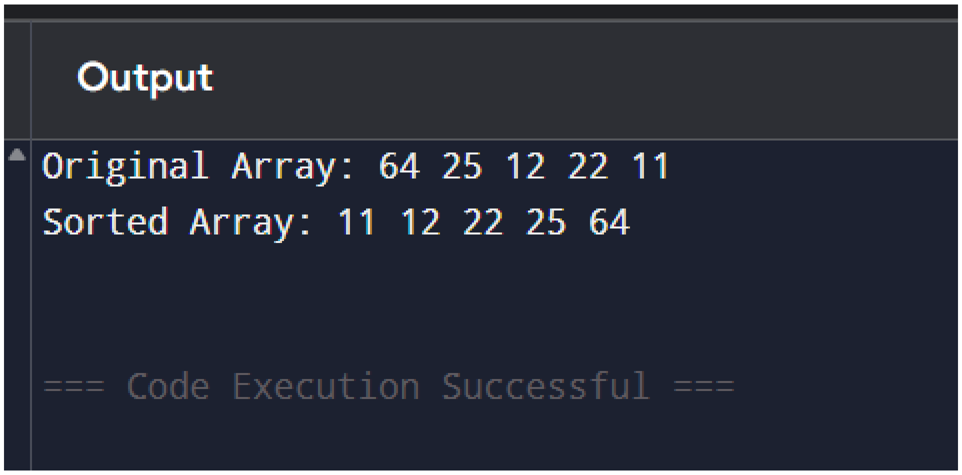
DSA ASSIGNMENT 7

QUESTION 1

```
#include <iostream>
using namespace std;

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}
void selectionSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        int minIndex = i;

        for (int j = i + 1; j < n; j++)
            if (arr[j] < arr[minIndex])
                minIndex = j;

        swap(arr[i], arr[minIndex]);
    }
}

void insertionSort(int arr[], int n) {
    for (int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i - 1;

        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j--;
        }

        arr[j + 1] = key;
    }
}

void bubbleSort(int arr[], int n) {
    for (int i = 0; i < n - 1; i++) {
        bool swapped = false;

        for (int j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                swap(arr[j], arr[j + 1]);
                swapped = true;
            }
        }

        if (!swapped) break;
    }
}
void merge(int arr[], int l, int mid, int r) {
    int n1 = mid - l + 1;
    int n2 = r - mid;

    int a[n1], b[n2];

    for (int i = 0; i < n1; i++) a[i] = arr[l + i];
    for (int i = 0; i < n2; i++) b[i] = arr[mid + 1 + i];

    int i = 0, j = 0, k = l;
    while (i < n1 && j < n2) {
        if (a[i] <= b[j]) arr[k++] = a[i++];
        else arr[k++] = b[j++];
    }

    while (i < n1) arr[k++] = a[i++];
    while (j < n2) arr[k++] = b[j++];
}

void mergeSort(int arr[], int l, int r) {
    if (l >= r) return;

    int mid = l + (r - l) / 2;

    mergeSort(arr, l, mid);
    mergeSort(arr, mid + 1, r);
    merge(arr, l, mid, r);
}
int partition(int arr[], int l, int r) {
    int pivot = arr[r];
    int i = l - 1;

    for (int j = l; j < r; j++) {
        if (arr[j] <= pivot) {
            i++;
            swap(arr[i], arr[j]);
        }
    }

    swap(arr[i + 1], arr[r]);
    return i + 1;
}
```

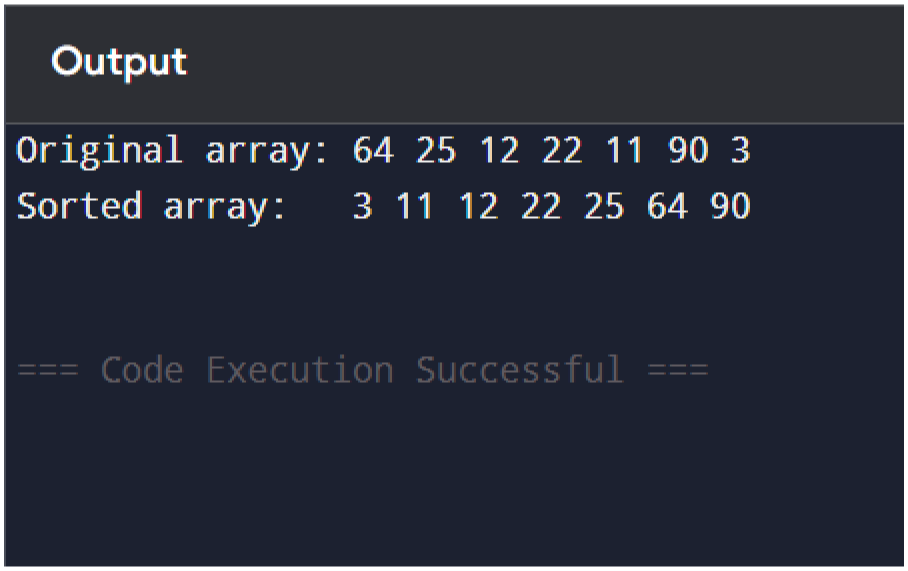
```
void quickSort(int arr[], int l, int r) {
    if (l < r) {
        int pi = partition(arr, l, r);
        quickSort(arr, l, pi - 1);
        quickSort(arr, pi + 1, r);
    }
}
int main() {
    int arr[] = {64, 25, 12, 22, 11};
    int n = 5;

    cout << "Original Array: ";
    printArray(arr, n);
    selectionSort(arr, n);
    // insertionSort(arr, n);
    // bubbleSort(arr, n);

    // mergeSort(arr, 0, n-1);

    quickSort(arr, 0, n-1);

    cout << "Sorted Array: ";
    printArray(arr, n);

    return 0;
}
```



```
Original Array: 64 25 12 22 11
Sorted Array: 11 12 22 25 64


=== Code Execution Successful ===
```

QUESTION 2

```
#include <iostream>
using namespace std;

void improvedSelectionSort(int arr[], int n) {
    int left = 0, right = n - 1;

    while (left < right) {

        int minIndex = left;
        int maxIndex = right;

        if (arr[minIndex] > arr[maxIndex])
            swap(arr[minIndex], arr[maxIndex]);

        for (int i = left + 1; i < right; i++) {

            if (arr[i] < arr[minIndex])
                minIndex = i;

            else if (arr[i] > arr[maxIndex])
                maxIndex = i;
        }

        swap(arr[left], arr[minIndex]);

        if (maxIndex == left)
            maxIndex = minIndex;

        swap(arr[right], arr[maxIndex]);
        left++;
        right--;
    }
}
void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main() {
    int arr[] = {64, 25, 12, 22, 11, 90, 3};
    int n = sizeof(arr) / sizeof(arr[0]);

    cout << "Original array: ";
    printArray(arr, n);

    improvedSelectionSort(arr, n);

    cout << "Sorted array:   ";
    printArray(arr, n);

    return 0;
}
```



Output

```
Original array: 64 25 12 22 11 90 3
Sorted array:   3 11 12 22 25 64 90


=== Code Execution Successful ===
```