

## Lab #9: Stack, Queue

The main aim of the lab is to get familiar with some methods in **Stack** and **Queue**.

=====

### Task 1: Stack

=====

By using Stack, implement the following methods in **MyLIFO\_App** class:

```
public class MyLIFO_App {
    // This method reserves the given array
    public static <E> void reserve(E[] array) {
        // TODO
    }

    // This method checks the correctness of the
    given input
    // i.e. () (()) [] { (()) } ==> true, ) { [] } () ==>
    false
    public static boolean isCorrect(String input) {
        // TODO
        return false;
    }

    // This method evaluates the value of expression
    // i.e. 51 + (54 * (3+2)) = 321
    public static int evaluateExpression(String
expression) {
        // TODO
        return 0;
    }

    // This method checks the well-defined of a given
    HTML or XML file
    public static boolean isWell_Defined(String
fileName) {
        // TODO
        return false;
    }
}
```

**Hint** (for `evaluateExpression` method):

### Phase 1: Scanning the expression

The program scans the expression from left to right to extract operands, operators, and the parentheses.

- 1.1. If the extracted item is an operand, push it to **operandStack**.
- 1.2. If the extracted item is a + or - operator, process all the operators at the top of **operatorStack** and push the extracted operator to **operatorStack**.
- 1.3. If the extracted item is a \* or / operator, process the \* or / operators at the top of **operatorStack** and push the extracted operator to **operatorStack**.
- 1.4. If the extracted item is a ( symbol, push it to **operatorStack**.
- 1.5. If the extracted item is a ) symbol, repeatedly process the operators from the top of **operatorStack** until seeing the ( symbol on the stack.

### Phase 2: Clearing the stack

Repeatedly process the operators from the top of **operatorStack** until **operatorStack** is empty.

---

## Task 2: Queue

---

By using Queue, implement the following methods in **MyFIFO\_App** class:

```
public class MyLIFO_App {
    // method stutter that accepts a queue of integers as
    // a parameter and replaces
    // every element of the queue with two copies of that
    // element
    // front [1, 2, 3] back
    // becomes
    // front [1, 1, 2, 2, 3, 3] back
    public static <E> void stutter(Queue<E> input) {
        // TODO
    }
    // Method mirror that accepts a queue of strings as a
    // parameter and appends the
    // queue's contents to itself in reverse order
}
```

```
// front [a, b, c] back  
// becomes  
// front [a, b, c, c, b, a] back  
  
public static <E> void mirror(Queue<E> input) {  
    // TODO  
}  
}
```