Selection Sort e Insertion Sort

Prof. Andrey Masiero

15 de setembro de 2017

Agenda

Selection Sort

- Busca pelo menor elemento no vetor;

Selection Sort

- Busca pelo menor elemento no vetor;
- Depois posiciona ele na primeira posição;

Selection Sort

- Busca pelo menor elemento no vetor;
- Depois posiciona ele na primeira posição;
- Realiza novamente o processo para todos elementos até $n - 1$;

Selection Sort

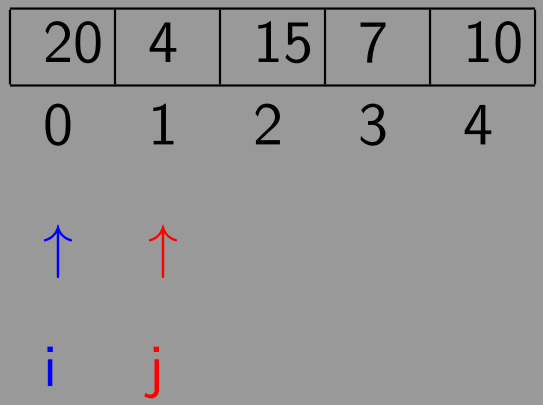
- Busca pelo menor elemento no vetor;
- Depois posiciona ele na primeira posição;
- Realiza novamente o processo para todos elementos até $n - 1$;
- Complexidade dele: $O(n^2)$.

Selection Sort

Vamos ordernar este vetor!

20	4	15	7	10
0	1	2	3	4

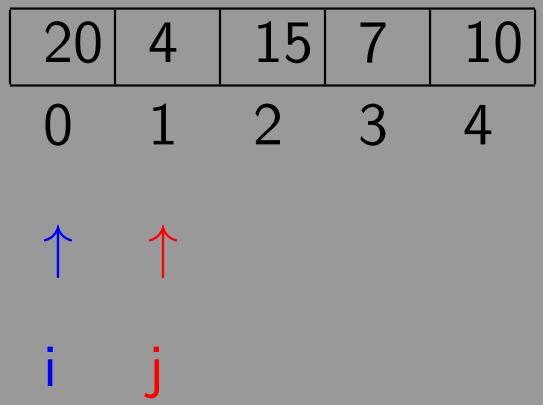
Selection Sort



$troca = 0$

- inicializa $i = 0$; $j = i + 1$;

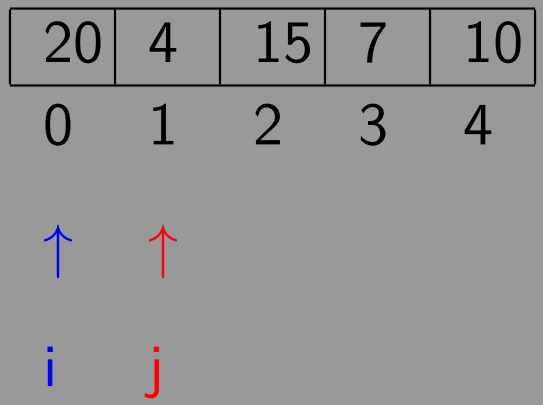
Selection Sort



$troca = 0$

- inicializa $i = 0$; $j = i + 1$;
- $20 > 4$? Sim

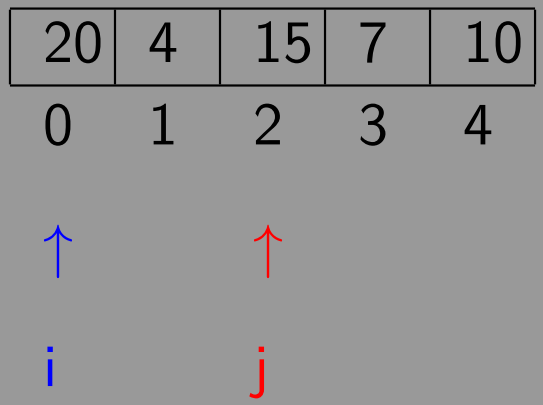
Selection Sort



$troca = 0$

- inicializa $i = 0$; $j = i + 1$;
- $20 > 4$? Sim
- $troca = j$

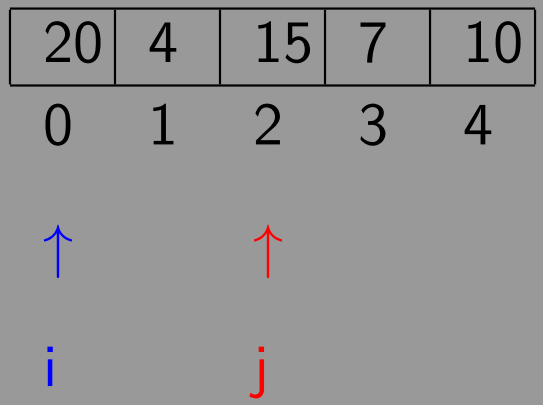
Selection Sort



$troca = 1$

- $j = j + 1;$

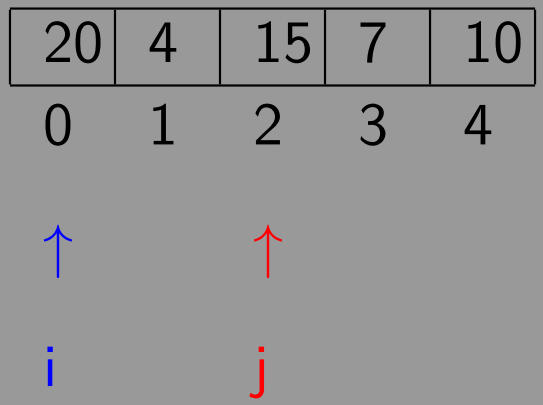
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 15$? Não

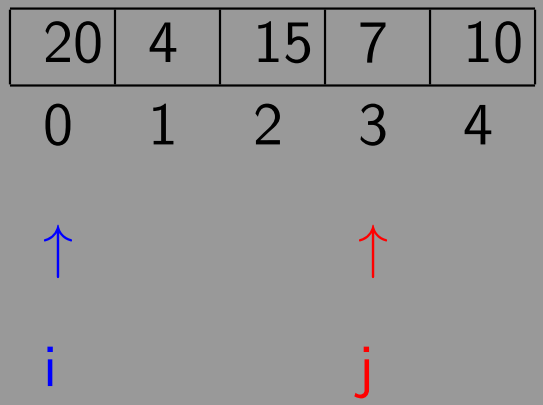
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 15$? Não
- próximo

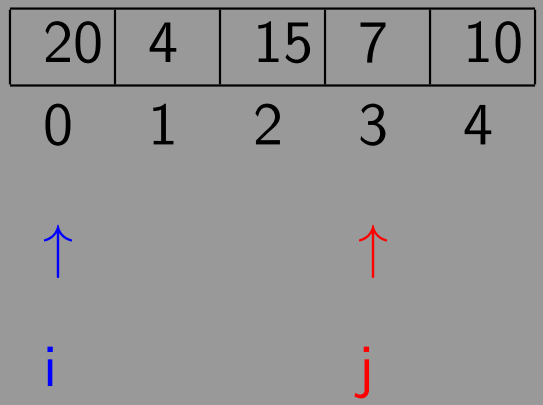
Selection Sort



$troca = 1$

- $j = j + 1;$

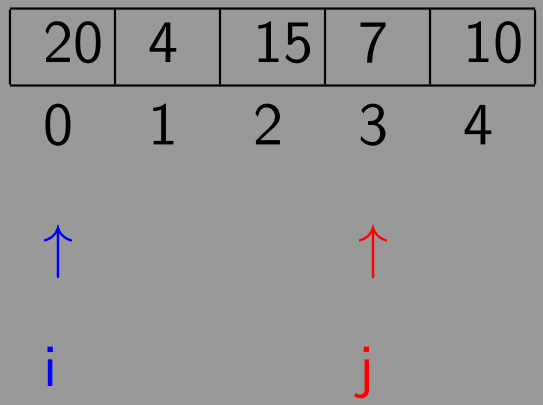
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 7$? Não

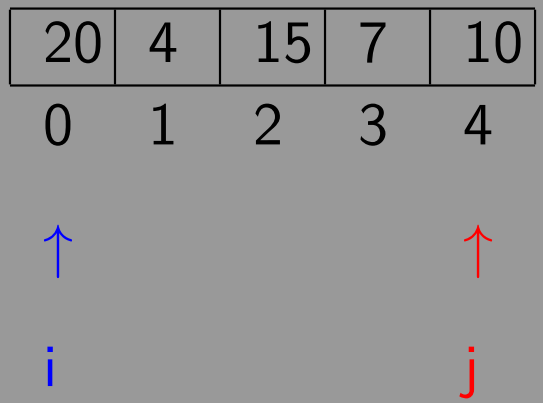
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 7$? Não
- próximo

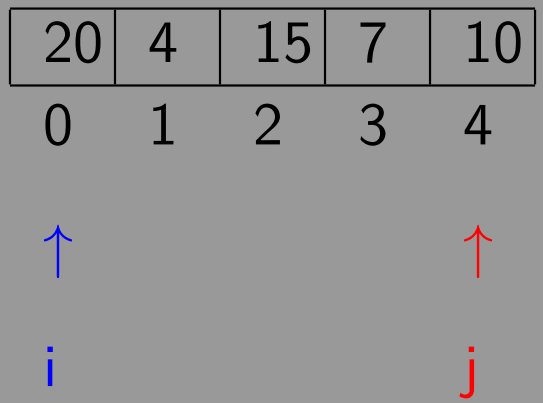
Selection Sort



$troca = 1$

- $j = j + 1;$

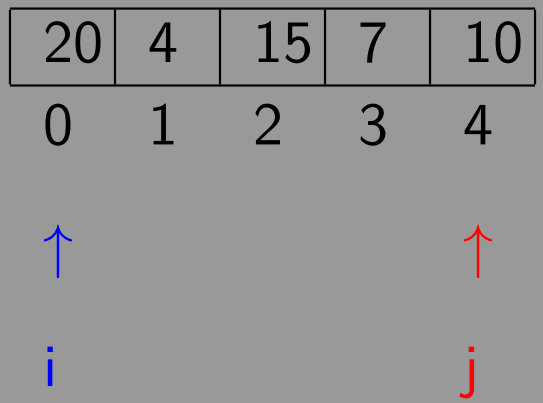
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 10$? Não

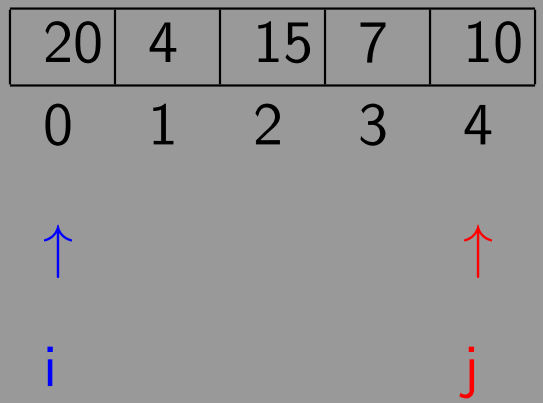
Selection Sort



$troca = 1$

- $j = j + 1$;
- $4 > 10$? Não
- próximo

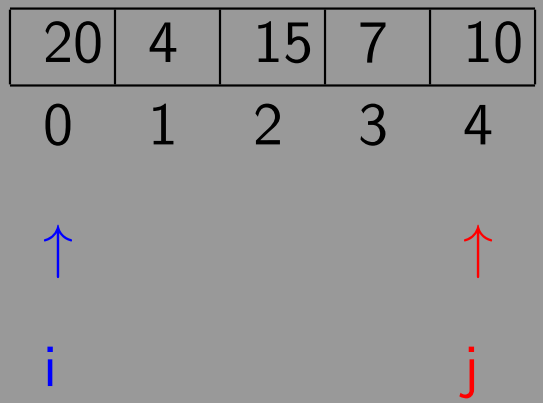
Selection Sort



$troca = 1$

- finalizou o vetor

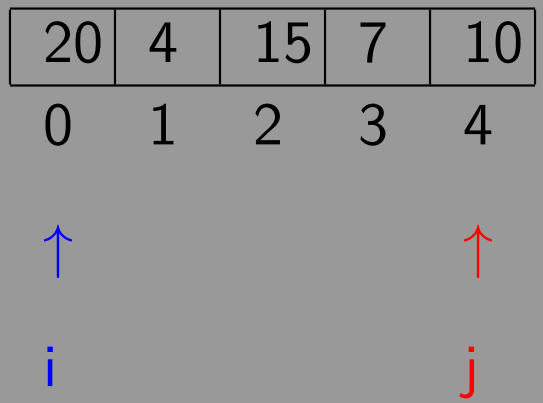
Selection Sort



$troca = 1$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$

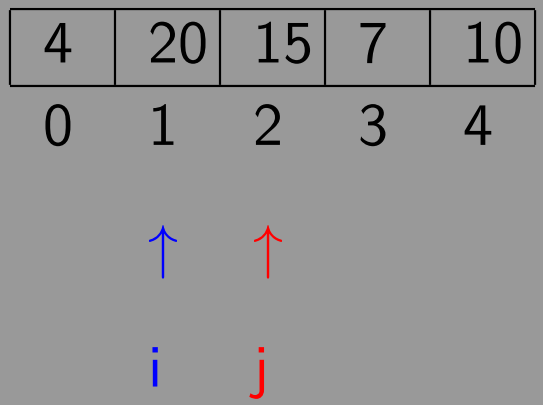
Selection Sort



$troca = 1$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$
- $i = i + 1$; $j = i + 1$; $troca = i$

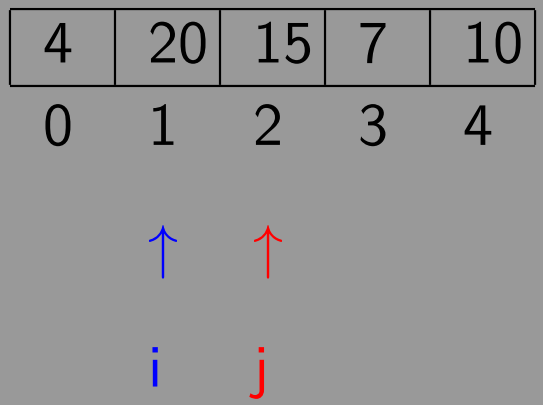
Selection Sort



$troca = 1$

- $20 > 15$? Sim

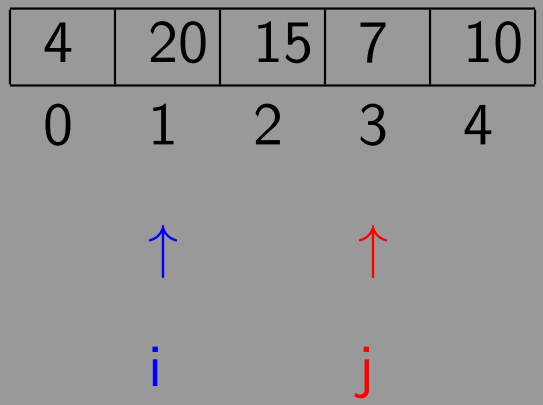
Selection Sort



$troca = 1$

- $20 > 15$? Sim
- $troca = j$

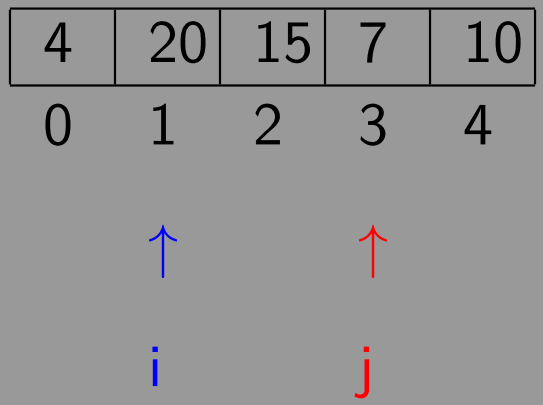
Selection Sort



$troca = 2$

- $j = j + 1;$

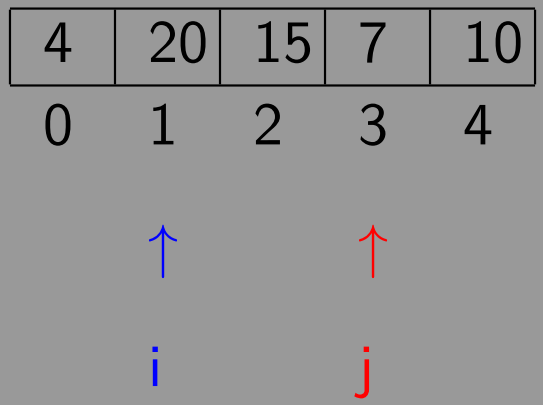
Selection Sort



$troca = 2$

- $j = j + 1$;
- $15 > 7$? Sim

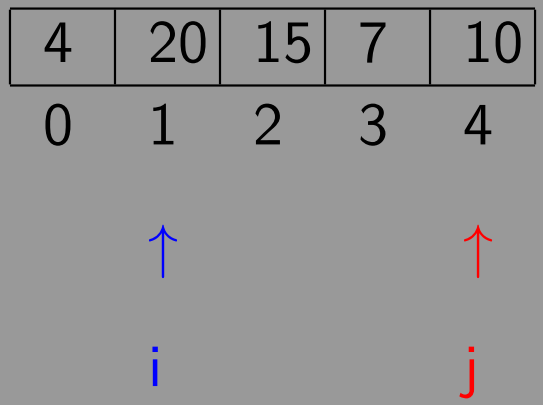
Selection Sort



$troca = 2$

- $j = j + 1$;
- $15 > 7$? Sim
- $troca = j$

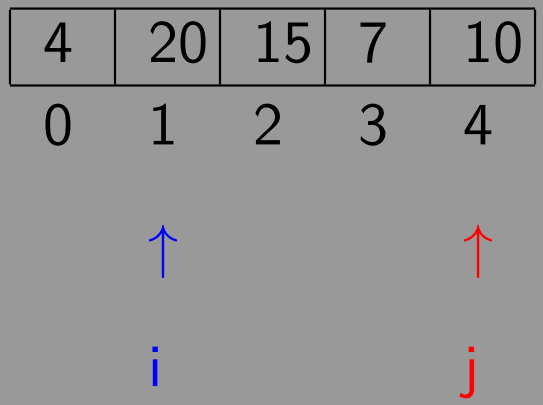
Selection Sort



$troca = 3$

- $j = j + 1;$

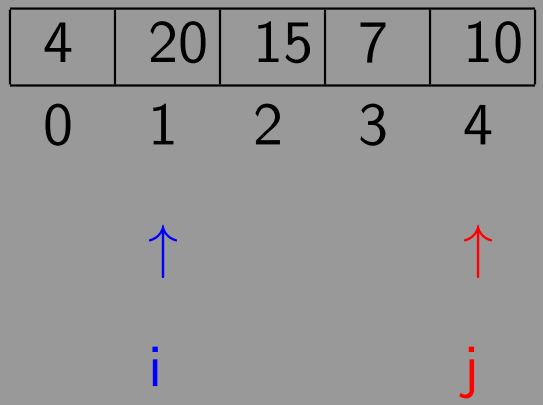
Selection Sort



$troca = 3$

- $j = j + 1$;
- $7 > 10$? Não

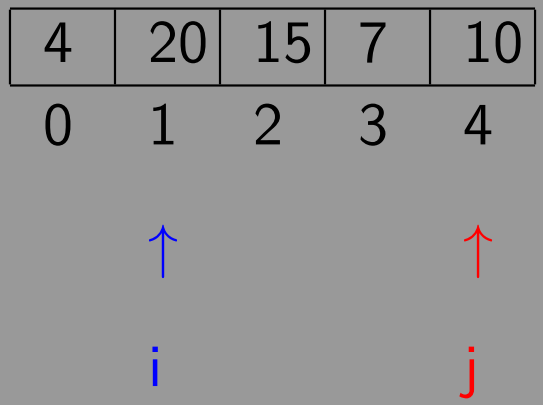
Selection Sort



$troca = 3$

- $j = j + 1$;
- $7 > 10$? Não
- próximo

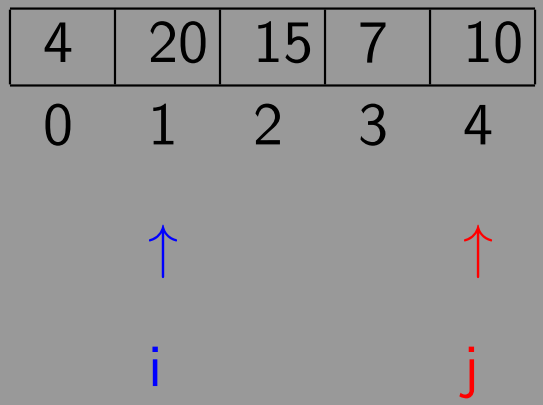
Selection Sort



$troca = 3$

- finalizou o vetor

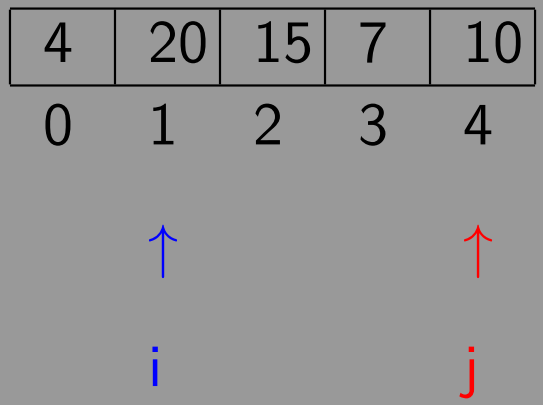
Selection Sort



$troca = 3$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$

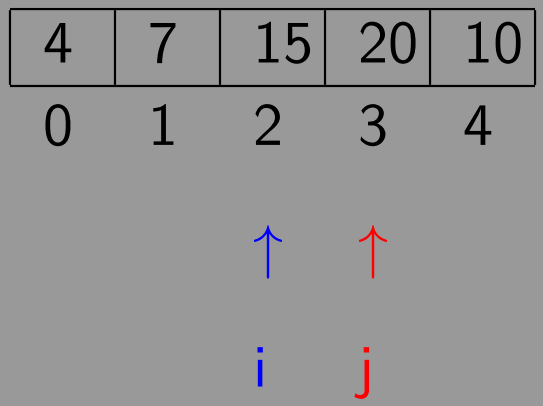
Selection Sort



$troca = 3$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$
- $i = i + 1$; $j = i + 1$; $troca = i$

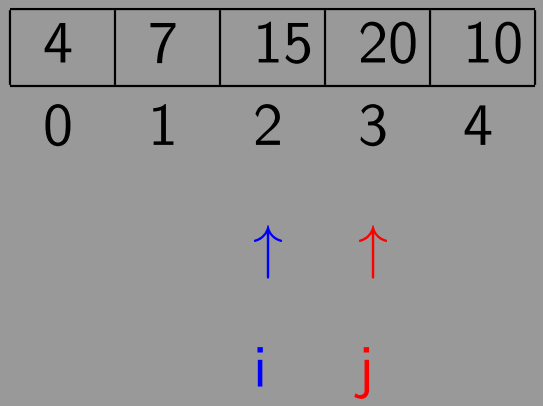
Selection Sort



$troca = 2$

- $15 > 20$? Não

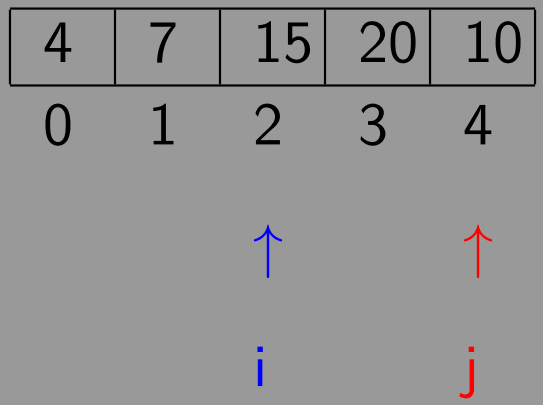
Selection Sort



$troca = 2$

- $15 > 20$? Não
- próximo

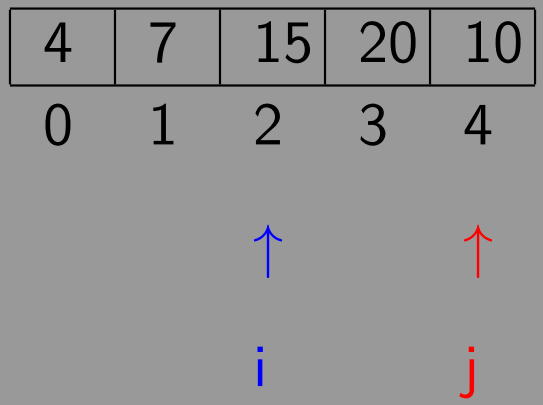
Selection Sort



$troca = 2$

- $j = j + 1;$

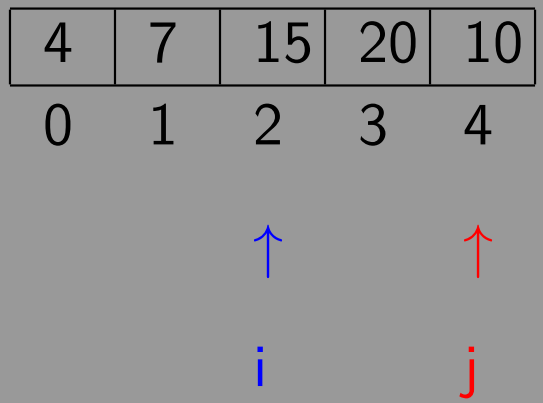
Selection Sort



$troca = 2$

- $j = j + 1$;
- $15 > 10$? Sim

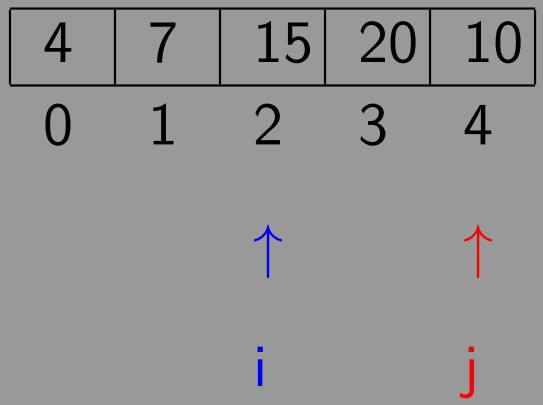
Selection Sort



$troca = 2$

- $j = j + 1;$
- $15 > 10$? Sim
- $troca = j$

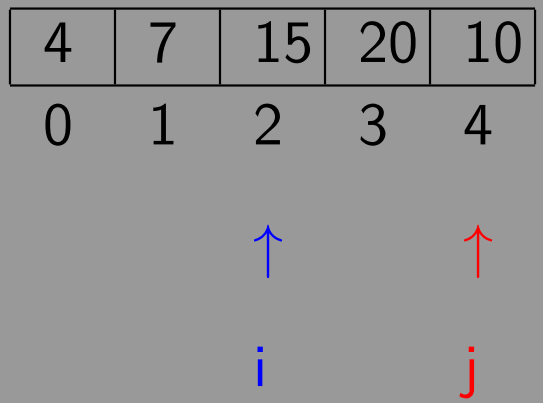
Selection Sort



$troca = 4$

- finalizou o vetor

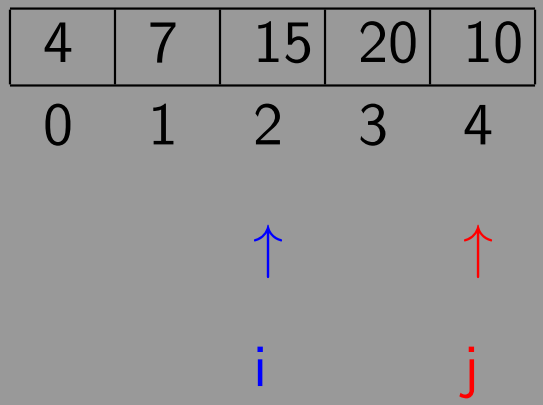
Selection Sort



$troca = 4$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$

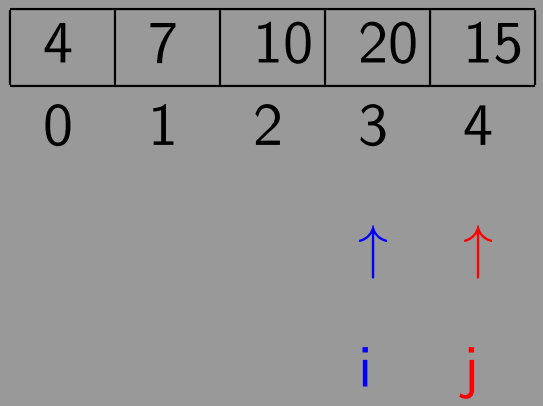
Selection Sort



$troca = 4$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$
- $i = i + 1$; $j = i + 1$; $troca = i$

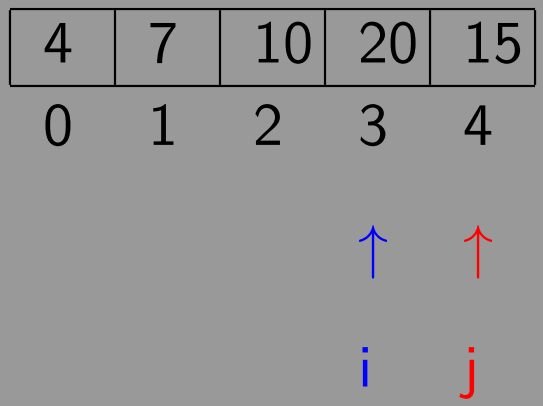
Selection Sort



$troca = 3$

- $20 > 15$? Sim

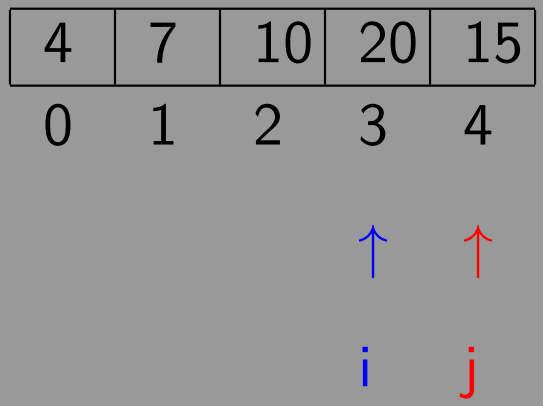
Selection Sort



$troca = 3$

- $20 > 15$? Sim
- $troca = j$

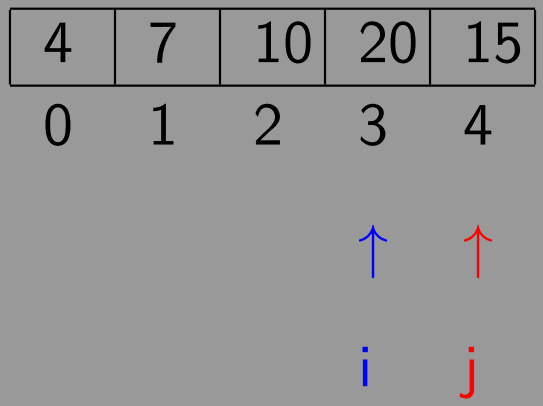
Selection Sort



$troca = 4$

- finalizou o vetor

Selection Sort



$troca = 4$

- finalizou o vetor
- $swap(vetor[i], vetor[troca])$

Selection Sort

4	7	10	15	20
0	1	2	3	4

Vetor Ordenado

Selection Sort

```
vetor[] = {20, 4, 15, 7, 10}

for (i = 0; i < vetor.length - 1; i++)
    indice_menor = i
    for (j = i + 1; j < vetor.length; j++)
        if (vetor[j] < vetor[indice_menor])
            indice_menor = j
    swap(vetor[j], vetor[j + 1])
```

Insertion Sort

- Utilizando um vetor auxiliar;

Insertion Sort

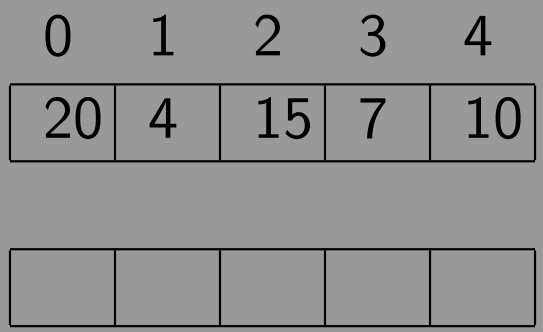
- Utilizando um vetor auxiliar;
- Os elementos são inseridos na posição correta;

Insertion Sort

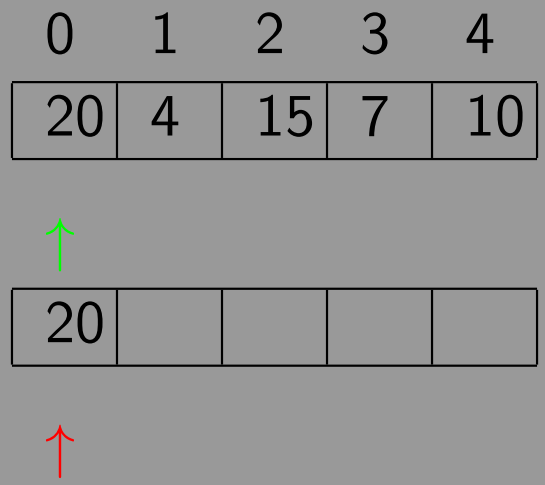
- Utilizando um vetor auxiliar;
- Os elementos são inseridos na posição correta;
- Complexidade dele: $O(n^2)$.

Insertion Sort

Vamos ordernar este vetor!



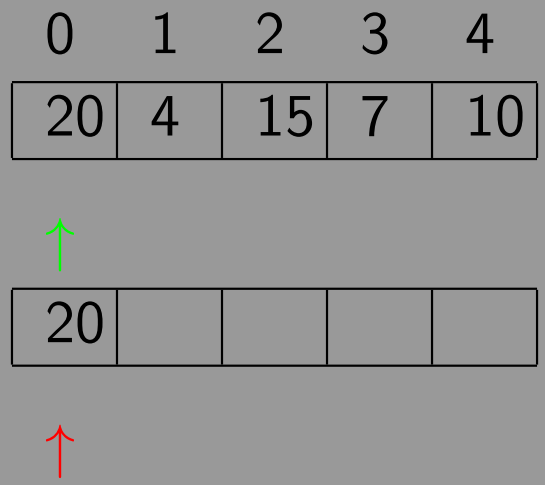
Insertion Sort



$ordenando = 20$

- inicializa $i = 0$; inicializa $j = i - 1$

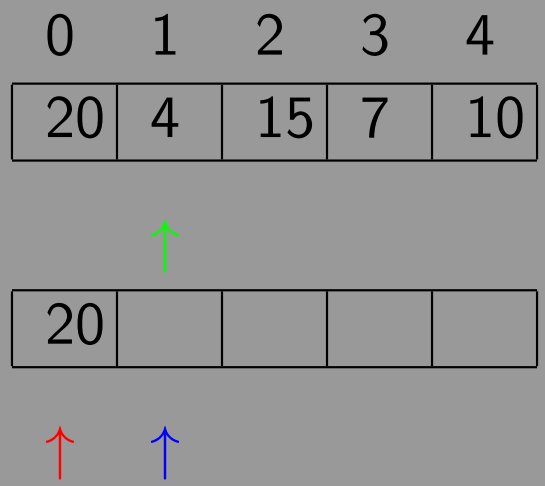
Insertion Sort



$ordenando = 20$

- inicializa $i = 0$; inicializa $j = i - 1$
- $j = -1$ então copia $ordenando$ na posição $j + 1$

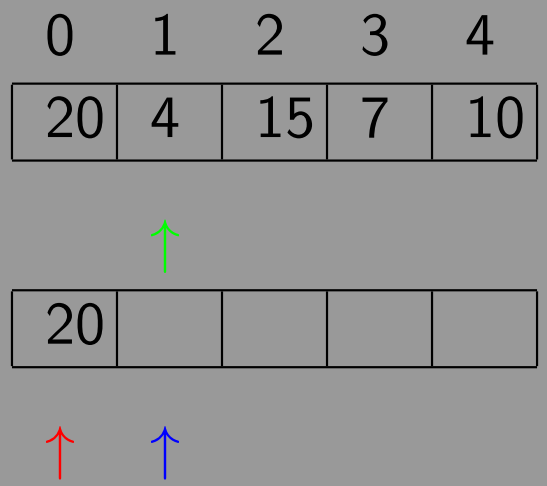
Insertion Sort



$ordenando = 4$

- $i = i + 1$; inicializa $j = i - 1$

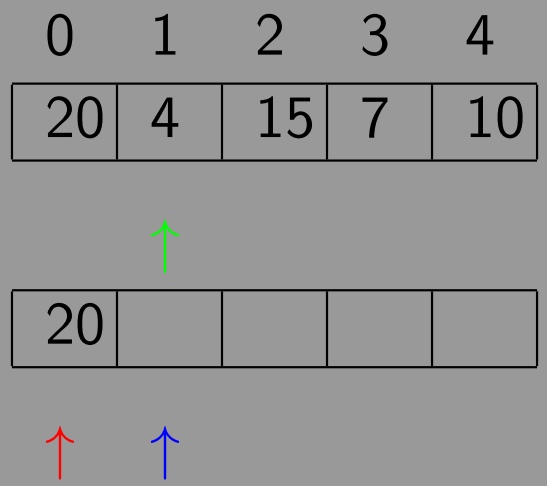
Insertion Sort



$ordenando = 4$

- $i = i + 1$; inicializa $j = i - 1$
- $4 < 20$? Sim

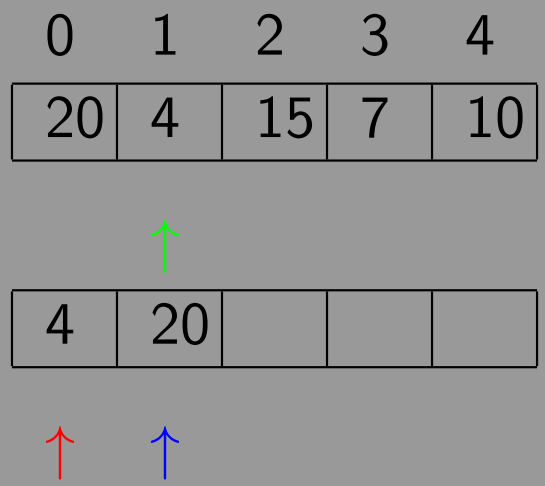
Insertion Sort



$ordenando = 4$

- $i = i + 1$; inicializa $j = i - 1$
- $4 < 20$? Sim
- copia 20 para a posição i

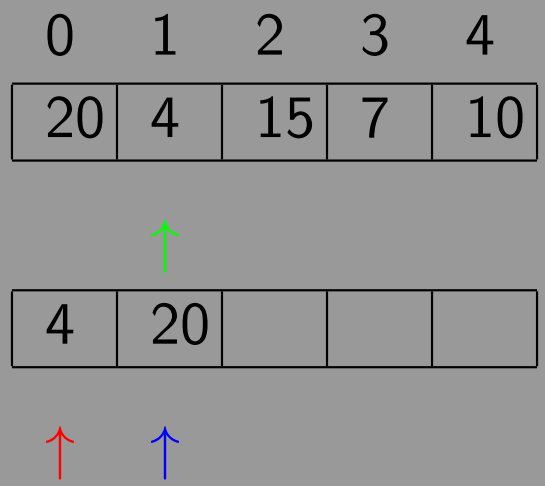
Insertion Sort



$ordenando = 4$

- $j = j - 1$

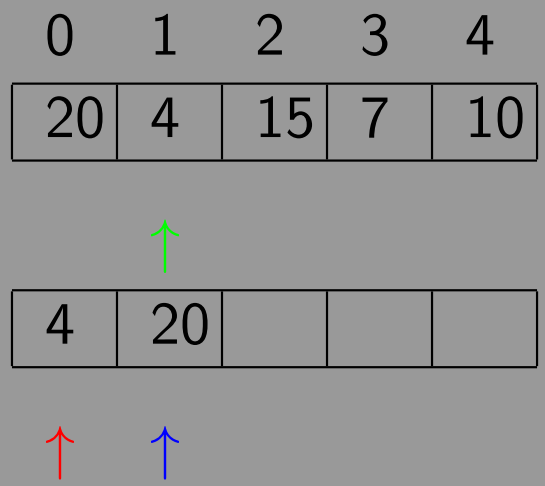
Insertion Sort



$ordenando = 4$

- $j = j - 1$
- acabou vetor

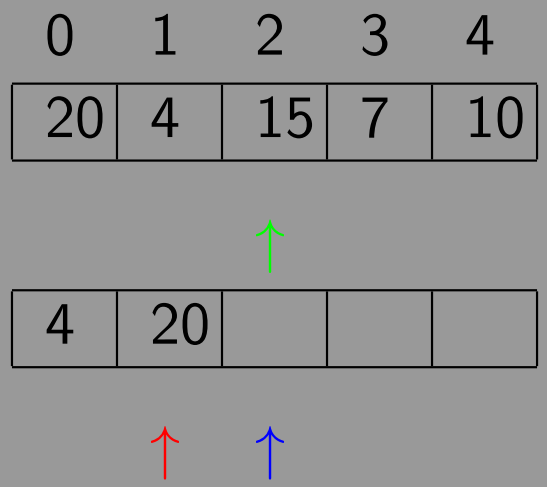
Insertion Sort



$ordenando = 4$

- $j = j - 1$
- acabou vetor
- copia $ordenando$ na posição $j + 1$

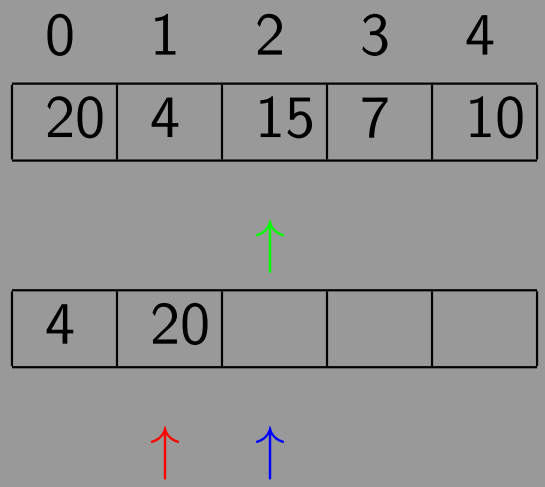
Insertion Sort



$ordenando = 15$

- $i = i + 1;\ j = i - 1$

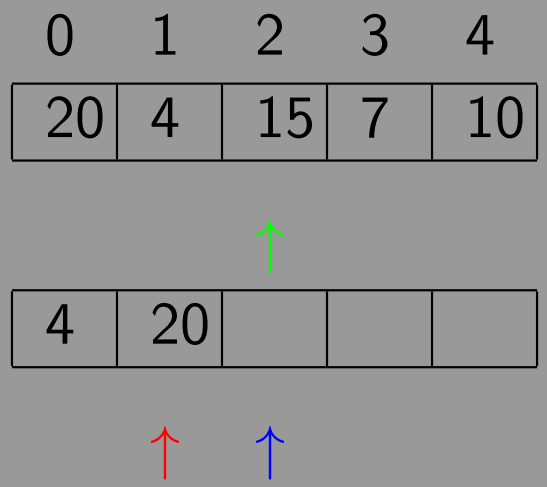
Insertion Sort



$ordenando = 15$

- $i = i + 1$; $j = i - 1$
- $15 < 20$? Sim

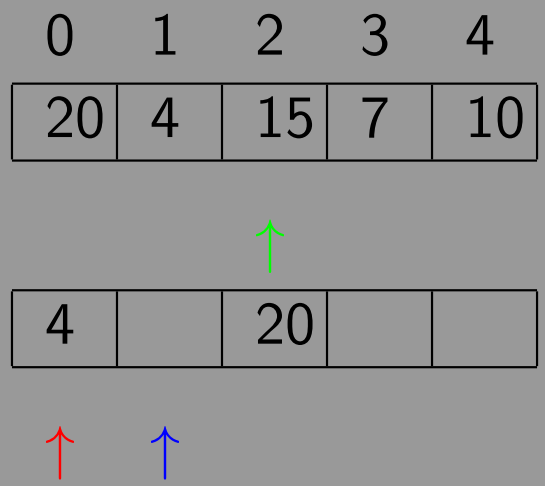
Insertion Sort



$ordenando = 15$

- $i = i + 1$; $j = i - 1$
- $15 < 20$? Sim
- copia 20 para a posição i

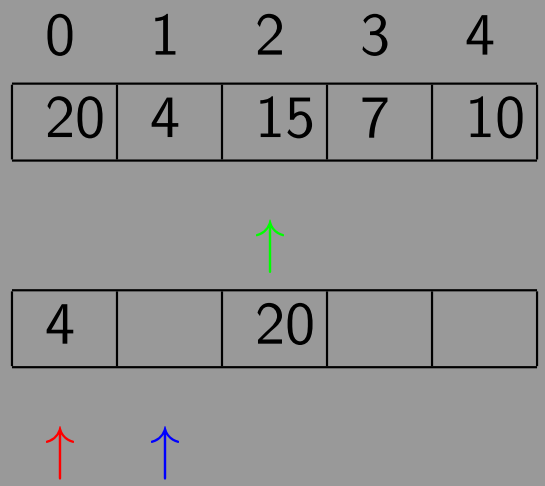
Insertion Sort



$ordenando = 15$

- $j = j - 1$

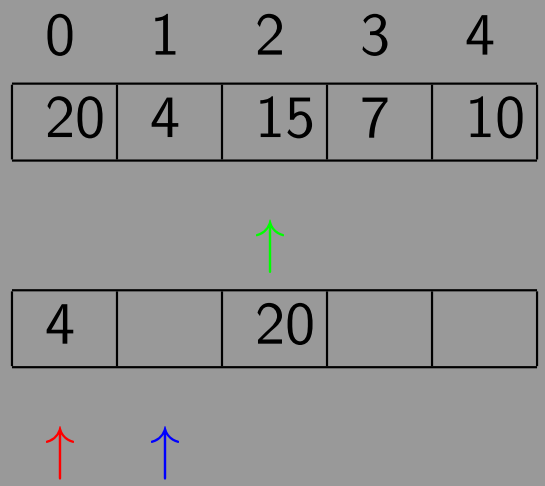
Insertion Sort



$ordenando = 15$

- $j = j - 1$
- $15 < 4$? Não

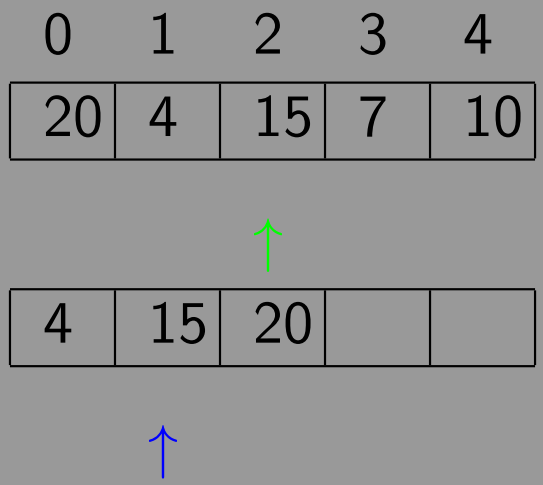
Insertion Sort



$ordenando = 15$

- $j = j - 1$
- $15 < 4$? Não
- acabou iteração

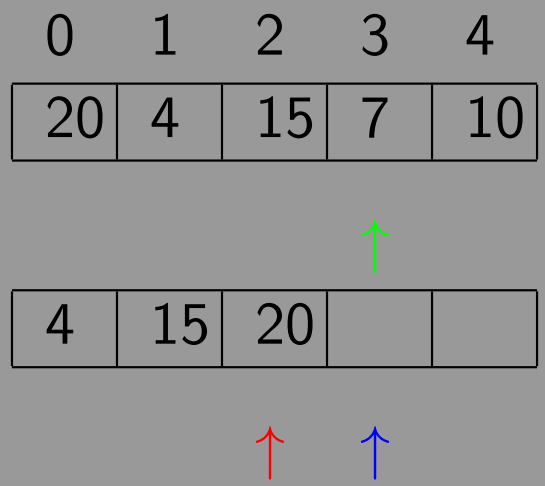
Insertion Sort



$ordenando = 15$

- copia $ordenando$ na posição $j + 1$

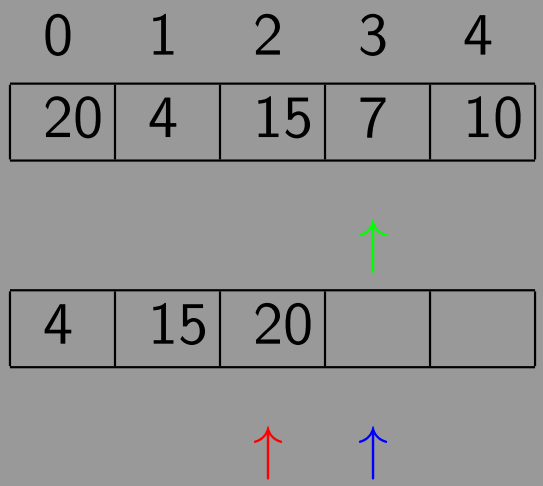
Insertion Sort



$ordenando = 7$

- $i = i + 1$; $j = i - 1$

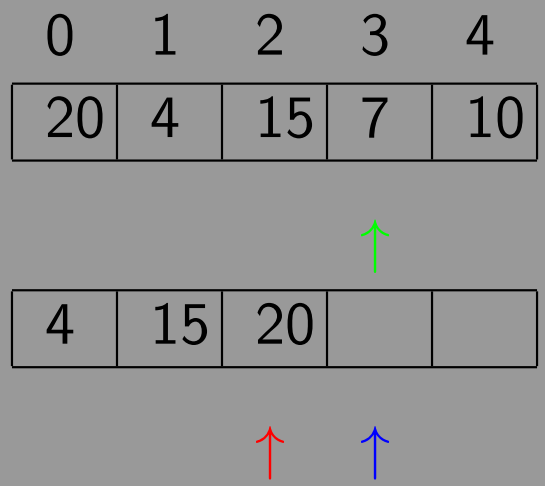
Insertion Sort



$ordenando = 7$

- $i = i + 1$; $j = i - 1$
- $7 < 20$? Sim

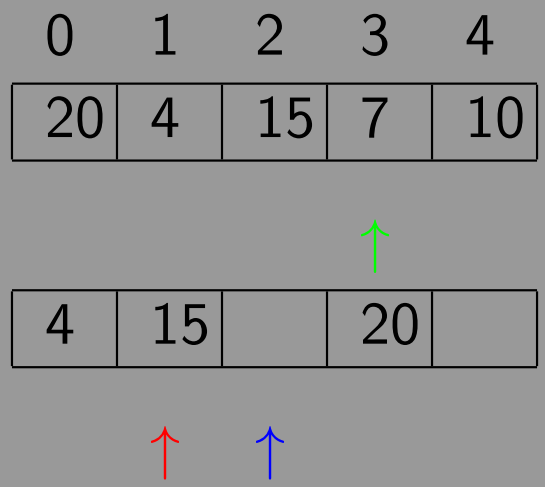
Insertion Sort



$ordenando = 7$

- $i = i + 1$; $j = i - 1$
- $7 < 20$? Sim
- copia 20 para a posição i

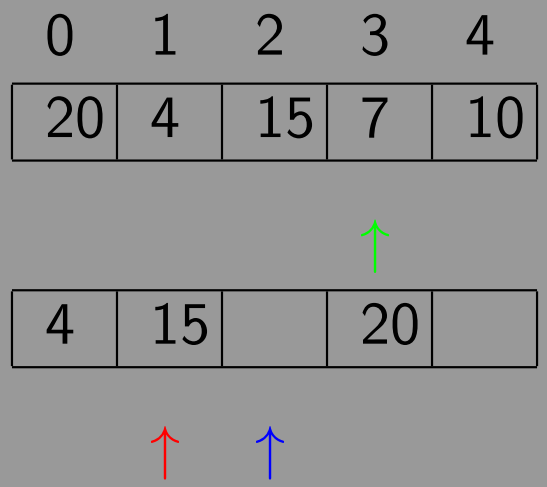
Insertion Sort



$ordenando = 7$

- $j = i - 1$

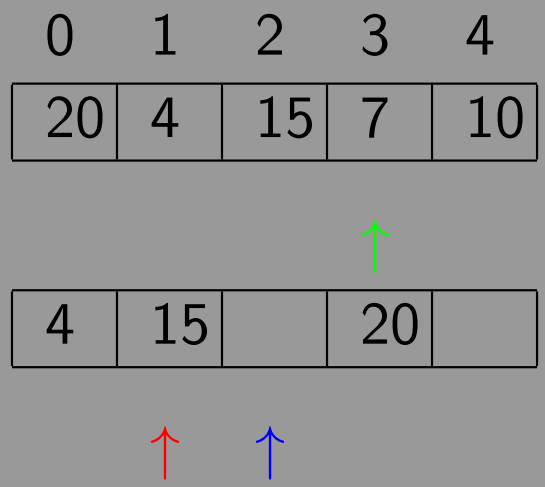
Insertion Sort



$ordenando = 7$

- $j = i - 1$
- $7 < 15$? Sim

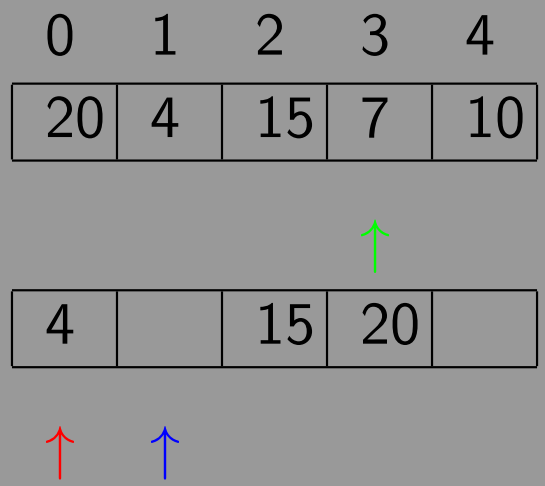
Insertion Sort



$ordenando = 7$

- $j = i - 1$
- $7 < 15$? Sim
- copia 15 para a posição $i - 1$

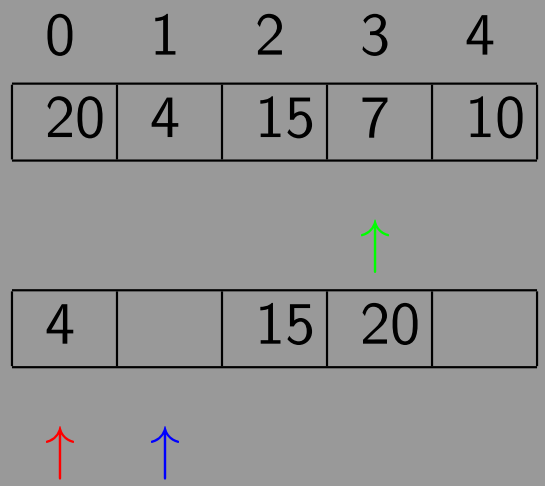
Insertion Sort



$ordenando = 7$

- $j = i - 1$

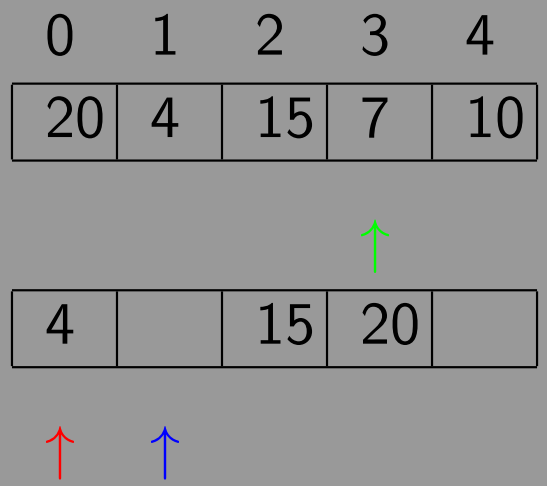
Insertion Sort



$ordenando = 7$

- $j = i - 1$
- $7 < 4$? Não

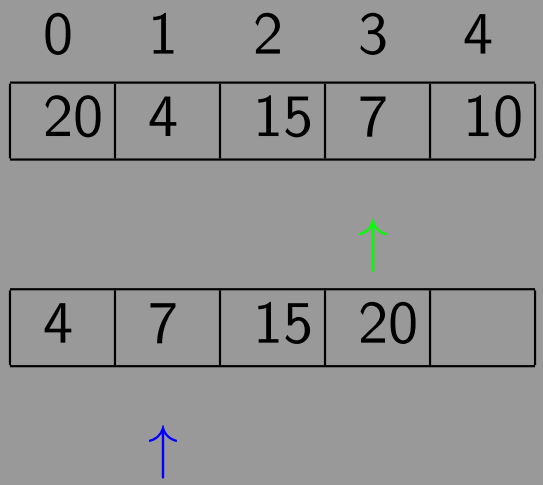
Insertion Sort



$ordenando = 7$

- $j = i - 1$
- $7 < 4$? Não
- acabou iteração

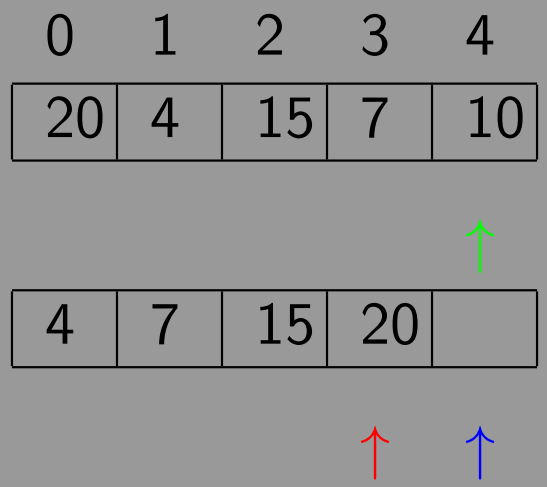
Insertion Sort



$ordenando = 7$

- copia $ordenando$ na posição $j + 1$

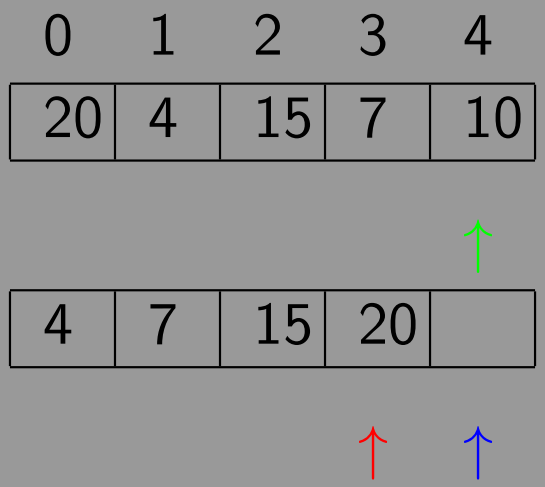
Insertion Sort



$ordenando = 10$

- $i = i + 1$; $j = i - 1$

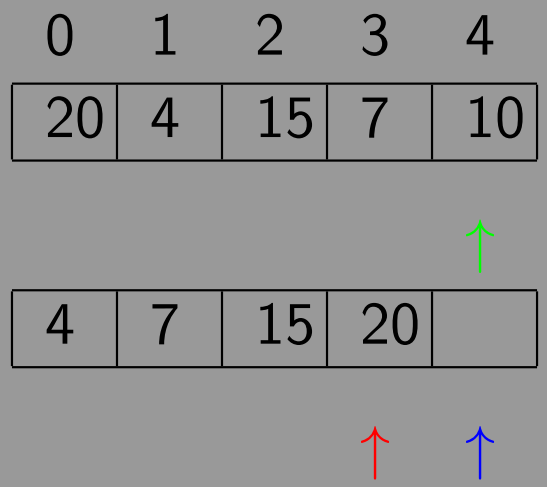
Insertion Sort



$ordenando = 10$

- $i = i + 1$; $j = i - 1$
- $10 < 20$? Sim

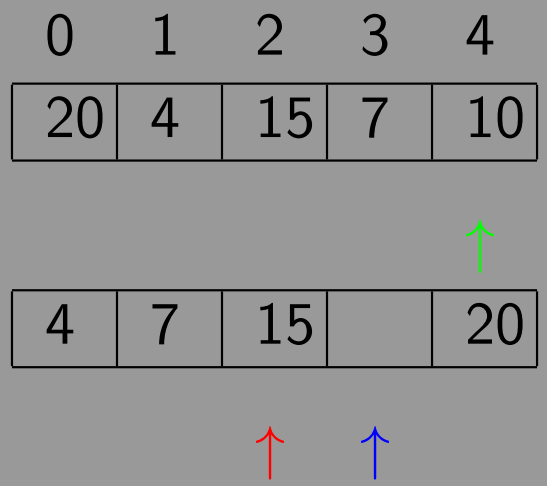
Insertion Sort



$ordenando = 10$

- $i = i + 1$; $j = i - 1$
- $10 < 20$? Sim
- copia 20 para a posição i

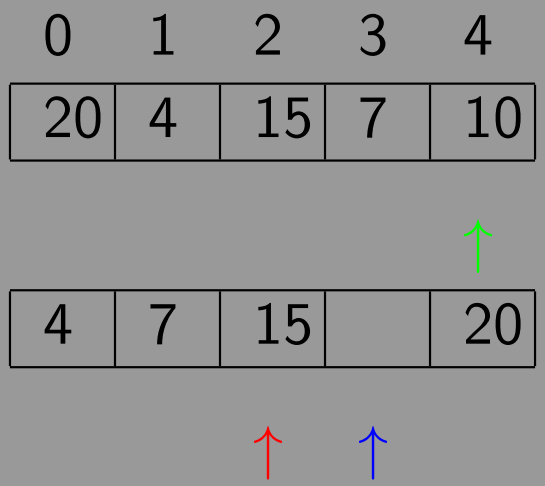
Insertion Sort



$ordenando = 10$

- $j = i - 1$

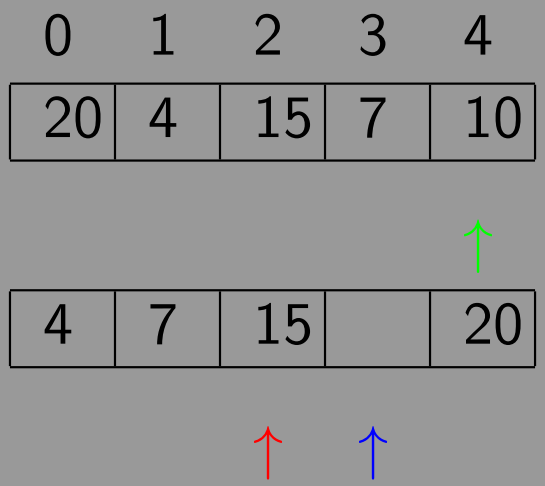
Insertion Sort



$ordenando = 10$

- $j = i - 1$
- $10 < 15$? Sim

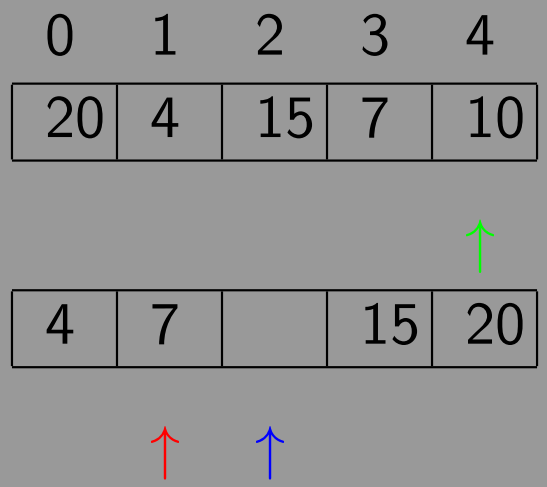
Insertion Sort



$ordenando = 10$

- $j = i - 1$
- $10 < 15$? Sim
- copia 15 para a posição $i - 1$

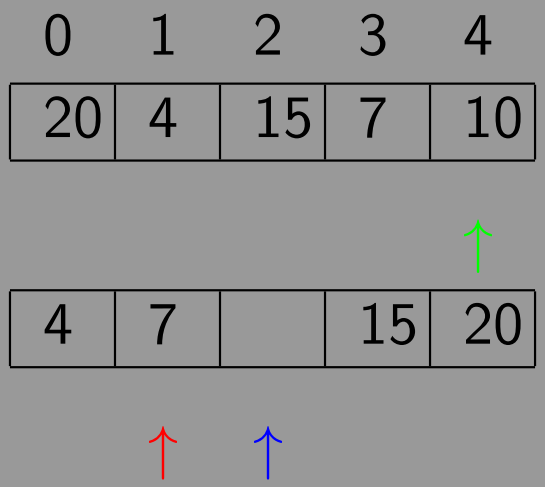
Insertion Sort



$ordenando = 10$

- $j = i - 1$

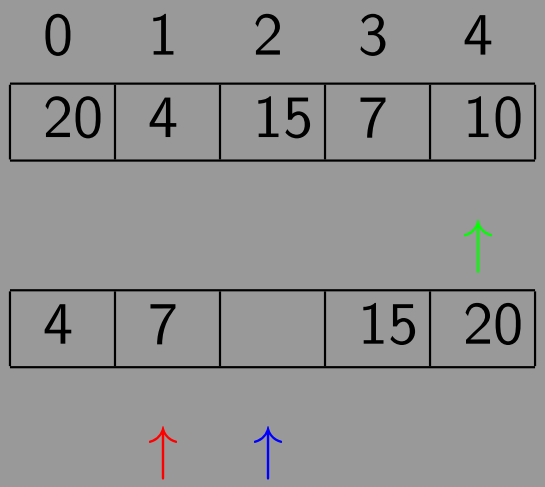
Insertion Sort



$ordenando = 10$

- $j = i - 1$
- $10 < 7$? Não

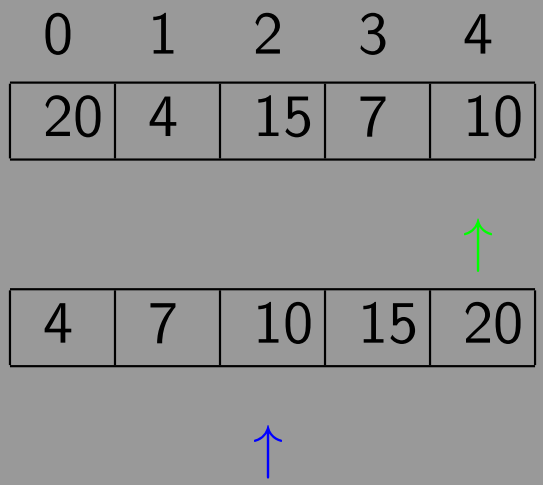
Insertion Sort



$ordenando = 10$

- $j = i - 1$
- $10 < 7$? Não
- acabou iteração

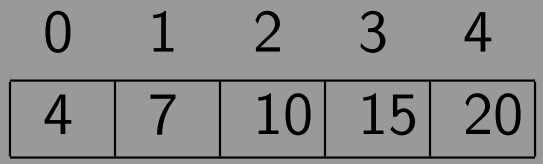
Insertion Sort



$ordenando = 10$

- copia $ordenando$ na posição $j + 1$

Insertion Sort



Acabou! Vetor Auxiliar Ordenado!

Insertion Sort

```
vetor[] = {20, 4, 15, 7, 10}

for (int i = 0; i < array.length; i++)
    int a = array[i]
    for (int j = i - 1; j >= 0 & array[j] > a; j--)
        array[j + 1] = array[j]
        array[j] = a
```

Exercícios

1. Implemente o método de ordenação Selection Sort.
2. Implemente o método de ordenação Insertion Sort.
3. Teste todos os algoritmos em um algoritmo principal

Referências Bibliográficas

1. Cormen, Thomas H., Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. "Introduction to algorithms second edition." (2001).
2. Tamassia, Roberto, and Michael T. Goodrich. "Estrutura de Dados e Algoritmos em Java." Porto Alegre, Ed. Bookman 4 (2007).
3. Ascencio, Ana Fernanda Gomes, and Graziela Santos de Araújo. "Estruturas de Dados: algoritmos, análise da complexidade e implementações em JAVA e C/C++." São Paulo: Perarson Prentice Halt 3 (2010).