DHRUV GARG 16 BCE 1190

Data Structures and Algorithms

LAB 8 – Sorting Algorithms

QUICK SORT

Code for sorting an array using quick sort

```
#include<stdio.h>

void quickSort(int a[10],int b,int c);

int main()
{
   int arr[20], n, i;
   printf("\n\n\t     IMPLEMENTATION OF QUICK SORT\n\n");
   printf("\n\tSize of the array (max size = 20) : ");
   scanf("%d", &n);

   printf("\n\n\tEnter the elements : \n\n\t");
   for(i = 0; i < n; i++)
      {
         scanf("%d", &arr[i]);
         printf("\n\t");
      }

   quickSort(arr, 0, n-1);

   printf("\n\n\tSORTED ARRAY USING QUICK SORT IS : \n\n\t");
   for(i = 0; i < n; i++)
      printf("  %d  ", arr[i]);
   printf("\n\n");
   return 0;
}

void quickSort(int arr[10], int first, int last)
{
```

```
    int pivot, j, temp, i;

    if(first < last)
    {
        pivot = first;
        i = first;
        j = last;

        while(i < j)
        {
            while((arr[i] <= arr[pivot]) && (i < last))
                i++;
            while(arr[j] > arr[pivot])
                j--;
            if(i < j)
            {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        temp = arr[pivot];
        arr[pivot] = arr[j];
        arr[j] = temp;
        quickSort(arr, first, j-1);
        quickSort(arr, j+1, last);
    }
}
```

Screenshot for sorting using Quick sort method

C:\Users\Dhruv\Documents\C\quickSort.exe

```
            IMPLEMENTATION OF QUICK SORT

        Size of the array (max size = 20) : 10

        Enter the elements :

        -67

        84

        12

        59

        231

        -81

        34

        50

        11

        121


        SORTED ARRAY USING QUICK SORT IS :

          -81    -67    11    12    34    50    59    84    121    231

Process returned 0 (0x0)   execution time : 40.025 s
Press any key to continue.
```

MERGE SORT

Code for sorting an array using merge sort

```
#include<stdio.h>

void merge_sort(int a, int b);
void merge_array(int a, int b, int c, int d);
int arr_sort[20];

int main()
{
   int i, n;

   printf("\n\n\t     IMPLEMENTATION OF MERGE SORT\n\n");
   printf("\n\tNumber of elements (max size = 20) : ");
   scanf("%d", &n);
   printf("\n\n\tEnter the elements : \n\n\t");

   for (i = 0; i < n; i++)
   {
      scanf("%d", &arr_sort[i]);
      printf("\n\t");
   }
   merge_sort(0, n - 1);

   printf("\n\n\t     SORTED ARRAY AFTER MERGE SORT : \n\n");
   for (i = 0; i < n; i++)
   {
      printf("\t%d", arr_sort[i]);
   }
   printf("\n\n");
   return 0;
}
```

```
void merge_sort(int i, int j)
{
   int m;

   if (i < j)
   {
      m = (i + j) / 2;
      merge_sort(i, m);
      merge_sort(m + 1, j);
      merge_array(i, m, m + 1, j);
   }
}

void merge_array(int a, int b, int c, int d)
{
   int t[50];
   int i = a, j = c, k = 0;
   while (i <= b && j <= d)
   {
      if (arr_sort[i] < arr_sort[j])
         t[k++] = arr_sort[i++];
      else
         t[k++] = arr_sort[j++];
   }

   while (i <= b)
      t[k++] = arr_sort[i++];

   while (j <= d)
      t[k++] = arr_sort[j++];

   for (i = a, j = 0; i <= d; i++, j++)
      arr_sort[i] = t[j];
}
```

Screenshot for sorting using Merge sort method

C:\Users\Dhruv\Documents\C\mergeSort.exe

```
            IMPLEMENTATION OF MERGE SORT

        Number of elements (max size = 20) : 10

        Enter the elements :

        -34

        68

        93

        12

        -97

        121

        44

        253

        -146

        59


            SORTED ARRAY AFTER MERGE SORT :

        -146    -97     -34     12      44      59      68      93      121     253

Process returned 0 (0x0)   execution time : 40.341 s
Press any key to continue.
```