Experiment 1

Student Name: Virat Samdarshi **UID: 22BCS12648**
Branch: CSE **Section/Group:22BCS_IOT_627/B**
Semester: 5th **Date of Performance: 17/07/2024**
Subject Name: DAA **Subject Code:22CSH-311**

1. **Aim:** Analyze if stack Isempty, Isfull and if elements are present then return top element in stacks using templates and also perform push and pop operation in stack.

2. **Objective:** The objective of this project is to create a template-based stack data structure in C++ that supports the following operations:
   1. Check if the Stack is Empty (**isEmpty**): Determine whether the stack contains any elements.
   2. Check if the Stack is Full (**isFull**): Determine whether the stack has reached its maximum capacity.
   3. Retrieve the Top Element (**top**): If the stack is not empty, return the element at the top of the stack without removing it.
   4. Push Operation (**push**): Add a new element to the top of the stack, provided the stack is not full.
   5. Pop Operation (**pop**): Remove the element from the top of the stack, provided the stack is not empty.

3. **Implementation/Code:**

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

int main() {
    stack<int> s;
    s.push(10);
    s.push(20);
    s.push(30);

```

```
10
11      // Check if the stack is empty
12      cout << "Stack is empty: " << s.empty() << endl;
13      // Get the top element
14      cout << "Top element: " << s.top() << endl;
15      // Perform pop operation
16      s.pop();
17      cout << "Top element after pop: " << s.top() << endl;
18      // Perform another pop operation
19      s.pop();
20      cout << "Top element after another pop: " << s.top() << endl;
21      // Empty the stack
22      while (!s.empty()) {
23          s.pop();
24      }
25      // Check if the stack is empty after emptying it
26      cout << "Stack is empty after popping all elements: " << s.empty() << endl;
27      return 0;
28  }
29
```

4.Output:

```
Stack is empty: 0
Top element: 30
Top element after pop: 20
Top element after another pop: 10
Stack is empty after popping all elements: 1
```

5. Time Complexity

1.Time complexity for insertion in stack: O (1).

2.Time complexity for deletion in stack: O (1).

3.Time complexity for seeking the top element in stack: O (1).

4.Time complexity for checking the size of the stack: O (1).

5.Time complexity for checking if the size is empty or not: O (1)