

PESQUISA

Goiânia 2024

Professor: Jacson Rodrigues

Aluno: Hugo Pereira Borges (hugoborges@discente.ufg.br)

O algoritmo de ordenação por inserção inicia tomando o elemento da primeira posição da lista como ordenado e, em seguida, compara cada um dos elementos subsequentes com os elementos já ordenados, colocando-os em sua posição correta entre os ordenados.

```

1  Receba um vetor com  $n$  elementos.
2  Para  $i = 1$  até  $n-1$  faça:
3      atual = vetor[i]
4       $j = i - 1$ 
5      Enquanto  $j > 0$  e vetor[j] > atual faça:
6          vetor[j + 1] = vetor[j]
7           $j = j - 1$ 
8      fim-enquanto
9      A[j + 1] = atual
10 fim-para
11 Retorne o vetor ordenado.

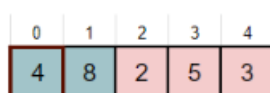
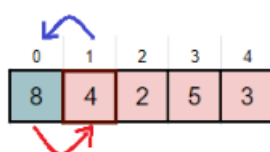
```

O *insertion sort* divide a lista em duas partes, uma ordenada e a outra desordenada. Assim, ele vai pegando sempre o primeiro elemento da lista desordenada e colocando em sua posição correta na lista ordenada

Primeira iteração

No início, são consideradas duas sublistas:

- lista ordenada: [8],
- lista não ordenada: [4, 2, 5, 3],



atual = 4;
Insira o elemento atual na posição correta entre os elementos que estão à sua esquerda (ordenados).

Comparações e movimentações:
 $8 > 4$? SIM, então $\text{vetor}[1] = 8$;
 $\text{vetor}[0] = \text{atual} = 4$;

O elemento inserido fará parte da lista ordenada:

- lista ordenada: [4, 8],
- lista não ordenada: [2, 5, 3],

```

package main

import "fmt"

func main() {
    var N int
    fmt.Scanln(&N)
    elementos := make([]int, N)
    for l := 0; l < N; l++ {
        fmt.Scan(&elementos[l])
    }
    for i := 1; i < N; i++ {
        atual := elementos[i]
        j := i - 1
        for j >= 0 && elementos[j] > atual {
            elementos[j+1] = elementos[j]
            j--
        }
        elementos[j+1] = atual
    }
    fmt.Println(elementos)
}

```

Fontes : <https://www.estrategiaconcursos.com.br/blog/algorithmo-ordenacao-insercao/>