



# Estrutura de Dados

Prof. Amarildo Lucena

# Métodos de Ordenação

## Tipos de Dados



# Ordenação

---

Ordenação é o processo de organizar dados de um mesmo tipo em uma ordem crescente ou decrescente. Existem, basicamente, quatro tipos de ordenação:

- ✓ Bubble sort;
- ✓ Insertion sort;
- ✓ Selection sort;
- ✓ Quick sort;



# Bubble Sort (Bolha)

---

A ordenação por bolha (bubble sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior.

5	7	6	4	2	3	1
---	---	---	---	---	---	---



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

5	7	6	4	2	3	1
---	---	---	---	---	---	---

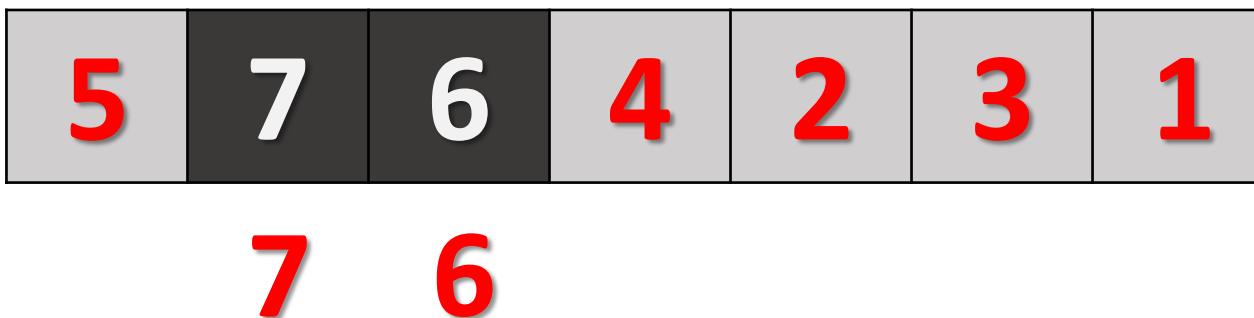
5 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



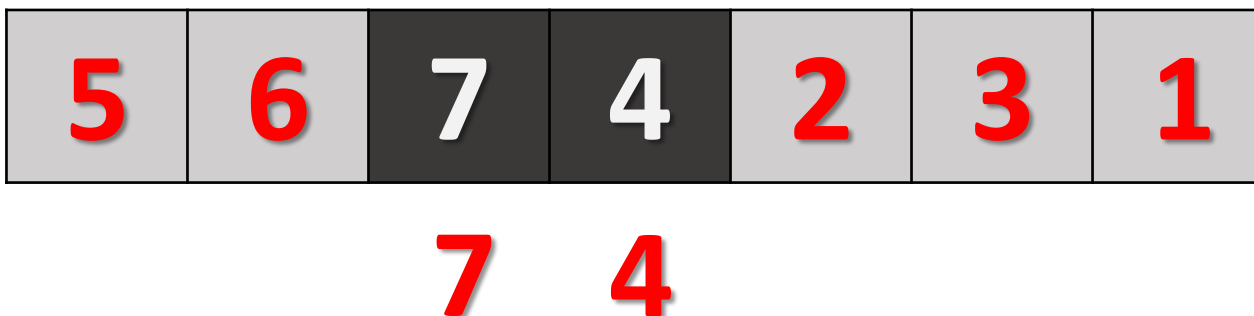
6 ↔ 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.







# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



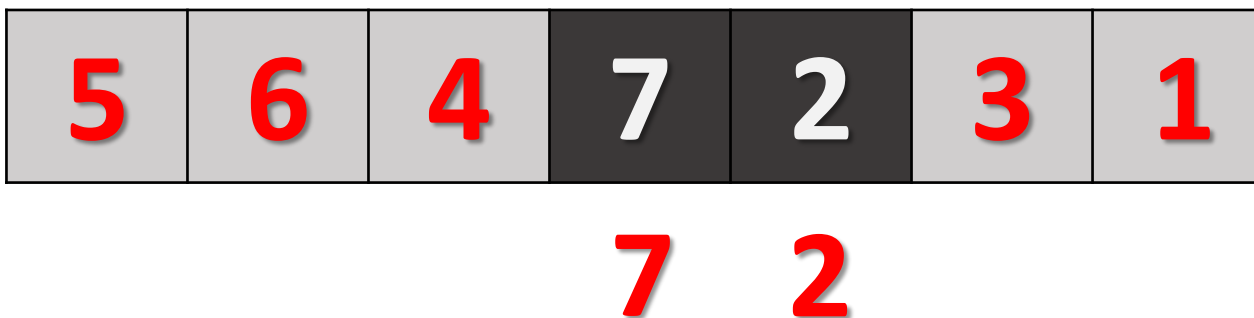
4 ↔ 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



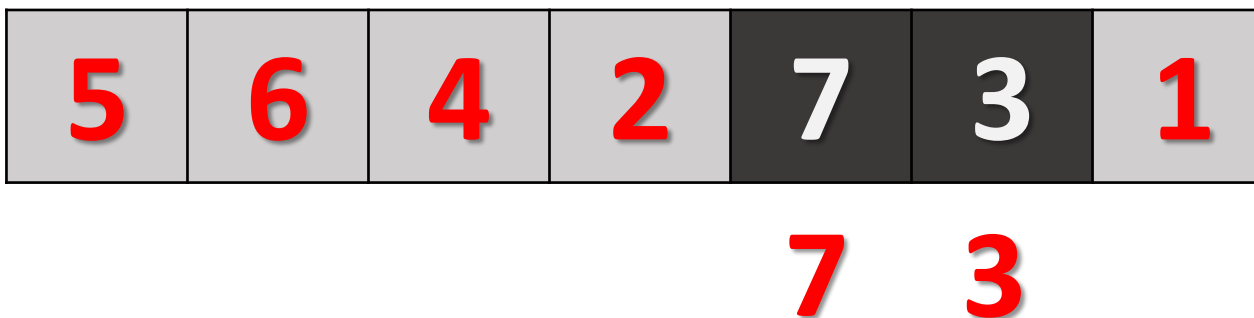
2 ↔ 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



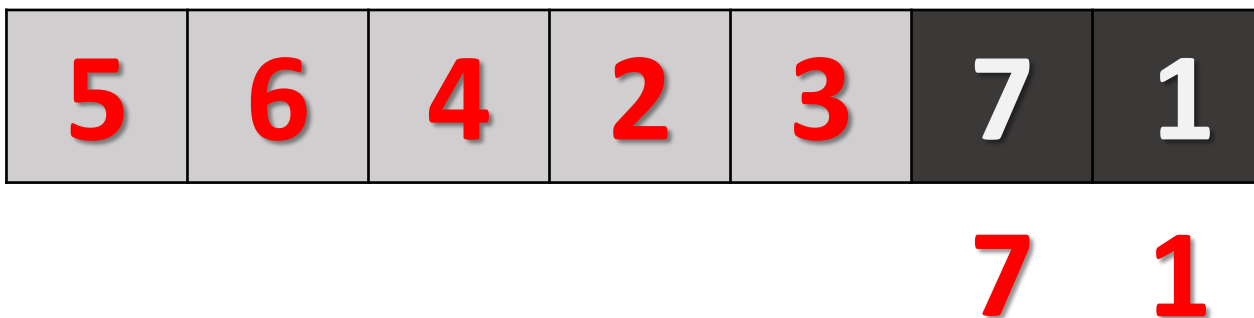
3 ↔ 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

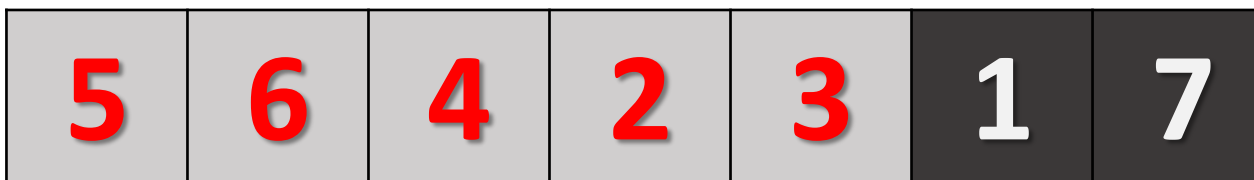




# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



1 ↔ 7



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



7





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



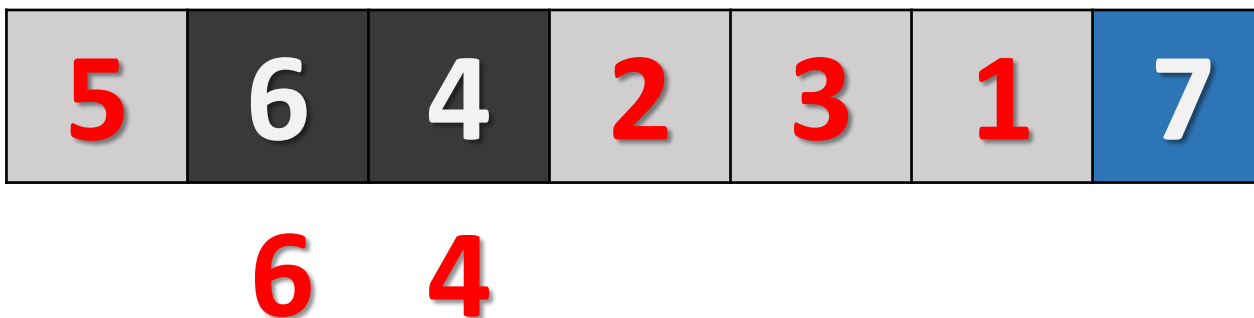
5 6



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



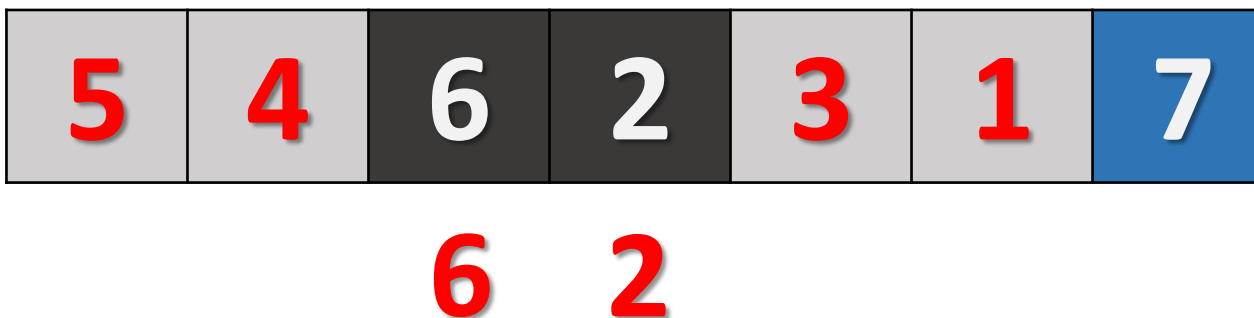
4 ↔ 6



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



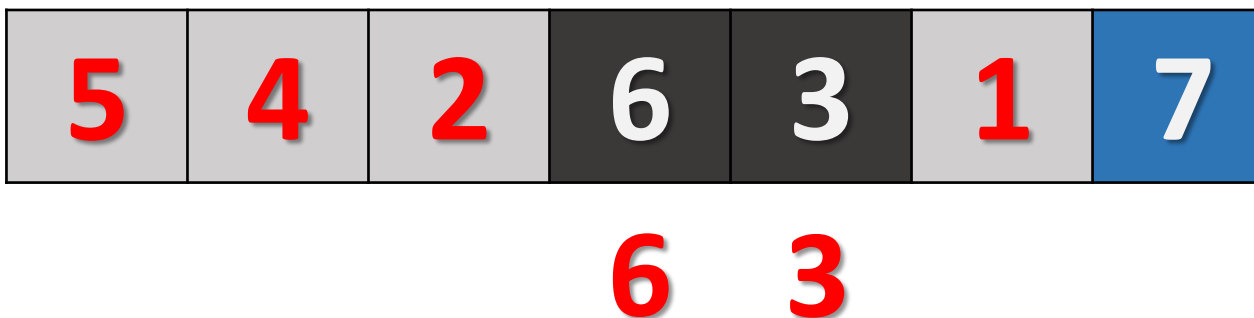
2 ↔ 6



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



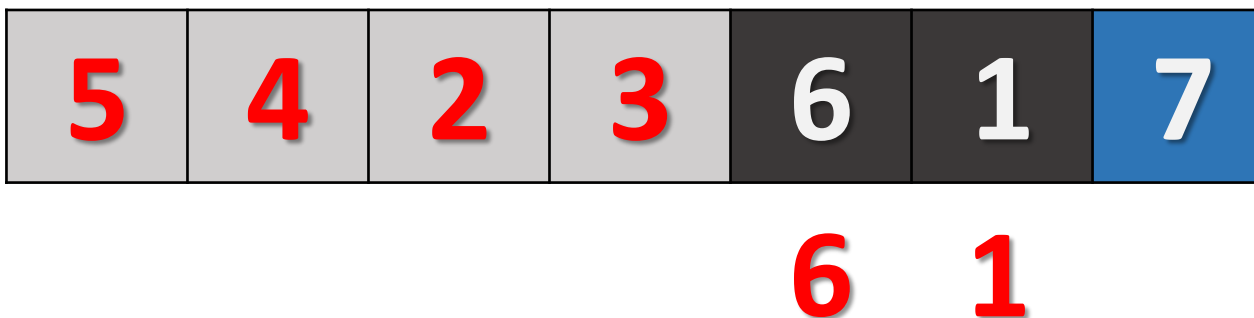
3 ↔ 6



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.







# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



1 ↔ 6



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

5	4	2	3	1	6	7
---	---	---	---	---	---	---



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



5 4



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



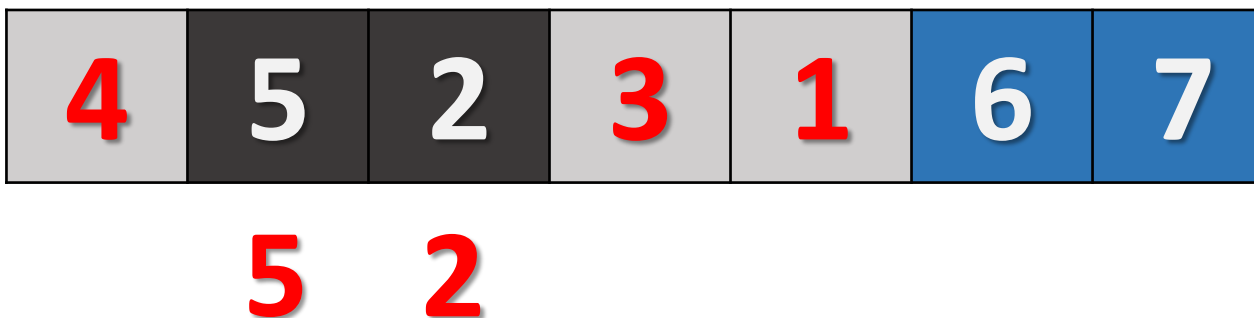
4 ↔ 5



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



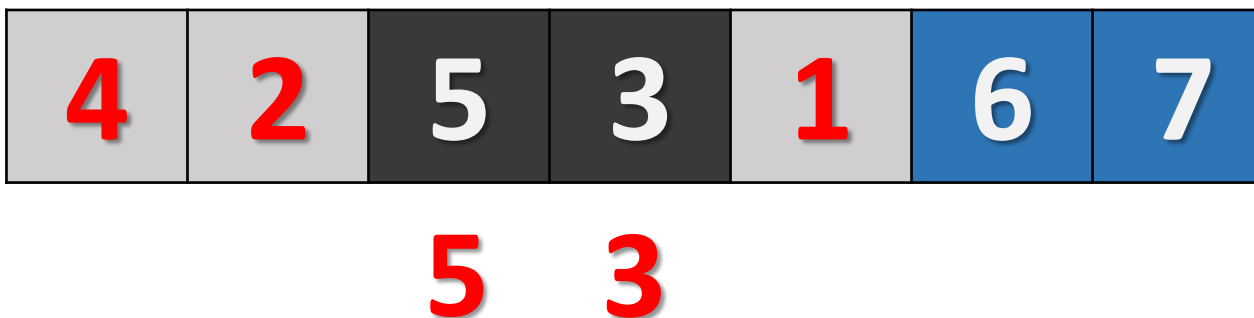
2 ↔ 5



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



3 ↔ 5

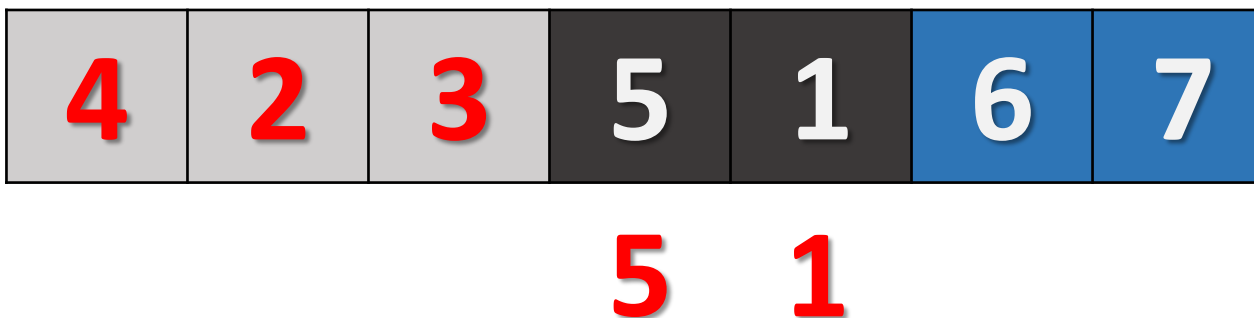




# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



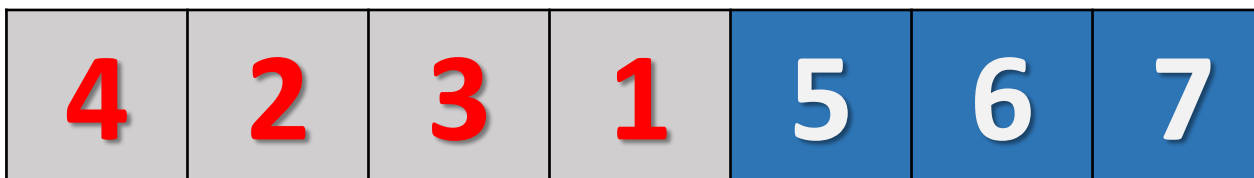
1 ↔ 5



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

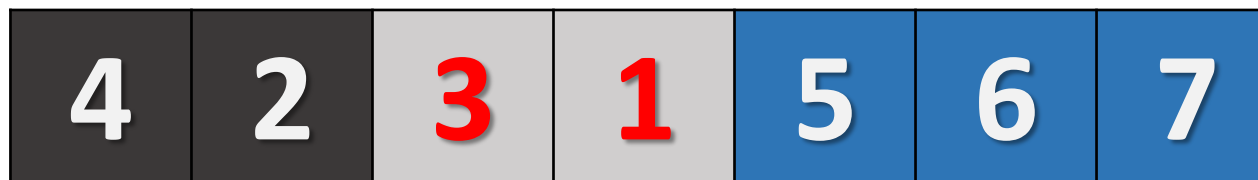




# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



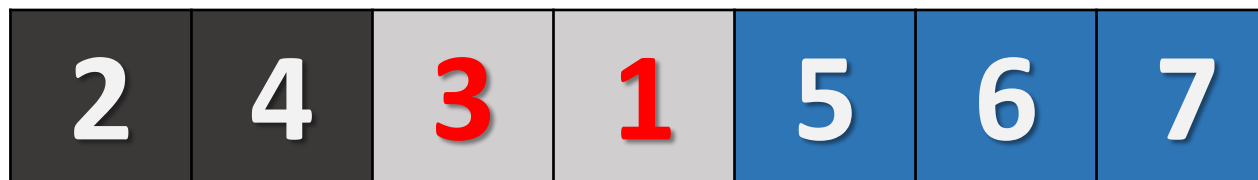
4 2



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



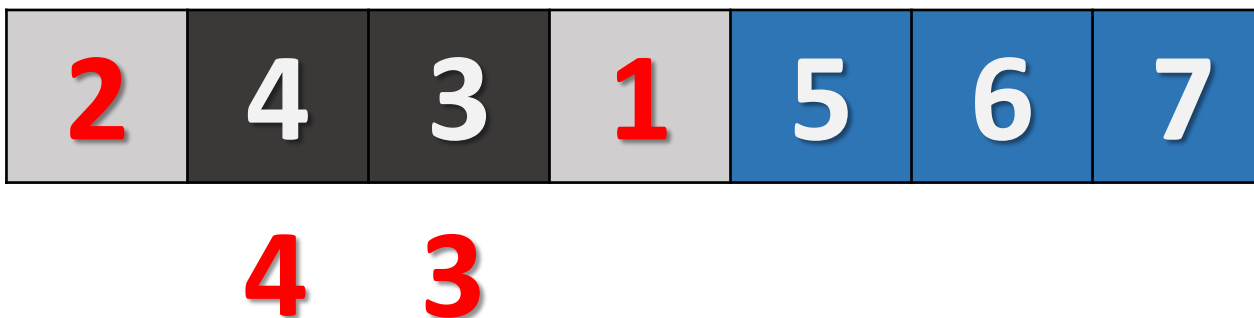
**2** ↔ **4**



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



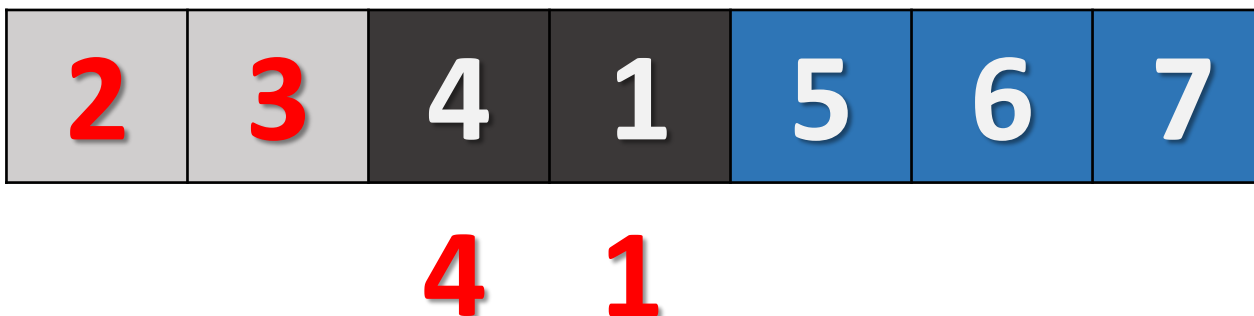
3 ↔ 4



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.







# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



1 ↔ 4



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



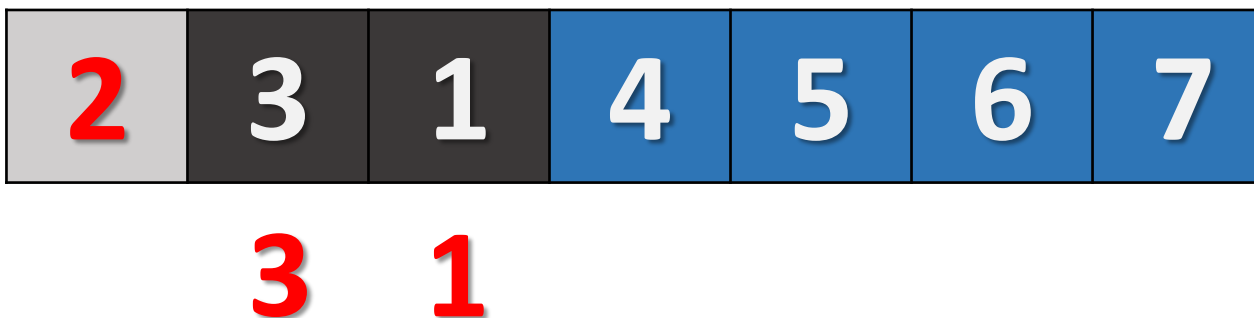
2 3



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

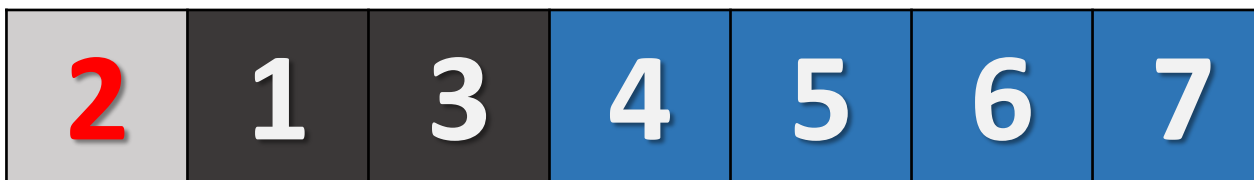




# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



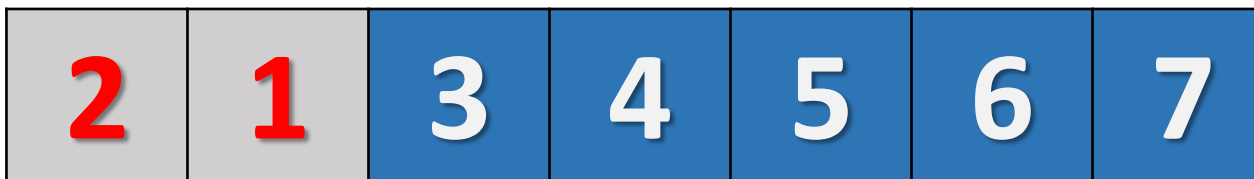
1 ↔ 3



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.





# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.



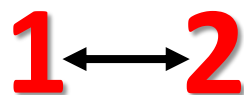
**2 1**



# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.







# Bubble Sort (Bolha)

---

Na primeira passagem são visitados todos os dados e comparados dois a dois a partir do primeiro, identificando o maior entre eles. Na segunda passagem, é repetido o processo a partir do segundo dado, depois do terceiro e assim por diante até o final da classificação.

1	2	3	4	5	6	7
---	---	---	---	---	---	---



```
#include <stdio.h>
```

```
int main() {
```

```
    int x, y, t, v[7] = {5, 7, 6, 4, 2, 3, 1};
```

```
    for (x = 6; x > 0; x--) {
```

```
        for (y = 0; y < x; y++) {
```

```
            if (v[y] > v[y+1]) {
```

```
                t = v[y];
```

```
                v[y] = v[y+1];
```

```
                v[y+1] = t;
```

```
            }
```

```
        }
```

```
    }
```

```
    for (x = 0; x < 7; x++)
```

```
        printf("%i ", v[x]);
```

```
    printf("\n\n\n");
```

```
    return 0;
```

```
}
```



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

5	7	6	4	2	3	1
---	---	---	---	---	---	---



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

5	7	6	4	2	3	1
---	---	---	---	---	---	---

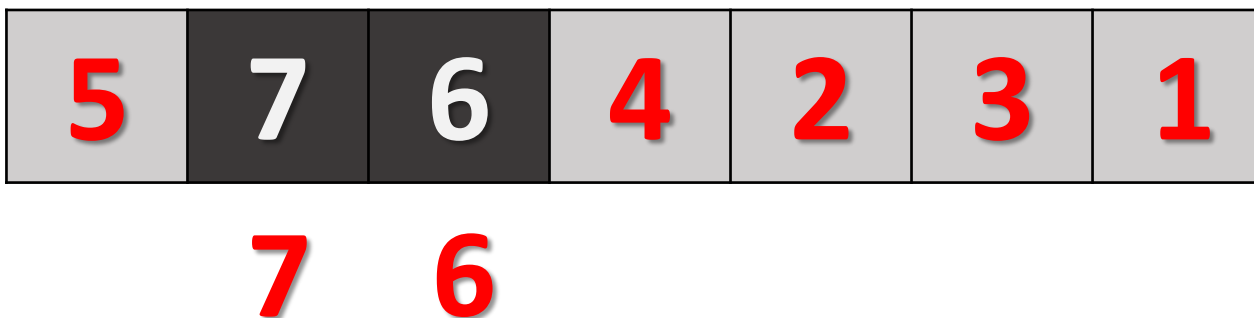
5 7



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

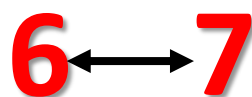




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

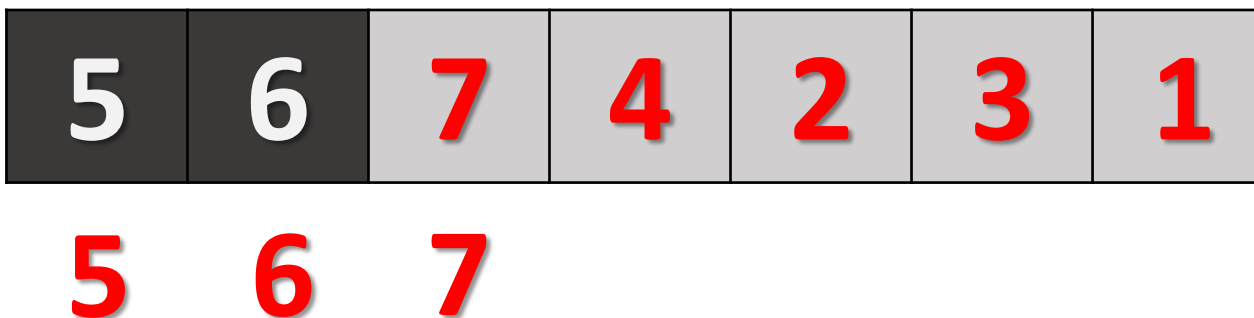




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

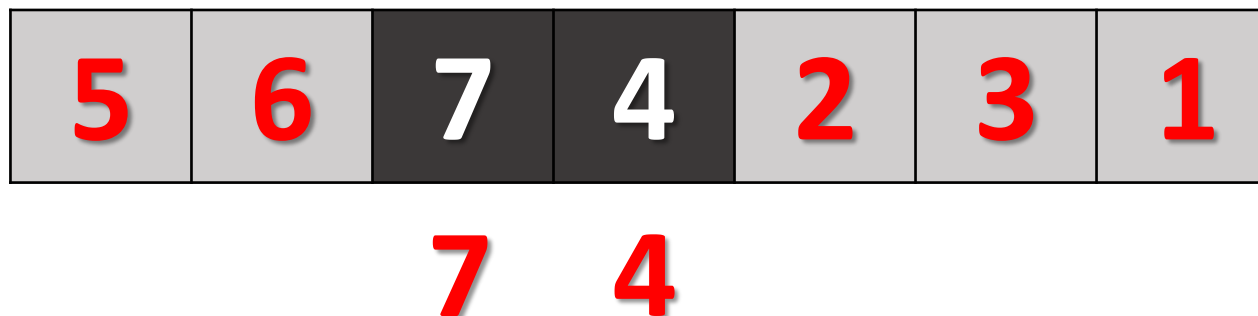




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



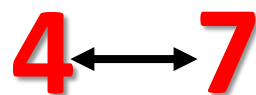




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

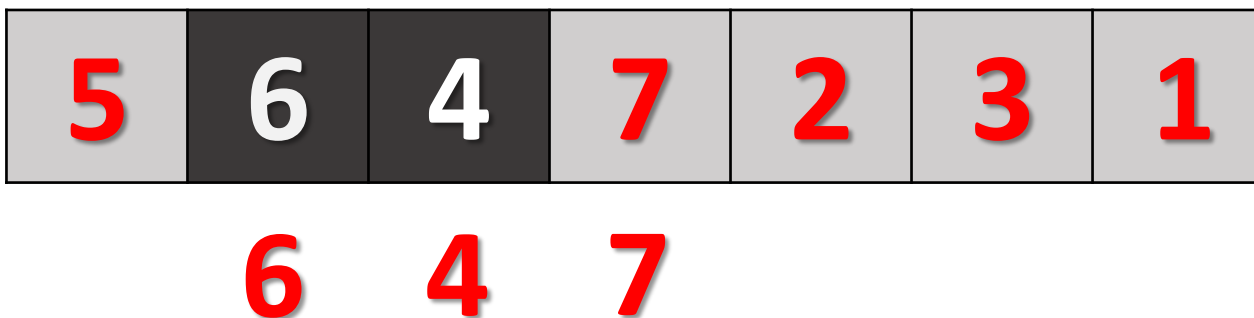




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

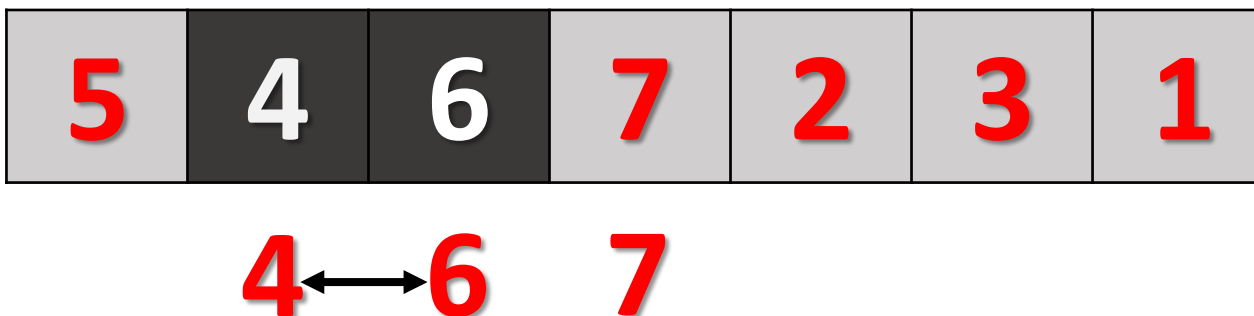




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

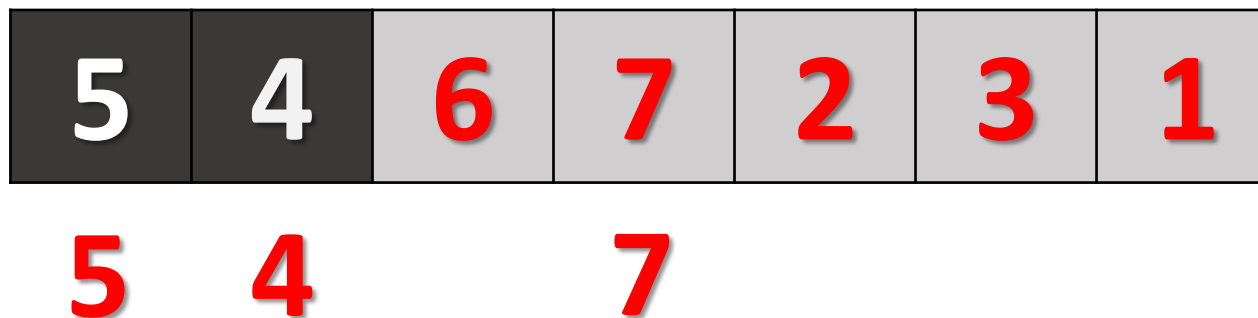




# Insertion Sort (Inserção)

---

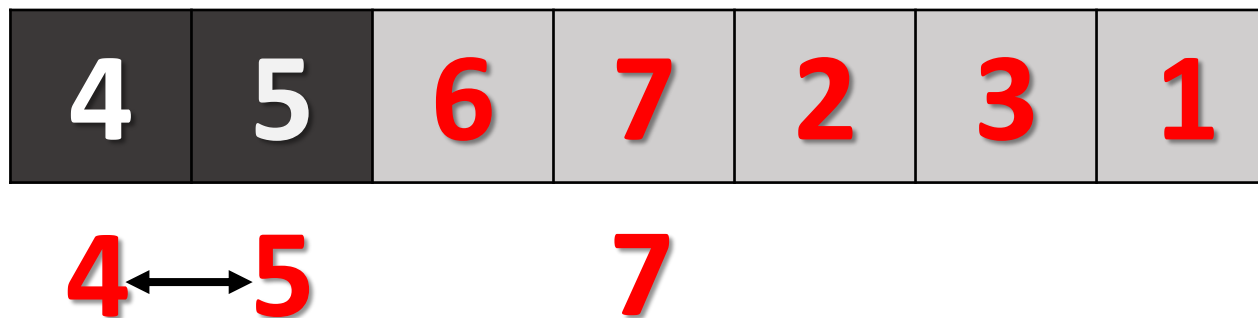
A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.





# Insertion Sort (Inserção)

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

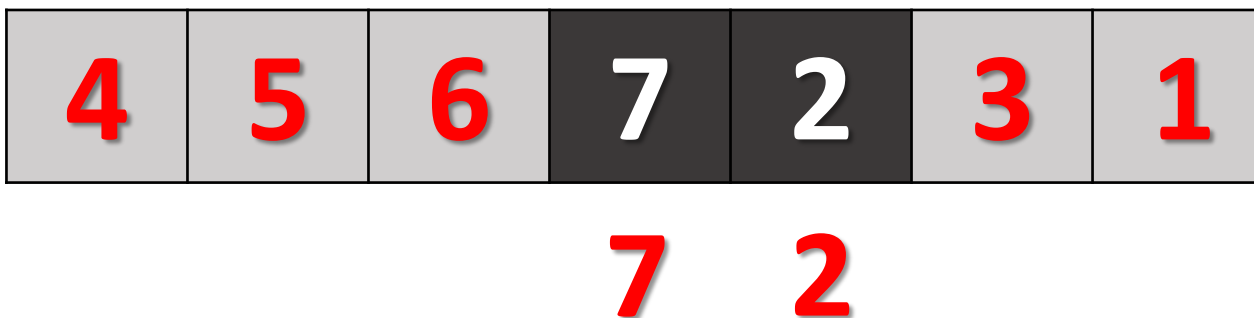




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.





# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



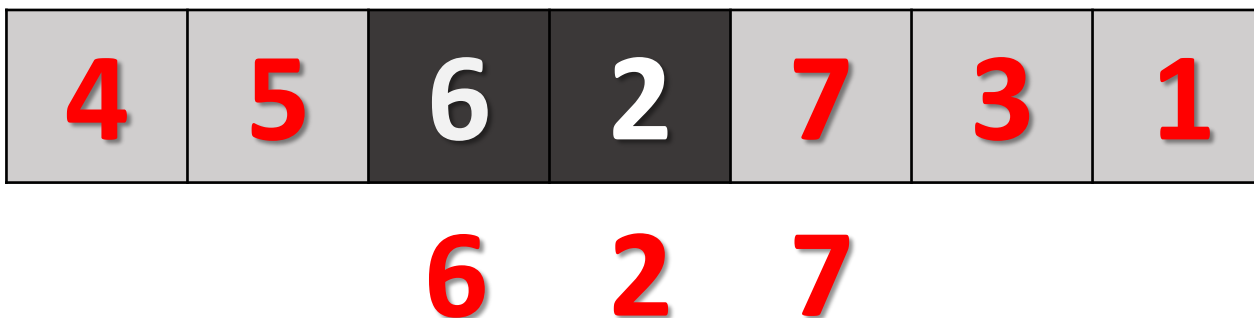
2 ↔ 7



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.







# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



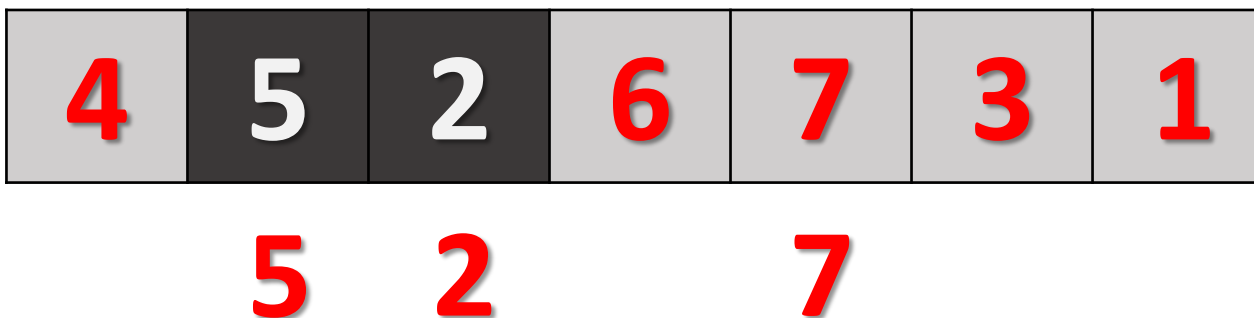
2 ↔ 6    7



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

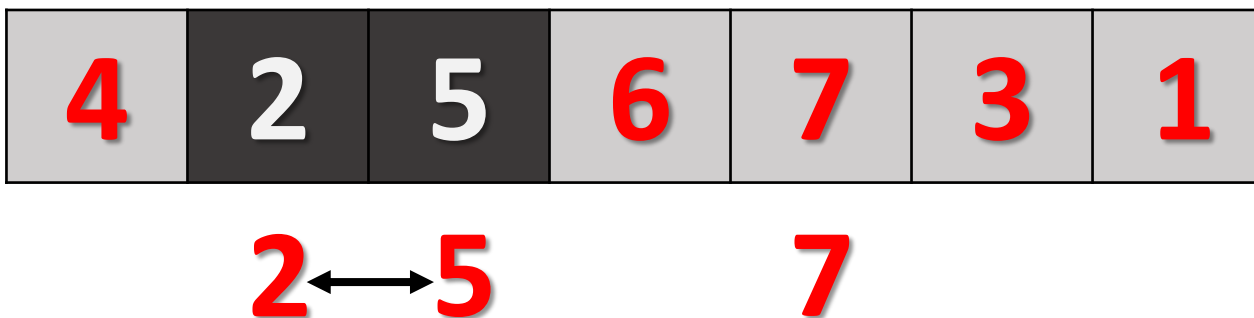




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

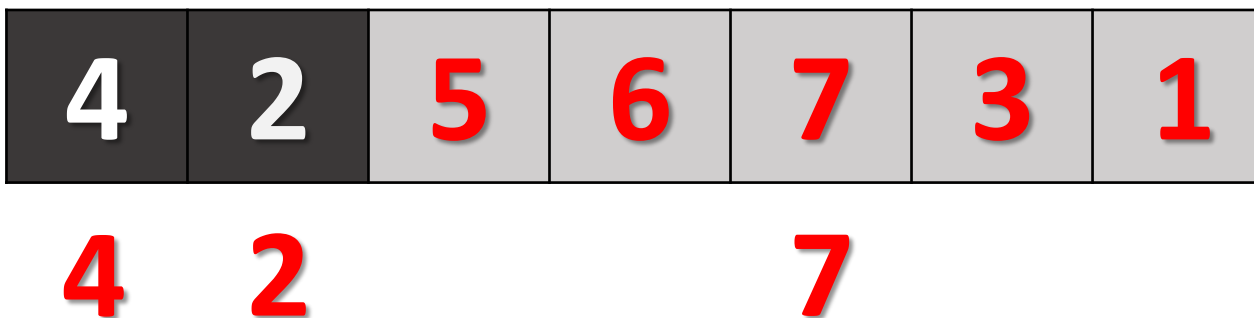




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

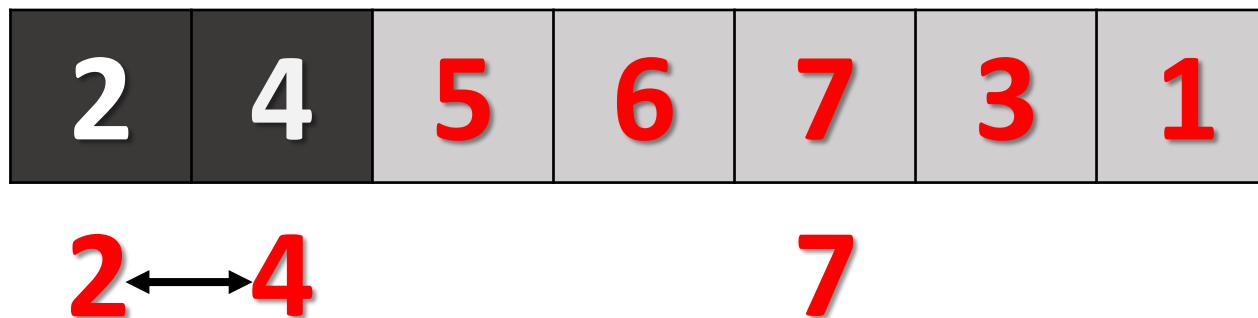




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

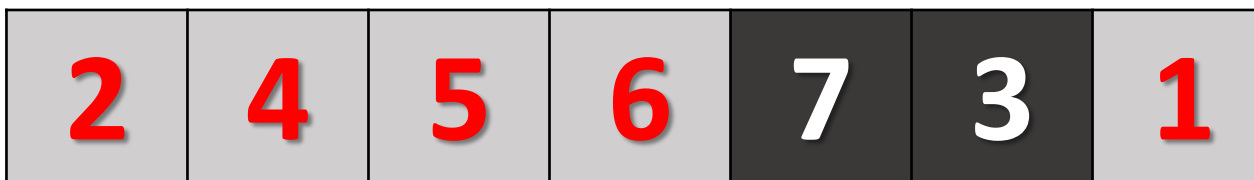




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



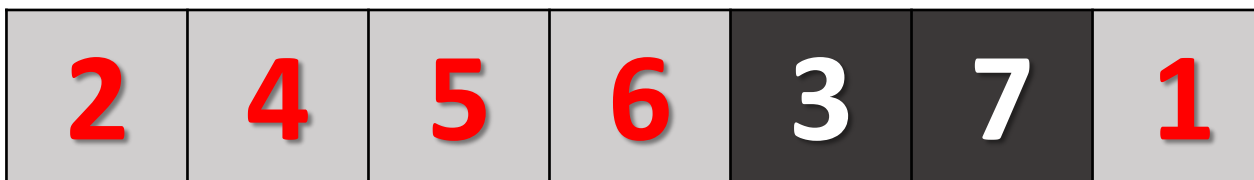
7 ↔ 3



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



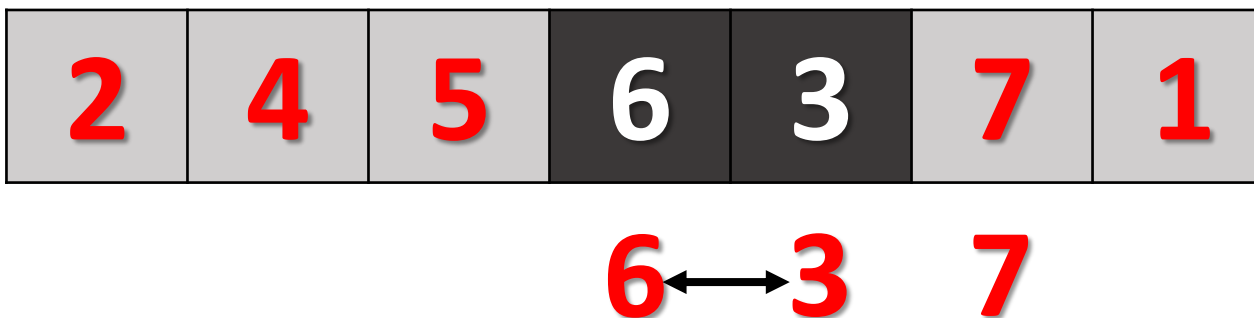
3 ↔ 7



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.







# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

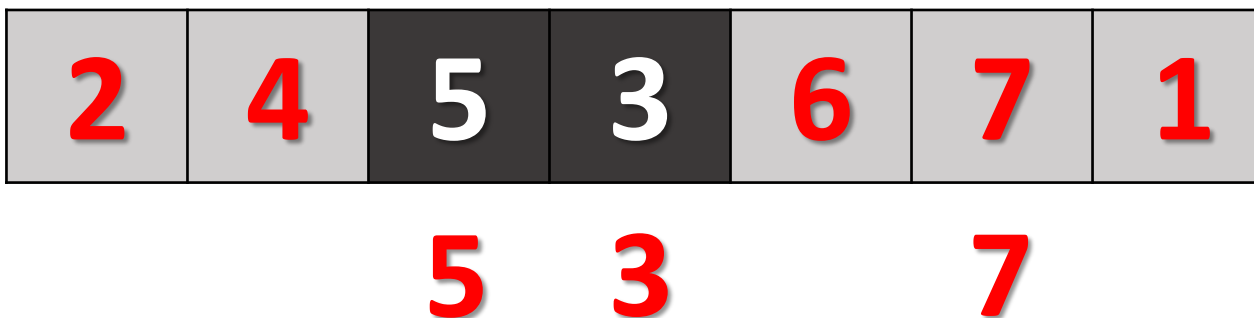




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

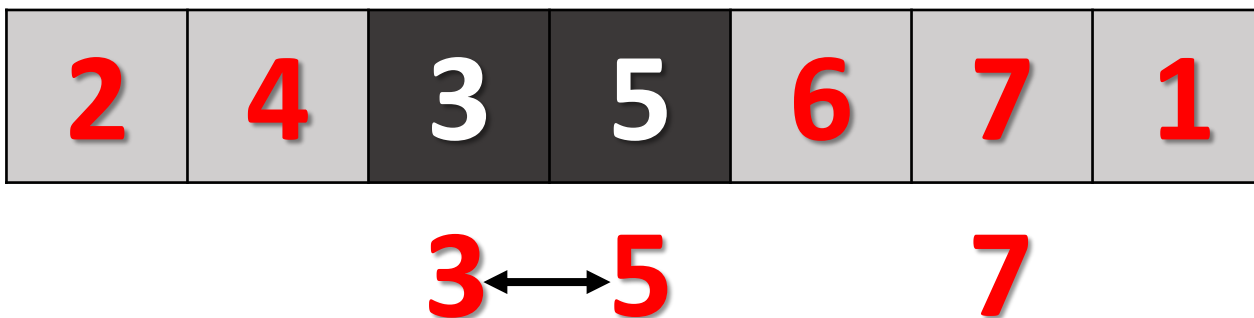




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

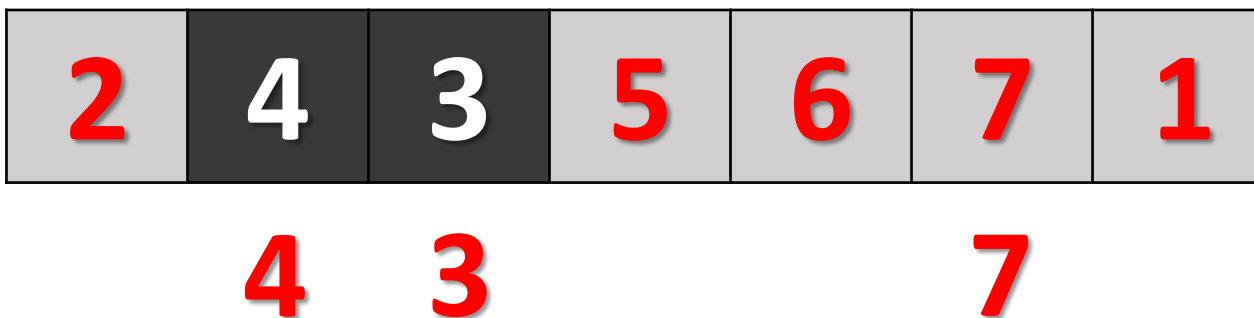




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

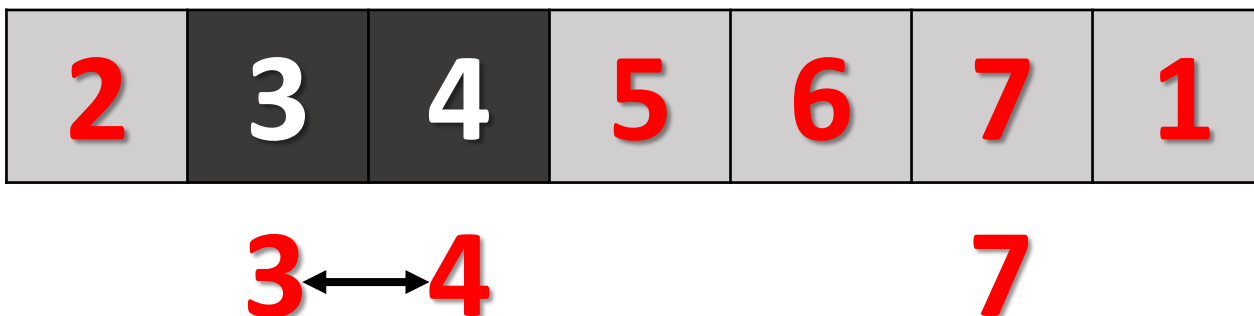





# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.





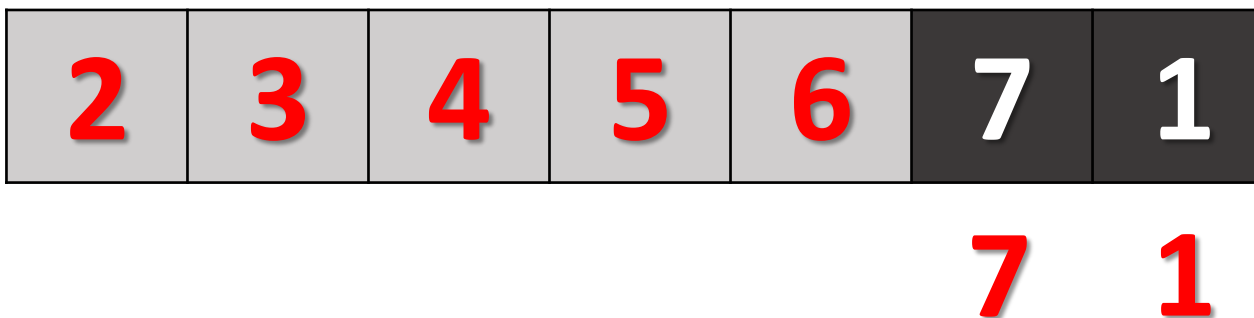
A diagram showing a horizontal array of seven boxes containing the numbers 2, 3, 4, 5, 6, 7, and 1. The boxes for 2 and 3 are dark gray, while the others are light gray. Below the array, the numbers 2, 3, and 7 are written in red, indicating they are the elements being compared.



# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

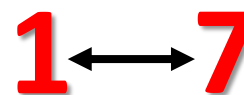




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



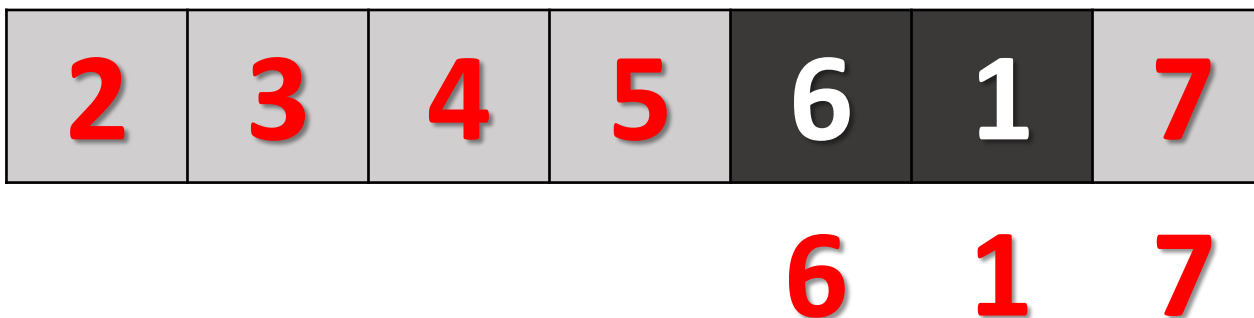




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

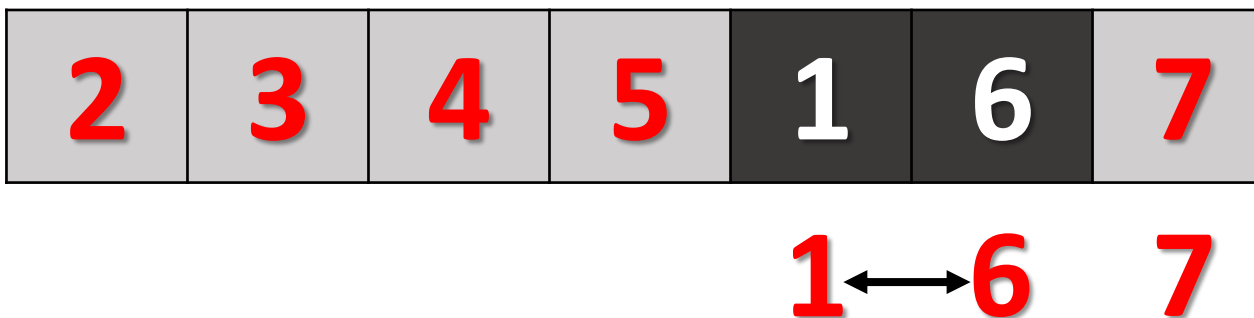




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

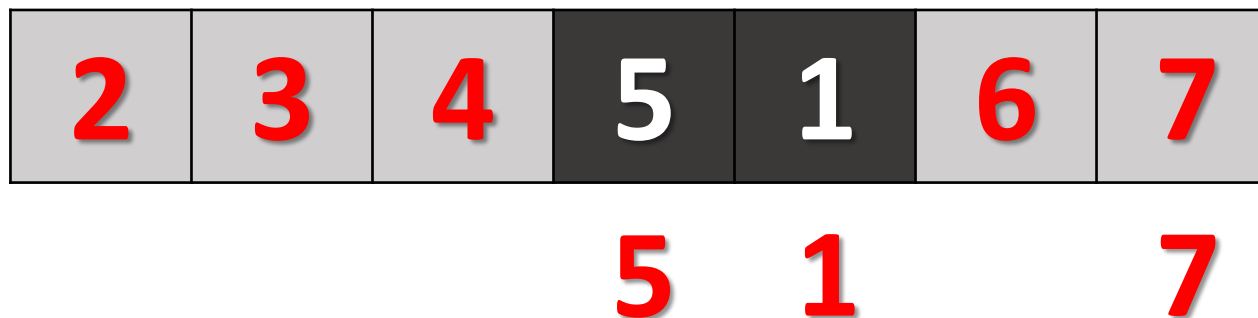




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

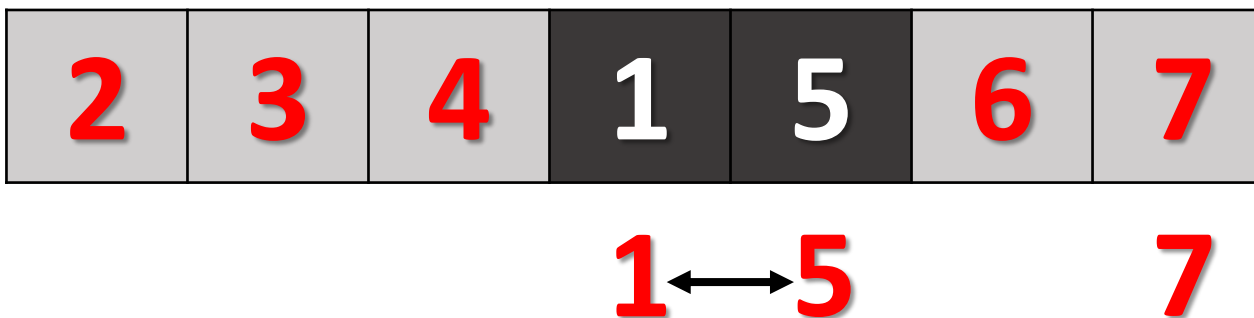




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

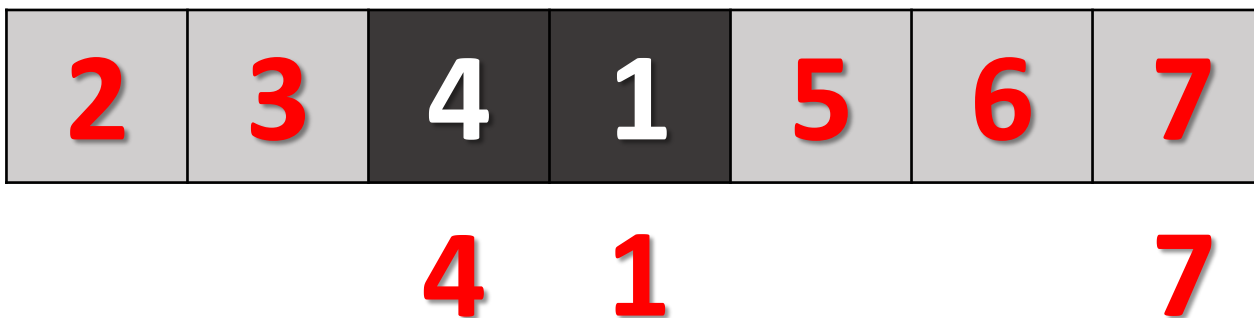




# Insertion Sort (Inserção)

---

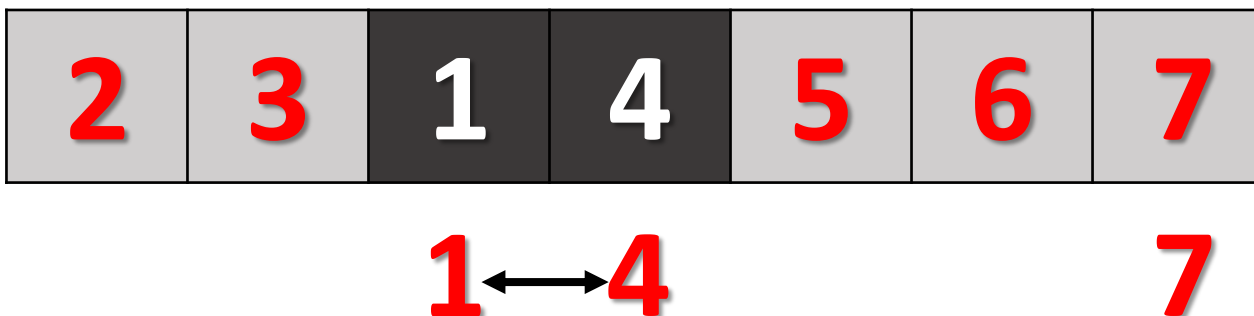
A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.





# Insertion Sort (Inserção)

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

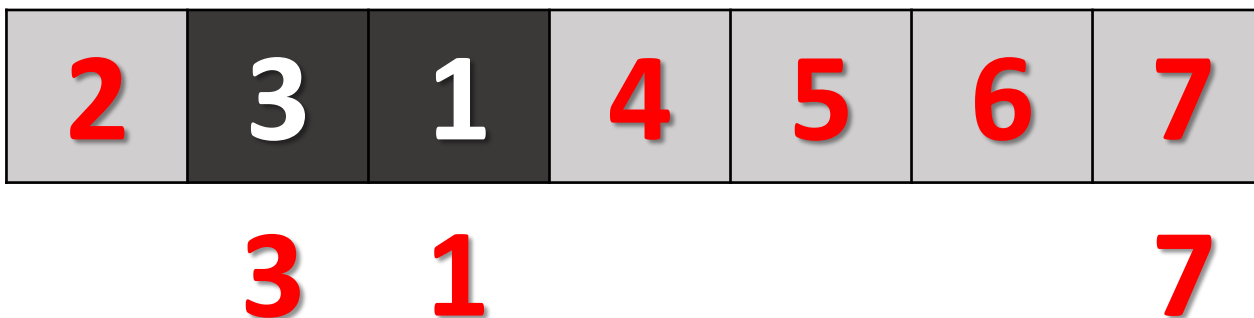




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

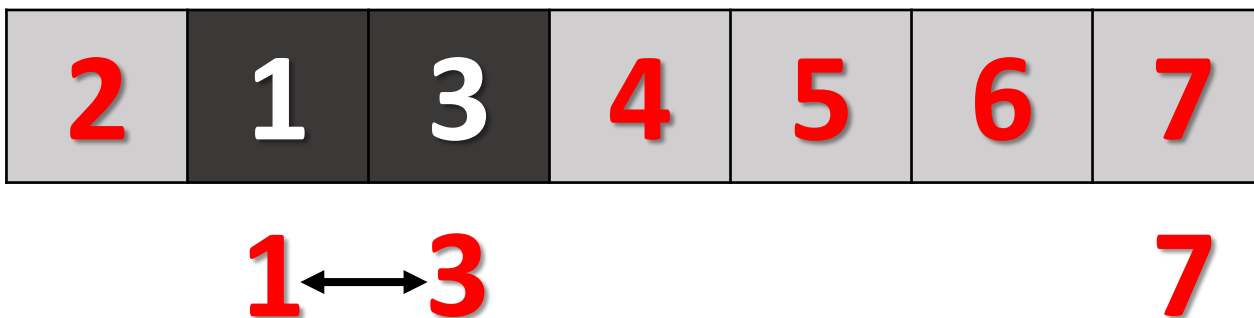




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.



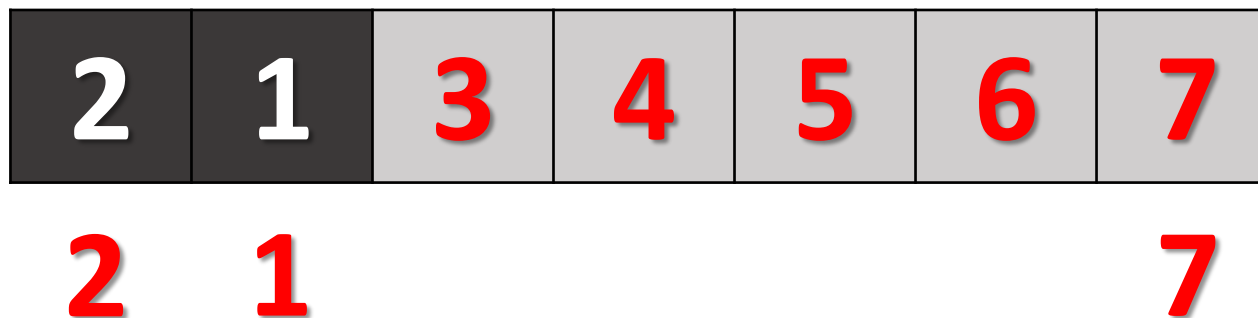




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

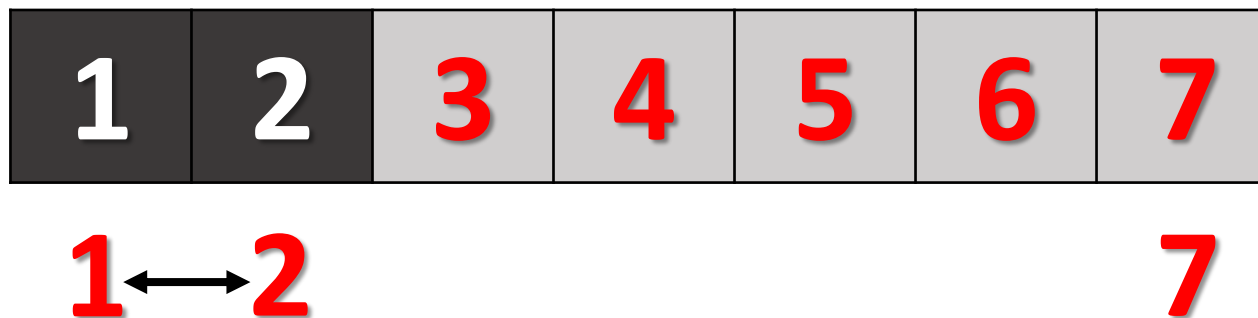




# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.





# Insertion Sort (Inserção)

---

A ordenação por inserção (insertion sort) se baseia em comparações sucessivas entre dois elementos adjacentes e a troca quando o elemento seguinte é menor que o anterior. Em caso de troca, os elementos anteriores deverão ser comparados novamente.

1	2	3	4	5	6	7
---	---	---	---	---	---	---



```
#include <stdio.h>
```

```
int main() {
```

```
    int x, y, t, v[7] = {5, 7, 6, 4, 2, 3, 1};
```

```
    for (x = 1; x < 7; x++) {
```

```
        if (v[x] < v[x-1]) {
```

```
            for (y = x; y != 0; y--) {
```

```
                t = v[y];
```

```
                v[y] = v[y-1];
```

```
                v[y-1] = t;
```

```
            }
```

```
        }
```

```
    }
```

```
    for (x = 0; x < 7; x++)
```

```
        printf("%i ", v[x]);
```

```
    printf("\n\n\n");
```

```
    return 0;
```

```
}
```