Experiment 1

Student Name: ISHITA NISHANT **UID: 23BCS11354**
Branch: BE CSE **Section/Group: KRG 2A**
Semester: 5 **DateofPerformance:25/7/2025**
Subject Name: DAA **Subject Code: 23CSH-301**

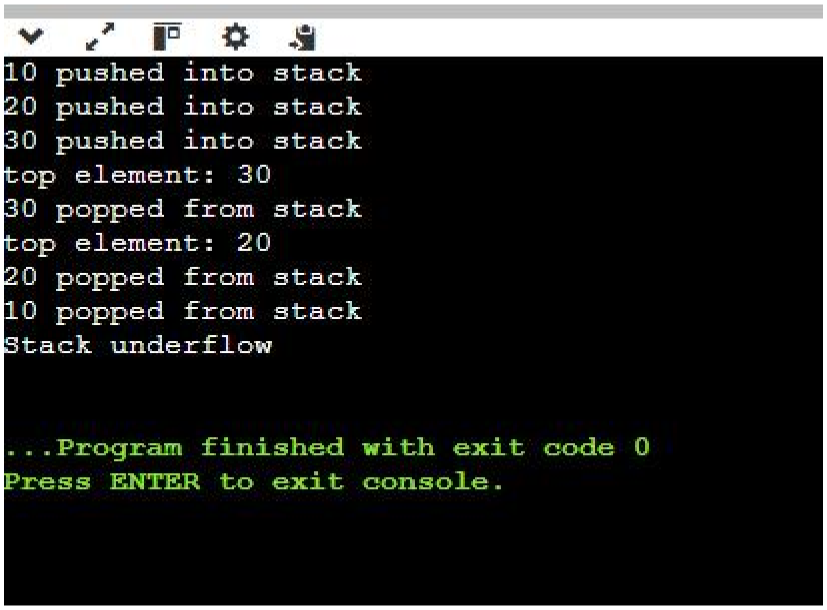
1. **Aim: Analyze if the stack is empty or full, and if elements are present, return the top element in the stack using templates. Also, perform push and pop operations on the stack.**

2. **Objective:** To understand the concept and implementation of stacks.

3. **Implementation/Code:**

```
#include <iostream>
using namespace std;
const int size=5;
template <class T>
class Stack{
   private:
   T arr[size];
   int top;
   public:
   Stack(){
      top=-1;
   }
   bool isFull(){
      return top==size-1;
   }
   bool isEmpty(){
      return top==-1;
   }
   void push(T value){
      if (isFull()){
          cout<<"Stack Overflow!\n";
      }
      else{
          arr[++top]=value;
          cout<<value<<" pushed into stack\n";
      }
   }
   void pop(){
      if (isEmpty()){
          cout<<"Stack underflow\n";
      }
      else{
          cout<<arr[top--]<<" popped from stack\n";
      }
   }
   void peek(){
      if(isEmpty()){
          cout<<"Stack Underflow!\n";
      }
      else{
          cout<<"top element: "<<arr[top]<<endl;
      }
   }
};
int main(){
   Stack <int>s;
   s.push(10);
   s.push(20);
   s.push(30);
   s.peek();
   s.pop();
   s.peek();
   s.pop();
   s.pop();
   s.pop();
   return 0;
}
```

4. Output



```
10 pushed into stack
20 pushed into stack
30 pushed into stack
top element: 30
30 popped from stack
top element: 20
20 popped from stack
10 popped from stack
Stack underflow


...Program finished with exit code 0
Press ENTER to exit console.
```

5. Learning Outcome

- Understand the basic concept of stack (LIFO).
- Implement stack operations using C++ templates.
- Handle stack overflow and underflow conditions.
- Use generic code for different data types.
- Analyse stack operations with O(1) time complexity