CENTRO UNIVERSITÁRIO ANHANGUERA

ERICK JONATHAN ALVES VIANA DA SILVA

HENRY TERUO KIMURA

**IMPLEMENTAÇÃO DE UM ALGORITMO DE ORDENAÇÃO RÁPIDA EM JAVASCRIPT**

São Paulo

2024

**INTRODUÇÃO**

O algoritmo de ordenação rápida, também conhecido como Quicksort, é um método de ordenação eficiente e amplamente utilizado para ordenar elementos em uma lista. Ele utiliza uma estratégia de divisão e conquista para ordenar os elementos de forma recursiva.

**OBJETIVO**

O objetivo deste código em JavaScript é implementar o algoritmo Quicksort para ordenar uma lista de números fornecidos pelo usuário. Ele demonstra como o Quicksort pode ser aplicado para classificar eficientemente uma grande quantidade de dados.

**EXPLICAÇÃO DA IMPLEMENTAÇÃO**

O código consiste em duas funções principais: partition e quick\_sort. A função partition é responsável por selecionar um pivô e particionar a lista em torno desse pivô, enquanto a função quick\_sort utiliza a recursão para ordenar os subconjuntos da lista.

A função partition recebe uma lista arr, o índice mais baixo low e o índice mais alto high do subarray. Ela seleciona o último elemento como pivô e rearranja os elementos da lista de forma que os elementos menores que o pivô estejam à esquerda e os maiores à direita.

A função quick\_sort recebe os mesmos parâmetros e, utilizando a recursão, ordena os subarrays à esquerda e à direita do pivô.

Após a definição das funções, o código solicita ao usuário uma lista de números separados por espaços, converte essa entrada em uma lista de inteiros e calcula o tamanho da lista.

Em seguida, chama a função quick\_sort para ordenar a lista e imprime os elementos ordenados.

**A ESCOLHA DO ALGORITMO**

O Quicksort é escolhido devido à sua eficiência em média e ao seu desempenho geralmente melhor do que outros algoritmos de ordenação, como o Merge Sort e o Bubble Sort, especialmente para grandes conjuntos de dados.

**RESULTADO**

O resultado do código é a lista fornecida pelo usuário ordenada em ordem crescente, exibida na saída padrão.

Em resumo, este código em Javascript demonstra de forma clara e concisa a implementação do algoritmo de ordenação rápida (Quicksort), oferecendo uma solução eficiente para ordenação de listas de números.

**REFERÊNCIAS**

COELHO, Hebert; FÉLIX, Nádia. **Métodos de Ordenação**: Quick, Radix, Couting, Bucket (Sort). Disponível em: <https://ww2.inf.ufg.br/~hebert/disc/aed1/AED1_05_ordenacao2.pdf>. Acesso em: 11 de mai. de 2024.

ALVES, Julia Rezende. **Análise de Desempenho dos Algoritmos de Busca Sequencial e de Busca Binária como Ferramentas de um Sistema de Gerenciamento Bancário.** Disponível em: <https://www.researchgate.net/profile/Julia-Rezende-Alves/publication/325116327_Analise_de_Desempenho_dos_Algoritmos_de_Busca_Sequencial_e_de_Busca_Binaria_como_Ferramentas_de_um_Sistema_de_Gerenciamento_Bancario/>. Acesso em: 12 de mai. de 2024.

Björk, C. (2020). A study of the effects of different contextual variables on sorting algorithms (Dissertation). Disponível em: <https://urn.kb.se/resolve?urn=urn:nbn:se:bth-20528>. Acesso em 13 de mai. de 2024.

GRONER, Loiane. **Learning JavaScript data structures and algorithms**. Packt Publishing Ltd, 2016. Acesso em: <https://books.google.com.br/books?hl=pt-BR&lr=&id=kvxvDQAAQBAJ>. Acesso em 13 de mai. de 2024.