**Assignment 1: Lambdas**

Write the following methods that return a lambda expression performing a specified action:

PerformOperation isOdd(): The lambda expression must return if a number is odd or if it is even.

PerformOperation isPrime(): The lambda expression must return if a number is prime or if it is composite.

PerformOperation isPalindrome(): The lambda expression must return if a number is a palindrome or if it is not.

**Sample Input**

The first line contains an integer, (the number of test cases).

The subsequent lines each describe a test case in the form of space-separated integers:

The first integer specifies the condition to check for ( for Odd/Even, for Prime, or for Palindrome). The second integer denotes the number to be checked.

5

1 4

2 5

3 898

1 3

2 12

**Sample Output**

EVEN

PRIME

PALINDROME

ODD

COMPOSITE

**Assignment 2: Functional**

Given a list of non-negative integers, return an integer list of the rightmost digits. (Note: use %)

rightDigit([1, 22, 93]) → [1, 2, 3]

rightDigit([16, 8, 886, 8, 1]) → [6, 8, 6, 8, 1]

rightDigit([10, 0]) → [0, 0]

**Assignment 3: Functional**

Given a list of integers, return a list where each integer is multiplied by 2.

doubling([1, 2, 3]) → [2, 4, 6]

doubling([6, 8, 6, 8, -1]) → [12, 16, 12, 16, -2]

doubling([]) → []

**Assignment 4: Functional**

Given a list of strings, return a list where each string has all its "x" removed.

noX(["ax", "bb", "cx"]) → ["a", "bb", "c"]

noX(["xxax", "xbxbx", "xxcx"]) → ["a", "bb", "c"]

noX(["x"]) → [""]

**Assignment 5: Recursion**

Given an array of ints, is it possible to choose a group of some of the ints, such that the group sums to the given target, with this additional constraint: if there are numbers in the array that are adjacent and the identical value, they must either all be chosen, or none of them chosen. For example, with the array {1, 2, 2, 2, 5, 2}, either all three 2's in the middle must be chosen or not, all as a group. (one loop can be used to find the extent of the identical values).

groupSumClump(0, [2, 4, 8], 10) → true

groupSumClump(0, [1, 2, 4, 8, 1], 14) → true

groupSumClump(0, [2, 4, 4, 8], 14) → false

**Assignment 6: Singleton**

Fix the below Singleton class:

import java.math.BigDecimal;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.ResultSet;

import java.sql.Statement;

public static class SampleSingleton {

private static Connection conn = null;

private static SampleSingleton instance = null;

public static SampleSingleton getInstance() {

return instance;

}

}