Experiment 1.1

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.*

Objectives: *To understand stacks.*

Input/Apparatus Used: *VS CODE*

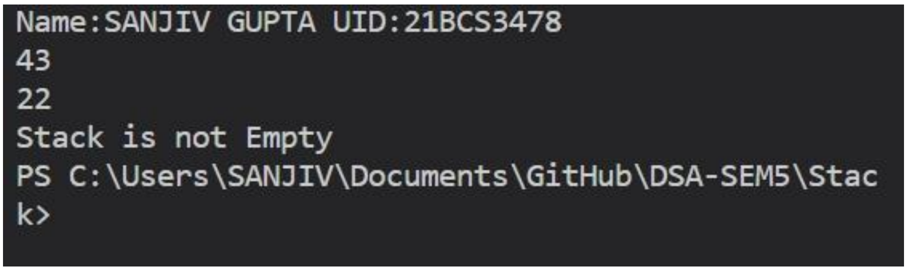
Procedure/Algorithm:

Step1: Create stack.
Step2: Check underflow and overflow condition.
Step3: Increment top to store element in stack.
Step4: Decrement top after removing element form stack.
Step5: Check is stack empty or not.

Sample Code:

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

class Stack {
    public:
        int *arr;
        int top;
        int size;

    Stack(int size) {
        this -> size = size;
        arr = new int[size];
        top = -1;
    }

    void push( int element) {
        if(size - top > 1) {
            top++;
            arr[top] = element;
        }
        else{
            cout << "Stack OverFlow" << endl;
        }
    }

    void pop() {
        if(top >=0 ) {
            top--;
        }
        else{
            cout << "Stack UnderFlow" << endl;
        }
    }

    int peek() {
        if(top >=0 )
            return arr[top];
        else
        {
            cout << "Stack is Empty" << endl;
            return -1;
        }
    }

    bool isEmpty() {
        if( top == -1) {
            return true;
        }
        else{
            return false;
        }
    }
};

int main() {
    Stack st(5);
    cout<<"Name:SANJIV GUPTA"<<" "<<"UID:21BCS3478"<<endl;
    st.push(22);
    st.push(43);
    st.push(44);
    st.push(22);
    st.push(43);

    cout<<st.peek()<<endl;
    st.pop();
    cout<<st.peek()<<endl;

    if(st.isEmpty()) {
        cout << "Stack is Empty" << endl;
    }
    else{
        cout << "Stack is not Empty" << endl;
    }
    return 0;
}
```

Observations/Outcome :



```
Name:SANJIV GUPTA UID:21BCS3478
43
22
Stack is not Empty
PS C:\Users\SANJIV\Documents\GitHub\DSA-SEM5\Stac
k>
```

Time Complexity: O(1)

Stack work on LIFO(Last in first out). The time complexity of all stack operations is constant.