Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: Summer 2022, B.Sc. in CSE (DAY)

LAB REPORT NO # 03

Course Title: Data Structure Lab
Course Code: CSE 106 **Section: CSE 213 - DA (PC)**

Lab Experiment Name(s):

- Merge Sort in C
- Quick Sort in C

Student Details

Name	ID
Md. Shahidul Islam Prodhan	213902017

Lab Date: 06 July 2022

Submission Date: 22 July 2022

Course Teacher's Name: Ms Farhana Akter Sunny, Senior Lecturer.

[For Teacher's use only: Don't write anything inside this box]

Lab Report Status

Marks:	**Signature:**
Comments:	**Date:**

1. TITLE OF THE LAB EXPERIMENT

Lab Report of Problem-Solving Using merge sort and quick sort in C

2. OBJECTIVES

Implementing merge sort and quick sort in C

3. PROCEDURE/ ANALYSIS / DESIGN

Problem 1: Merge sort

STEPS	PROCEDURES
1	Get the value of elements of an array.
2	Declare left ,right and mid variable.

3	Perform Marge function
	
	margesort(array, left, right)
	margesort(array, left, right)
	If left < right Mid= (left+right)/2
	margesort(array, left, mid)
	margesort(array, mid+1, right)
	marge(array, left, mid, right)

4	In marge function we pass the arr[], l, m, r;
5	Declare two new array L[] & R[] and n1 & n2 n1 = m-l+1 n2 = r-m
6	Copy the element in L[n1] & R[n2] For i-0 to n1 L[i]=arr[l+i] For j-0 to n2 R[i]=arr[m+1+j]

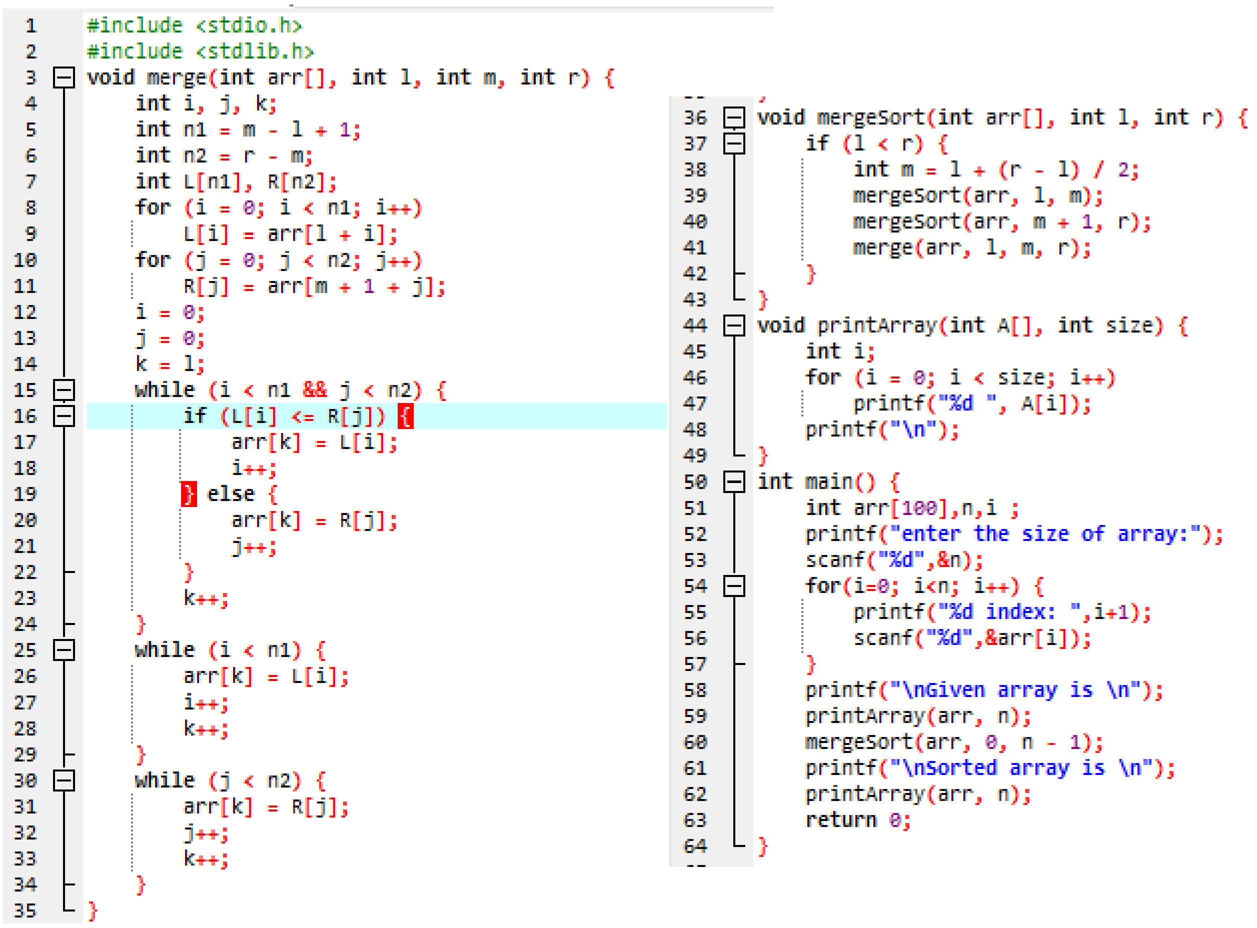
7	i=0,j=0,k=l
	Repeat while (i<= R[j])
	arr[k]=L[i] i++ else arr[k]=R[j]
	j++ end of if k++ end of loop

8	repeat while(i<n1)
	Arr[k]=L[i]
	i++
	K++

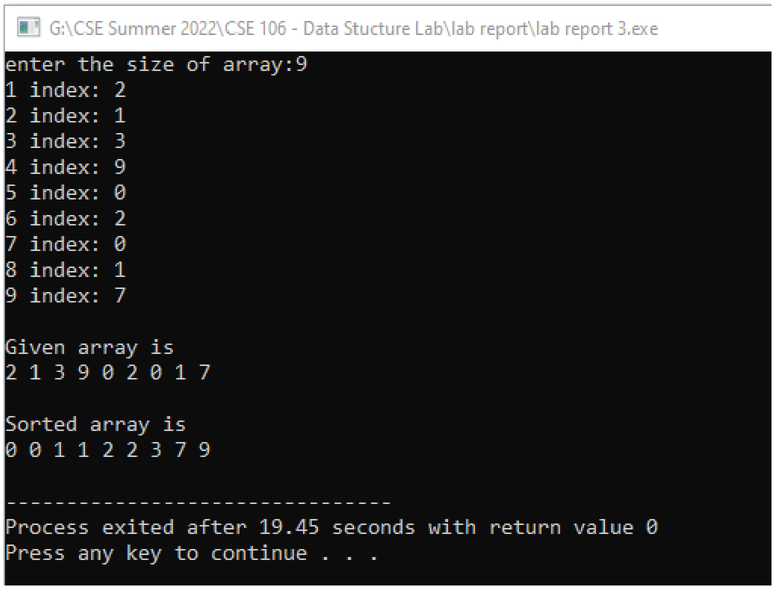
9	Repeat while(j<n2) Arr[k]=R[j] j++ k++
10	Exit

4. IMPLEMENTATION & TEST RESULT

Problem 1: Merge sort



```c
#include <stdio.h>
#include <stdlib.h>
void merge(int arr[], int l, int m, int r) {
    int i, j, k;
    int n1 = m - l + 1;
    int n2 = r - m;
    int L[n1], R[n2];
    for (i = 0; i < n1; i++)
        L[i] = arr[l + i];
    for (j = 0; j < n2; j++)
        R[j] = arr[m + 1 + j];
    i = 0;
    j = 0;
    k = l;
    while (i < n1 && j < n2) {
        if (L[i] <= R[j]) {
            arr[k] = L[i];
            i++;
        } else {
            arr[k] = R[j];
            j++;
        }
        k++;
    }
    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
    }
    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}
void mergeSort(int arr[], int l, int r) {
    if (l < r) {
        int m = l + (r - l) / 2;
        mergeSort(arr, l, m);
        mergeSort(arr, m + 1, r);
        merge(arr, l, m, r);
    }
}
void printArray(int A[], int size) {
    int i;
    for (i = 0; i < size; i++)
        printf("%d ", A[i]);
    printf("\n");
}
int main() {
    int arr[100],n,i ;
    printf("enter the size of array:");
    scanf("%d",&n);
    for(i=0; i<n; i++) {
        printf("%d index: ",i+1);
        scanf("%d",&arr[i]);
    }
    printf("\nGiven array is \n");
    printArray(arr, n);
    mergeSort(arr, 0, n - 1);
    printf("\nSorted array is \n");
    printArray(arr, n);
    return 0;
}
```



G:\CSE Summer 2022\CSE 106 - Data Stucture Lab\lab report\lab report 3.exe

```
enter the size of array:9
1 index: 2
2 index: 1
3 index: 3
4 index: 9
5 index: 0
6 index: 2
7 index: 0
8 index: 1
9 index: 7

Given array is
2 1 3 9 0 2 0 1 7

Sorted array is
0 0 1 1 2 2 3 7 9

--------------------------------
Process exited after 19.45 seconds with return value 0
Press any key to continue . . .
```

1. TITLE OF THE LAB EXPERIMENT

Lab Report of Problem-Solving Using merge sort and quick sort in C

2. OBJECTIVES

Implementing merge sort and quick sort in C

3. PROCEDURE/ ANALYSIS / DESIGN

Problem 1: Quick sort

STEPS	PROCEDURES
1	Get the value of elements of an array.
2	Declare first and last.
3	Perform quicksort function quicksort(number[],first,last) declare I,j,pivot and temp.

4	if (first<last)
	Pivot = first
	i = first
	j = last

4.1	Repeat while (i < j)
	while(number[i]<=number[pivot
	]&&i<last)
	i++
	while(number[j]>number[pivot])
	j—
	if(i<j)
	temp=number[i]
	number[i]=number[j]
	number[j]=temp
	end of if
	end of loop

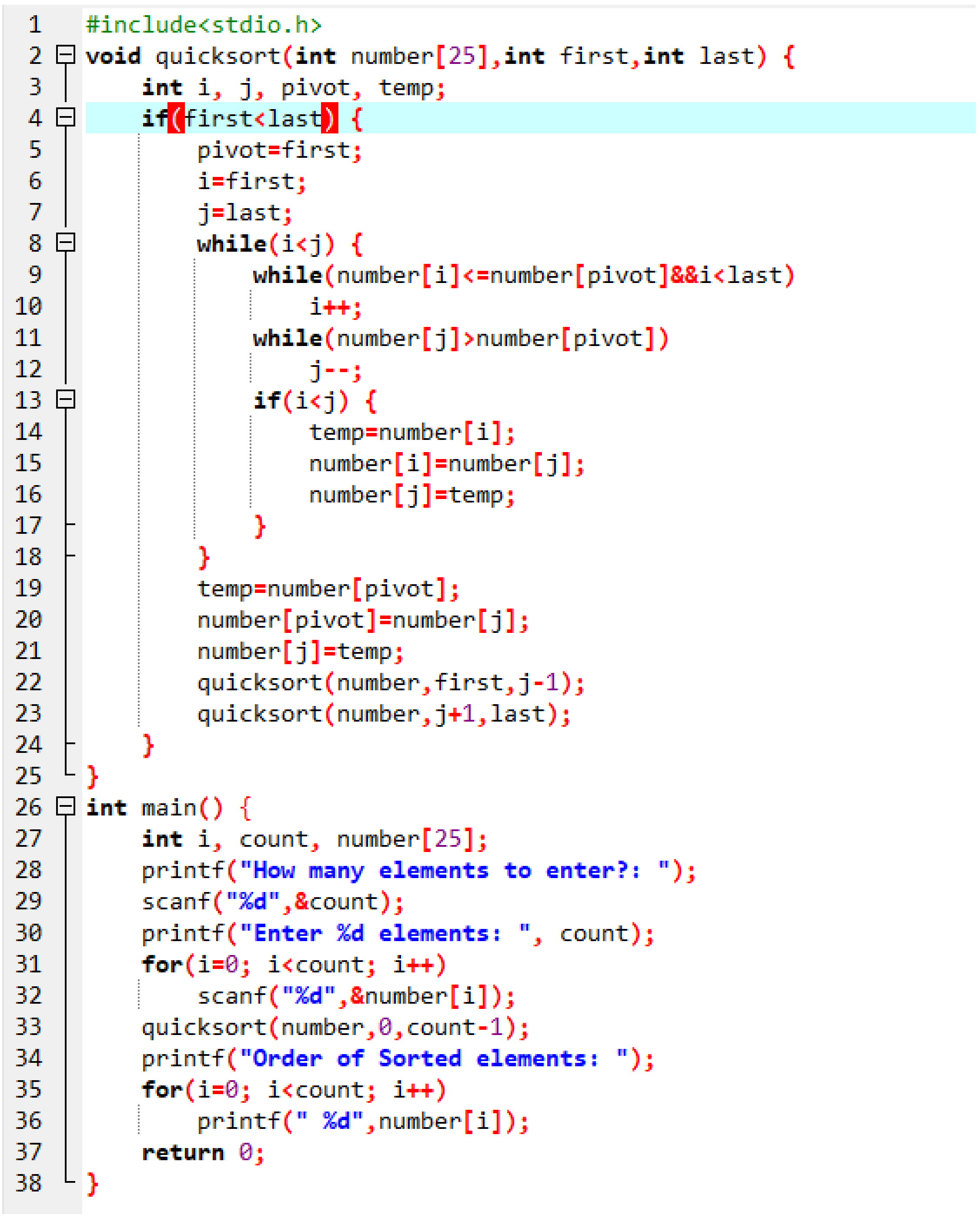
STE PS	PROCEDURES

4.3	temp=number[pivot]
	number[pivot]=number[j]
	number[j]=temp

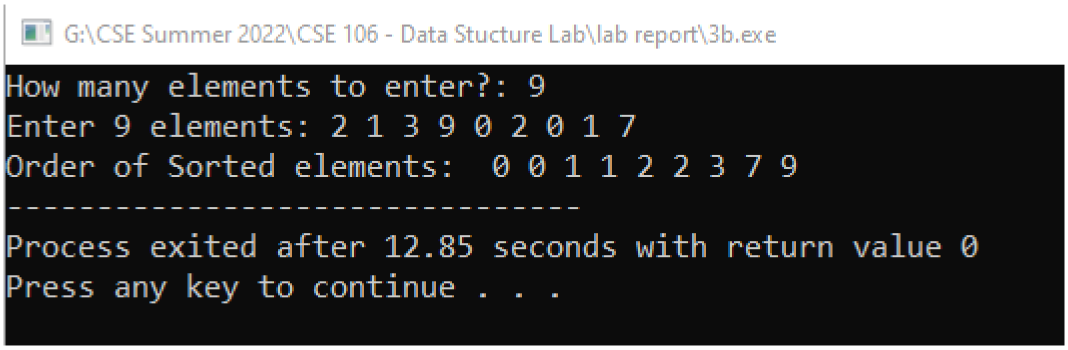
4.3	call quicksort function again
	quicksort(number,first,j-1)
	quicksort(number,j+1,last)

5	Back to main function and print the array
10	Exit

Problem 2: Quick sort



```c
#include<stdio.h>
void quicksort(int number[25],int first,int last) {
    int i, j, pivot, temp;
    if(first<last) {
        pivot=first;
        i=first;
        j=last;
        while(i<j) {
            while(number[i]<=number[pivot]&&i<last)
                i++;
            while(number[j]>number[pivot])
                j--;
            if(i<j) {
                temp=number[i];
                number[i]=number[j];
                number[j]=temp;
            }
        }
        temp=number[pivot];
        number[pivot]=number[j];
        number[j]=temp;
        quicksort(number,first,j-1);
        quicksort(number,j+1,last);
    }
}
int main() {
    int i, count, number[25];
    printf("How many elements to enter?: ");
    scanf("%d",&count);
    printf("Enter %d elements: ", count);
    for(i=0; i<count; i++)
        scanf("%d",&number[i]);
    quicksort(number,0,count-1);
    printf("Order of Sorted elements: ");
    for(i=0; i<count; i++)
        printf(" %d",number[i]);
    return 0;
}
```



G:\CSE Summer 2022\CSE 106 - Data Stucture Lab\lab report\3b.exe

```
How many elements to enter?: 9
Enter 9 elements: 2 1 3 9 0 2 0 1 7
Order of Sorted elements:  0 0 1 1 2 2 3 7 9
--------------------------------
Process exited after 12.85 seconds with return value 0
Press any key to continue . . .
```

6. ANALYSIS AND DISCUSSION

1) This problem is solved by using c program. In this program we implement marge sort. marge-sort is more efficient way to sort the data of an array.
2) This problem is solved by using c program. In this program we implement quick sort. quick-sort is more efficient way to sort the data of an array.

7. SUMMARY

1. marge is more efficient way to sort the data of an array. We done this problem in c programming language .
2. Quick sort is more efficient way to sort the data of an array. We done this problem in c programming language