ATIVIDADE 3

QUICKSORT RECURSIVO COM VETORES

Desenvolva um programa capaz de ler um array de n valores (a ser definido pelo usuário). Após a leitura, o programa deve utilizar o algoritmo de quicksort recursivo para ordenar o array em ordem crescente. Cada chamada da função quicksort deve ser exibida ao usuário, assim como o estado do vetor no momento.

É **obrigatório** o desenvolvimento das seguintes subrotinas:

- **le_vetor** - Função responsável pela leitura dos valores de um vetor até que o valor -100 seja lido OU o vetor chegue no tamanho máximo definido pela constante `MAX_ARR`. Deve receber uma referência ao vetor a ser preenchido e retornar a quantidade de valores preenchidos pelo usuário.

- **troca_elementos** - Procedimento que recebe como parâmetros um vetor e dois índices (i e j). A subrotina deve trocar os elementos i e j do vetor de posição.

- **particionar** - Função de apoio do procedimento quicksort. Deve receber como parâmetros um vetor, o começo e o fim de um intervalo a ser considerado. A função deve implementar a seguinte lógica:
    - Escolhe valor central como pivô.
    - Posiciona pivô ao final do vetor.
    - Inicia variável j = -1
    - Para cada valor do vetor, verifica se o valor atual é menor que o pivô. Caso seja, j deve ser incrementado e o valor deve ir para a posição j.
    - Ao final, incrementa j novamente e coloca o pivô na posição j.
    - Retorna j.

- **quicksort -** Procedimento responsável pela implementação do algoritmo quicksort recursivo. Recebe como parâmetros o vetor, o tamanho do vetor, o começo e o fim de um intervalo a ser considerado. Cada vez que a função quicksort for chamada ou sua execução for encerrada, deverão ser exibidos os parâmetros começo e fim e o estado atual do vetor. Deve-se também exibir se trata-se de uma nova chamada à função (">>") ou se a função está se encerrando ("**"). Mais detalhes no exemplo a seguir.

Exemplos:

```
Digite os valores do vetor: 1 2 3 -100
>> Quicksort(começo = 0, fim = 2). Vetor = [1, 2, 3]
>> Quicksort(começo = 0, fim = 0). Vetor = [1, 2, 3]
** Quicksort(começo = 0, fim = 0). Vetor = [1, 2, 3]
>> Quicksort(começo = 2, fim = 2). Vetor = [1, 2, 3]
** Quicksort(começo = 2, fim = 2). Vetor = [1, 2, 3]
** Quicksort(começo = 0, fim = 2). Vetor = [1, 2, 3]

Digite os valores do vetor: 3 2 1 -100
>> Quicksort(começo = 0, fim = 2). Vetor = [3, 2, 1]
>> Quicksort(começo = 0, fim = 0). Vetor = [1, 2, 3]
** Quicksort(começo = 0, fim = 0). Vetor = [1, 2, 3]
>> Quicksort(começo = 2, fim = 2). Vetor = [1, 2, 3]
** Quicksort(começo = 2, fim = 2). Vetor = [1, 2, 3]
** Quicksort(começo = 0, fim = 2). Vetor = [1, 2, 3]

Digite os valores do vetor: 15 200 13 -100
>> Quicksort(começo = 0, fim = 2). Vetor = [15, 200, 13]
>> Quicksort(começo = 0, fim = 1). Vetor = [15, 13, 200]
>> Quicksort(começo = 0, fim = 0). Vetor = [13, 15, 200]
** Quicksort(começo = 0, fim = 0). Vetor = [13, 15, 200]
>> Quicksort(começo = 2, fim = 1). Vetor = [13, 15, 200]
** Quicksort(começo = 2, fim = 1). Vetor = [13, 15, 200]
** Quicksort(começo = 0, fim = 1). Vetor = [13, 15, 200]
>> Quicksort(começo = 3, fim = 2). Vetor = [13, 15, 200]
** Quicksort(começo = 3, fim = 2). Vetor = [13, 15, 200]
** Quicksort(começo = 0, fim = 2). Vetor = [13, 15, 200]
```

Por quê você está aqui? O enunciado tá lá em cima, seu vagabundo!