**DSA LAB**

**Lab Assignment number 18**

**Name:** Aamir Ansari  **Batch:** A **Roll no:** 01

**Aim:** Implement Merge Sort and Quick Sort

**Program:**

#include<stdio.h>

/\*Array to store the list\*/

int array[1000];

/\*Merge Sort\*/

void merge(int first, int last)

{

int mid = (first+last)/2;

int i = first;

int j = mid+1;

int k = first;

int temp[100];

while(i<=mid && j<=last)

{

if(array[i] < array[j])

{

temp[k++] = array[i++];

}

else

{

temp[k++] = array[j++];

}

}

while(i<=mid)

{

temp[k++] = array[i++];

}

while(j<=last)

{

temp[k++] = array[j++];

}

/\*Copy all element to original array\*/

for(int i=first;i<=last;i++)

{

array[i] = temp[i];

}

}

void merge\_sort(int first,int last)

{

/\*Base case - 1 or 0 elements\*/

if(first>=last)

{

return;

}

int mid = (first+last)/2; /\*Divide\*/

/\*Recursively the arrays - first,mid and mid+1,last\*/

merge\_sort(first,mid);

merge\_sort(mid+1,last);

/\*Merge the two parts\*/

merge(first,last);

}

/\*Quick Sort\*/

void quick\_sort(int first,int last)

{

int i, j, pivot, temp;

if(first<last)

{

pivot=first;

i=first;

j=last;

while(i<j)

{

while(array[i]<=array[pivot]&&i<last)

{

i++;

}

while(array[j]>array[pivot])

{

j--;

}

if(i<j)

{

temp=array[i];

array[i]=array[j];

array[j]=temp;

}

}

temp=array[pivot];

array[pivot]=array[j];

array[j]=temp;

quick\_sort(first,j-1);

quick\_sort(j+1,last);

}

}

/\*Print a sorted array\*/

void print\_sorted\_array(int n)

{

int i;

printf("Sorted Array:");

for(i=0;i<n;i++)

{

printf("%d ",array[i]);

}

}

void main()

{

int n,i, choice;

printf("Enter number of elements in the List : ");

scanf("%d", &n);

printf("Enter %d integers\n", n);

for (i = 0; i < n; i++)

{

scanf("%d", &array[i]);

}

printf("Type of sort to perform:\n1Merge Sort\n2.Quick Sort\n3.Exit");

printf("\nEnter the choice to be performed: ");

scanf("%d",&choice);

switch(choice)

{

case 1:

merge\_sort(0,n-1);

print\_sorted\_array(n);

break;

case 2:

quick\_sort(0,n-1);

print\_sorted\_array(n);

break;

case 3:

default:

printf("Thank You!!");

}

}

**Output:**



