**Lab 10. (Stacks and Queues)**

**Problem A**

You are given a list of Strings as input. Check whether each string is a palindrome or not using a stack data structure. The first line of input ***n*** is the number of test cases followed by n strings.

**Sample Input:**

3

test

baab

uigui

No

Yes

Yes

Solution: <http://www.javaproblems.com/2013/12/check-if-string-is-palindrome-using.html>

**Problem B**

You are given a list of postfix expressions. Calculate the result of the expressions using a stack. The first line of input ***n*** is the number of test cases followed by n expressions. Each expression begins with an integer ***t*** representing the number of characters in the expression, and then ***t*** characters follow which are either a number or an operand.

**Sample Input:**

3

2 3 \*

5 4 2 + -

5 6 2 + \*12 4 / -

6

1

37

Solution: <http://www.javaproblems.com/2013/12/calculate-postfix-expressions-using.html>

**Problem C**

You are given a list of integers as input that you should add to a queue. If the integer at the beginning of the queue is an even number, you should remove it from the queue; else, if the integer at the beginning of the queue is odd, you should remove to elements from the queue. Keep repeating this until the queue is empty. The input guarantees that the queue will be empty in the end and no runtime errors will occur. The first line of input ***n*** is the number of test cases followed by ***n*** lists. Each list begins with an integer ***t*** representing the number of integers in the list, and then ***t*** integers follow. As output, you should print on each line the number (or numbers) removed.

**Sample Input:**

3

2 3 4

1 2 3 5 4 8

8 4 2 1 6

**Sample Output:**

Test case 1:

2

3 4

Test case 2:

1 2

3 5

4

8

Test case 3:

8

4

2

16

**Solution:** [**http://www.javaproblems.com/2013/12/adding-or-removing-odd-and-even-numbers.html**](http://www.javaproblems.com/2013/12/adding-or-removing-odd-and-even-numbers.html)

**Problem D:**

To convert a number from decimal to binary, you simply divide by two until a quotient of zero is reached, then use the successive remainders in reverse order as the binary representation. Use a stack to store the remainder of the division and finally print the binary representation. The first line of input n is the number test cases. Each test case begins with an integer *t* representing the number of integers in the list, then ***t*** integers follow.

**Sample Input:**

3

5

7

10

101

111

1010

Solution: <http://www.javaproblems.com/2013/12/convert-from-decimal-to-binary-using.html>