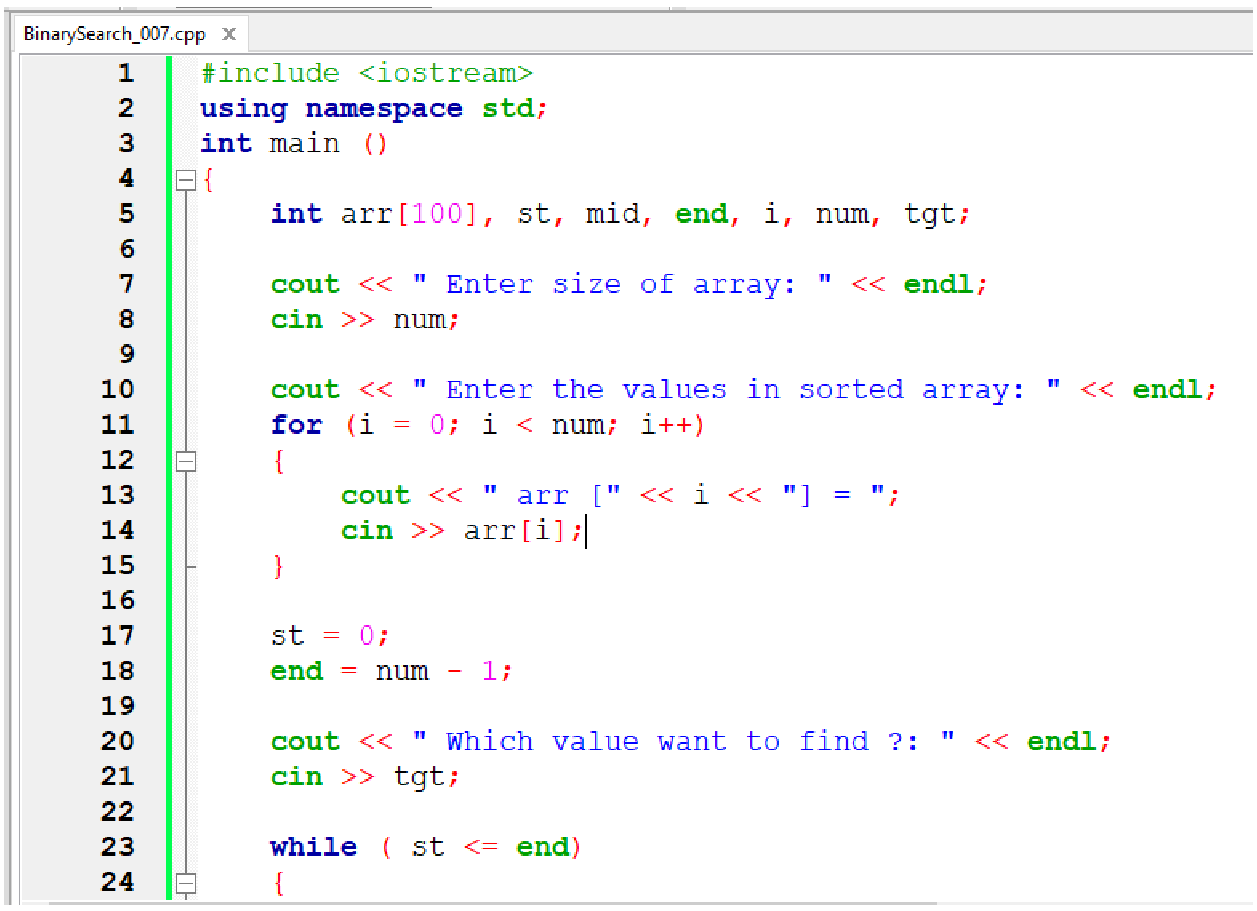
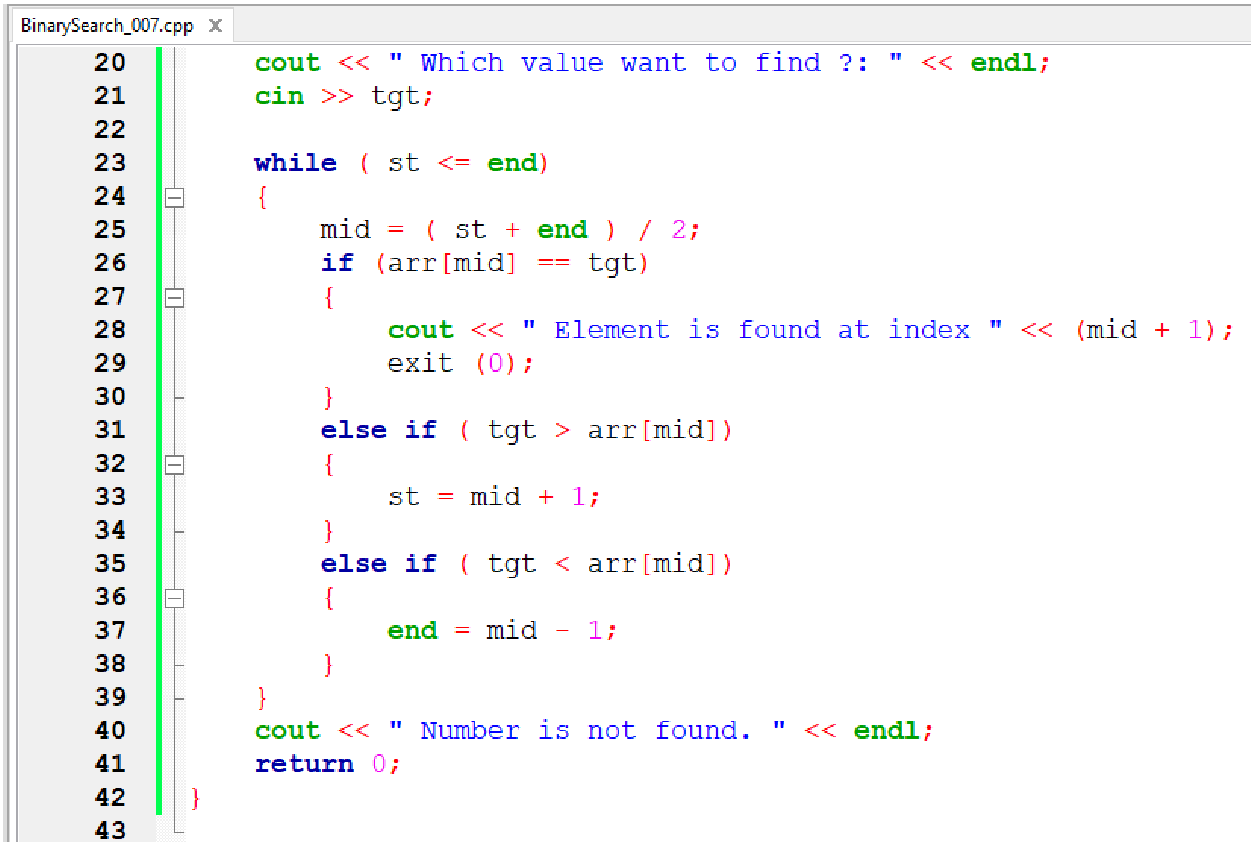
Name : Riyadul Islam
Roll : 007
Section : 1 , Intake : 50
Dept: CSE

Binary Search Algorithm :

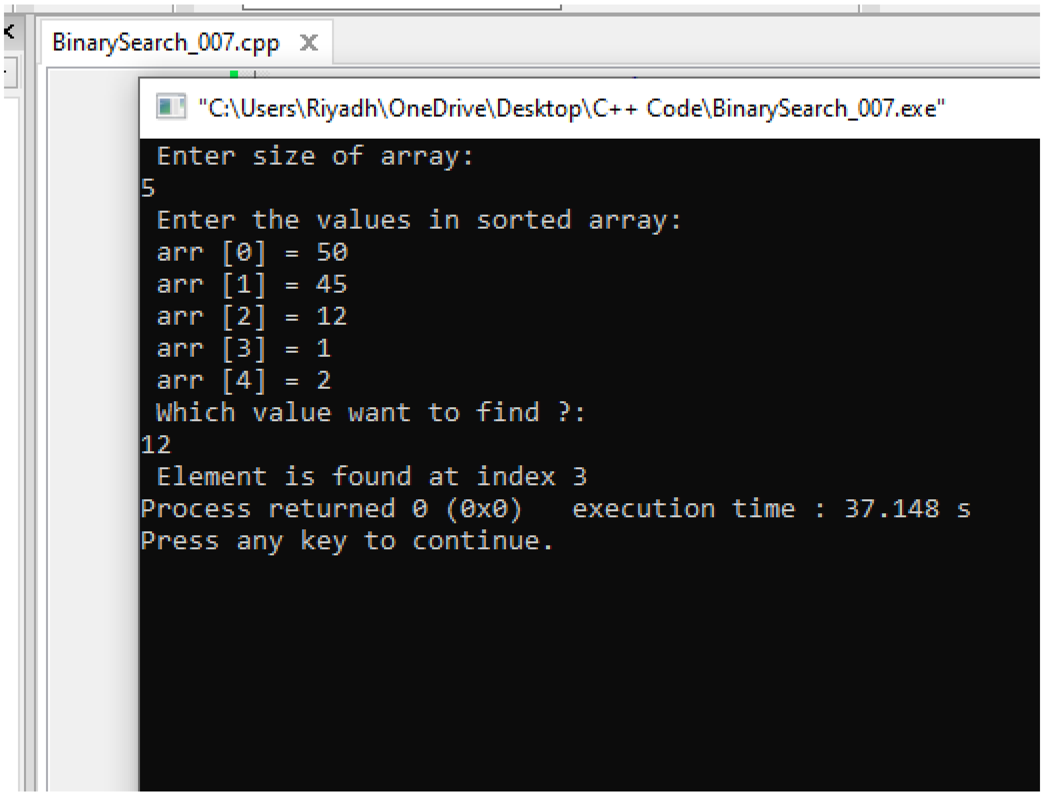

BinarySearch_007.cpp

```
#include <iostream>
using namespace std;
int main ()
{
    int arr[100], st, mid, end, i, num, tgt;

    cout << " Enter size of array: " << endl;
    cin >> num;

    cout << " Enter the values in sorted array: " << endl;
    for (i = 0; i < num; i++)
    {
        cout << " arr [" << i << "] = ";
        cin >> arr[i];
    }

    st = 0;
    end = num - 1;

    cout << " Which value want to find ?: " << endl;
    cin >> tgt;

    while ( st <= end)
    {
```



BinarySearch_007.cpp

```
        cout << " Which value want to find ?: " << endl;
        cin >> tgt;

        while ( st <= end)
        {
            mid = ( st + end ) / 2;
            if (arr[mid] == tgt)
            {
                cout << " Element is found at index " << (mid + 1);
                exit (0);
            }
            else if ( tgt > arr[mid])
            {
                st = mid + 1;
            }
            else if ( tgt < arr[mid])
            {
                end = mid - 1;
            }
        }
        cout << " Number is not found. " << endl;
        return 0;
    }
```

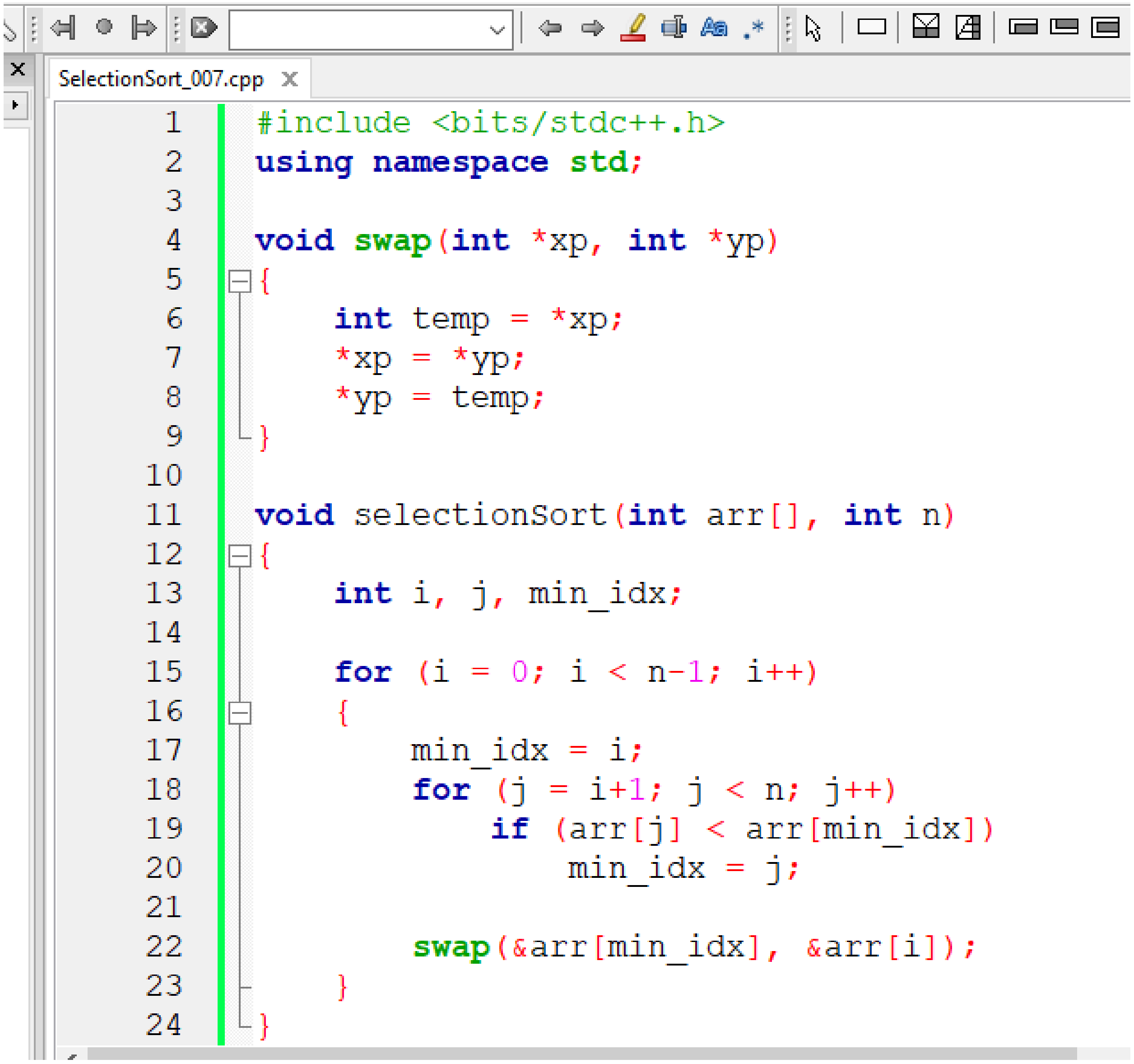


BinarySearch_007.cpp

"C:\Users\Riyadh\OneDrive\Desktop\C++ Code\BinarySearch_007.exe"

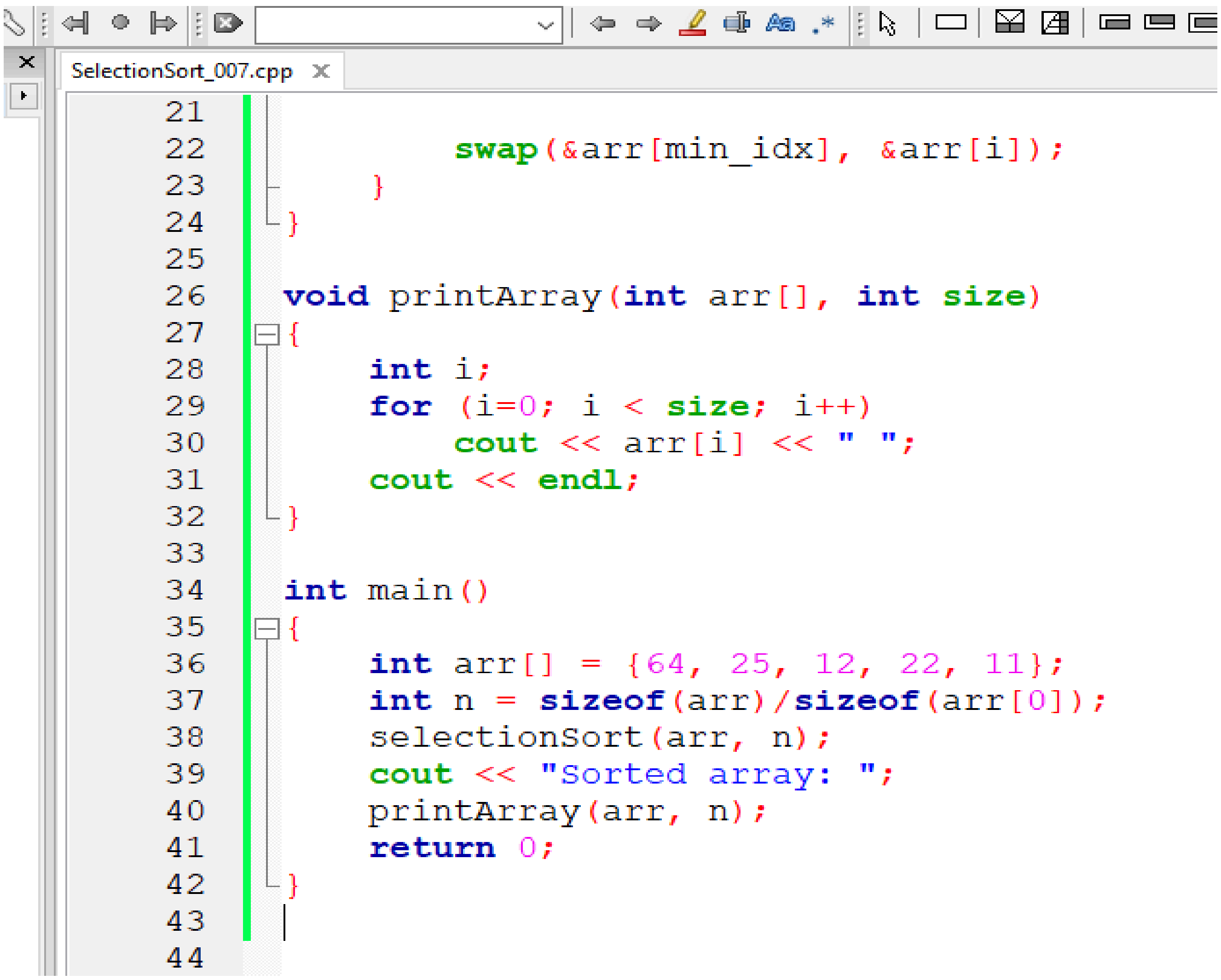
```
 Enter size of array:
5
 Enter the values in sorted array:
 arr [0] = 50
 arr [1] = 45
 arr [2] = 12
 arr [3] = 1
 arr [4] = 2
 Which value want to find ?:
12
 Element is found at index 3
Process returned 0 (0x0)   execution time : 37.148 s
Press any key to continue.
```

Selection sort :

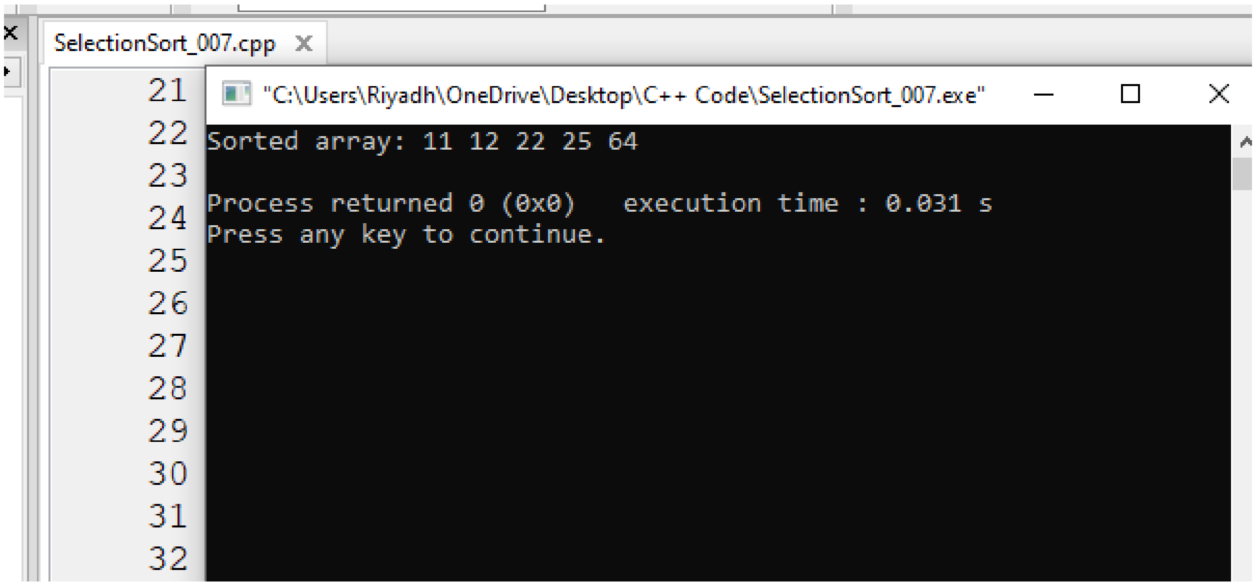


SelectionSort_007.cpp

```
#include <bits/stdc++.h>
using namespace std;

void swap(int *xp, int *yp)
{
    int temp = *xp;
    *xp = *yp;
    *yp = temp;
}

void selectionSort(int arr[], int n)
{
    int i, j, min_idx;

    for (i = 0; i < n-1; i++)
    {
        min_idx = i;
        for (j = i+1; j < n; j++)
            if (arr[j] < arr[min_idx])
                min_idx = j;

        swap(&arr[min_idx], &arr[i]);
    }
}
```



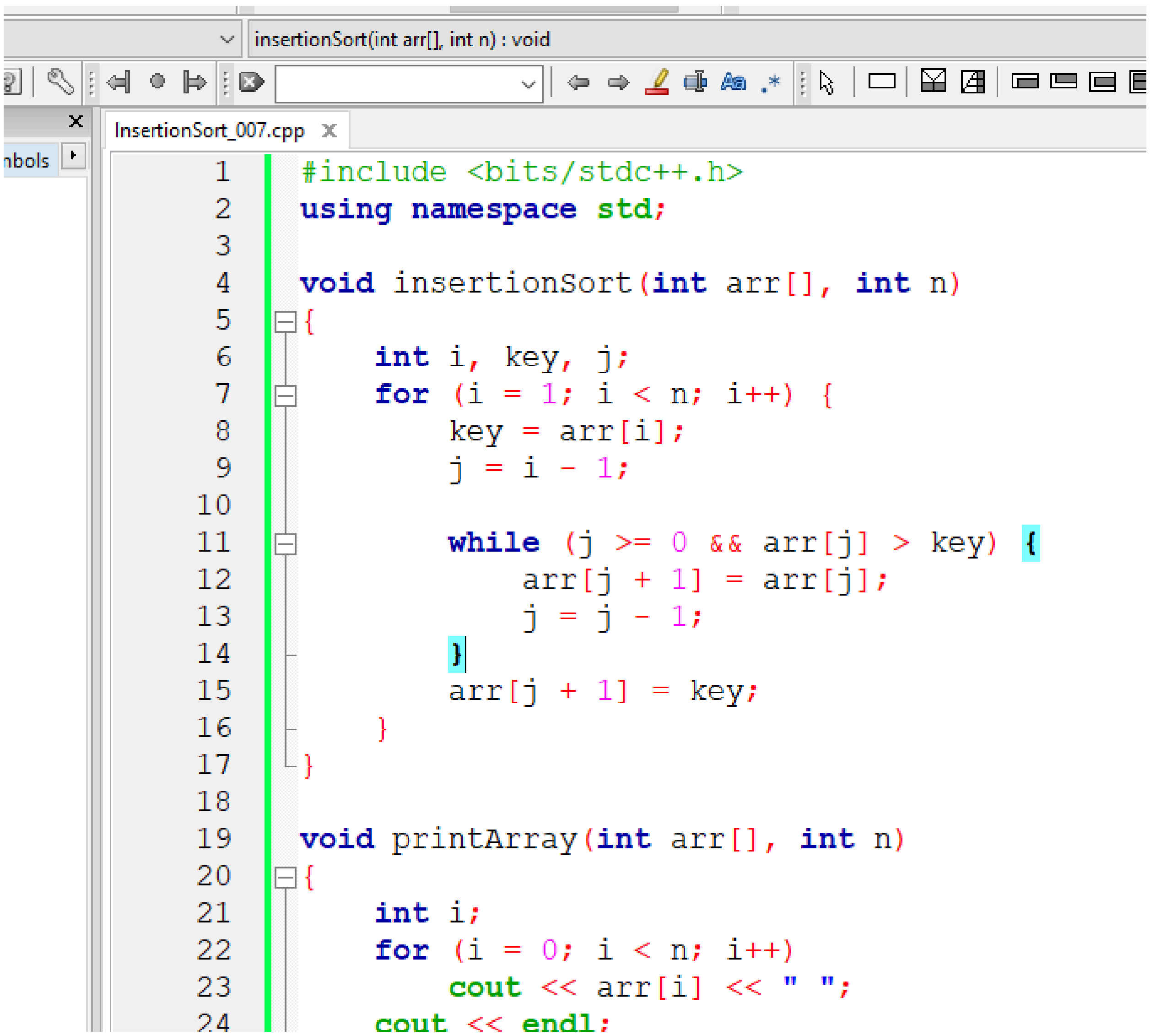
```
            swap(&arr[min_idx], &arr[i]);
        }
}

void printArray(int arr[], int size)
{
    int i;
    for (i=0; i < size; i++)
        cout << arr[i] << " ";
    cout << endl;
}

int main()
{
    int arr[] = {64, 25, 12, 22, 11};
    int n = sizeof(arr)/sizeof(arr[0]);
    selectionSort(arr, n);
    cout << "Sorted array: ";
    printArray(arr, n);
    return 0;
}
```



"C:\Users\Riyadh\OneDrive\Desktop\C++ Code\SelectionSort_007.exe"

```
Sorted array: 11 12 22 25 64

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

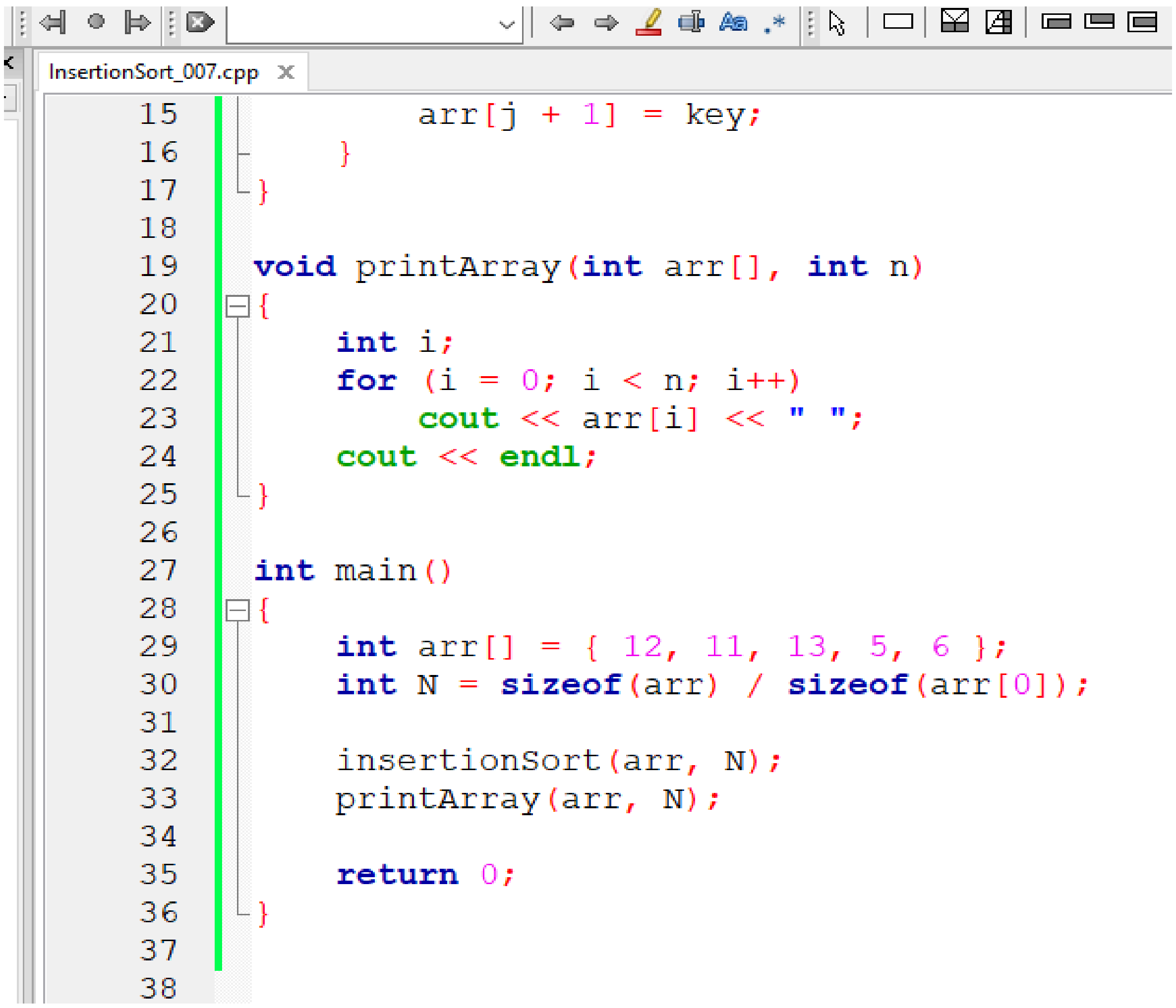
Insertion Sort :



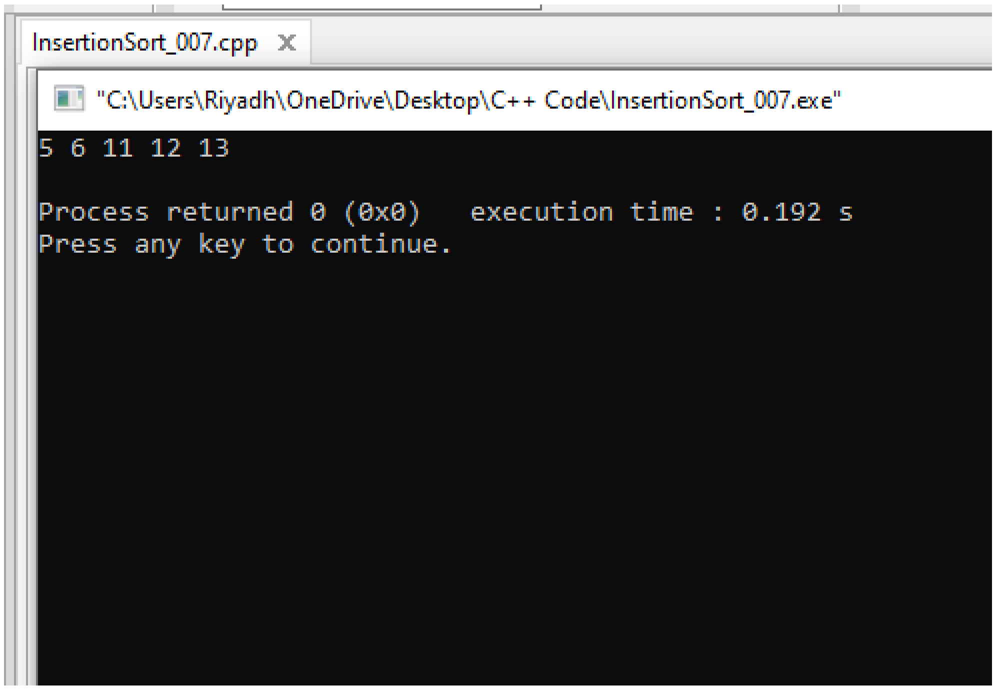
insertionSort(int arr[], int n) : void

InsertionSort_007.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;

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

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

void printArray(int arr[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        cout << arr[i] << " ";
    cout << endl;
```

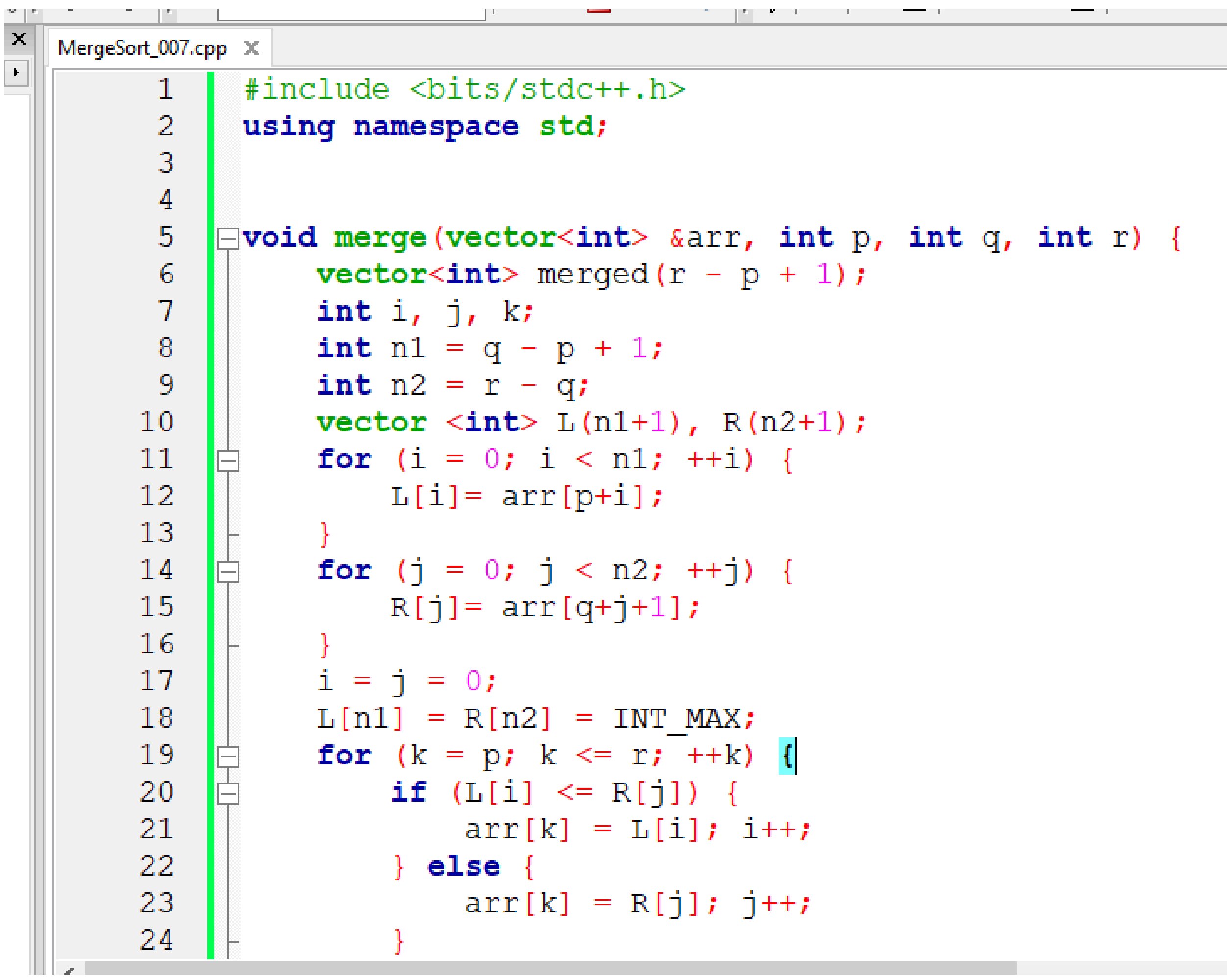


```cpp
            arr[j + 1] = key;
        }
}

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

int main()
{
    int arr[] = { 12, 11, 13, 5, 6 };
    int N = sizeof(arr) / sizeof(arr[0]);

    insertionSort(arr, N);
    printArray(arr, N);

    return 0;
}

```



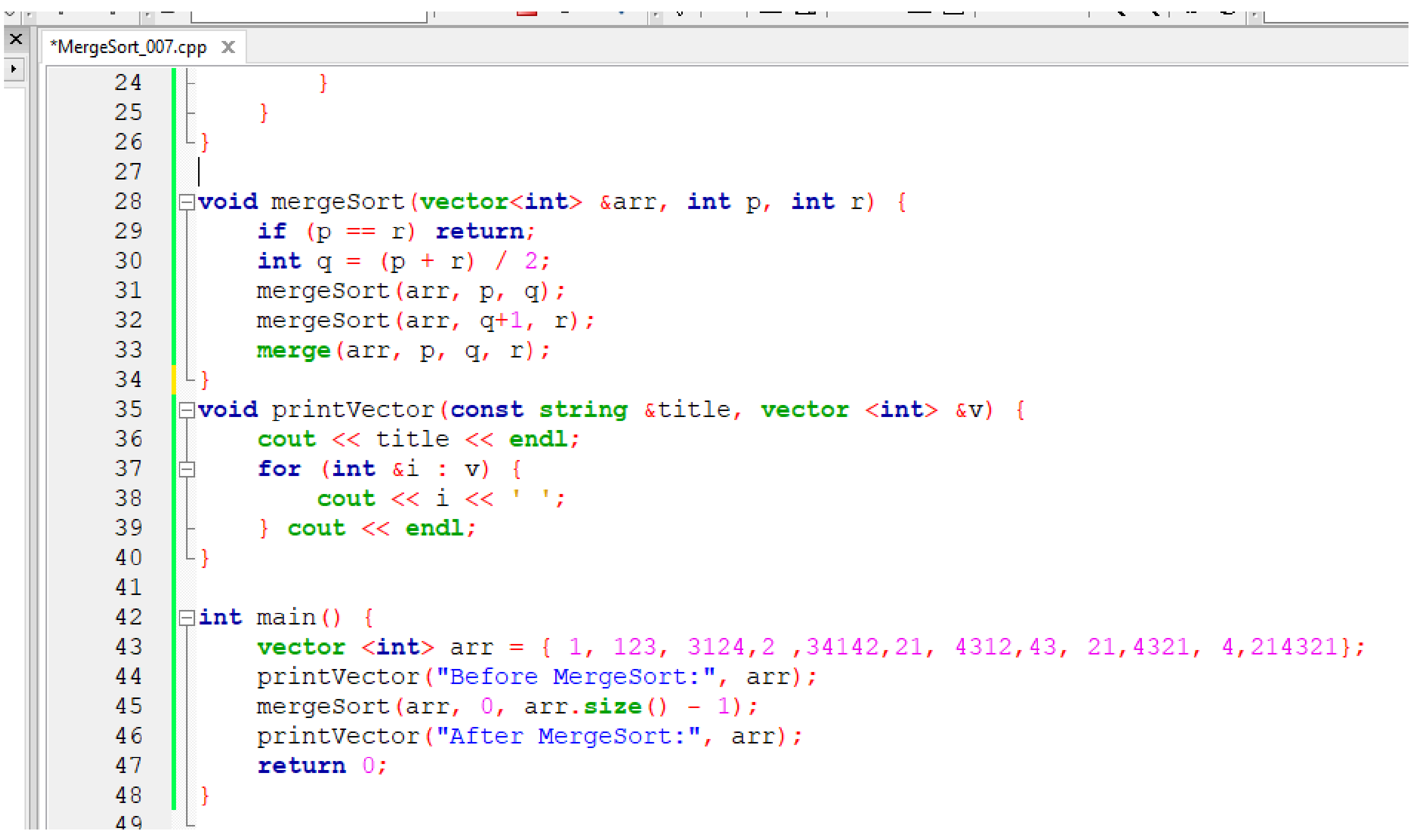
```
"C:\Users\Riyadh\OneDrive\Desktop\C++ Code\InsertionSort_007.exe"
5 6 11 12 13

Process returned 0 (0x0)   execution time : 0.192 s
Press any key to continue.
```

Merge Sort :

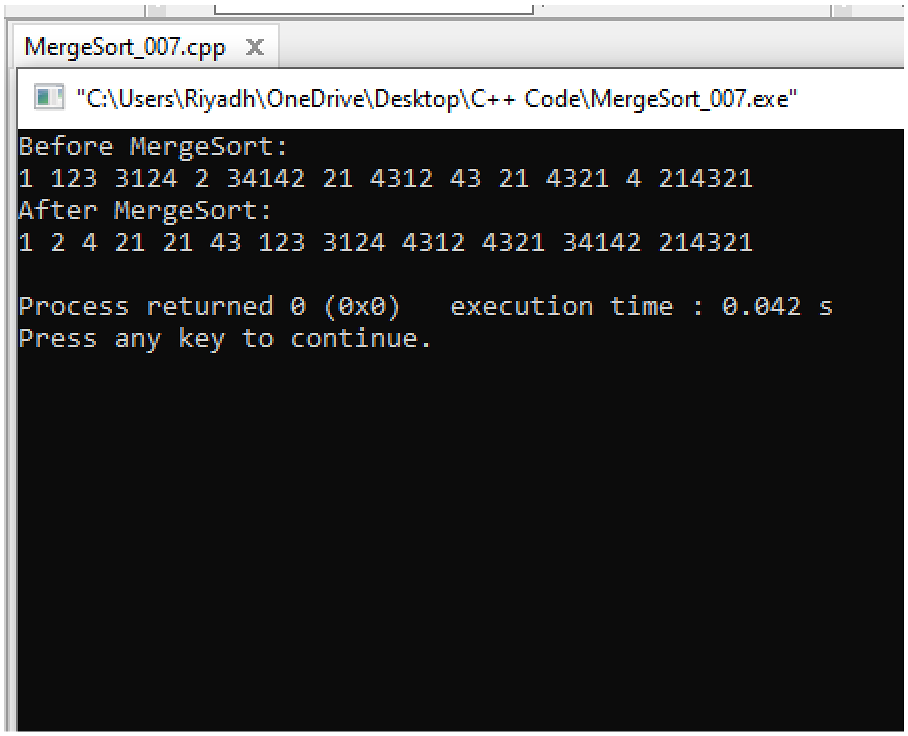


MergeSort_007.cpp

```cpp
#include <bits/stdc++.h>
using namespace std;


void merge(vector<int> &arr, int p, int q, int r) {
    vector<int> merged(r - p + 1);
    int i, j, k;
    int n1 = q - p + 1;
    int n2 = r - q;
    vector <int> L(n1+1), R(n2+1);
    for (i = 0; i < n1; ++i) {
        L[i]= arr[p+i];
    }
    for (j = 0; j < n2; ++j) {
        R[j]= arr[q+j+1];
    }
    i = j = 0;
    L[n1] = R[n2] = INT_MAX;
    for (k = p; k <= r; ++k) {
        if (L[i] <= R[j]) {
            arr[k] = L[i]; i++;
        } else {
            arr[k] = R[j]; j++;
        }
```


*MergeSort_007.cpp

```cpp
            }
        }
    }

void mergeSort(vector<int> &arr, int p, int r) {
        if (p == r) return;
        int q = (p + r) / 2;
        mergeSort(arr, p, q);
        mergeSort(arr, q+1, r);
        merge(arr, p, q, r);
    }
void printVector(const string &title, vector <int> &v) {
        cout << title << endl;
        for (int &i : v) {
            cout << i << ' ';
        } cout << endl;
    }

int main() {
        vector <int> arr = { 1, 123, 3124,2 ,34142,21, 4312,43, 21,4321, 4,214321};
        printVector("Before MergeSort:", arr);
        mergeSort(arr, 0, arr.size() - 1);
        printVector("After MergeSort:", arr);
        return 0;
    }
```


MergeSort_007.cpp

"C:\Users\Riyadh\OneDrive\Desktop\C++ Code\MergeSort_007.exe"

```
Before MergeSort:
1 123 3124 2 34142 21 4312 43 21 4321 4 214321
After MergeSort:
1 2 4 21 21 43 123 3124 4312 4321 34142 214321

Process returned 0 (0x0)   execution time : 0.042 s
Press any key to continue.
```