Computer Theory Project #1: Recursive Definitions of Languages

In this project, you will write a program to implement various recursive definitions of the following languages. Your program should allow the user to run through all six languages, providing test input for each. (If you want, you can hard code all test input, so that your program simply prints true or false for each test input for all six languages. This would mean the user doesn't have to manually enter any data.) You can do this program in any approved language (Java, C, C++, .NET, Bash, Python, Perl, and JavaScript.)

Output should look similar to the following:

Enter test for DivisibleBy7Language: 28
X = 28 Member? Yes

1. Language **DivisibleBy7** is defined recursively as:
   (1) Integer 7 is in DivisibleBy7.
   (2) If x is in DivisibleBy7, then so is x + 7.
   Encode DivisibleBy7 as a boolean recursive function. Test your function for input values: 382, 749, 2977, 9989, 52878.

2. Language **PowerOf2** is defined recursively as:
   (1) Integer 1 is in PowerOf2.
   (2) If x is in PowerOf2, then so is 2x.
   Encode PowerOf2 as a boolean recursive function. Test your function for input values: 128, 257, 1023, 8192, 65536.

3. Language **SumXY** is defined recursively as:
   (1) Integers 7 and 43 are in SumXY.
   (2) If x and y are in SumXY, then so is x + y.
   Encode SumXY as a boolean recursive function. Test your function for input values: 12, 51, 137, 364, 589.

4. For alphabet `{a, b}`, the language **OddPalindrome** is defined recursively as:
   (1) Words `a` and `b` are in OddPalindrome.
   (2) If `w` is a word in OddPalindrome, then so are `awa` and `bwb` (concatenation).
   Encode OddPalindrome as a boolean recursive function. Test your function for input values: abaaba, bbaabbb, abababa, aabcbaa, baabbbaab.

5. For alphabet `{a, b}`, the language **ContainsABA** is defined recursively as:
   (1) Word `aba` is in ContainsABA
   (2) If `w` is a word in ContainsABA, then so are `aw, bw, wa,` and `wb`.
   Encode ContainsABA as a boolean recursive function. Test your function for input values: abaaba, bbaabbb, abababa, aabcbaa, baabbbaab.

6. For alphabet `{a, b, +}`, the language **ABPlus** is defined recursively as:
   (1) Words `a` and `b` are in ABPlus.
   (2) If `w` and `v` are words in ABPlus, then so are `vw` and `v+w`.
   Encode ABPlus as a boolean recursive function. Test your function for input values: aba+ba, b+ba+bb, ab++ba+a, +aab+bba, b+abb+ab+.

Sample Problem

Language **EndsWith8** is the set of postive integers having 8 as the last digit. It can be defined recursively as:

(1) Integer 8 is in EndsWith8.

(2) If x is in EndsWith8, then so is x + 10.

Encode EndsWith8 as a boolean recursive function.

Sample Java code

```
import java.io.*;

public class TestEndsWith8
{
  public static void main(String args[])
  {
    //Code goes here to call the EndsWith8 method passing in a test number
    //as an argument.
  }

  public static boolean EndsWith8(int x)
  {
    if (x < 8)
      return false;
    if (x == 8)
      return true;
    return EndsWith8(x - 10);
  }
}
```