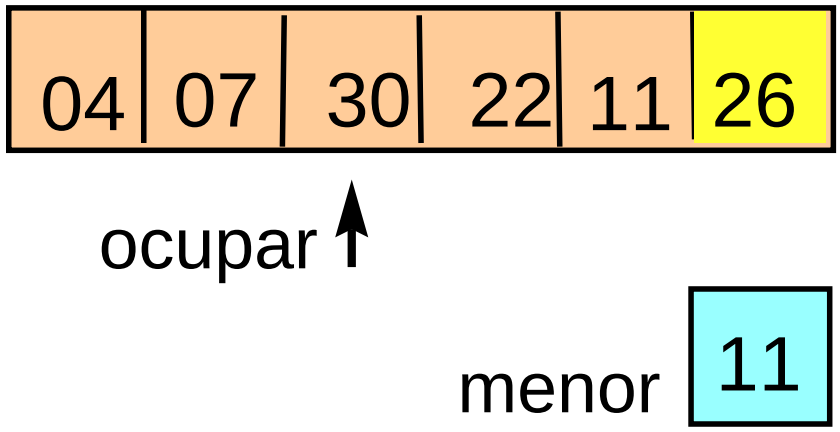
Selection Sort



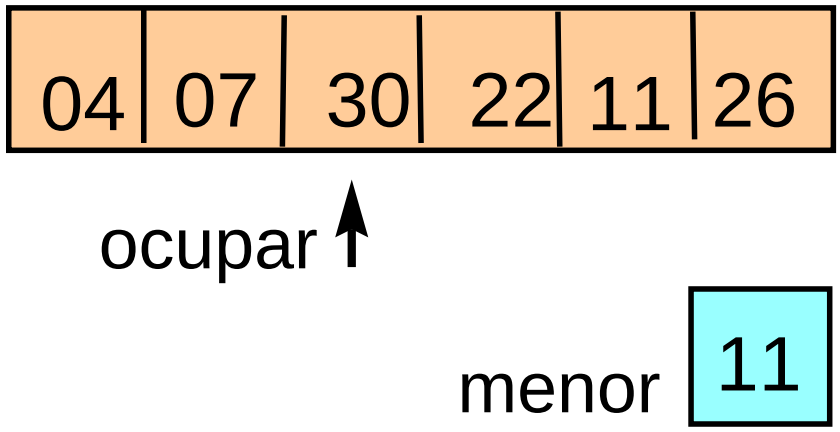
Selection Sort



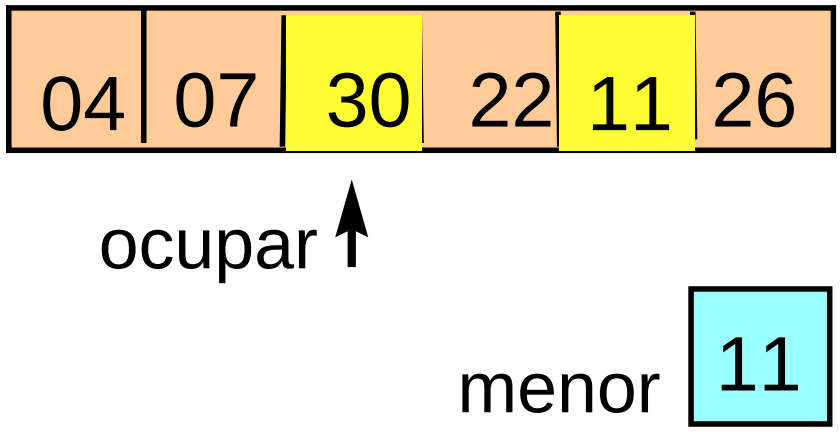
Selection Sort



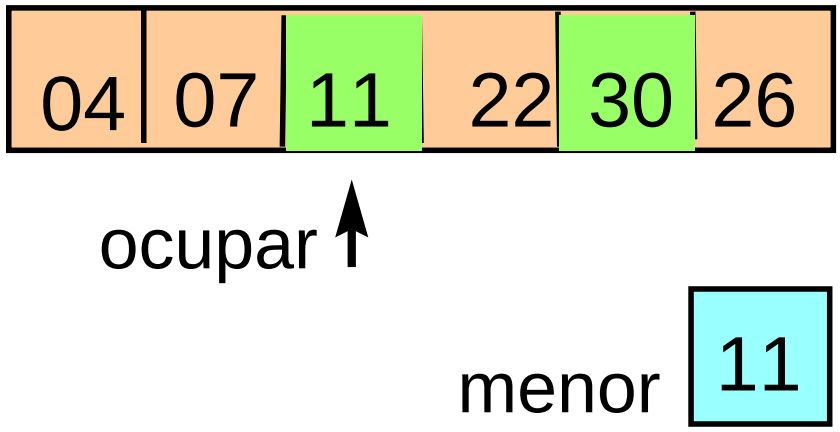
Selection Sort



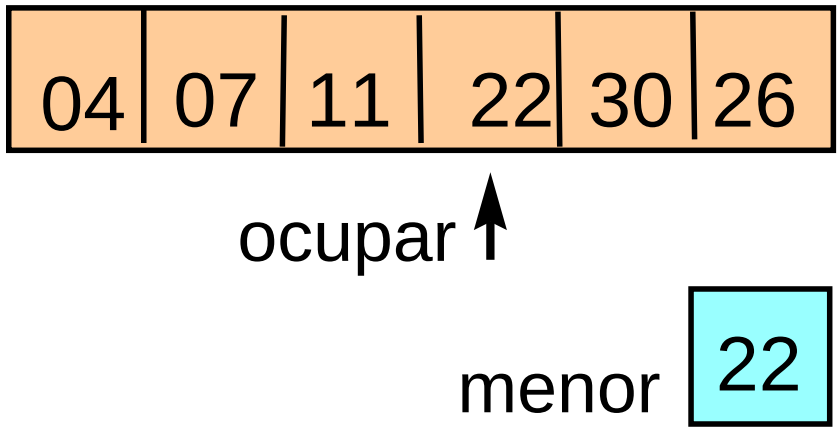
Selection Sort



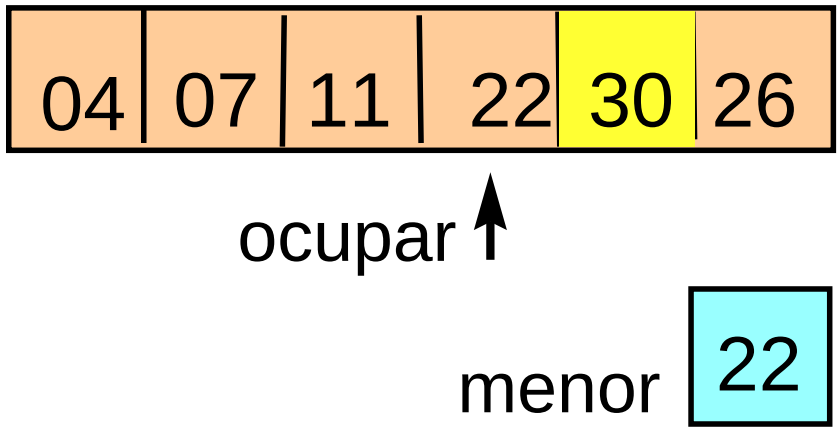
Selection Sort



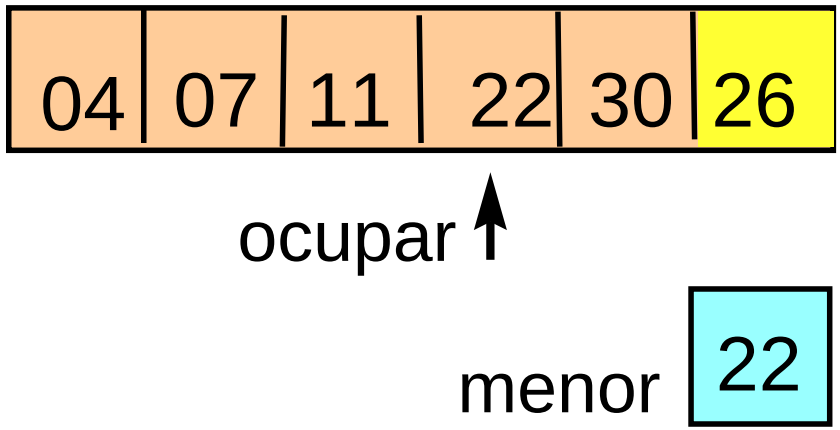
Selection Sort



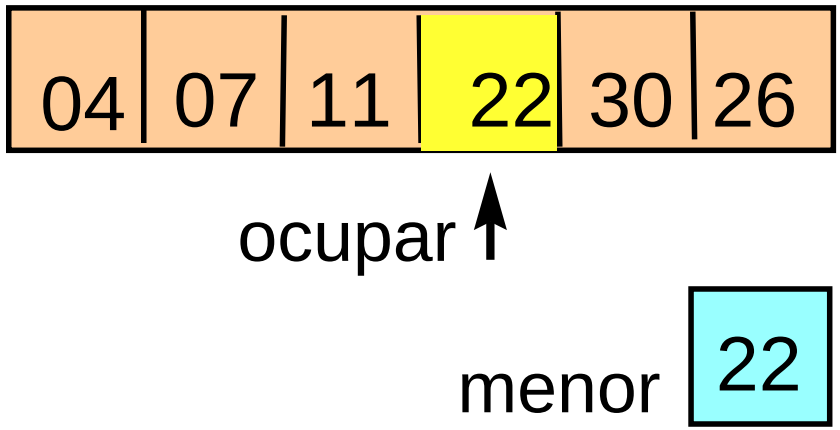
Selection Sort



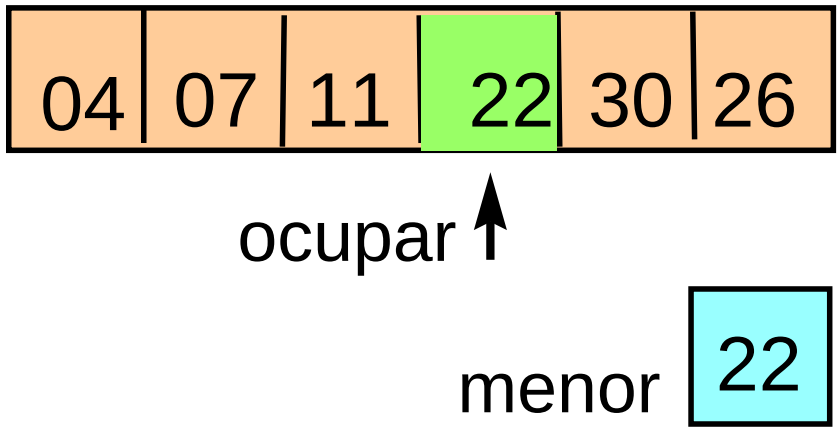
Selection Sort



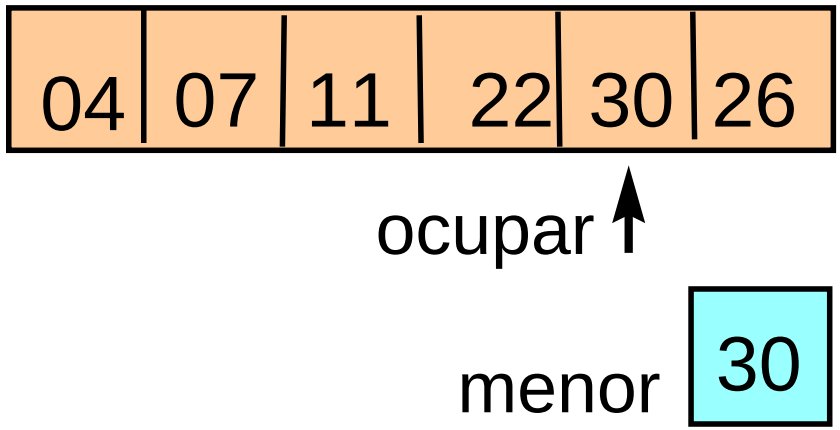
Selection Sort



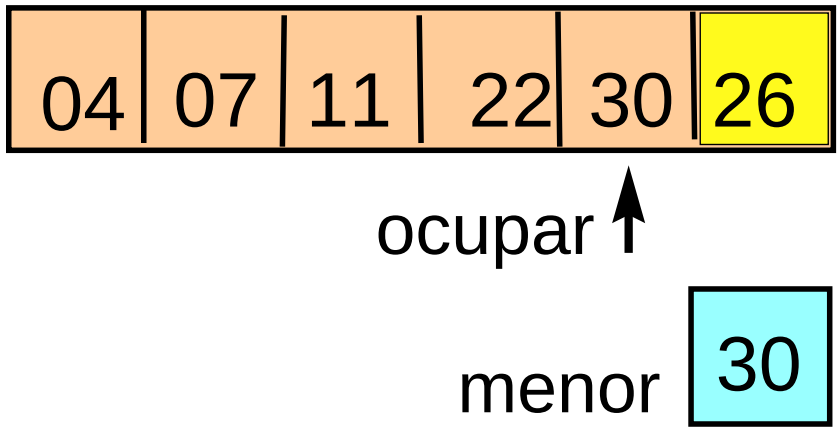
Selection Sort



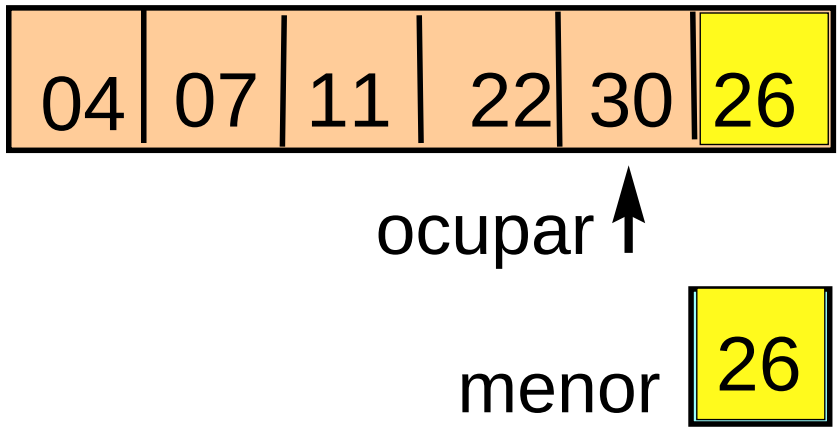
Selection Sort



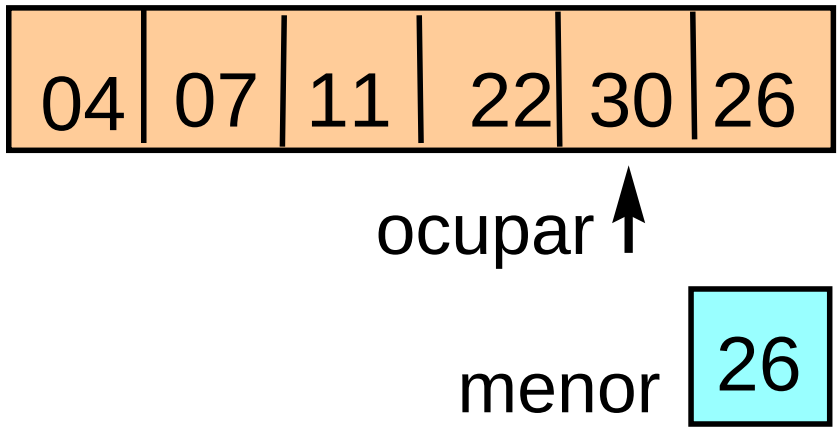
Selection Sort



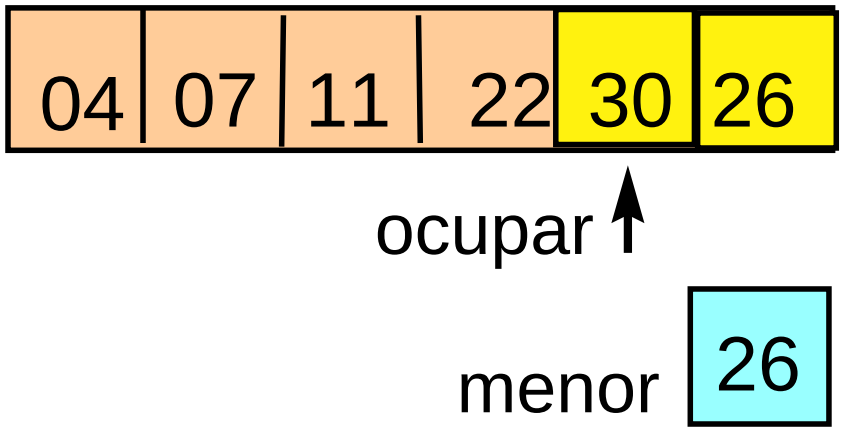
Selection Sort



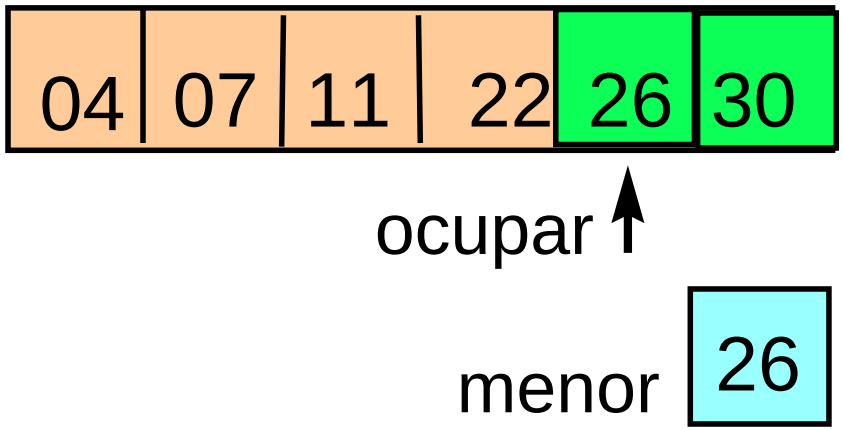
Selection Sort



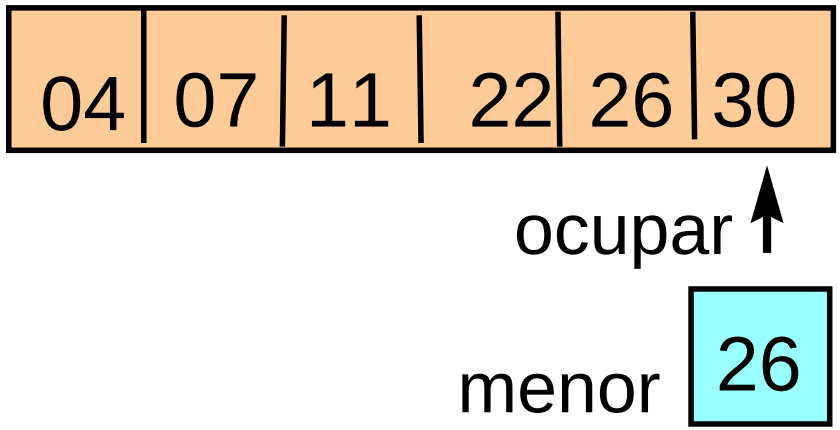
Selection Sort



Selection Sort



Selection Sort



Selection Sort