Week 7 Lab 7:
Name: Saif Majid Khan.
SAP-ID: 57114.
DSA Lab.

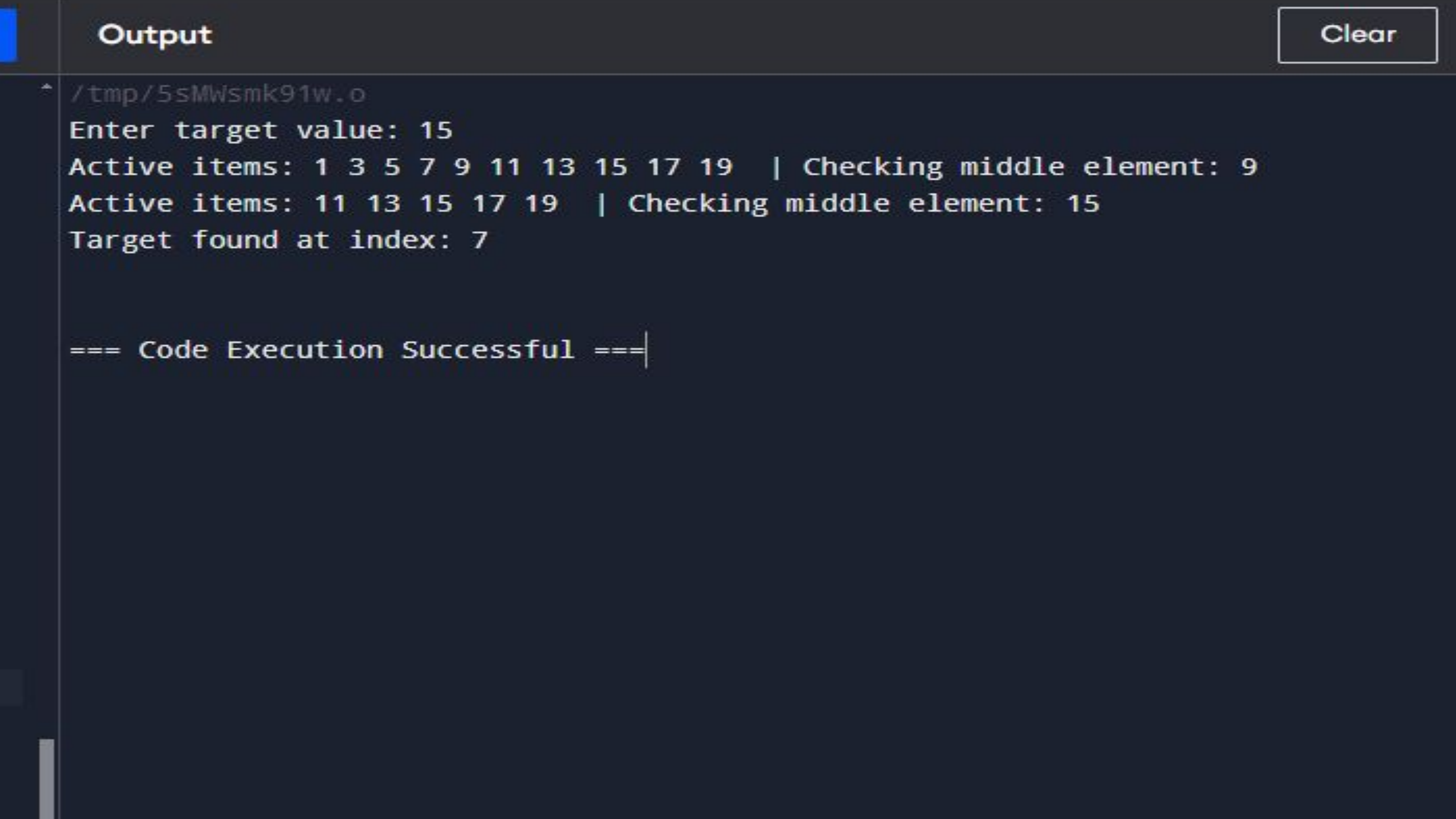
<u>GitHub Link:</u>

Q1)

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

int binarySearch(int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;

    while (left <= right) {
        int mid = left + (right - left) / 2;
//show active items
      cout << "Active items: ";
      for (int i = left; i <= right; ++i) {
         cout << arr[i] << " ";
      }
      cout << " | Checking middle element: " << arr[mid] << endl;

      if (arr[mid] == target) {
         return mid;     //target found
      } else if (arr[mid] < target) {
         left = mid + 1;          //earch right half
      } else {
         right = mid - 1;     //search left half
      }
   }
```

```
 return -1;    //target not found
}

int main() {
    int arr[] = {1, 3, 5, 7, 9, 11, 13, 15, 17, 19};
    int size = sizeof(arr) / sizeof(arr[0]);
    int target;

    cout << "Enter target value: ";
    cin >> target;

    int result = binarySearch(arr, size, target);
```

```
if (result != -1) {
        cout << "Target found at index: " << result << endl;
    } else {
        cout << "Target not found." << endl;
    }

    return 0;
}
```

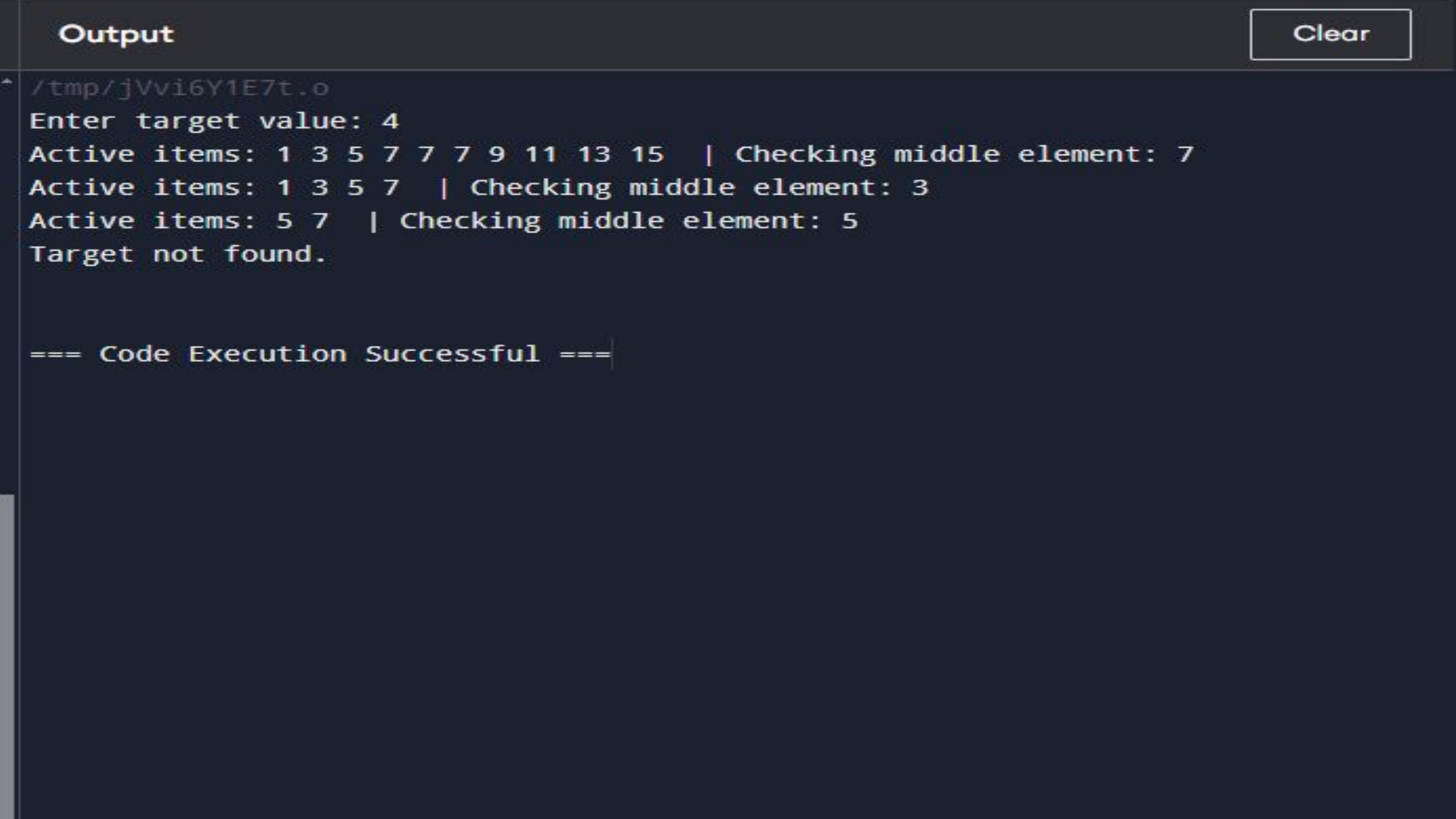


```
/tmp/5sMWsmk91w.o
Enter target value: 15
Active items: 1 3 5 7 9 11 13 15 17 19  | Checking middle element: 9
Active items: 11 13 15 17 19  | Checking middle element: 15
Target found at index: 7


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

Q2)

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

int findFirstOccurrence(int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;
    int result = -1; // To store the index of the first occurrence

    while (left <= right) {
        int mid = left + (right - left) / 2;
cout << "Active items: ";
        for (int i = left; i <= right; ++i) {
            cout << arr[i] << " ";
        }
        cout << " | Checking middle element: " << arr[mid] << endl;

        if (arr[mid] == target) {
            result = mid;  //store index and continue search in left half
            right = mid - 1;
        } else if (arr[mid] < target) {
            left = mid + 1;   //search right half
        } else {
            right = mid - 1;     //search left half
        }
    }
```

```
return result;    //return index of first ocr or -1 if not found
}
int main() {
    int arr[] = {1, 3, 5, 7, 7, 7, 9, 11, 13, 15};
    int size = sizeof(arr) / sizeof(arr[0]);
    int target;
    cout << "Enter target value: ";
    cin >> target;

    int result = findFirstOccurrence(arr, size, target);

    if (result != -1) {
        cout << "First occurrence of target found at index: " << result << endl;
    } else {
        cout << "Target not found." << endl;
    }
    return 0;
}
```

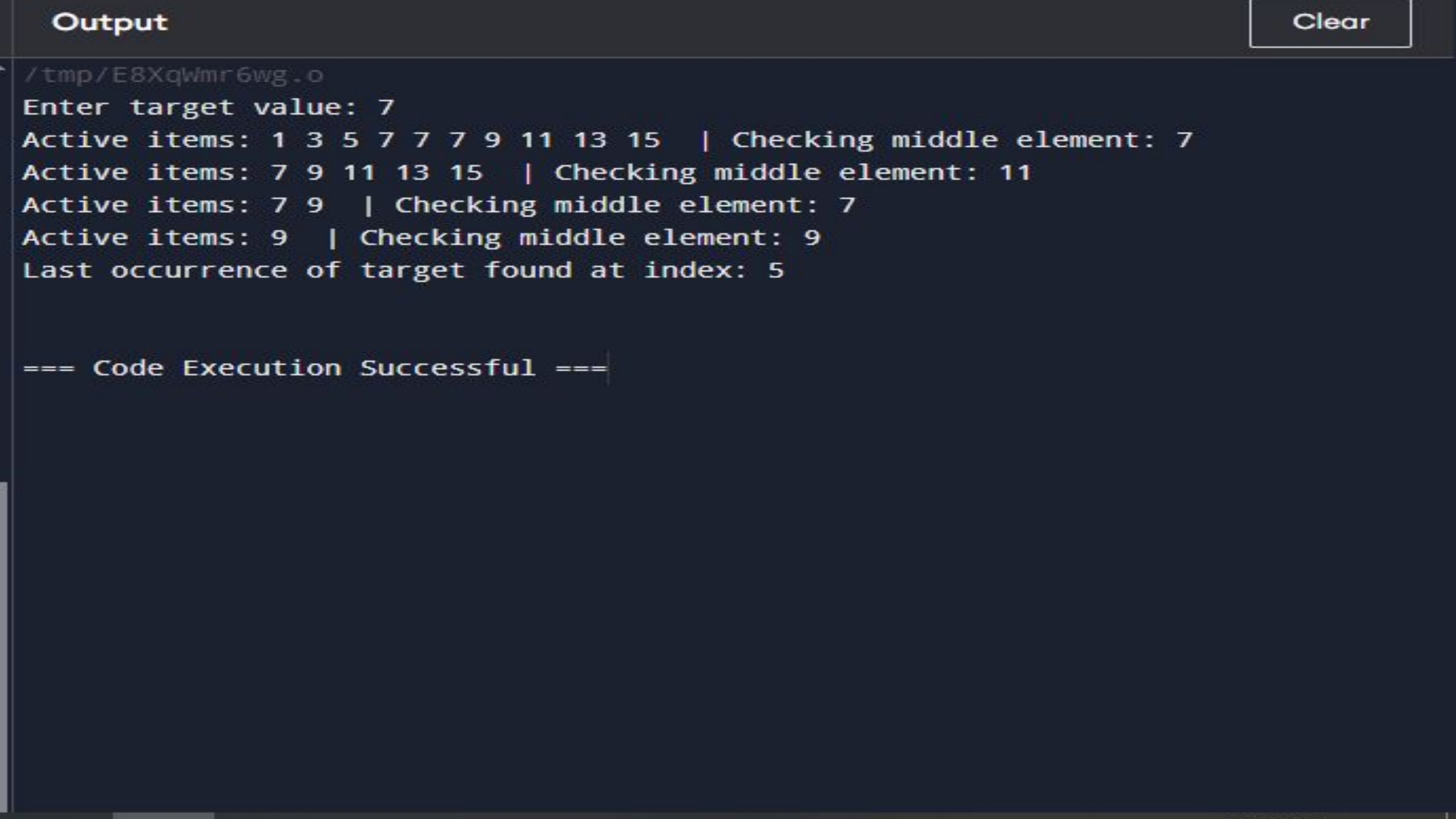


```
/tmp/jVvi6Y1E7t.o
Enter target value: 4
Active items: 1 3 5 7 7 7 9 11 13 15  | Checking middle element: 7
Active items: 1 3 5 7  | Checking middle element: 3
Active items: 5 7  | Checking middle element: 5
Target not found.


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

Q3)

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

int findLastOccurrence(int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;
    int result = -1;

    while (left <= right) {
        int mid = left + (right - left) / 2;
cout << "Active items: ";
        for (int i = left; i <= right; ++i) {
            cout << arr[i] << " ";
        }
        cout << " | Checking middle element: " << arr[mid] << endl;
if (arr[mid] == target) {
          result = mid;  //store index and continue searcg in right half
          left = mid + 1;
       } else if (arr[mid] < target) {
          left = mid + 1; // Search right half
       } else {
          right = mid - 1; // Search left half
       }
   }

   return result; //returns index of last occurrence or -1 if not found
}
int main() {
   int arr[] = {1, 3, 5, 7, 7, 7, 9, 11, 13, 15};
   int size = sizeof(arr) / sizeof(arr[0]);
   int target;
```

```
 cout << "Enter target value: ";
    cin >> target;

    int result = findLastOccurrence(arr, size, target);

    if (result != -1) {
        cout << "Last occurrence of target found at index: " << result <<
endl;
    } else {
        cout << "Target not found." << endl;
    }

    return 0;
}
```


Output

Clear

```
/tmp/E8XqWmr6wg.o
Enter target value: 7
Active items: 1 3 5 7 7 7 9 11 13 15  | Checking middle element: 7
Active items: 7 9 11 13 15  | Checking middle element: 11
Active items: 7 9  | Checking middle element: 7
Active items: 9  | Checking middle element: 9
Last occurrence of target found at index: 5


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

Q4)

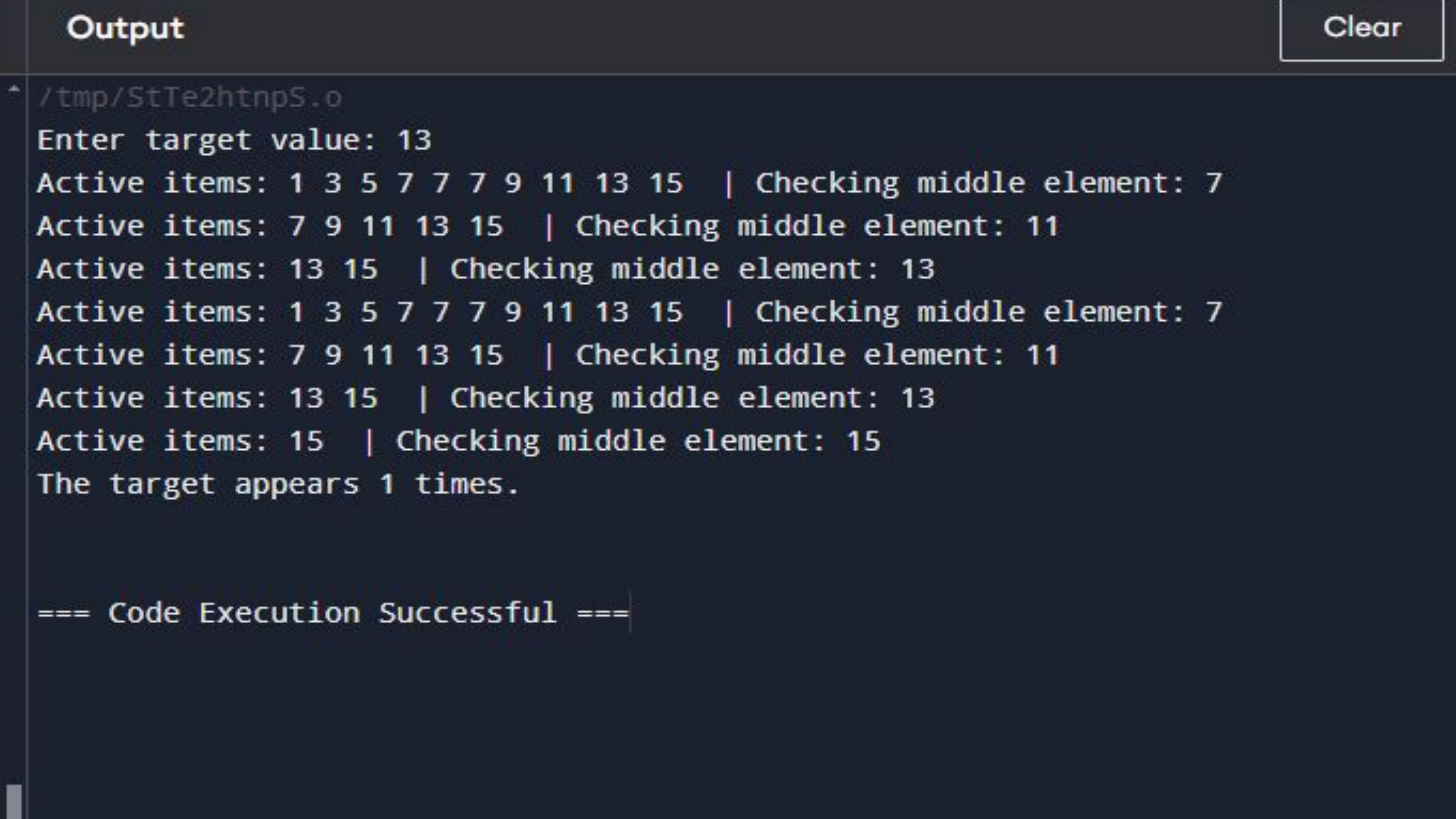
```
#include <iostream>
using namespace std;

int findFirstOccurrence(int arr[], int size, int target) {
    int left = 0, right = size - 1, result = -1;
    while (left <= right) {
        int mid = left + (right - left) / 2;

        cout << "Active items: ";
        for (int i = left; i <= right; ++i) {
            cout << arr[i] << " ";
        }
        cout << " | Checking middle element: " << arr[mid] << endl;
```

```
 if (arr[mid] == target) {
            result = mid;
            right = mid - 1;  //search left half for the first ocr
        } else if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return result;
}

int findLastOccurrence(int arr[], int size, int target) {
    int left = 0, right = size - 1, result = -1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
```

```
cout << "Active items: ";
        for (int i = left; i <= right; ++i) {
            cout << arr[i] << " ";
        }
        cout << " | Checking middle element: " << arr[mid] << endl;
        if (arr[mid] == target) {
            result = mid;
            left = mid + 1;  //search right half for the last occur
        } else if (arr[mid] < target) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return result;
}
```

```
int countOccurrences(int arr[], int size, int target) {
    int first = findFirstOccurrence(arr, size, target);
    if (first == -1) {
        return 0;  //target not found
    }
    int last = findLastOccurrence(arr, size, target);
    return last - first + 1;  //no. of occurrences
}
int main() {
    int arr[] = {1, 3, 5, 7, 7, 7, 9, 11, 13, 15};
    int size = sizeof(arr) / sizeof(arr[0]);
    int target;
```

```
 cout << "Enter target value: ";
    cin >> target;

    int count = countOccurrences(arr, size, target);

    if (count > 0) {
        cout << "The target appears " << count << " times." << endl;
    } else {
        cout << "Target not found." << endl;
    }
    return 0;
}
```



Output

Clear

```
/tmp/StTe2htnpS.o
Enter target value: 13
Active items: 1 3 5 7 7 7 9 11 13 15  | Checking middle element: 7
Active items: 7 9 11 13 15  | Checking middle element: 11
Active items: 13 15  | Checking middle element: 13
Active items: 1 3 5 7 7 7 9 11 13 15  | Checking middle element: 7
Active items: 7 9 11 13 15  | Checking middle element: 11
Active items: 13 15  | Checking middle element: 13
Active items: 15  | Checking middle element: 15
The target appears 1 times.


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