# Resumo sobre o Algoritmo Insertion Sort

Aluna: Luyze Paula Bueno Rodrigues

Videira, outubro de 2025

## Funcionamento

O algoritmo Insertion Sort organiza uma lista de elementos separando-a em duas partes: uma ordenada e outra ainda desordenada. Inicialmente, apenas o primeiro item é considerado ordenado. A cada passo, o próximo elemento da parte desordenada é comparado e inserido na posição correta dentro da sequência já ordenada. Esse processo continua até que toda a lista esteja em ordem crescente.

## Exemplo ilustrativo

Considere o vetor: [8, 3, 5, 2, 9]  
  
1. [8] | [3, 5, 2, 9] → (8 já está na parte ordenada)  
2. [3, 8] | [5, 2, 9] → 3 inserido antes de 8  
3. [3, 5, 8] | [2, 9] → 5 colocado na posição correta  
4. [2, 3, 5, 8] | [9] → 2 movido para o início  
5. [2, 3, 5, 8, 9] → vetor totalmente ordenado.

## Exemplo de Implementação em C#

public class InsertionSorter  
{  
 public static void Ordenar<T>(T[] vetor) where T : IComparable<T>  
 {  
 for (int i = 1; i < vetor.Length; i++)  
 {  
 T atual = vetor[i];  
 int j = i - 1;  
  
 while (j >= 0 && vetor[j].CompareTo(atual) > 0)  
 {  
 vetor[j + 1] = vetor[j];  
 j--;  
 }  
  
 vetor[j + 1] = atual;  
 }  
 }  
}

## Complexidade Computacional

• Melhor caso: O(n) — ocorre quando a lista já está ordenada.  
• Pior caso: O(n²) — acontece quando os elementos estão em ordem inversa.  
• Caso médio: O(n²) — cada elemento precisa ser comparado com parte da lista já ordenada.

Apesar de sua baixa eficiência para listas grandes, o Insertion Sort é útil para coleções pequenas ou quase ordenadas, além de ser fácil de implementar.

## Resumo curto

O Insertion Sort é um método de ordenação simples que constrói gradualmente uma lista ordenada, inserindo cada novo elemento na posição adequada. É intuitivo, mas pouco eficiente em grandes volumes de dados.