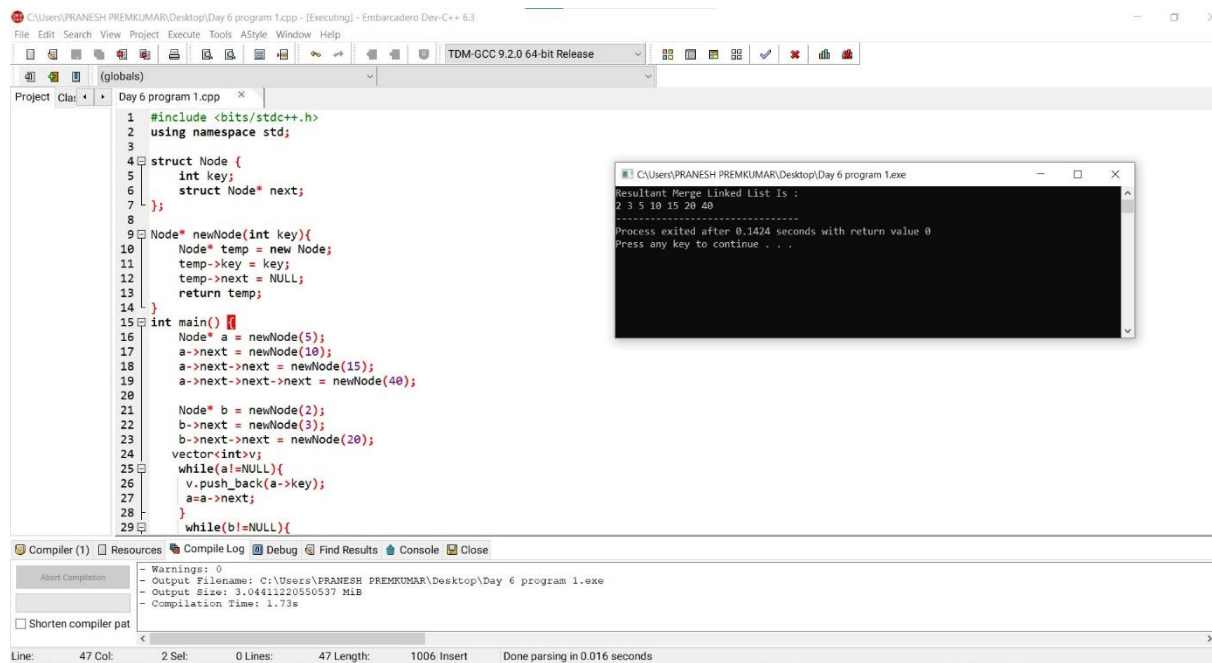


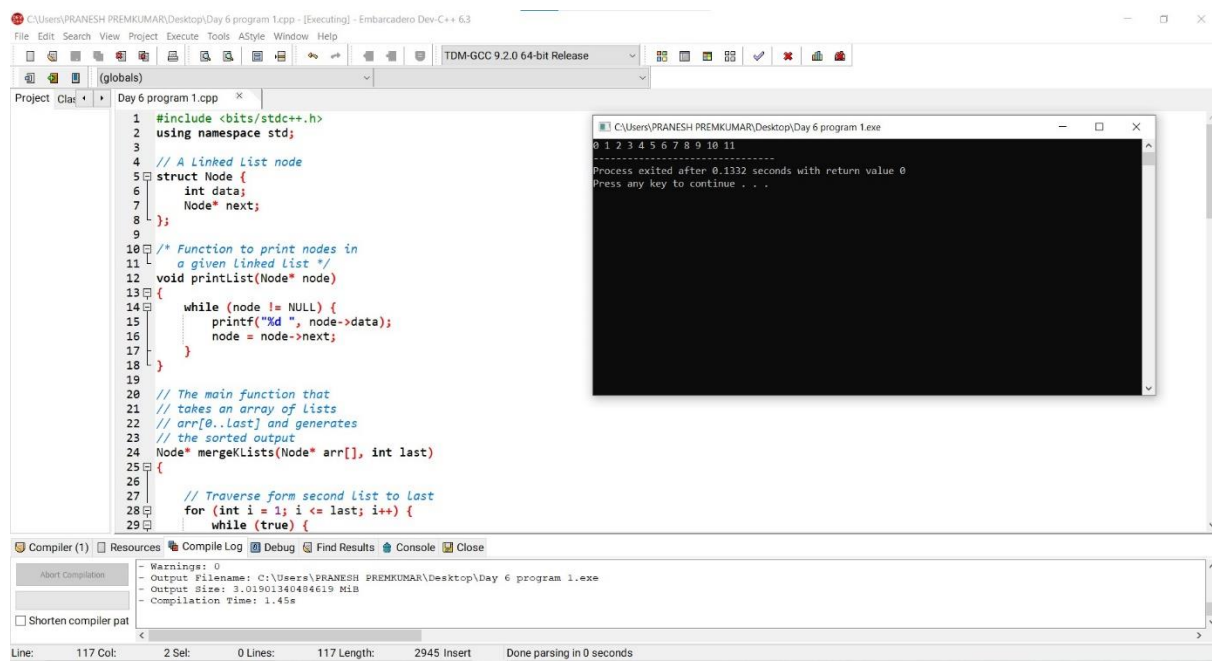
1.



```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 struct Node {
5     int key;
6     struct Node* next;
7 };
8
9 Node* newNode(int key){
10     Node* temp = new Node;
11     temp->key = key;
12     temp->next = NULL;
13     return temp;
14 }
15
16 int main() {
17     Node* a = newNode(5);
18     a->next = newNode(10);
19     a->next->next = newNode(15);
20     a->next->next->next = newNode(40);
21
22     Node* b = newNode(2);
23     b->next = newNode(3);
24     b->next->next = newNode(20);
25
26     vector<int> v;
27     while(a!=NULL){
28         v.push_back(a->key);
29         a=a->next;
30     }
31     while(b!=NULL){
32         v.push_back(b->key);
33         b=b->next;
34     }
35
36     sort(v.begin(), v.end());
37
38     for(int i=0; i<v.size(); i++){
39         cout<<v[i]<<" ";
40     }
41     cout<<endl;
42
43     return 0;
44 }
```

Resultant Merge Linked List Is :
2 3 5 10 15 20 40
.....
Process exited after 0.1424 seconds with return value 0
Press any key to continue . . .

2.



```
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 // A Linked List node
5 struct Node {
6     int data;
7     Node* next;
8 };
9
10 /* Function to print nodes in
11 a given Linked List */
12 void printList(Node* node)
13 {
14     while (node != NULL) {
15         printf("%d ", node->data);
16         node = node->next;
17     }
18 }
19
20 // The main function that
21 // takes an array of Lists
22 // arr[0..Last] and generates
23 // the sorted output
24 Node* mergeKLists(Node* arr[], int last)
25 {
26     // Traverse from second list to last
27     for (int i = 1; i <= last; i++) {
28         while (true) {
29             // Find the minimum node
30             Node* minNode = arr[0];
31             int minIndex = 0;
32             for (int j = 1; j <= last; j++) {
33                 if (arr[j]->data < minNode->data) {
34                     minNode = arr[j];
35                     minIndex = j;
36                 }
37             }
38             // Add the minimum node to the result list
39             Node* temp = minNode;
40             minNode = minNode->next;
41             temp->next = NULL;
42             arr[minIndex] = minNode;
43             last = minIndex;
44         }
45     }
46     return arr[0];
47 }
```

0 1 2 3 4 5 6 7 8 9 10 11
.....
Process exited after 0.1332 seconds with return value 0
Press any key to continue . . .

3.

C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.cpp - [Executing] - Embarcadero Dev-C++ 6.3

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 9.2.0 64-bit Release

Project Claws Day 6 program 1.cpp

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int removeDuplicates(int arr[], int n)
4 {
5     if (n == 0 || n == 1)
6         return n;
7
8     int temp[n];
9     int j = 0;
10
11     for (int i = 0; i < n - 1; i++)
12         if (arr[i] != arr[i + 1])
13             temp[j++] = arr[i];
14     temp[j++] = arr[n - 1];
15     for (int i = 0; i < j; i++)
16         arr[i] = temp[i];
17
18     return j;
19 }
20
21 // Driver code
22 int main()
23 {
24     int arr[] = { 1, 2, 2, 3, 4, 4, 4, 5, 5 };
25     int n = sizeof(arr) / sizeof(arr[0]);
26
27     // removeDuplicates() returns new size of array.
28     n = removeDuplicates(arr, n);
29 }
```

Compiler (1) Resources Compile Log Debug Find Results Console Close

Abort Compilation

Shorten compiler path

Warnings: 0
Output Filename: C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.exe
Output Size: 2,991,357,106,936 KiB
Compilation Time: 2.53s

C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.exe

1 2 3 4 5

Process exited after 0.1176 seconds with return value 0
Press any key to continue . . .

4.

[*] murai.cpp murai.cpp

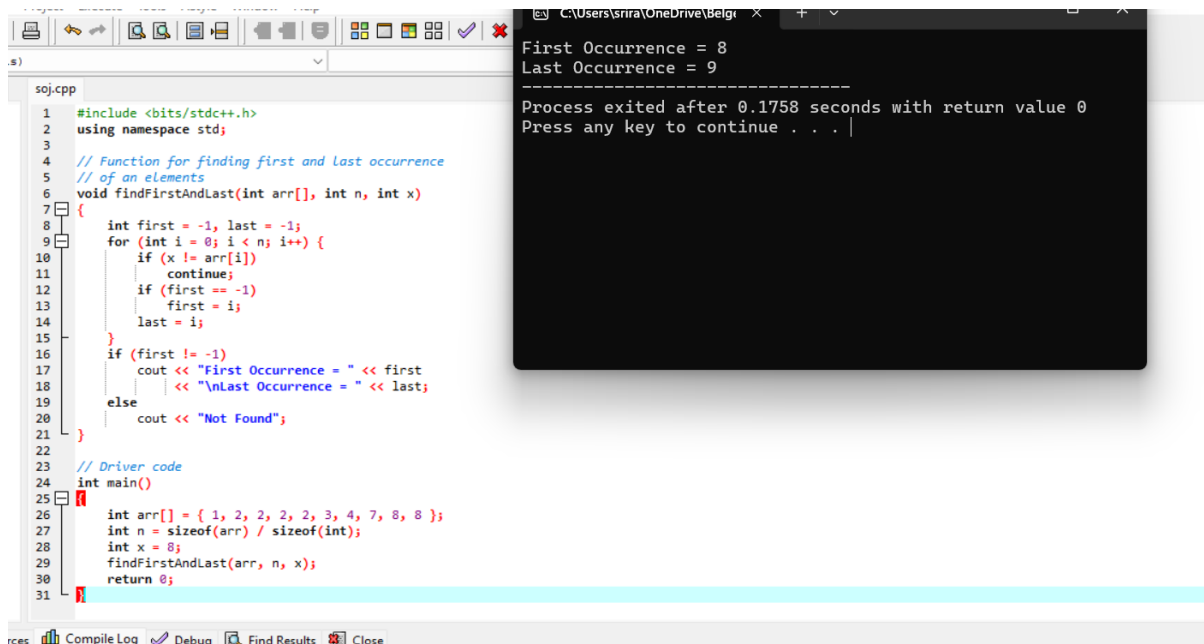
```
1 #include <stdio.h>
2
3 int search(int* nums, int numsSize, int target) {
4     int left = 0, right = numsSize - 1;
5
6     while (left <= right) {
7         int mid = left + (right - left) / 2;
8
9         if (nums[mid] == target) {
10             return mid;
11         }
12
13         if (nums[left] <= nums[mid]) {
14             if (nums[left] <= target && target < nums[mid]) {
15                 right = mid - 1;
16             } else {
17                 left = mid + 1;
18             }
19         } else {
20             if (nums[mid] < target && target <= nums[right]) {
21                 left = mid + 1;
22             } else {
23                 right = mid - 1;
24             }
25         }
26     }
27
28     return -1;
29 }
30
31 int main() {
32     int nums[] = { 4, 5, 6, 7, 0, 1, 2 };
33     int target = 0;
34     int numsSize = sizeof(nums) / sizeof(nums[0]);
35
36     int result = search(nums, numsSize, target);
37
38     printf("Output: %d\n", result);
39
40     return 0;
41 }
42 }
```

C:\Users\murai\OneDrive\Des

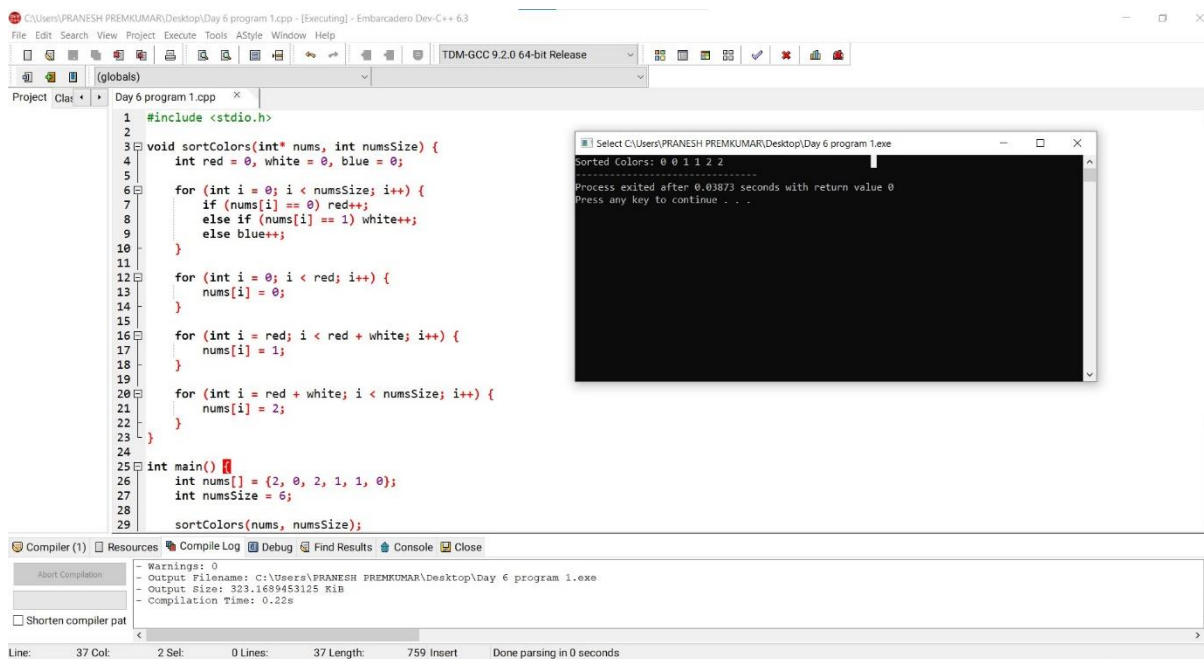
Output: 4

Process exited after 1.674 seconds with return value 0
Press any key to continue . . .

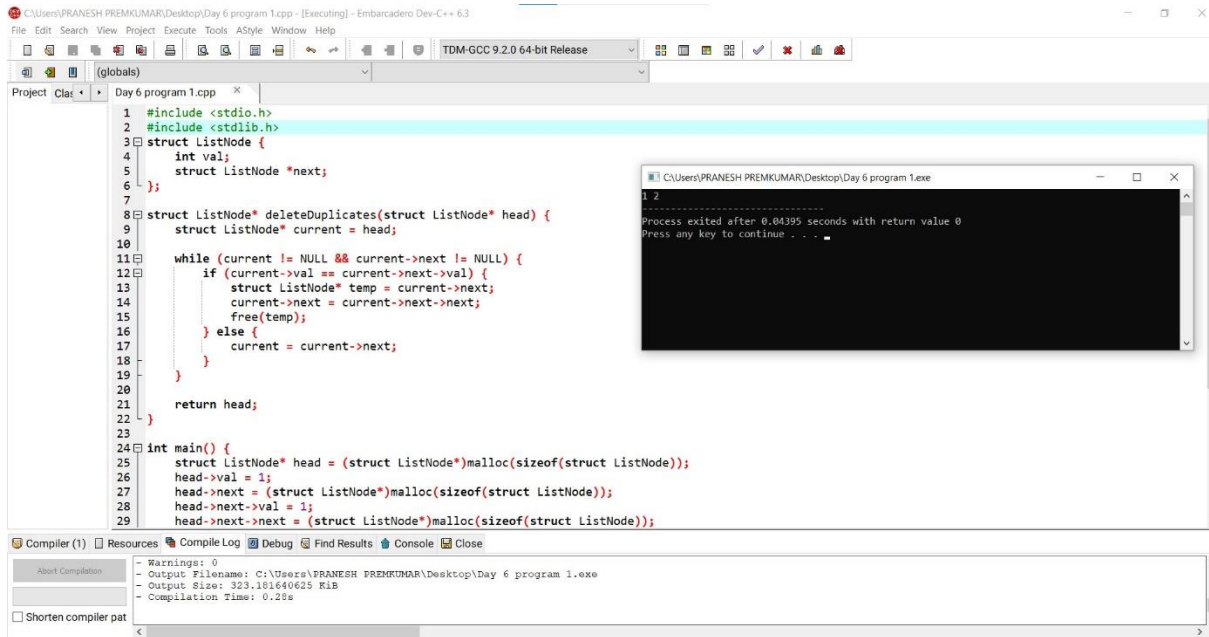
5.



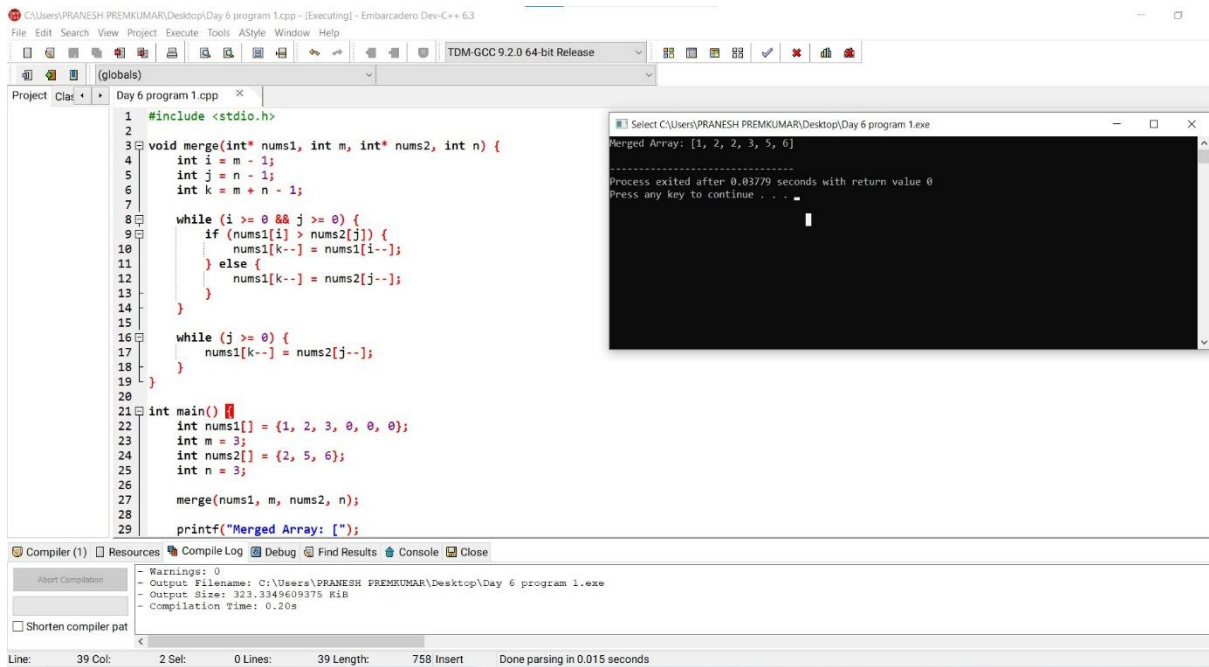
6.



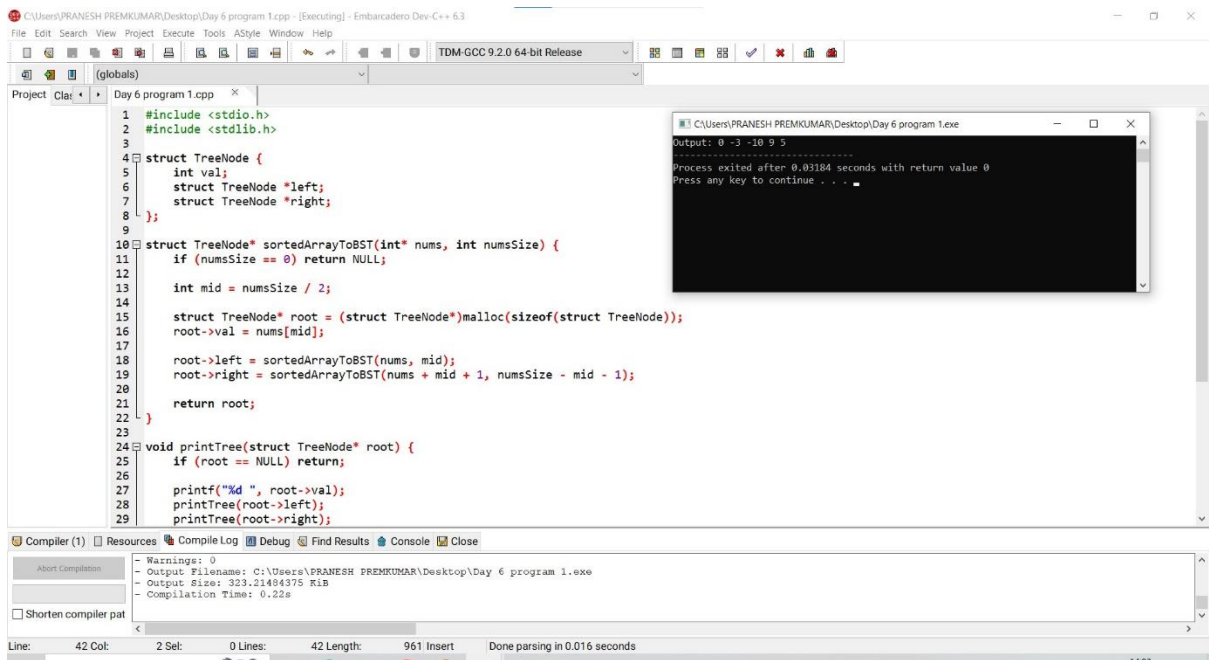
7.



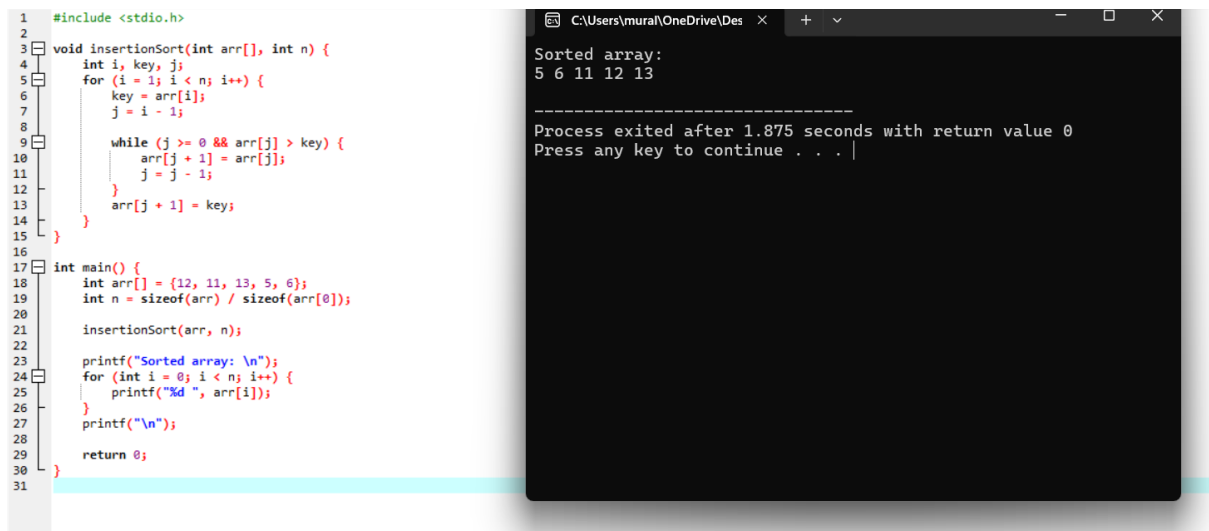
8.



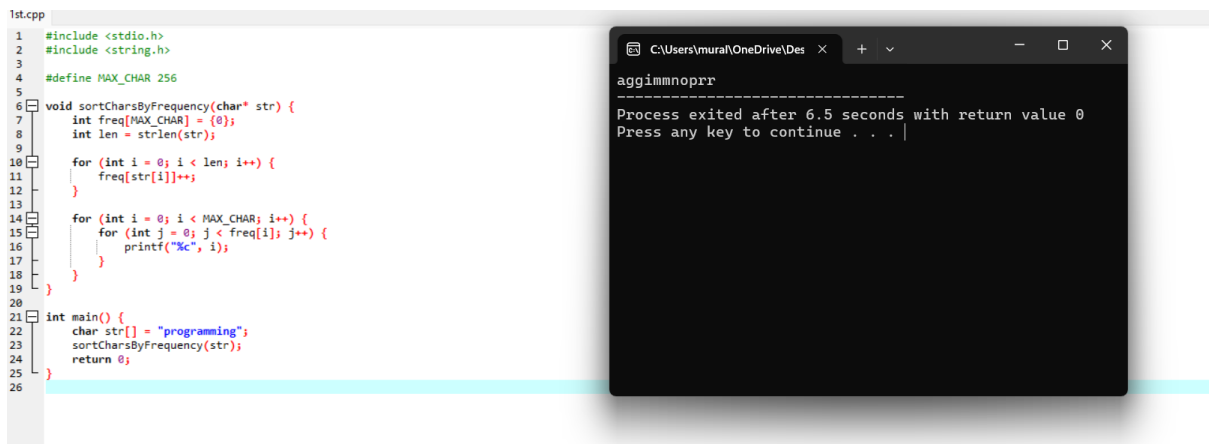
9.



10.



11.



12.

C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.cpp - [Executing] - Embarcadero Dev-C++ 6.3

```

1 #include <bits/stdc++.h>
2 using namespace std;
3 int maxPartitions(int arr[], int n)
4 {
5     int ans = 0, max_so_far = 0;
6     for (int i = 0; i < n; ++i) {
7         max_so_far = max(max_so_far, arr[i]);
8     }
9     if (max_so_far == i)
10        ans++;
11    return ans;
12 }
13
14 // Driver code
15 int main()
16 {
17     int arr[] = { 1, 0, 2, 3, 4 };
18     int n = sizeof(arr) / sizeof(arr[0]);
19     cout << maxPartitions(arr, n);
20     return 0;
21 }

```

Process exited after 0.1251 seconds with return value 0
Press any key to continue . . .

Compiler (2) | Resources | Compile Log | Debug | Find Results | Console | Close

Warnings: 0
Output Filename: C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.exe
Output Size: 2.9910888671875 MiB
Compilation Time: 1.53s

13.

C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.cpp - [Executing] - Embarcadero Dev-C++ 6.3

```

1 #include <stdio.h>
2
3 void findIntersection(int arr1[], int arr2[], int arr3[], int n1, int n2, int n3) {
4     int i = 0, j = 0, k = 0;
5     while (i < n1 && j < n2 && k < n3) {
6         if (arr1[i] == arr2[j] && arr2[j] == arr3[k]) {
7             printf("%d ", arr1[i]);
8             i++;
9             j++;
10            k++;
11        } else if (arr1[i] < arr2[j]) {
12            i++;
13        } else if (arr2[j] < arr3[k]) {
14            j++;
15        } else {
16            k++;
17        }
18    }
19 }
20
21 int main() {
22     int arr1[] = {1, 5, 10, 20, 40, 80};
23     int arr2[] = {6, 7, 20, 80, 100};
24     int arr3[] = {3, 4, 15, 20, 30, 70, 80, 120};
25     int n1 = sizeof(arr1) / sizeof(arr1[0]);
26     int n2 = sizeof(arr2) / sizeof(arr2[0]);
27     int n3 = sizeof(arr3) / sizeof(arr3[0]);
28 }

```

20 80
Process exited after 0.04879 seconds with return value 0
Press any key to continue . . .

Compiler (1) | Resources | Compile Log | Debug | Find Results | Console | Close

Warnings: 0
Output Filename: C:\Users\PRANESH PREMUMAR\Desktop\Day 6 program 1.exe
Output Size: 323.1806640625 KiB
Compilation Time: 0.27s

Line: 33 Col: 2 Sel: 0 Line: 33 Length: 86.3 Insert Done parsing in 0.015 seconds

14.

```
1 #include <stdio.h>
2
3 void sortDiagonal(int matrix[3][3], int rows, int cols) {
4     int temp;
5     for (int k = 0; k < rows; k++) {
6         for (int i = 0; i < rows - 1; i++) {
7             for (int j = 0; j < cols - 1; j++) {
8                 if (matrix[i][j] > matrix[i + 1][j + 1]) {
9                     temp = matrix[i][j];
10                    matrix[i][j] = matrix[i + 1][j + 1];
11                    matrix[i + 1][j + 1] = temp;
12                }
13            }
14        }
15    }
16 }
17
18 int main() {
19     int matrix[3][3] = {{3, 2, 1},
20                        {6, 5, 4},
21                        {9, 8, 7}};
22
23     sortDiagonal(matrix, 3, 3);
24
25     printf("Sorted Matrix Diagonally:\n");
26     for (int i = 0; i < 3; i++) {
27         for (int j = 0; j < 3; j++) {
28             printf("%d ", matrix[i][j]);
29         }
30     }
```

Sorted Matrix Diagonally:
3 2 1
6 5 4
9 8 7

Process exited after 0.04281 seconds with return value 0
Press any key to continue . . .