DESIGN AND ANALYSIS OF ALGORITHMS

LAB WORKBOOK

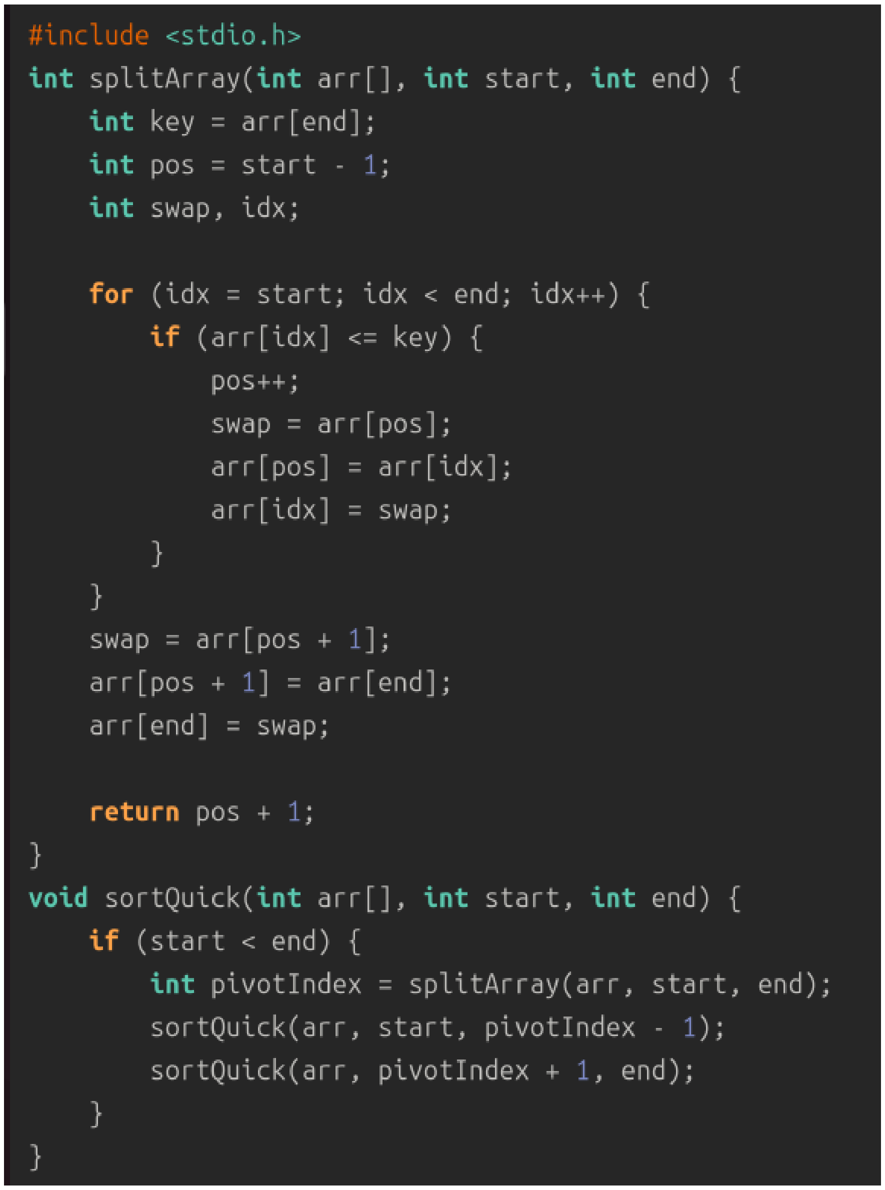
NAME : B.SUMITRANAND SHARMA

ROLL NUMBER : CH.SC.U4CSE24162

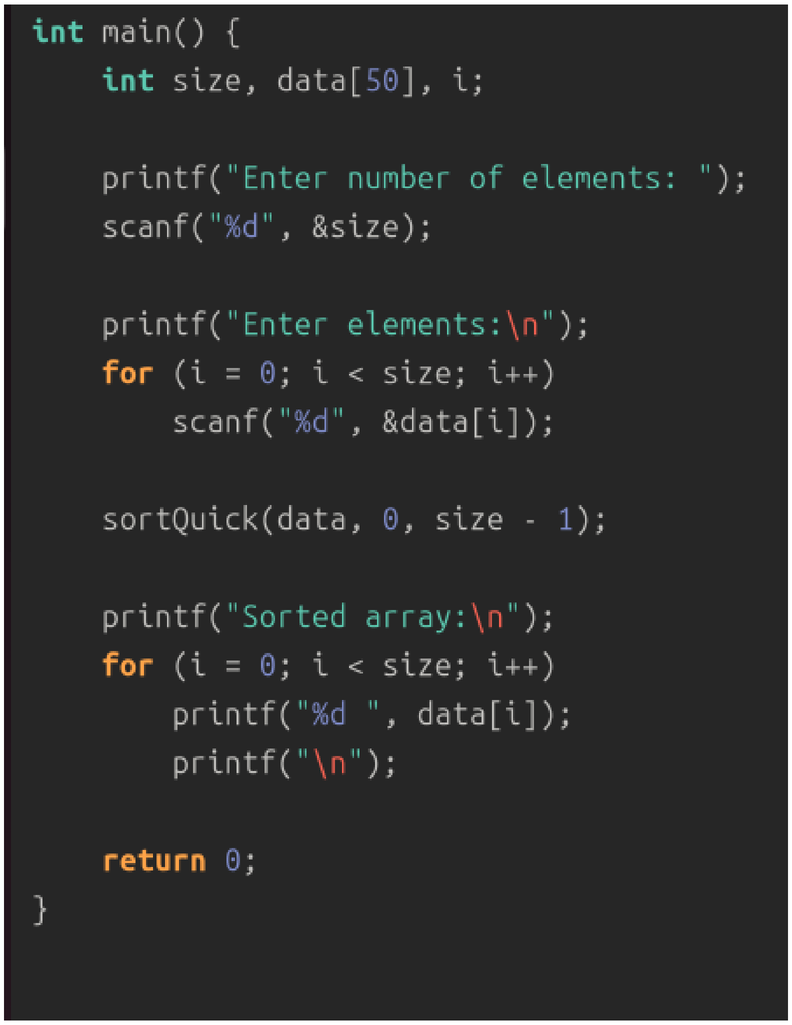
CLASS : CSE-B

1.QUICK SORT

CODE:

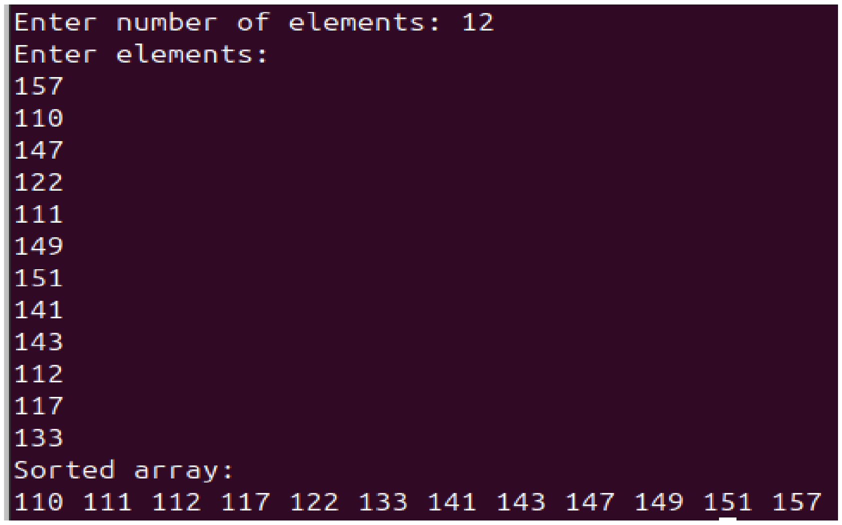


```c
#include <stdio.h>
int splitArray(int arr[], int start, int end) {
    int key = arr[end];
    int pos = start - 1;
    int swap, idx;

    for (idx = start; idx < end; idx++) {
        if (arr[idx] <= key) {
            pos++;
            swap = arr[pos];
            arr[pos] = arr[idx];
            arr[idx] = swap;
        }
    }
    swap = arr[pos + 1];
    arr[pos + 1] = arr[end];
    arr[end] = swap;

    return pos + 1;
}
void sortQuick(int arr[], int start, int end) {
    if (start < end) {
        int pivotIndex = splitArray(arr, start, end);
        sortQuick(arr, start, pivotIndex - 1);
        sortQuick(arr, pivotIndex + 1, end);
    }
}
```



```c
int main() {
    int size, data[50], i;

    printf("Enter number of elements: ");
    scanf("%d", &size);

    printf("Enter elements:\n");
    for (i = 0; i < size; i++)
        scanf("%d", &data[i]);

    sortQuick(data, 0, size - 1);

    printf("Sorted array:\n");
    for (i = 0; i < size; i++)
        printf("%d ", data[i]);
        printf("\n");

    return 0;
}
```

OUTPUT:



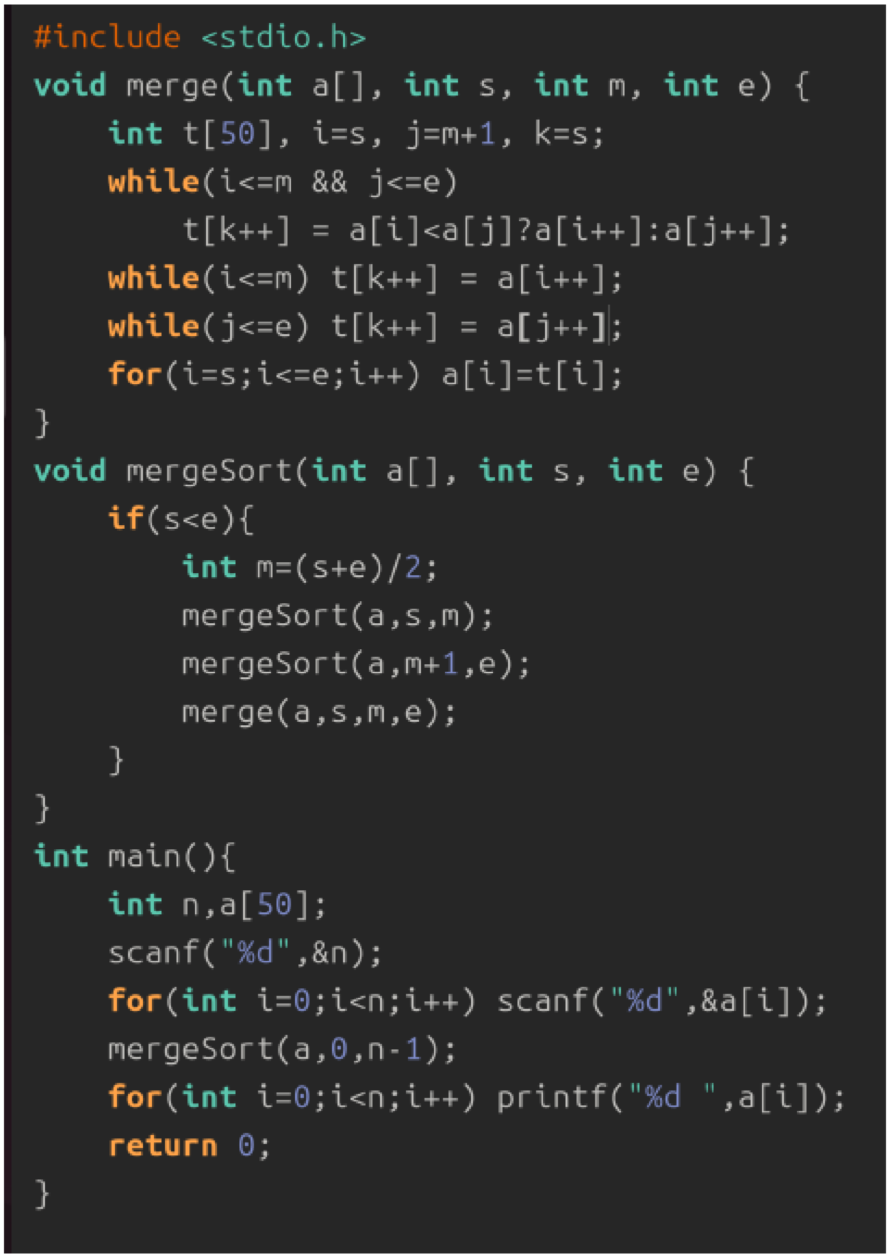
```
Enter number of elements: 12
Enter elements:
157
110
147
122
111
149
151
141
143
112
117
133
Sorted array:
110 111 112 117 122 133 141 143 147 149 151 157
```

Time Complexity

- The pivot divides the array into two equal halves.
- At each level, all n elements are compared once.
- The array is divided log n times.
- So,
  Time = n × log n = O(n log n)

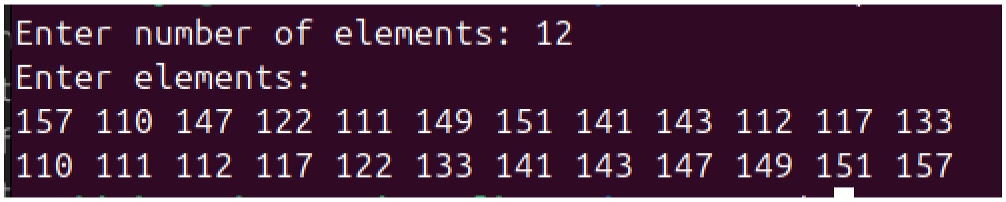
2.MERGE SORT

CODE:



```c
#include <stdio.h>
void merge(int a[], int s, int m, int e) {
    int t[50], i=s, j=m+1, k=s;
    while(i<=m && j<=e)
        t[k++] = a[i]<a[j]?a[i++]:a[j++];
    while(i<=m) t[k++] = a[i++];
    while(j<=e) t[k++] = a[j++];
    for(i=s;i<=e;i++) a[i]=t[i];
}
void mergeSort(int a[], int s, int e) {
    if(s<e){
        int m=(s+e)/2;
        mergeSort(a,s,m);
        mergeSort(a,m+1,e);
        merge(a,s,m,e);
    }
}
int main(){
    int n,a[50];
    scanf("%d",&n);
    for(int i=0;i<n;i++) scanf("%d",&a[i]);
    mergeSort(a,0,n-1);
    for(int i=0;i<n;i++) printf("%d ",a[i]);
    return 0;
}
```

OUTPUT:



```
Enter number of elements: 12
Enter elements:
157 110 147 122 111 149 151 141 143 112 117 133
110 111 112 117 122 133 141 143 147 149 151 157
```

Time Complexity

- The array is always divided into two equal halves.
- Number of divisions = log n.
- At each level, all n elements are merged.
- So,
  Time = n × log n = O(n log n)