5 DATA STRUCTURES

Lab Exercise # 5: Stack

In the lab exercise, you will complete a C++ program that enables users to perform various operations on a Stack. Download the starter code from Brightspace and complete the following tasks:

1. **Task-1: MyStack Class** **(2 Points)**

   Create a templated **array-based** stack class, called **MyStack** that should contain at least the following methods:

   - `push(e):` push an element to the top of the stack. This method must throw an **exception** if the stack is full.
   - `pop():` Remove the element from the top of the stack. This method must throw an **exception** if the stack is empty.
   - `top():` Return a reference to the element at the top of the stack without removing it. This method must throw an **exception** if the stack is empty.
   - `size():` Return the size (number of current elements) of the stack.
   - `empty():` Return true if the stack is empty, false otherwise.

   Note: The generated **exception** must be derived from the C++ "**exception**" class so that the **catch** block in the **main()** function will be able to catch it. You can also use the exception classes from the C++ standard library.

2. **Task-2: isBalanced Expression** **(2 Points)**

   Write a function called **"isBalanced (expression)"** that takes an expression as an argument and checks for balanced parentheses in that expression using stack. An expression is a string comprising of opening and closing parentheses( ), you need to check whether parentheses are balanced or not. For example:

   - `(2+3)*5` is a balanced expression
   - `((2+3)*(5-2))` is a balanced expression
   - `(2+3)*(5` is not a balanced expression
   - `)2+3(` is not a balanced expression

   Algorithm to Check if an expression is Balanced (Parentheses) using Stack

   1. **Initialize** an empty stack.
   2. **Scan** the expression from left to right and **repeat steps 2a to 2b** for each element of the expression.
      a. If '**(**' is found then push it on the stack
      b. else if '**)**' is found then pop an element from the top of the stack. (return false if the stack is empty)
   3. If the stack is **empty**, then the expression is balanced: return **true**, else return **false**

3. Task-3: Infix to Postfix (3 Points)

Complete the function **infix2postfix(infix)** which converts an Infix expression into a Postfix expression using stack.

Infix is an expression in which the operator is written between two operands, e.g.,

- `2+3`
- `2+3*5`
- `(2+3)*5`

Postfix is an expression in which the operator is written after its operands, e.g.,

- `23+`
- `235*+`
- `23+5*`

Note: Your program should support only **single-digit numbers (0-9**) and **binary operators +** , **-**, ***** , **/** , **^**.

Algorithm to convert Infix to Postfix

1. Create a stack of type char to store operators and parenthesis.
2. Scan **INFIX** from left to right and repeat steps 3 to 6 for each element of **INFIX**
3. If an **operand** is found, add it to **POSTFIX.**
4. If **(** is found, push it onto the Stack.
5. If an **operator** is found:
    a. **Pop** from **Stack** and add to **POSTFIX** each operator from the top of the Stack with the same or higher precedence until **(** or an operator with lower precedence is found or the stack becomes empty.
    b. **Push** the **operator** to Stack.
    [end of if]
6. If **)** is found:
    a. **pop** from the **Stack** and add to **POSTFIX** each operator until **(** is found.
    b. Pop the **(** from **Stack**.
    [end of if]
7. **Pop** all remaining operators from the **stack** and add them to the **POSTFIX**.
8. END.

4. Task-4: Postfix Evaluation (2 Points)

Complete the function **evaluate(string postfix)** to your program, which evaluates the postfix expression using a Stack and returns the result to the calling program.

Note: For simplicity, you can assume that each number and operator in the input string is only one letter long.

Algorithm to evaluate a postfix expression

1. Create a **Stack** to store the operands (values)
2. Scan the **POSTFIX** expression from left to right for every element.
    a. if an **operand** is found, push it to the stack.
    b. if an **operator** is found, pop 2 elements from the stack, apply the operator on it, and push the result back into the stack.
3. The number (element) left in the **Stack** is the final answer of the expression.

Comments (1 Point)

Comments are a very important part of any program. You must always write comments in your code.

Desired Output

```
Enter an Infix Expression: (2+3)*5
The postfix form is: 23+5*
(2+3)*5=25
Enter an Infix Expression: ((2+3)*(5-2))
The postfix form is: 23+52-*
((2+3)*(5-2))=15
Enter an Infix Expression: 5/2
The postfix form is: 52/
5/2=2.5
Enter an Infix Expression: (2+3)*(5
Expression is not Balanced
Enter an Infix Expression: )2+3(
Expression is not Balanced
Enter an Infix Expression: exit
```

CODE OF CONDUCT

All assignments are graded, meaning we expect you to adhere to the academic integrity standards of NYU Abu Dhabi. To avoid any confusion regarding this, we will briefly state what is and isn't allowed when working on an assignment/lab-task.

Any documents and program code that you submit must be fully written by yourself. You can, of course, discuss your ideas with fellow students, as long as these discussions are restricted to general solution techniques. Put differently, these discussions should not be about concrete code you are writing, nor about specific results you wish to submit. When discussing an assignment with others, this should never lead to you possessing the complete or partial solution of others, regardless of whether the solution is in paper or digital form, and independent of who made the solution, meaning you are also not allowed to possess solutions by someone from a different year or course, by someone from another university, or code from the Internet, etc. This also implies that there is never a valid reason to share your code with fellow students, and that there is no valid reason to publish your code online in any form.

Every student is responsible for the work they submit. If there is any doubt during the grading about whether a student created the assignment themselves (e.g. if the solution matches that of others), we reserve the option to let the student explain why this is the case. In case doubts remain, or we decide to directly escalate the issue, the suspected violations will be reported to the academic administration according to the policies of NYU Abu Dhabi.

https://students.nyuad.nyu.edu/campus-life/community-standards/policies/academic-integrity/