**EXPERIMENT 1.1**

**Name:** Pranav Aggarwal **UID:** 21BCS3197

**Branch:** CSE 3rd Year **Section/Group:** 619-B

**Semester:** 5th  **Subject Code:** 21CSH-311

**Subject:** Design and Analysis **Date of Performance:** 07/08/23  of Algorithms

**AIM:** Analyse 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:** Visual Studio Code **PROCEDURE/ ALGORITHM:**

1. Include necessary headers:
   * Include the iostream header for input and output.
   * Use the std namespace for convenience.

1. Define the Stack class template:
   * Define a class template named Stack that takes a template parameter T for the element type.
   * Private members:

 data: A dynamic array to hold the stack elements.

 top: An integer representing the index of the top element in the stack.

 maxSize: An integer representing the maximum size of the stack.

* + Public methods:

 Constructor:

* + - Accepts the maximum size of the stack as a parameter.
    - Initializes the maxSize, creates a dynamic array of type T to store the data, and initializes top to -1.

 push method:

* + - Accepts an element of type T as a parameter.
    - Checks if the stack is full using the isFull method.
    - If the stack is full, print an error message.
    - Otherwise, increment top, and store the element in the data array.

 pop method:

* + - Checks if the stack is empty using the isEmpty method.
    - If the stack is empty, print an error message and return a default-constructed T.
    - Otherwise, retrieve the top element, decrement top, and return the element.

 isEmpty method:

* + - Returns true if the stack is empty (i.e., top is -1), otherwise returns false.

 isFull method:

* + - Returns true if the stack is full (i.e., top is equal to maxSize - 1), otherwise returns false.

 topElement method:

* + - Checks if the stack is empty.
    - If the stack is empty, print an error message and return a default-constructed T.
    - Otherwise, return the top element of the stack.

1. Main function:
   * Create an instance of the Stack class with a maximum size of 10 and element type int.
   * Push three integers (1, 2, and 3) onto the stack.
   * Print the top element of the stack.
   * Pop one element from the stack.
   * Print the new top element of the stack.
   * Return 0 to indicate successful execution.

**CODE:** #include <iostream> using namespace std;

template <typename T> class Stack { private: T \*data; int top; int maxSize;

public: Stack(int maxSize) { this->maxSize = maxSize; data = new T[maxSize]; top = -1;

}

void push(T element) {

if (isFull()) { cout << "Stack is full!" << endl; return;

}

top++;

data[top] = element;

}

T pop() { if (isEmpty()) { cout << "Stack is empty!" << endl; return T();

}

T element = data[top]; top--; return element;

}

bool isEmpty() { return top == -1;

}

bool isFull() { return top == maxSize - 1;

}

T topElement() { if (isEmpty()) { cout << "Stack is empty!" << endl; return T();

}

return data[top];

}

};

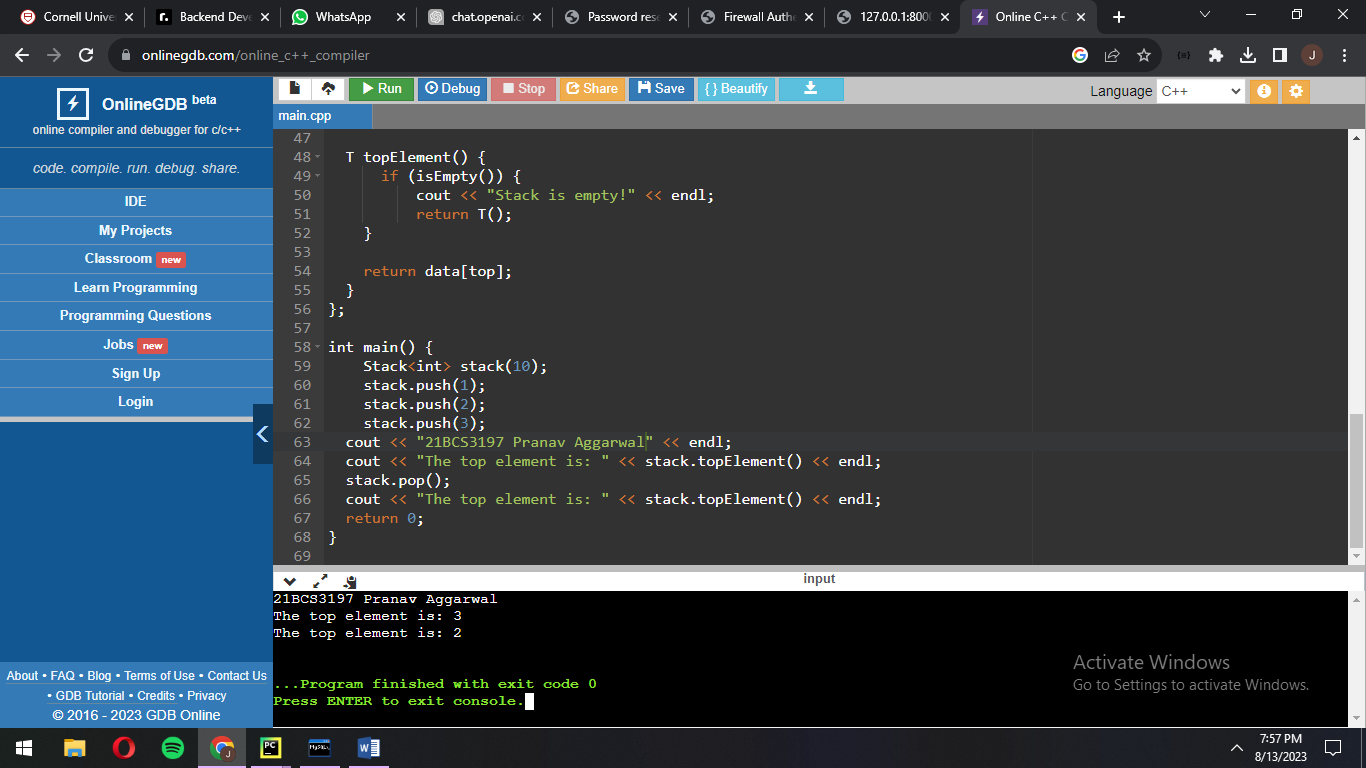
int main() { Stack<int> stack(10); stack.push(1); stack.push(2); stack.push(3);

cout << "The top element is: " << stack.topElement() << endl; stack.pop();

cout << "The top element is: " << stack.topElement() << endl; return 0;

}

**OUTCOME:**

****

**TIME COMPLEXITY:** O(n)