Question01.cpp

```cpp
// <--- DSA LAB 8 --->
// <--- Q1 --->

// Write a program to implement a recursive version of merge-sort. Run it for some sample
// data.

#include<iostream>
using namespace std;

void merge(int *arr, int s, int e) {

    int mid = (s+e)/2;

    int len1 = mid - s + 1;
    int len2 = e - mid;

    int *first = new int[len1];
    int *second = new int[len2];

    //copy values
    int mainArrayIndex = s;
    for(int i=0; i<len1; i++) {
        first[i] = arr[mainArrayIndex++];
    }

    mainArrayIndex = mid+1;
    for(int i=0; i<len2; i++) {
        second[i] = arr[mainArrayIndex++];
    }

    //merge 2 sorted arrays
    int index1 = 0;
    int index2 = 0;
    mainArrayIndex = s;

    while(index1 < len1 && index2 < len2) {
        if(first[index1] < second[index2]) {
            arr[mainArrayIndex++] = first[index1++];
        }
        else{
            arr[mainArrayIndex++] = second[index2++];
        }
    }

    while(index1 < len1) {
        arr[mainArrayIndex++] = first[index1++];
    }

    while(index2 < len2 ) {
        arr[mainArrayIndex++] = second[index2++];
    }

    delete []first;
```

```cpp
    delete []second;

}

void mergeSort(int *arr, int s, int e) {

    //base case
    if(s >= e) {
        return;
    }

    int mid = (s+e)/2;

    //left part sort karna h
    mergeSort(arr, s, mid);

    //right part sort karna h
    mergeSort(arr, mid+1, e);

    //merge
    merge(arr, s, e);

}

int main() {

    int arr[15] = {3,7,0,1,5,8000,3,2000,34,66,870,230,128,120,122};
    int n = 15;

    mergeSort(arr, 0, n-1);

    for(int i=0;i<n;i++){
        cout << arr[i] << " ";
    } cout << endl;

    return 0;
}
```

Question02.cpp

```cpp
// <--- DSA LAB 8 --->
// <--- Q2 --->

// Write a program to implement a recursive version of quicksort. Run it for some sample
// data.

#include<iostream>
using namespace std;


int partition( int arr[], int s, int e) {

    int pivot = arr[s];

    int cnt = 0;
    for(int i = s+1; i<=e; i++) {
        if(arr[i] <=pivot) {
            cnt++;
        }
    }

    //place pivot at right position
    int pivotIndex = s + cnt;
    swap(arr[pivotIndex], arr[s]);

    //left and right wala part smbhal lete h
    int i = s, j = e;

    while(i < pivotIndex && j > pivotIndex) {

        while(arr[i] <= pivot)
        {
            i++;
        }

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

        if(i < pivotIndex && j > pivotIndex) {
            swap(arr[i++], arr[j--]);
        }

    }

    return pivotIndex;

}

void quickSort(int arr[], int s, int e) {

    //base case
    if(s >= e)
```

```cpp
        return ;

    //partitioon karenge
    int p = partition(arr, s, e);

    //left part sort karo
    quickSort(arr, s, p-1);

    //right wala part sort karo
    quickSort(arr, p+1, e);

}

int main() {

    int arr[10] = {2,4,1,6,1000,8,5,0,8,10};
    int n = 10;

    quickSort(arr, 0, n-1);

    for(int i=0; i<n; i++)
    {
        cout << arr[i] << " ";
    } cout << endl;


    return 0;
}
```