

Name: Badal Prabhakar Wanjari

Branch: Computer Technology

Section: B

Roll No. 140

Reg No. 20011045

Subject: Data Structures

Practical – 6

Aim: Program for allocating memory dynamically for sing dimensional array and sort it using quick sort and merge sort.

Program:

Sorting Array using quicksort.

```
#include <stdio.h>
#include <stdlib.h>
void quicksort(int* arr, int first, int last){
    if(first<last){
        int pivot = first;
        int i = first;
        int j = last;
        while(i<j){
            while(arr[i]<=arr[pivot] && i < last){
                i++;
            }
            while(arr[j]>arr[pivot]){
                j--;
            }
            if(i<j){
                int temp = arr[i];
                arr[j] = arr[i];
                arr[i] = temp;
            }
        }
        int temp = arr[pivot];
        arr[pivot] = arr[j];
        arr[j] = temp;

        quicksort(arr, first, j-1);
        quicksort(arr, j+1, last);
    }
}

int main(){
    int n;
```

```

printf("Enter length of array: ");
scanf("%d", &n);
int *arr = (int*)malloc(n * sizeof(int));

printf("Enter elements of Array : ");
for(int i=0;i<n;i++){
    scanf("%d", &arr[i]);
}
quicksort(arr, 0, n-1);

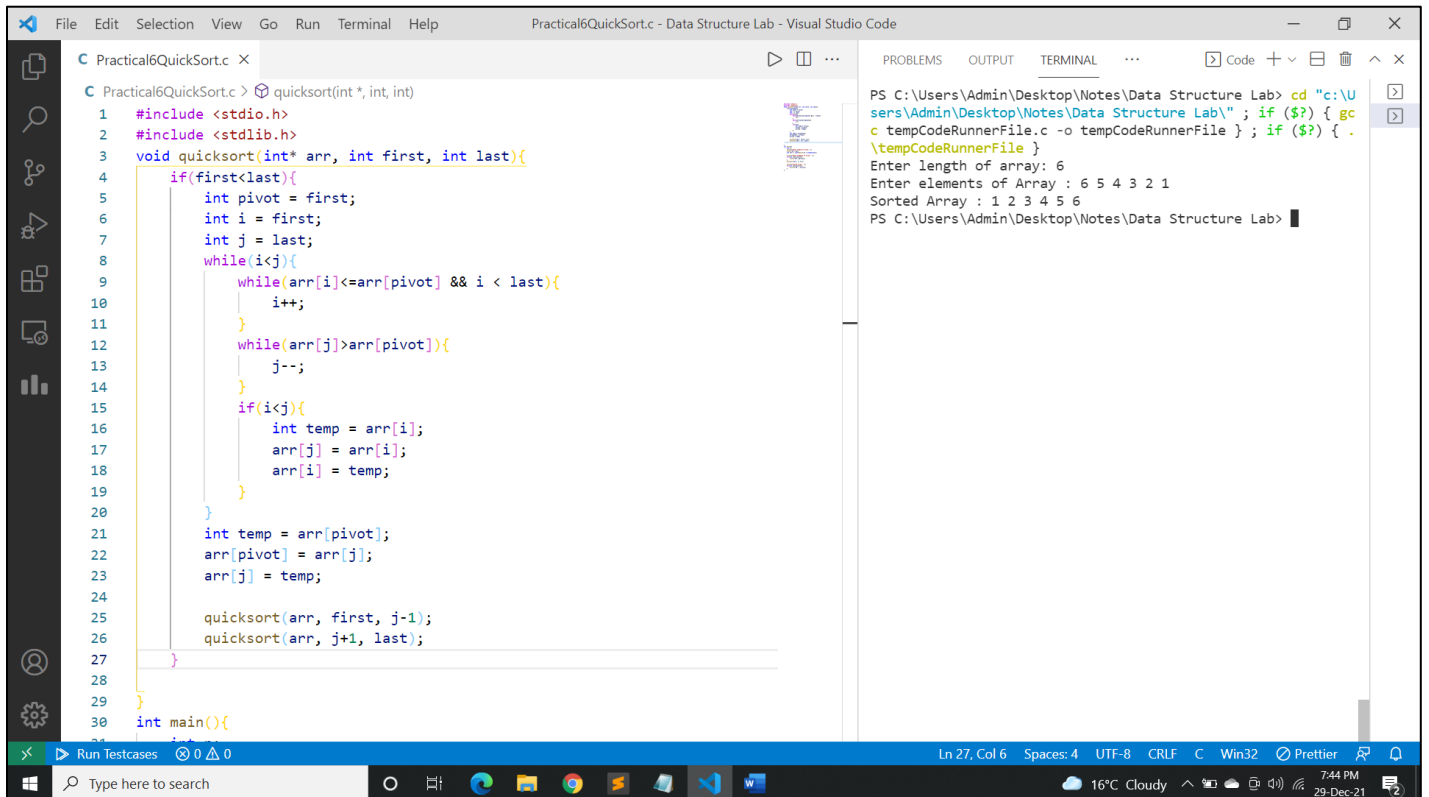
printf("Sorted Array : ");
for(int i=0;i<n;i++){
    printf("%d ", arr[i]);
}
}

```

Output:

Enter length of array: 6
 Enter elements of Array : 6 5 4 3 2 1
 Sorted Array : 1 2 3 4 5 6

Screenshot:



Sorting Array using mergesort.

```
#include <stdio.h>
#include <stdlib.h>
void merge(int *arr, int left, int mid, int right){
    int subArrOne = mid + 1 - left;
    int subArrTwo = right - mid;
    int*leftArr = (int*)malloc(subArrOne*subArrOne);
    int*rightArr = (int*)malloc(subArrTwo*subArrOne);
    for(int i=0;i<subArrOne;i++){
        leftArr[i]=arr[left+i];
    }
    for(int i=0;i<subArrTwo;i++){
        rightArr[i]=arr[mid+1+i];
    }
    int idxLeftArr=0, idxRightArr=0, idxSortedArr = left;
    while(subArrOne>idxLeftArr && subArrTwo>idxRightArr){
        if(leftArr[idxLeftArr]<rightArr[idxRightArr]){
            arr[idxSortedArr] = leftArr[idxLeftArr];
            idxLeftArr++;
            idxSortedArr++;
        }
        else{
            arr[idxSortedArr] = rightArr[idxRightArr];
            idxRightArr++;
            idxSortedArr++;
        }
    }
    while(subArrOne>idxLeftArr){
        arr[idxSortedArr] = leftArr[idxLeftArr];
        idxLeftArr++;
        idxSortedArr++;
    }
    while(subArrTwo>idxRightArr){
        arr[idxSortedArr] = rightArr[idxRightArr];
        idxRightArr++;
        idxSortedArr++;
    }
}

void mergesort(int *arr, int start, int end){
    if(start>=end){
        return;
    }
    int mid = (start + end)/2;
    mergesort(arr, start, mid);
    mergesort(arr, mid+1, end);
    merge(arr, start, mid, end);
}

int main(){
```

```

int n;
printf("Enter length of array: ");
scanf("%d", &n);
int *arr = (int*)malloc(n * sizeof(int));

printf("Enter elements of Array : ");
for(int i=0;i<n;i++){
    scanf("%d", &arr[i]);
}
mergesort(arr, 0, n-1);

printf("Sorted Array : ");
for(int i=0;i<n;i++){
    printf("%d ", arr[i]);
}
}

```

Output:

Enter length of array: 10
 Enter elements of Array : 9 8 7 6 5 4 3 2 1 10
 Sorted Array : 1 2 3 4 5 6 7 8 9 10

Screenshot:

The screenshot displays the Visual Studio Code interface with a C program for Merge Sort. The code is written in `Practical_6_MergeSort.c` and includes standard headers, a recursive merge function, and a mergesort function. The terminal window shows the execution of the program, where the user enters an array length of 10 and the elements 9 8 7 6 5 4 3 2 1 10. The output shows the sorted array: 1 2 3 4 5 6 7 8 9 10.

```

C Practical_6_MergeSort.c x
C Practical_6_MergeSort.c > main()
1  #include <stdio.h>
2  #include <stdlib.h>
3  void merge(int *arr, int left, int mid, int right){
4      int subArrOne = mid + 1 - left;
5      int subArrTwo = right - mid;
6      int*leftArr = (int*)malloc(subArrOne*subArrOne);
7      int*rightArr = (int*)malloc(subArrTwo*subArrOne);
8      for(int i=0;i<subArrOne;i++){
9          leftArr[i]=arr[left+i];
10     }
11     for(int i=0;i<subArrTwo;i++){
12         rightArr[i]=arr[mid+1+i];
13     }
14     int idxLeftArr=0, idxRightArr=0, idxSortedArr = left;
15     while(subArrOne>idxLeftArr && subArrTwo>idxRightArr){
16         if(leftArr[idxLeftArr]<rightArr[idxRightArr]){
17             arr[idxSortedArr] = leftArr[idxLeftArr];
18             idxLeftArr++;
19             idxSortedArr++;
20         }
21         else{
22             arr[idxSortedArr] = rightArr[idxRightArr];
23             idxRightArr++;
24             idxSortedArr++;
25         }
26     }
27     while(subArrOne>idxLeftArr){
28         arr[idxSortedArr] = leftArr[idxLeftArr];
29         idxLeftArr++;
30         idxSortedArr++;
31     }
32     while(subArrTwo>idxRightArr){
33         arr[idxSortedArr] = rightArr[idxRightArr];
34         idxRightArr++;
35         idxSortedArr++;
36     }
37 }
38 void mergesort(int *arr, int left, int right){
39     if(left<right){
40         int mid = (left+right)/2;
41         mergesort(arr, left, mid);
42         mergesort(arr, mid+1, right);
43         merge(arr, left, mid, right);
44     }
45 }
46 int main(){
47     int n;
48     printf("Enter length of array: ");
49     scanf("%d", &n);
50     int *arr = (int*)malloc(n * sizeof(int));
51     printf("Enter elements of Array : ");
52     for(int i=0;i<n;i++){
53         scanf("%d", &arr[i]);
54     }
55     mergesort(arr, 0, n-1);
56     printf("Sorted Array : ");
57     for(int i=0;i<n;i++){
58         printf("%d ", arr[i]);
59     }
60     printf("\n");
61     return 0;
62 }

```

Windows PowerShell
 Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\Users\Admin\Desktop\Notes\Data Structure Lab> cd "c:\Users\Admin\Desktop\Notes\Data Structure Lab" ; if (\$?) { gc c:\Practical_6_MergeSort.c -o Practical_6_MergeSort } ; if (\$?) { .\Practical_6_MergeSort }
 Enter length of array: 10
 Enter elements of Array : 9 8 7 6 5 4 3 2 1 10
 Sorted Array : 1 2 3 4 5 6 7 8 9 10
 PS C:\Users\Admin\Desktop\Notes\Data Structure Lab>

Conclusion: I have successfully completed practical 6.