Assignment No. 1

Name: Bhavin Ratansing Patil

Roll No.: 26 SEDA

Q.1 Write a program in c to implement Quick and Merge sort using structure array. Structure should contain at least one integer; one string consists of two words.

Quick Sort

Quick sort is an algorithm based on divide and conquers approach in which the array is split into sub arrays and these sub-arrays are recursively called to sort the elements.

Algorithm for Quick sort:

Step 1: Choose the highest index value has pivot

Step 2: Take two variables to point left and right of the list excluding pivot

Step 3: left points to the low index

Step 4: right points to the high

Step 5: while value at left is less than pivot move right

Step 6: while value at right is greater than pivot move left

Step 7: if both step 5 and step 6 does not match swap left and right

Step 8: if left ≥ right, the point where they met is new pivot

Merge Sort:

Merge Sort is a Divide and Conquer algorithm. It splits the input array in half, calls itself for each half, and then merges the two sorted parts. Merging two halves is done with the merge () method. merge (arr, l, m, r) is a key process that assumes arr[l..m] and arr[m+1..r] are both sorted sub-arrays and merges them into one.

Algorithm for Merge sort:

Step 1: Find the middle index of the array.

Middle = 1 + (last – first)/2

Step 2: Divide the array from the middle.

Step 3: Call merge sort for the first half of the array

MergeSort(array, first, middle)

Step 4: Call merge sort for the second half of the array.

MergeSort(array, middle+1, last)

Step 5: Merge the two sorted halves into a single sorted array.

Program:

```c
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef struct
{
    int rollno;
    char name[20];
    float marks;
} stud;

void quickSort(stud[], int, int);
int partition(stud[], int, int);
void mergeSort(stud[], int, int);
void merge(stud[], int, int, int);


int main()
{
    int t[5], n, index = 1;
    printf("Enter the no. of Student: ");
    scanf("%d", &n);
    stud student[n];
    for (int i = 0; i < n; i++)
    {
        printf("Enter Name of Student %d: ", index);
        scanf("\n");
        scanf("%[^\n]%*c", student[i].name);

        printf("Enter Roll No of Student %d: ", index);
        scanf("%d", &student[i].rollno);

        printf("Enter Marks of Student %d: ", index);
        scanf("%f", &student[i].marks);
        printf("\n");
        index++;
    }
```

```
    printf("\nData Before Sorting: \n"); //printing unsorted data

    printf("\n\tName\t|\tRoll No.|\tMarks\n");
    printf("\t---------------------------------\n");
    for (int i = 0; i < n; i++)
    {
        printf("%s\t|\t%d\t|\t%f\n", student[i].name, student[i].rollno,
student[i].marks);
    }

    int choice;
    do
    {
        printf("\nEnter the coice \n1.Quick Sort\t2.Merge Sort\t3.Exit\n\n");
        scanf("%d", &choice);
        switch (choice)
        {
        case 1:
            printf("\nYour data is sorting using Qiuck Sort algorithm....\n");
            quickSort(student, 0, n - 1);
            for (int i = 0; i < n; i++)
            {
                printf("\nName : %s\tRoll No. : %d\tMarks : %f",
student[i].name, student[i].rollno, student[i].marks);
            }
            break;
        case 2:
            printf("\nYour data is sorting using Merge Sort algorithm....\n");
            mergeSort(student, 0, n - 1);
            for (int i = 0; i < n; i++)
            {
                printf("\nName : %s\tRoll No. : %d\tMarks : %f",
student[i].name, student[i].rollno, student[i].marks);
            }
            break;
        case 3:
            exit(1);
        default:
            printf("Enter a valid choice!");
            break;
        }
    } while (choice != 3);
    return 0;
}
// quick sort code
int partition(stud arr[], int low, int high)
{
    int pivot = arr[low].rollno;
```

```
    int i = low + 1;
    int j = high;
    stud temp[3];
    int k = 0;
    do
    {
        while (arr[i].rollno <= pivot)
        {
            i++;
        }

        while (arr[j].rollno > pivot)
        {
            j--;
        }

        if (i < j)
        {
            temp[k] = arr[i];
            arr[i] = arr[j];
            arr[j] = temp[k];
        }
    } while (i < j);

    temp[k] = arr[low];
    arr[low] = arr[j];
    arr[j] = temp[k];
    return j;
}
void quickSort(stud arr[], int low, int high)
{
    int partitionIndex;

    if (low < high)
    {
        partitionIndex = partition(arr, low, high);
        quickSort(arr, low, partitionIndex - 1);
        quickSort(arr, partitionIndex + 1, high);
    }
}
// merge sort code
void merge(stud A[], int low, int mid, int high)
{
    int i, j, k;
    stud B[high + 1];
    i = low;
    j = mid + 1;
    k = low;
```

```
    while (i <= mid && j <= high)
    {
        if (A[i].rollno < A[j].rollno)
        {
            B[k] = A[i];
            k++;
            i++;
        }
        else
        {
            B[k] = A[j];
            k++;
            j++;
        }
    }

    while (i <= mid)
    {
        B[k] = A[i];
        k++;
        i++;
    }
    while (j <= high)
    {
        B[k] = A[j];
        k++;
        j++;
    }
    for (int i = low; i <= high; i++)
    {
        A[i] = B[i];
    }
}
void mergeSort(stud A[], int low, int high)
{
    if (low < high)
    {
        int mid = (low + high) / 2;
        mergeSort(A, low, mid);
        mergeSort(A, mid + 1, high);
        merge(A, low, mid, high);
    }
}
```

Output:

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19043.1348]
(c) Microsoft Corporation. All rights reserved.

Data Before Sorting:

        Name    |       Roll No.|       Marks
        ---------------------------------
Mini Diva       |       26      |       92.000000
Jia Lisa        |       28      |       95.000000
Eva Elfie       |       32      |       96.750000
Elsa Jean       |       11      |       91.199997
Kimmy Gray      |       19      |       89.970001

Enter the coice
1.Quick Sort    2.Merge Sort    3.Exit

1

Your data is sorting using Qiuck Sort algorithm....

Name : Elsa Jean        Roll No. : 11   Marks : 91.199997
Name : Kimmy Gray       Roll No. : 19   Marks : 89.970001
Name : Mini Diva        Roll No. : 26   Marks : 92.000000
Name : Jia Lisa Roll No. : 28   Marks : 95.000000
Name : Eva Elfie        Roll No. : 32   Marks : 96.750000
Enter the coice
1.Quick Sort    2.Merge Sort    3.Exit

2

Your data is sorting using Merge Sort algorithm....

Name : Elsa Jean        Roll No. : 11   Marks : 91.199997
Name : Kimmy Gray       Roll No. : 19   Marks : 89.970001
Name : Mini Diva        Roll No. : 26   Marks : 92.000000
Name : Jia Lisa Roll No. : 28   Marks : 95.000000
Name : Eva Elfie        Roll No. : 32   Marks : 96.750000
Enter the coice
1.Quick Sort    2.Merge Sort    3.Exit

3

E:\DS Lab\Assignment No.1>
```