# Exercícios Práticos – QuickSort em C

## Exercício 1 – Implementação com Ponteiros

Objetivo:

Implemente o algoritmo QuickSort em C usando ponteiros para manipular um array de inteiros com 20 elementos fornecidos pelo usuário.

Requisitos:

- Utilizar uma função quicksort(int \*arr, int inicio, int fim) com recursão.

- Utilizar a função particionar com ponteiros para troca de posições.

- Usar scanf e printf com aritmética de ponteiros (ex: \*(ptr + i)).

- Exibir o array ordenado após a ordenação.

## Exercício 2 – QuickSort sem Recursão (iterativo com pilha)

Objetivo:

Implemente o algoritmo QuickSort de forma iterativa, sem usar chamadas recursivas. Utilize uma estrutura de pilha para simular a recursão.

Requisitos:

- Criar uma estrutura de pilha (stack) com dois arrays auxiliares: inicio[] e fim[].

- Armazenar os intervalos dos subarrays a serem ordenados.

- Usar laço while para simular a recursão com chamadas de particionar.

Dica:

Use a técnica de simulação de pilha para armazenar os subarrays pendentes de ordenação.

## Exercício 3 – QuickSort com Alocação Dinâmica

Objetivo:

Crie um programa que aloque dinamicamente um vetor de n inteiros, leia os valores, e utilize QuickSort para ordená-los.

Requisitos:

- Usar malloc para criar o vetor com n posições.

- Implementar QuickSort com ponteiros.

- Liberar a memória com free ao final da execução.

Exemplo de entrada esperada:

Digite o número de elementos: 10

Digite os 10 valores: 5 3 7 1 9 2 8 6 0 4

Saída esperada:

Array ordenado: 0 1 2 3 4 5 6 7 8 9