University of Central Punjab
(Incorporated by Ordinance No. XXIV of 2002 promulgated by Government of the Punjab)
FACULTY OF INFORMATION TECHNOLOGY

Data Structures and Algorithms - Lab

Lab 5			
CLO NO	**CLO STATEMENT**	**Blooms Taxonomy Level**	**PLO**
1	Solve real-world problems skillfully with precision using programming constructs learned in theory with the course toolkit.	P3	5

Task 1

Create a generic class `AbstractStack` with the following attributes:

```
public:
    virtual void push(T value) = 0;
    virtual T pop() = 0;
    virtual T top() const = 0;
    virtual bool isEmpty() const = 0;
    virtual bool isFull() const = 0;
    virtual ~AbstractStack() {}     // Virtual destructor
```

Create a child class myStack which implements the following functions of the parent class:

- **`push(T value)`**: Adds an element to the top of the stack.
- **`T pop()`**: Removes the top element and returns it.
- **`T top()`**: Returns the top element.
- **`isEmpty()`**: Checks if the stack is empty.
- **`isFull()`**: Check if the stack is full.

More functions to implement:

- **`display():`** To display the values present in the stack – this is an exclusive function of the class myStack.
- Main Function: Provides a menu for users to perform stack operations interactively.

Task 2

You are required to perform the addition of two very large non-negative integers that cannot be stored in standard integer data types. To achieve this, implement the logic using **stacks**, where **each digit of the numbers is stored separately**.

Requirements:

- Read **two large numbers** as **strings** from the user.
- Push **each digit** of the numbers into **two separate stacks**, so that the least significant digit is at the top.
- Perform the addition **digit by digit** using a **third stack** to store the result.
- Convert each character digit into an integer by subtracting '0' (or 48) before performing the addition.
- **Handle carry** from the addition properly across digits.
- Finally, **display the sum** as a properly formatted number by popping digits from the result stack.

Task 3

You are required to determine whether a given string is a **palindrome** using a stack. A palindrome is a word, number, or sequence that reads the same forward and backward.

Read a **string** from the **user**. Use a **stack** to reverse the string by pushing each character onto the stack and then popping them one by one to form the reversed version. Finally, compare the reversed string with the original input. If both are the same, the input is a **palindrome**.

Task 4

Write a program to convert an **infix** expression to a **postfix** expression using a **stack**. The infix expression is the standard mathematical notation (e.g., A + B).

Example:
Enter an infix expression: (A+B)*(C-D)

Postfix expression: AB+CD-*