



# **Processamento da Informação – Teoria –**

## **Algoritmos de ordenação (segunda parte)**

Semana 09  
Prof. Jesús P. Mena-Chalco

22/06/2013

## B. Ordenação por troca (Bubble sort)



O algoritmo de ordenação baseado em troca, consiste em intercalar pares de elementos que não estão em ordem até que não exista mais pares.

O princípio do **bolha** é a **troca de valores** entre posições **consecutivas** fazendo com que **os valores mais altos “borbulhem”** para o final da lista.

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 9 3 7 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 9 3 7 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 9 3 7 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 9 7 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 9 7 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 9 4 2



## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 9 4 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 9 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 9 2

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 9 3 7 4 2

5 6 3 7 4 2 9



Maior elemento

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 2 **9**

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 6 3 7 4 2 9



## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 7 4 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 7 4 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 7 4 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 4 7 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 4 7 2 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 4 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 4 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

5 3 6 4 2 7 9



## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 6 4 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 6 4 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 6 4 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 6 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 6 2 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 2 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 2 **6 7 9**

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 2 6 7 9



## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 5 4 2 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 5 2 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 5 2 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 2 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 2 **5 6 7 9**

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 2 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 4 2 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 2 4 5 6 7 9



## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 2 4 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

3 2 4 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

2 3 4 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

2 3 4 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

**2** 3 4 5 6 7 9

## B. Bubble sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

Após ser feita essa operação n vezes, a lista estará ordenada.

**2 3 4 5 6 7 9**

## B. Bubble sort

Crie uma função que permita ordenar de forma ascendente uma lista dada como entrada.

Use o algoritmo de ordenação bolha.

**Cabeçalho:** `def bubble_sort(L):`

## B. Bubble sort

```
def bubble_sort(L):  
    j = len(L)-1  
    while j>0:  
        for i in range(0,j):  
            if L[i]>L[i+1]:  
                L[i],L[i+1] = L[i+1],L[i]  
            j = j-1  
    return L
```



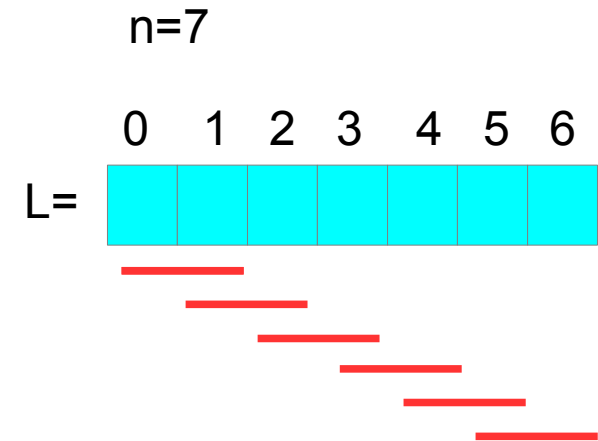
## B. Bubble sort

```
def bubble_sort(L):  
    j = len(L)-1  
    while j>0:  
        for i in range(0,j):  
            if L[i]>L[i+1]:  
                L[i],L[i+1] = L[i+1],L[i]  
            j = j-1  
    return L
```

```
>>> bubble_sort([9,8,7,6,5,4,3,2,1])  
[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

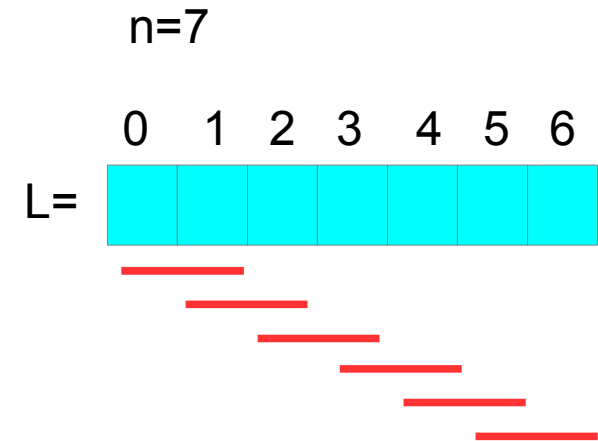
## B. Bubble sort (tempo)

```
def bubble_sort(L):  
    j = len(L)-1  
    while j>0:  
        for i in range(0,j):  
            if L[i]>L[i+1]:  
                L[i],L[i+1] = L[i+1],L[i]  
            j = j-1  
    return L
```



## B. Bubble sort (tempo)

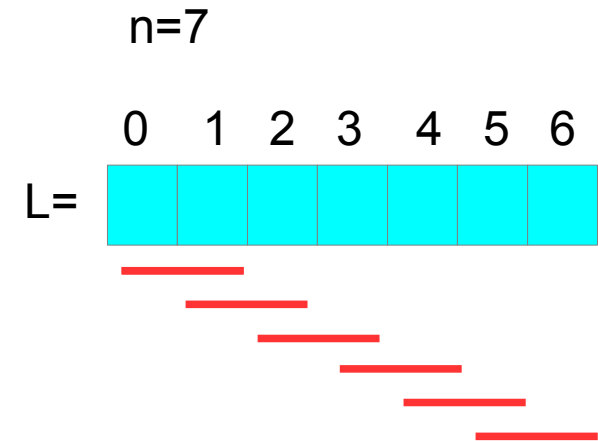
```
def bubble_sort(L):  
    j = len(L)-1  
    while j>0:  
        for i in range(0,j):  
            if L[i]>L[i+1]:  
                L[i],L[i+1] = L[i+1],L[i]  
            j = j-1  
    return L
```



	Comparações
j=n-1	n-1
j=n-2	n-2
j=n-3	n-3
...	
j=1	1

## B. Bubble sort (tempo)

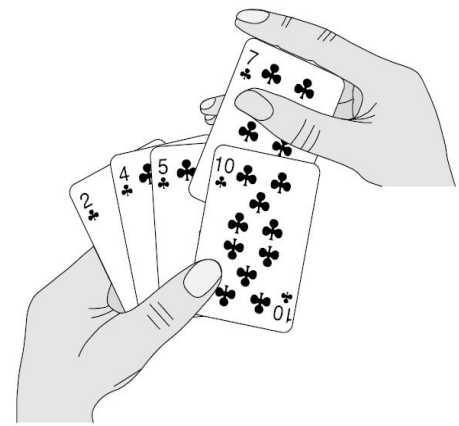
```
def bubble_sort(L):  
    j = len(L)-1  
    while j>0:  
        for i in range(0,j):  
            if L[i]>L[i+1]:  
                L[i],L[i+1] = L[i+1],L[i]  
            j = j-1  
    return L
```



	Comparações
j=n-1	n-1
j=n-2	n-2
j=n-3	n-3
...	
j=1	1

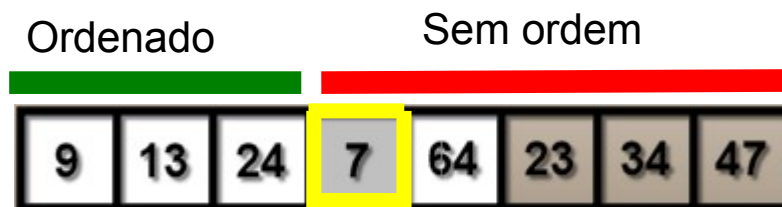
$$\text{Tempo} = (n-1)(n)/2$$
$$\text{Tempo} = n^2/2 - n/2$$

## C. Ordenação por inserção (Insertion sort)



A principal característica deste algoritmo consiste em ordenar a lista utilizando uma sublista ordenada em seu início.

A cada novo passo, acrescentamos a esta sublista mais um elemento até atingirmos o último elemento de um arranjo.



## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 6 9 3 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 6 9 3 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 6 9 3 7 4 2



Ordenado

Sem ordem



## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 6 9 3 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 6 3 9 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

5 3 6 9 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

A cada novo passo, acrescentamos a esta sublista mais um elemento.

3 5 6 9 7 4 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 6 9 7 4 2



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 6 7 9 4 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 6 7 9 4 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 6 7 4 9 2



Ordenado

Sem ordem



## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 6 4 7 9 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 5 4 6 7 9 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 5 6 7 9 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 5 6 7 9 2

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 5 6 7 2 9

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 5 6 2 7 9

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 5 2 6 7 9

Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

A cada novo passo, acrescentamos a esta sublista mais um elemento.

5 6 9 3 7 4 2

3 4 2 5 6 7 9

Ordenado

Sem ordem




## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

A cada novo passo, acrescentamos a esta sublista mais um elemento.

3 2 4 5 6 7 9



Ordenado

Sem ordem


## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

A cada novo passo, acrescentamos a esta sublista mais um elemento.

2 3 4 5 6 7 9



Ordenado

Sem ordem

## C. Insertion sort

A ideia é ir comparando elementos dois-a-dois e trocá-los em ordem.

5 6 9 3 7 4 2

A cada novo passo, acrescentamos a esta sublista mais um elemento.

2 3 4 5 6 7 9



Ordenado

## C. Insertion sort

Crie uma função que permita ordenar de forma ascendente uma lista dada como entrada.

Use o algoritmo de ordenação por inserção.

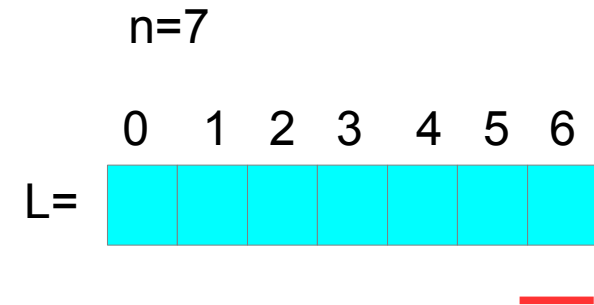
**Cabeçalho:** `def insertion_sort(L):`

## C. Insertion sort

```
def insertion_sort(L):  
    for i in range(1,len(L)):  
        j = i  
        while j >= 1 and L[j-1] > L[j]:  
            L[j-1], L[j] = L[j], L[j-1]  
            j = j-1  
    return L
```

# C. Insertion sort (tempo)

```
def insertion_sort(L):  
    for i in range(1, len(L)):  
        j = i  
        while j >= 1 and L[j-1] > L[j]:  
            L[j-1], L[j] = L[j], L[j-1]  
            j = j - 1  
    return L
```



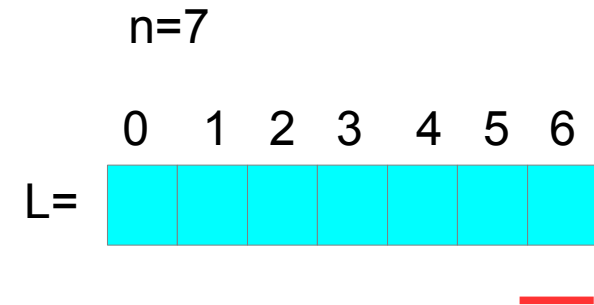
	Comparações
i=1	1
i=2	2
i=3	3
...	
i=n-1	n-1

$$\text{Tempo} = (n-1)(n)/2$$
$$\text{Tempo} = n^2/2 - n/2$$

# C. Insertion sort (tempo)

```
def insertion_sort(L):  
    for i in range(1, len(L)):  
        j = i  
        while j >= 1 and L[j-1] > L[j]:  
            L[j-1], L[j] = L[j], L[j-1]  
            j = j - 1  
    return L
```

Este algoritmo é o mais apropriado quando a lista estiver semi-ordenada.



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

$$\text{Tempo} = (n-1)(n)/2$$
$$\text{Tempo} = n^2/2 - n/2$$

# Atividade em aula

## Questão única:

Preencha a seguinte tabela com apenas o número de trocas que serão necessárias para ordenar as listas dadas como entrada.

<b>Lista</b>	<b>Bubble-sort</b>	<b>Selection-sort</b>	<b>Insertion-sort</b>
[2, 4, 6, 8, 10, 12]			
[12, 10, 8, 6, 4, 2]			
[5, 2, 8, 4, 6, 1]			
[1, 2, 3, 9, 8, 7]			



# Atividade em aula

## Questão única:

Preencha a seguinte tabela com apenas o número de trocas que serão necessárias para ordenar as listas dadas como entrada.

<b>Lista</b>	<b>Bubble-sort</b>	<b>Selection-sort</b>	<b>Insertion-sort</b>
[2, 4, 6, 8, 10, 12]	0	5	0
[12, 10, 8, 6, 4, 2]	15	5	15
[5, 2, 8, 4, 6, 1]	9	5	9
[1, 2, 3, 9, 8, 7]	3	5	3

# Lista 06

## Questão única.

- (a) Implemente o algoritmo de ordenação **Cocktail-sort** [1].
- (b) Explique brevemente a abordagem considerada para ordenar uma lista de tamanho  $n$ .
- (c) Apresente três exemplos de ordenação.
- (d) Preencha a seguinte tabela com o número de trocas necessárias para ordenar as listas.

Lista	Cocktail-sort
[2, 4, 6, 8, 10, 12]	
[12, 10, 8, 6, 4, 2]	
[5, 2, 8, 4, 6, 1]	
[1, 2, 3, 9, 8, 7]	

[1] [http://pt.wikipedia.org/wiki/Cocktail\\_sort](http://pt.wikipedia.org/wiki/Cocktail_sort)

# Lista 06

A entrega da Lista 06 será através do Tidia-ae: [Seção Atividades/lista-06](#). Até 29/06 (23h50) – Sábado.

Esta atividade pode ser realizada por grupos de **1, 2 ou 3 alunos**.

Apenas deve ser enviado um arquivo PDF contendo a solução das questões, nome completo e RA dos integrantes do grupo.

O documento deve ter o seguinte nome:  
RA-NomeCompleto-Lista-06.pdf

**OBS: Todos** os membros do grupo devem obrigatoriamente enviar o relatório em formato PDF.