# Algoritmo Bubble Sort em C - Explicação e Código

## O que é o Bubble Sort?

Bubble Sort é um algoritmo de ordenação simples baseado na repetição de comparações entre elementos adjacentes de um vetor. Se os elementos estiverem fora de ordem, eles são trocados. Esse processo é repetido até que o vetor esteja completamente ordenado.

## Funcionamento do Algoritmo

1. Percorre o vetor comparando elementos adjacentes.

2. Troca os elementos de posição se estiverem fora de ordem.

3. Repete o processo para todos os elementos até que nenhuma troca seja necessária.

4. A cada iteração, o maior elemento "borbulha" para o final do vetor.

## Complexidade do Algoritmo

- Melhor caso (vetor já ordenado): O(n)

- Caso médio: O(n²)

- Pior caso (vetor em ordem reversa): O(n²)

- Espaço adicional: O(1), pois é in-place

## Implementação em C (com ponteiros)

#include <stdio.h>

void bubbleSort(int \*arr, int n) {

for (int i = 0; i < n - 1; i++) {

for (int j = 0; j < n - i - 1; j++) {

if (\*(arr + j) > \*(arr + j + 1)) {

int temp = \*(arr + j);

\*(arr + j) = \*(arr + j + 1);

\*(arr + j + 1) = temp;

}

}

}

}

void imprimir(int \*arr, int n) {

for (int i = 0; i < n; i++)

printf("%d ", \*(arr + i));

printf("\n");

}

int main() {

int vetor[10] = {5, 1, 4, 2, 8, 0, 3, 9, 7, 6};

int n = 10;

printf("Original: ");

imprimir(vetor, n);

bubbleSort(vetor, n);

printf("Ordenado: ");

imprimir(vetor, n);

return 0;

}

## Vantagens e Desvantagens

Vantagens:

- Fácil de implementar.

- Útil para listas pequenas ou já quase ordenadas.

Desvantagens:

- Ineficiente para grandes volumes de dados.

- Muitas comparações desnecessárias em vetores desordenados.

## Referências

- Cormen, T. H. et al. Introduction to Algorithms.

- Sedgewick, R. Algorithms in C.

- Visualização online: https://visualgo.net/en/sorting