Due: 14/12/2024

Lab Assignment One: Sequential Stack Applications

Part One: Balanced Parenthesis Check

You are required to implement an application based on stack for testing balanced parenthesis (),{},[]. The input to your program is a ***parenthesis*** string.

1. Implement the basic stack operations **based on sequential allocation paradigm**:
   - Create: creates and initializes an empty stack
   - Push: pushes and element on to the stack.
   - Pop: pops an element from the top of stack and returns it.
   - Is Empty: checks if the stack is empty.
   - Is Full: checks if the stack is full.
2. Use the implemented stack to realize the solution to the problem mentioned.
   Example runs are shown below (expect the string to be as much as 80 characters long)
   Hint: Refer to the following C++ code and try to re-implement it in any language of your choice.

Example or Run

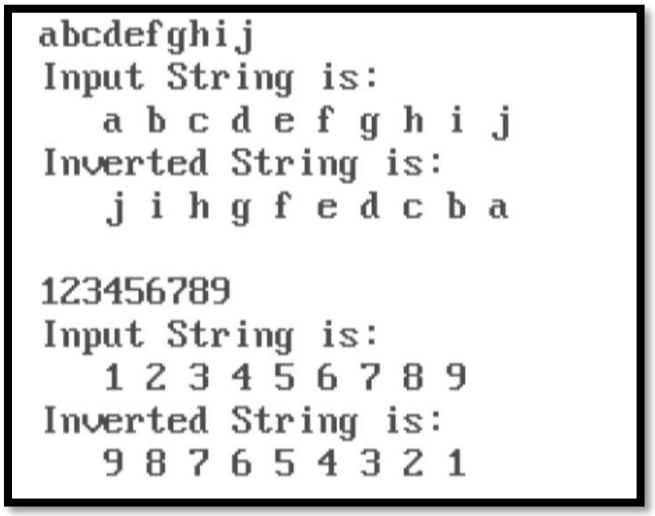
Underlined text is user's input

```
({()[()]})
Input is: ({()[()]})
Properly Nested structure
((0
illegal char in input
(()))
Input is: (()))
Not Ballanced
(()]
Input is: (()]
Not Ballanced
```

Part Two: String Inversion

You are required to implement an application based on stack for inverting a string. The input is expected to be a character string (including numbers, special characters and English alphabet).

1. Implement the basic stack operations:
   - You can reuse the code you implemented in part one.
2. Use the implemented stack to realize the solution to the problem mentioned.
   Example runs are shown below (expect the string to be as much as 80 characters long)
   Hint: Refer to the following C++ code and try to re-implement it in any language of your choice.



Deliverable

- You are required to submit a working source code, in addition to a report showing your own test cases. Further details will follow.

Policy

- You are required to work in groups of maximum five students.
- Plagiarism will be severely penalized so it is better to deliver nothing than to deliver a copied code and/or report.