Design and Analysis of Algorithms

Assignment 1

Write a program to implemenet the following algorithms using divide-and-conquer approach:

Consider an array `arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}` , using Binary Search Method, find the target 23.

Solution:

```cpp
#include <bits/stdc++.h>

using namespace std;

int main() {
        vector<int> v = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91};
        int target = 23;
        int low = 0;
        int high = v.size() - 1;

        int idx = 0;
        bool found = false;

        while(low <= high) {
                int mid = low + (high - low) / 2;
                if(v[mid] == target) {
                        found = true;
                        idx = mid;
                        break;
                }

                else if(v[mid] < target) {
                        low = mid + 1;
                }

                else {
                        high = mid - 1;
                }
        }

        if(found) cout << "Element found at index " << idx << '\n';
        else cout << "Element not found" << "\n";
}
```

Output:

```
Element found at index 5
```

Implement Merge Sort for the given array `int arr[] = {12, 11, 13, 5, 6, 7}`.

```
#include <bits/stdc++.h>

using namespace std;

void merge(vector<int>& v, int l, int m, int r) {
        int n1 = m - l + 1;
        int n2 = r - m;

        vector<int> L(n1);
        vector<int> R(n2);

        for(int i = 0; i < n1; i++) {
                L[i] = v[l + i];
        }

        for(int i = 0; i < n2; i++) {
                R[i] = v[m + 1 + i];
        }

        int i = 0;
        int j = 0;
        int k = l;

        while(i < n1 && j < n2) {
                if(L[i] <= R[j]) {
                        v[k] = L[i];
                        i++;
                } else {
                        v[k] = R[j];
                        j++;
                }
                k++;
        }

        while(i < n1) {
                v[k] = L[i];
                i++;
                k++;
        }

        while(j < n2) {
                v[k] = R[j];
                j++;
                k++;
        }
}

void mergeSort(vector<int>& v, int l, int r) {
        if(l >= r) {
                return;
        }

        int m = l + (r - l) / 2;
        mergeSort(v, l, m);
        mergeSort(v, m + 1, r);
        merge(v, l, m, r);
}

```

```cpp
int main() {
        vector<int> v = {12, 11, 13, 5, 6, 7};
        mergeSort(v, 0, v.size() - 1);
        for(int i : v) {
                cout << i << " ";
        }
}
```

Output:

```
5 6 7 11 12 13
```

Implement Quick Sort for the given array `int arr[] = {3, 2, 6, 9, 2}`.

```cpp
#include <bits/stdc++.h>

using namespace std;

int partition(vector<int> &v, int low, int high) {
        int pivot = v[high];
        int i = low - 1;
        for (int j = low; j < high; j++) {
                if (v[j] < pivot) {
                        i++;
                        swap(v[i], v[j]);
                }
        }
        swap(v[i + 1], v[high]);
        return i + 1;
}

void quick_sort(vector<int> &v, int low, int high) {
        if (low < high) {
                int pi = partition(v, low, high);
                quick_sort(v, low, pi - 1);
                quick_sort(v, pi + 1, high);
        }
}

int main() {
        vector<int> v = {4, 2, 6, 9, 2};
        quick_sort(v, 0, v.size() - 1);

        for (int i = 0; i < v.size(); i++) {
                cout << v[i] << " ";
        }
}
```

Output:

```
2 2 4 6 9
```

You are given a one dimensional array that may contain both positive and negative integers, find the sum of contiguous subarray of numbers which has the largest sum.

```
#include <bits/stdc++.h>

using namespace std;

int main() {
        vector<int> v = {-2, -5, 6, -2, -3, 1, 5, -6};
        int currSum = 0;
        int maxSum = INT_MIN;

        for(int i : v) {
                currSum += i;
                if(currSum < 0) {
                        currSum = 0;
                }
                maxSum = max(maxSum, currSum);
        }

        cout << maxSum;
}
```

Output:

```
7
```