

BubbleSort:

Baseado na **troca de elementos adjacentes**.

Cada iteração leva o elemento maior para o final do vetor.

Bubble sort

```
bubbleSort( A : list of sortable items )
  swapped = true;
  n = length(A)
  while swapped {
    swapped = false;
    for each i in 1 to n-1{
      if(A[i] > A[i+1]){
        swap(A[i],A[i+1])
        swapped = true
      }
    }
  }
}
```

Pior caso

i=1 → n-1
i=2 → n-2
i=3 → n-3
...
i=n-1 → 1

$$\begin{array}{r} n + n + \dots + n - (1+2+3+\dots) \\ \hline n^2 - S_n \\ \hline n^2 - n(n+1)/2 \\ \hline (n^2 - 1)/2 = \Theta(n^2) \end{array}$$

Melhor caso é na ordem de n ; mas vai iterar na ordem de n vezes.

SelectionSort:

Baseado na seleção do maior e seu posicionamento no final.

Primeiro seleciona, depois leva para a posição correta.

Selection sort

```
SELECTION-SORT(A)
  n = length(A)
  for i ← 1 to n {
    min ← i
    for j ← (i + 1) to n {
      if A[j] < A[min] {
        min ← j
      }
    }
    swap (A[i], A[min])
  }
}
```

(n - 1)
(n - 2)
(n - 3)
...
(n - n)

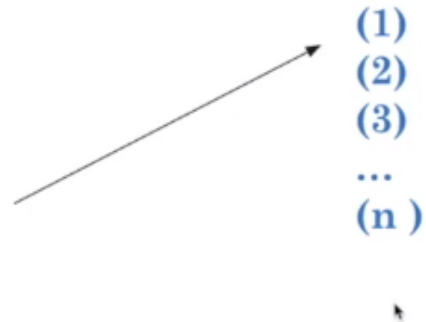
$$\begin{array}{r} n^2 - S_n \\ \hline (n^2 - 1)/2 = \Theta(n^2) \end{array}$$

InsertionSort:

Baseado na ideia de inserir um elemento em uma lista ordenada.

Insertion sort

```
INSERTION-SORT(A, n)
  for j ← 2 to n do
    key ← A[j]
    i ← j-1
    while i > 0 and A[i] > key do
      A[i+1] ← A[i]
      i ← i-1
    A[i+1] = key
```



Comparação

Algoritmo	melhor tempo	tempo médio	pior tempo	espaço
Insertion sort	$O(n)$	$O(n^2)$	$O(n^2)$	in-place
Selection sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	in-place
Bubble sort	$O(n)$	$O(n^2)$	$O(n^2)$	in-place

BubbleSort muitas comparação e troca.

SelectionSort muitas comparações e menos trocas.

InsertionSort menos comparações (último da lista ordenada) e reduz a quantidade de troca (faz trocas suficientes para encaixar a posição do elemento na lista ordenada).

Questões de Implementação:

Como ordenar apenas uma faixa de um array ?



Questões de implementação

```
bubbleSort( A : list of Comparable items
           leftIndex, rightIndex )
    swapped = true;
    n = length(A)
    while swapped {
        swapped = false;
        for each i in leftIndex to rightIndex-1{
            if(A[i].compareTo(A[i+1])) > 0){
                swap(A[i],A[i+1])
                swapped = true
            }
        }
    }
}
```

Como ordenar apenas
uma faixa de um array?