

• Insertion Sort (Buckets)

→ Ordenação por Inserção

→ Ideia:

- similar a ordenação de cartas de baralho com as mãos
- pega - se uma carta de cada vez e a coloca em seu devido lugar, sempre deixando as cartas de mão em ordem

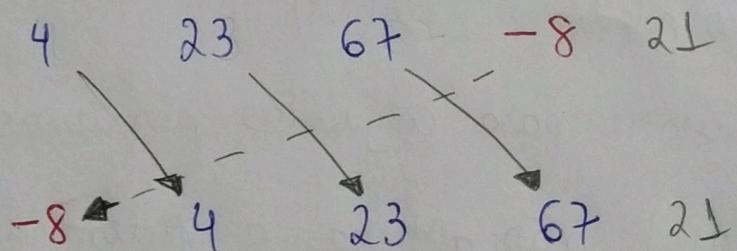
→ Performance

- melhor caso: $O(N)$
- pior caso : $O(N^2)$
- caso médio : $O(N^2)$
- Eficiente para conjuntos pequenos de dados
- Estável: não altera a ordem de dados iguais
- Capaz de ordenar os dados à medida em que os recebe (tempo real)

- Funcionamento

→ o algoritmo percorre o array e para cada posição X verifica-se o seu valor está na posição correta:

- isso é feito andando para o começo do array a partir da posição X e movimentando uma posição para frente os valores que são maiores do que o valor da posição X
- desse modo, teremos uma posição livre para inserir o valor da posição X em seu devido lugar



- Pseudocódigo

Insertion Sort (V, N)

$\rightarrow i, j$, eleito

Para i de 1 até $N-1$ faça:

eleito = $A[i]$

$j = i - 1$

Enqto $((j \geq 0) \wedge (\text{eleito} < A[j]))$ faça:

$A[j+1] = A[j]$

$j = j - 1$

$A[j+1] = \text{eleito}$

// fim para

fim algo

• Algoritmo (em C)

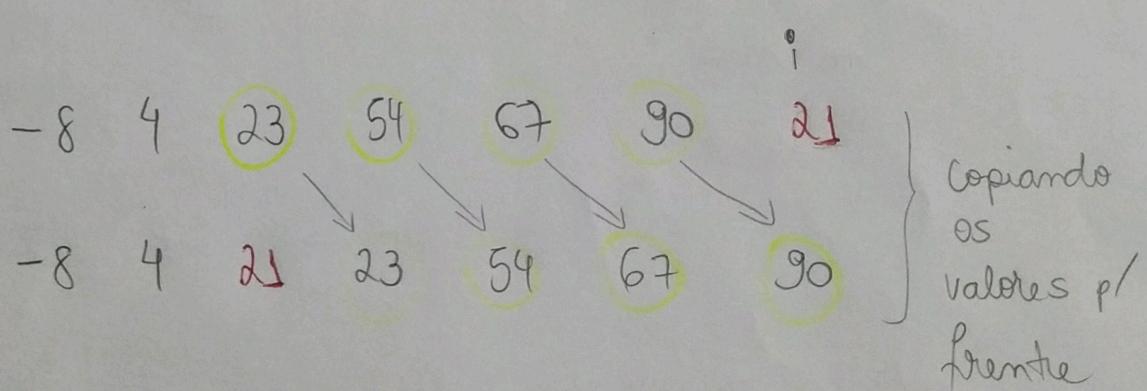
```

void InsertionSort( int *v, int n ) {
    int i, j, aux;
    for( i=1; i<n; i++ ) {
        aux = v[i];
        for( j=i; (j>0) && (aux < v[j-1]); j-- );
        v[j] = aux;
    }
}

```

move os
 cartas maiores
 para a frente

j volta



• Exemplo:

vetor não-ordenado						
23	4	67	-8	90	54	21

• iteração $i = 1$

$\leftarrow j$	i	23	4	67	-8	90	54	21	trocar!
4		23	67	-8	90	54	21		

• iteração $i = 2$

$\leftarrow j$	i	4	23	67	-8	90	54	21	ok
4		23	67	-8	90	54	21		

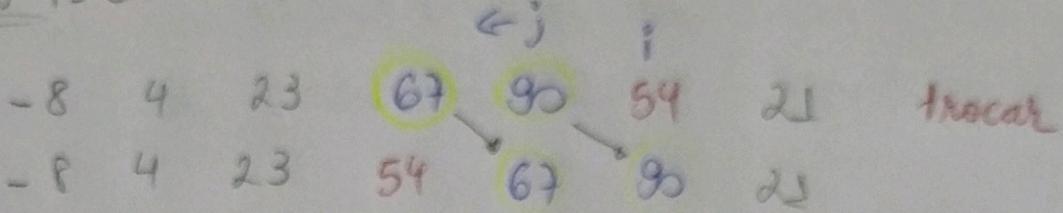
• iteração $i = 3$

$\leftarrow j$	i	4	23	67	-8	90	54	21	trocar
-8		4	23	67	-8	90	54	21	

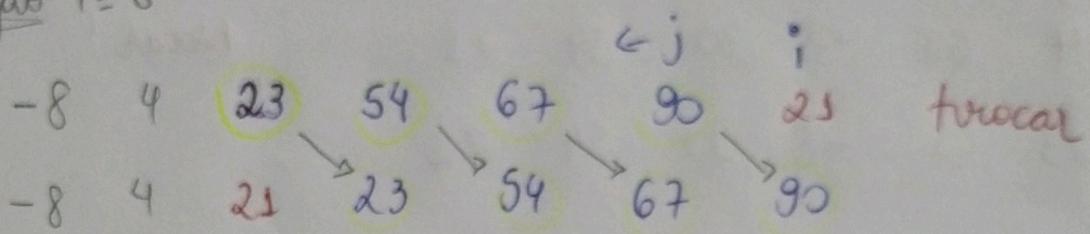
• iteração $i = 4$

$\leftarrow j$	i	-8	4	23	67	90	54	21	ok
-8		4	23	67	90	54	21		

1º troca $i = 5$



2º troca $i = 6$



* Vetor Ordenado

-8	4	21	23	54	67	90
0	1	2	3	4	5	6

• Observações Gerais

→ Fácil Implementação

→ Na prática é mais eficiente que a maioria dos algoritmos de ordem quadrática, como Selection e Bubble Sort

→ Um dos mais rápidos algoritmos para conjuntos pequenos (superando quick sort)

→ Estável: não altera a ordem de dados iguais

→ Online: pode ordenar elementos a medida que os recebe

- Estável: não altera a ordem de dados iguais
- Online: pode ordenar elementos a medida que os recebe