Resumo das funções e do funcionamento do algoritmo misto

FUNÇÃO: Main

Recebe: (v) O arquivo com o vetor a ser ordenado.

Objetivo: 1 - Ler o arquivo e repassar para um vetor.
2 - Ordenar o vetor por "Ordenação por Inserção" ou "QuickSort".
3 - Gravar o vetor ordenado para um arquivo.

Devolve: (v) Um arquivo do vetor ordenado.

Explicação:\
1 - Declarado "vetor" Vetor que vai receber o valores do arquivo para ordenação.
"entrada" Ponteiro para o arquivo de entrada.
"saida" Ponteiro para o arquivo de saída.
“linhas" Contador para o tamanho do vetor.
"num" Variável para auxilio a atribuição de valores ao vetor, e gravação do vetor ordenado no arquivo de saída.

2 - Contagem do tamanho do vetor "while ((ch = fgetc(entrada)) != EOF)".
3 - Leitura do arquivo passando os valores para o vetor "while (i < linhas)".
4 - Chamada para a ordenação do vetor "QuickInsert(vetor, indA, linhas - 1);".
5 - Gravação do vetor ordenado no arquivo de saída "while (i < linhas)".

FUNÇÃO: QuickInsert

Recebe: (v) O vetor a ser ordenado.
(a) e (b) Determinando o tamanho do vetor.

Objetivo:
1 - Dependendo do tamanho do vetor de entrada, ela escolhe o melhor algoritmo para ordenar o vetor.
2 - Recursivamente ela ordena o vetor pelo melhor algoritmo.

Devolve: (v) O vetor ordenado.

Explicação:
1 - Declarado “m” índice do pivô atualizado no vetor.
2 - Se o tamanho do vetor for menor que k, sendo k o valor ideal para a transição entre os dois algoritmos;
2.1 - Faz a "Ordenação por Inserção" se não faz o "QuickSort".
3 - Escolher o pivô e atualizar o vetor a partir deles "m = Particiona(v, a, b,pivot)”;
4 - Se na instância exite um vetor "if (a < b)".
4.1 - Chamadas recursivas para os elementos a esquerda e direita do pivô "QuickInsert(v, a, m-1); QuickInsert(v, m+1, b);".

FUNÇÃO: Insertion

Recebe: (v) O vetor a ser ordenado.
(a) e (b) Determinando o tamanho do vetor.

Objetivo: Ordenar o vetor pelo algoritmo de "Ordenação por Inserção" recursivamente.

Devolve: (v) O vetor ordenado pelo algoritmo de "Ordenação por Inserção".

Explicação:
1 - Critério de parada "if (a > b - 1)".
2 - Chamada recursiva com o vetor menor "Insertion(v, a, b - 1);".
3 - Chamada da função Insere "Insere(v, a, b);" para comparar e trocar os elementos para todas as instâncias na memória.

FUNÇÃO: Insere

Recebe: (v) O vetor a ser ordenado.
(a) e (b) Determinando o tamanho do vetor (na instância).

Objetivo: 1 - Busca um elemento menor que o ultimo elemento do vetor (na instância).
2 - Comparar e trocar os elementos (se necessário).

Devolve: (v) O vetor previamente ordenado (até a instância).

Explicação:
1 - Declarado "p": Elemento menor que o de referência.
"i": Índice do último elemento do vetor (na instância).
2 - Buscar elemento menor que o último do vetor (na instância) "p = Busca(v[b], v,a,b - 1);".
3 - Garantindo que o pivô não é negativo "if (p >= 0)".
4 - Enquanto não chegar no ultimo índice do vetor "while (i >= (p + 1))".
4.1 - Troca os elementos "Troca(v, i, i - 1);".
4.2 - Diminui o índice de i"i--;".

FUNÇÃO: Busca

Recebe: (x) Valor de referência (Último valor do vetor na instância).
(v) O vetor a ser ordenado.
(a) e (b) Determinando o tamanho do vetor (na instância).

Objetivo: Encontrar o índice da posição i cujo valor v[i] é maior do que x (na instância).

Devolve: Retorna o índice do último elemento que é maior do que o elemento de referência. Nesse caso a leitura do vetor ocorre da direita para a esquerda então quando dizemos o último elemento maior do que o x, na verdade se trata de um elemento que está perto do início do vetor.

Explicação:

Em cada instância compara-se o x, que é o elemento do vetor de índice [i + 1], com todos os elementos v[i] do vetor para i > 0.

FUNÇÃO: Troca

Recebe: (v) O vetor a ser ordenado.

(a) e (b) Índices dos elementos a serem trocados (Índice de (a) menor que (b) e v[a]>v[b]).

Objetivo: Trocar dois elementos do vetor.

Devolve: (v) O vetor com os elementos trocados.

Explicação:

1 - É declarado "temp".
2 - "temp" recebe o valor de v[a], que é maior que v[b].
3 - v[a] recebeu o menor valor no momento, que é v[b].
4 - Finalmente v[b] recebe o maior valor.

FUNÇÃO: Particiona

Recebe: (v) O vetor a ser ordenado.

(a) e (b) Determinando o tamanho do vetor (na instância).
(Pivot) Inteiro onde se armazena o valor do pivô

Objetivo:

1 - Escolher um número para ser o pivô (no caso primeiro elemento do vetor na instância).

2 - A partir desse número encontrar no vetor os número menores que ele, tendo assim, números a esquerda menores que o pivô, e a direita maiores que o pivô.

3 - E essa função nas chamadas recursivas, vai quebrando e parcialmente ordenando o vetor.

Devolve: (iPivot) Índice do pivô atualizado.

Explicação:

1 - Declarado "pivot=v[a];" Pivô recebendo o primeiro elemento do vetor (na instância).
"iPivot=a;" Índice do pivô.
"j=a+1;" j Recebendo o segundo índice do vetor (na instância).
2 - Enquanto j não chegar no final do vetor "while (j <= b)".
2.1 - Se o elemento for menor ou igual que o pivô "if (v[j] <= pivot)".
2.1.1 - Coloca o elemento atrás do pivô "Troca(v, iPivot, j);".
2.1.2 - Atualiza o índice do pivô "iPivot = iPivot + 1;".