Experiment 3(B)

Student Name: Tanmaya Kumar Pani **UID: 22BCS12986**
Branch: BE-CSE **Section/Group: IOT-613B**
Semester: 5 **Date of Performance: 06/8/24**
Subject Name: Advanced Programming Lab-1 **Subject Code: 22CSP-314**

1. Title: Inserting a Node into a Sorted Doubly Linked List.

2. Objective:

You Given a Reference to the head of Double linked list and an interger data create a new doublyLinkedListNode Object having data value data and insert it at proper location to maintain the sort.

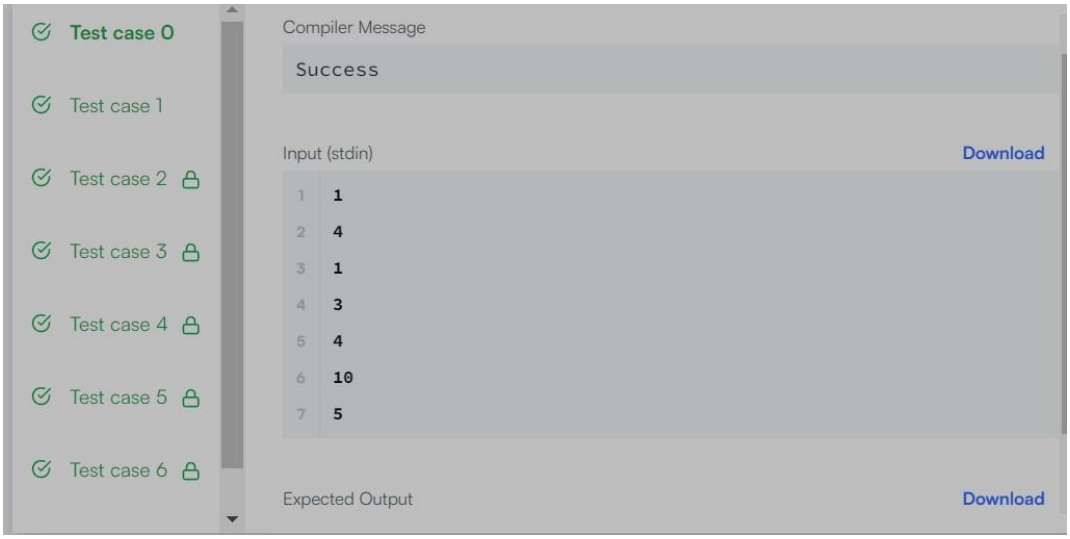
3. Algorithm:

- **Start with an Empty List**: Initialize an empty `std::list`.
- **Read Initial List and New Value**: Read the existing sorted list values and the new value to be inserted.
- **Find Insertion Point**: Traverse the list to find the correct position where the new value should be inserted to maintain sorted order.
- **Insert the New Value**: Insert the new value at the found position.
- **Print the Updated List**: Output the list with the newly inserted value.

4. Implementation/Code

```
#include <iostream>
#include <list>
using namespace std;
void insertSorted(list<int>&
     lst, int data) {
   auto it = lst.begin();
   while (it != lst.end() && *it
    < data) {
      ++it;
   }
   lst.insert(it, data);
}
void printList(const list<int>&
     lst) {
   for (int value : lst) {
      cout << value << " ";
   }
   cout << endl;
}
int main() {
   int t, n, data;
   cin >> t;
   while (t--) {
      cin >> n;
      list<int> lst;
      for (int i = 0; i < n; ++i) {
         int nodeValue;
         cin >> nodeValue;
      lst.push_back(nodeValue);
      }
      cin >> data;
      insertSorted(lst, data);
      printList(lst);
   }
   return 0;
    }
```

5. Output:



6. Learning Outcomes:

- Insert an element into a sorted std::list while maintaining order.
- Read and handle input values, including list elements and new data.
    - Use std::list to manage and manipulate a doubly linked list in C++.

7. Time Complexity: O(n)

8. Space Complexity: O(1)