

# Algorithm Design Techniques

## Column 8 - Programming Pearls

Sérgio Alexandre Gobbi Rebello  
Gabriel Henrique Garces de Deus  
Miguel Büge Paganini

26 de fevereiro de 2026

# Introdução

O design de algoritmos influencia diretamente o desempenho de um programa.

Este capítulo mostra como diferentes algoritmos podem resolver o mesmo problema com eficiências muito diferentes.

Objetivo:

- ▶ Demonstrar a importância da escolha do algoritmo
- ▶ Comparar diferentes complexidades
- ▶ Mostrar evolução de soluções

# O Problema

Problema estudado no capítulo:

Encontrar a maior soma de um subvetor contínuo dentro de um vetor de números.

Entrada:

- ▶ Vetor de números reais

Saída:

- ▶ Maior soma possível de um subvetor contínuo

Se todos os valores forem negativos, o resultado é zero.

## Algoritmo 1 — $O(n^3)$

- ▶ Testa todos os subvetores possíveis
- ▶ Muito custoso computacionalmente

```
int maxsum1(int x[], int n) {  
    int maxsofar = 0;  
  
    for (int i = 0; i < n; i++) {  
        for (int j = i; j < n; j++) {  
            int sum = 0;  
  
            for (int k = i; k <= j; k++) {  
                sum += x[k];  
            }  
  
            if (sum > maxsofar)  
                maxsofar = sum;  
        }  
    }  
  
    return maxsofar;  
}
```

## Algoritmo 2 — $O(n^2)$

- ▶ Reaproveita somas anteriores
- ▶ Evita recomputação

```
int maxsum2(int x[], int n) {  
    int maxsofar = 0;  
  
    for (int i = 0; i < n; i++) {  
        int sum = 0;  
  
        for (int j = i; j < n; j++) {  
            sum += x[j];  
  
            if (sum > maxsofar)  
                maxsofar = sum;  
        }  
    }  
  
    return maxsofar;  
}
```

## Algoritmo 3 — $O(n \log n)$

```
int max(int a, int b) {  
    return (a > b) ? a : b;  
}
```

```
int max3(int a, int b, int c) {  
    return max(max(a, b), c);  
}
```

```
int maxsum3(int x[], int l, int u) {  
    if (l > u)  
        return 0;  
  
    if (l == u)  
        return max(0, x[l]);  
  
    int m = (l + u) / 2;  
  
    int lmax = 0, sum = 0;  
  
    for (int i = m; i >= l; i--) {
```

## Algoritmo 4 — $O(n)$

- ▶ Algoritmo linear
- ▶ Apenas uma passagem no vetor

```
int maxsum4(int x[], int n) {  
    int maxsofar = 0;  
    int maxendinghere = 0;  
  
    for (int i = 0; i < n; i++) {  
        if (maxendinghere + x[i] > 0)  
            maxendinghere += x[i];  
        else  
            maxendinghere = 0;  
  
        if (maxendinghere > maxsofar)  
            maxsofar = maxendinghere;  
    }  
  
    return maxsofar;  
}
```

# Comparação de Complexidade

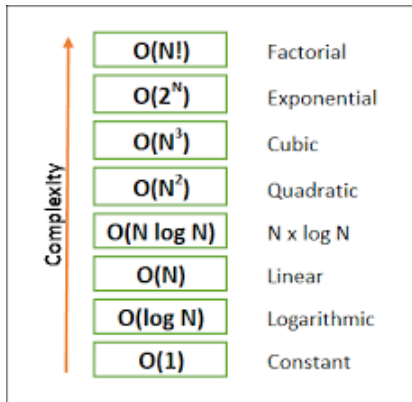


Figura: Complexidade Big O

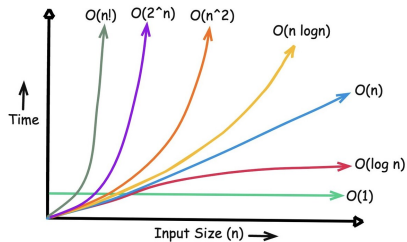


Figura: Gráficos Big O