

Name : Riyadul Islam

Roll : 007

Section : 1 , Intake : 50

Dept: CSE

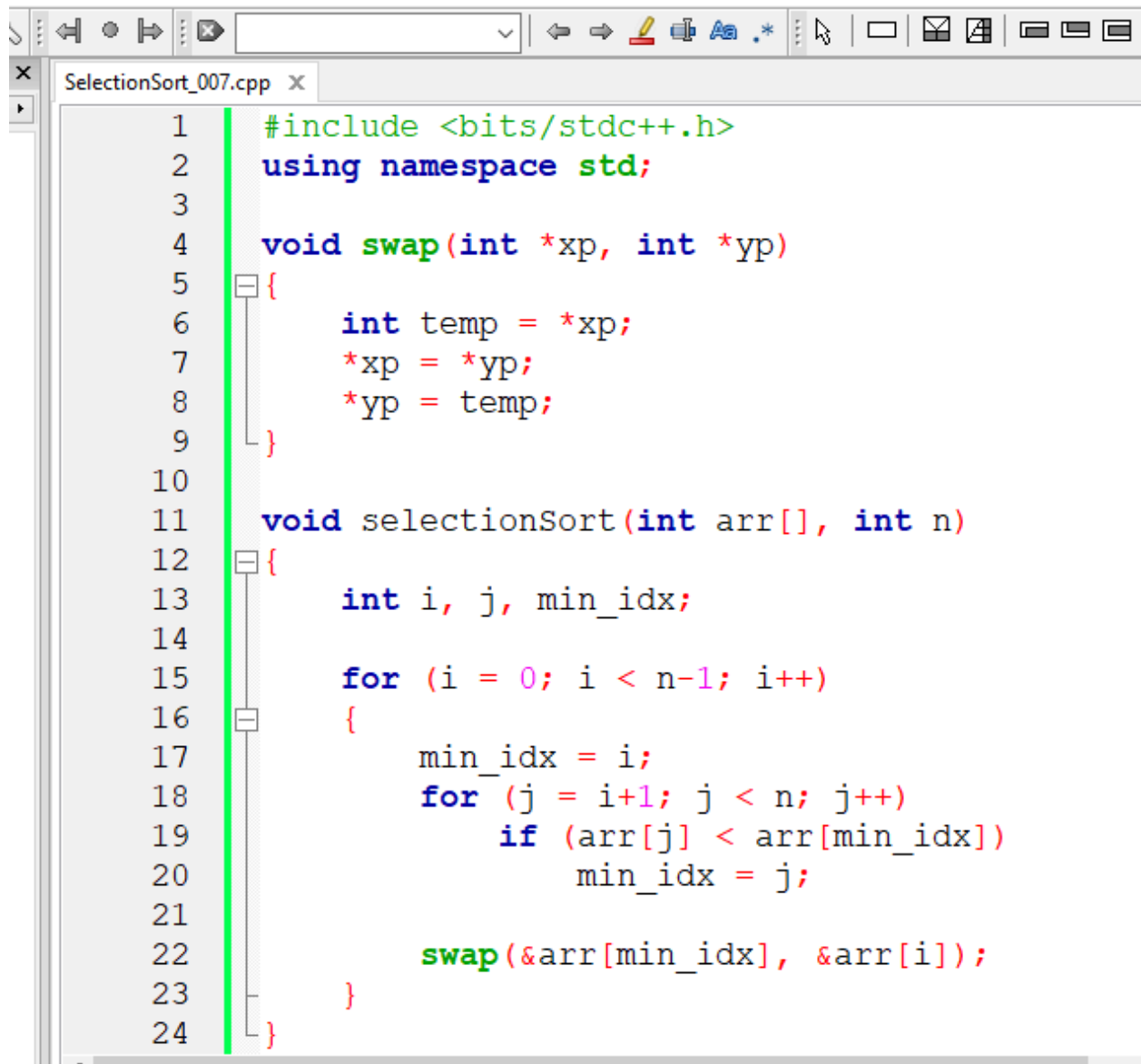
Binary Search Algorithm :

```
BinarySearch_007.cpp x
1  #include <iostream>
2  using namespace std;
3  int main ()
4  {
5      int arr[100], st, mid, end, i, num, tgt;
6
7      cout << " Enter size of array: " << endl;
8      cin >> num;
9
10     cout << " Enter the values in sorted array: " << endl;
11     for (i = 0; i < num; i++)
12     {
13         cout << " arr [" << i << "] = ";
14         cin >> arr[i];
15     }
16
17     st = 0;
18     end = num - 1;
19
20     cout << " Which value want to find ?: " << endl;
21     cin >> tgt;
22
23     while ( st <= end)
24     {
```

```
BinarySearch_007.cpp X
20     cout << " Which value want to find ?: " << endl;
21     cin >> tgt;
22
23     while ( st <= end)
24     {
25         mid = ( st + end ) / 2;
26         if (arr[mid] == tgt)
27         {
28             cout << " Element is found at index " << (mid + 1);
29             exit (0);
30         }
31         else if ( tgt > arr[mid])
32         {
33             st = mid + 1;
34         }
35         else if ( tgt < arr[mid])
36         {
37             end = mid - 1;
38         }
39     }
40     cout << " Number is not found. " << endl;
41     return 0;
42 }
43
```

```
BinarySearch_007.cpp X
"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 :

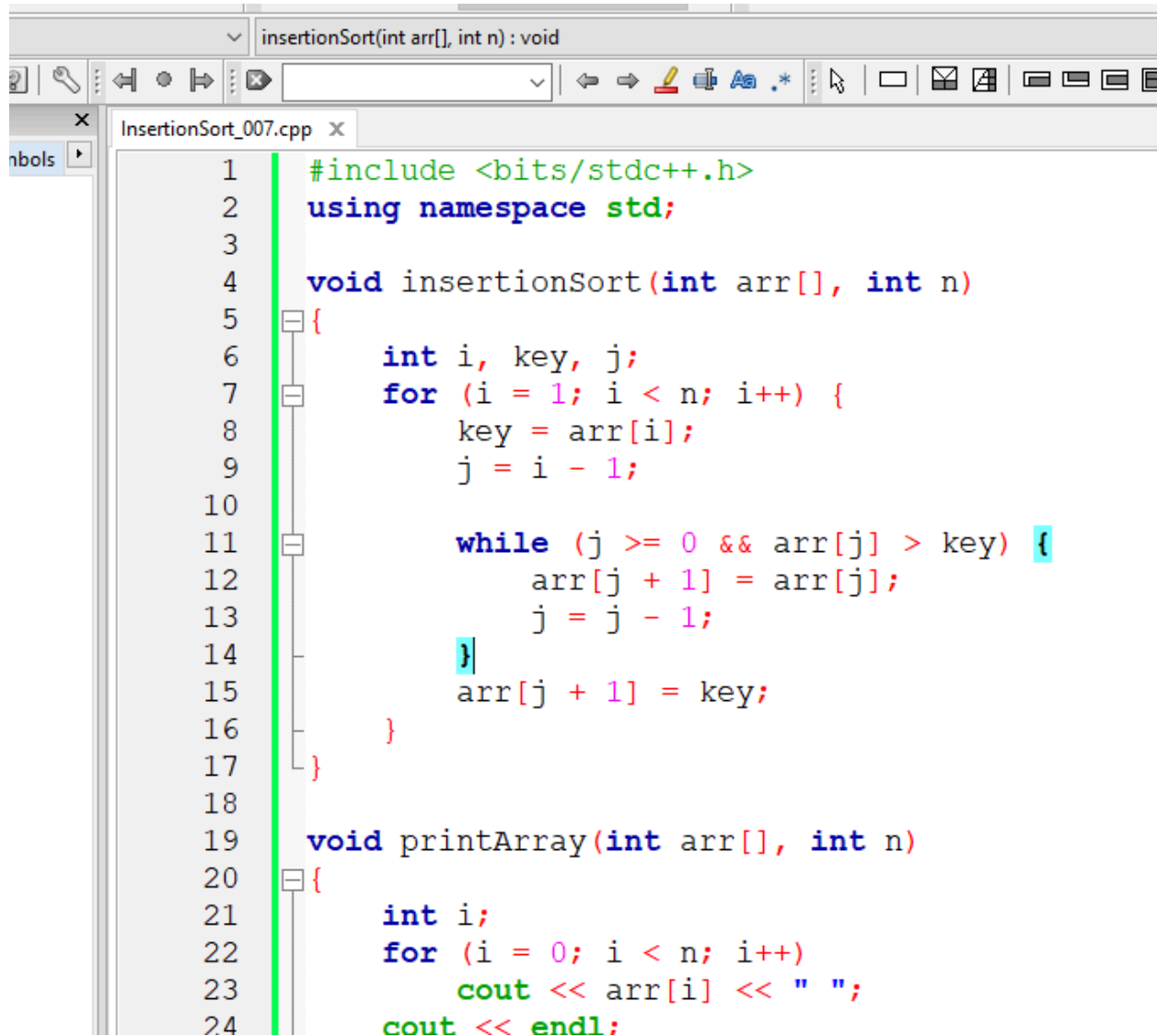


```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  void swap(int *xp, int *yp)
5  {
6      int temp = *xp;
7      *xp = *yp;
8      *yp = temp;
9  }
10
11 void selectionSort(int arr[], int n)
12 {
13     int i, j, min_idx;
14
15     for (i = 0; i < n-1; i++)
16     {
17         min_idx = i;
18         for (j = i+1; j < n; j++)
19             if (arr[j] < arr[min_idx])
20                 min_idx = j;
21
22         swap(&arr[min_idx], &arr[i]);
23     }
24 }
```

```
SelectionSort_007.cpp x
21
22         swap(&arr[min_idx], &arr[i]);
23     }
24 }
25
26 void printArray(int arr[], int size)
27 {
28     int i;
29     for (i=0; i < size; i++)
30         cout << arr[i] << " ";
31     cout << endl;
32 }
33
34 int main()
35 {
36     int arr[] = {64, 25, 12, 22, 11};
37     int n = sizeof(arr)/sizeof(arr[0]);
38     selectionSort(arr, n);
39     cout << "Sorted array: ";
40     printArray(arr, n);
41     return 0;
42 }
43
44
```

```
SelectionSort_007.cpp x
21 "C:\Users\Riyadh\OneDrive\Desktop\C++ Code\SelectionSort_007.exe" - □ X
22 Sorted array: 11 12 22 25 64
23
24 Process returned 0 (0x0) execution time : 0.031 s
25 Press any key to continue.
26
27
28
29
30
31
32
```

Insertion Sort :

A screenshot of a C++ IDE window titled 'InsertionSort_007.cpp'. The code implements the Insertion Sort algorithm. It includes the standard C++ header and namespace. The 'insertionSort' function takes an array and its size, iterates through each element, and shifts elements greater than the current element one position to the right to insert the current element in its sorted position. A 'printArray' function is also provided to display the array. The code is as follows:

```
1  #include <bits/stdc++.h>
2  using namespace std;
3
4  void insertionSort(int arr[], int n)
5  {
6      int i, key, j;
7      for (i = 1; i < n; i++) {
8          key = arr[i];
9          j = i - 1;
10
11         while (j >= 0 && arr[j] > key) {
12             arr[j + 1] = arr[j];
13             j = j - 1;
14         }
15         arr[j + 1] = key;
16     }
17 }
18
19 void printArray(int arr[], int n)
20 {
21     int i;
22     for (i = 0; i < n; i++)
23         cout << arr[i] << " ";
24     cout << endl;
```

```
InsertionSort_007.cpp x
15         arr[j + 1] = key;
16     }
17 }
18
19 void printArray(int arr[], int n)
20 {
21     int i;
22     for (i = 0; i < n; i++)
23         cout << arr[i] << " ";
24     cout << endl;
25 }
26
27 int main()
28 {
29     int arr[] = { 12, 11, 13, 5, 6 };
30     int N = sizeof(arr) / sizeof(arr[0]);
31
32     insertionSort(arr, N);
33     printArray(arr, N);
34
35     return 0;
36 }
37
38
```

```
InsertionSort_007.cpp x
"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 x
1  #include <bits/stdc++.h>
2  using namespace std;
3
4
5  void merge(vector<int> &arr, int p, int q, int r) {
6      vector<int> merged(r - p + 1);
7      int i, j, k;
8      int n1 = q - p + 1;
9      int n2 = r - q;
10     vector<int> L(n1+1), R(n2+1);
11     for (i = 0; i < n1; ++i) {
12         L[i] = arr[p+i];
13     }
14     for (j = 0; j < n2; ++j) {
15         R[j] = arr[q+j+1];
16     }
17     i = j = 0;
18     L[n1] = R[n2] = INT_MAX;
19     for (k = p; k <= r; ++k) {
20         if (L[i] <= R[j]) {
21             arr[k] = L[i]; i++;
22         } else {
23             arr[k] = R[j]; j++;
24         }
25     }
```

```
*MergeSort_007.cpp X
24     }
25     }
26 }
27
28 void mergeSort(vector<int> &arr, int p, int r) {
29     if (p == r) return;
30     int q = (p + r) / 2;
31     mergeSort(arr, p, q);
32     mergeSort(arr, q+1, r);
33     merge(arr, p, q, r);
34 }
35
36 void printVector(const string &title, vector<int> &v) {
37     cout << title << endl;
38     for (int &i : v) {
39         cout << i << ' ';
40     } cout << endl;
41 }
42
43 int main() {
44     vector<int> arr = { 1, 123, 3124, 2, 34142, 21, 4312, 43, 21, 4321, 4, 214321};
45     printVector("Before MergeSort:", arr);
46     mergeSort(arr, 0, arr.size() - 1);
47     printVector("After MergeSort:", arr);
48     return 0;
49 }
```

```
MergeSort_007.cpp X
"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.
```