Day 14: Scope | HackerRank

Terms you'll find helpful in completing today's challenge are outlined below, along with sample Java code (where appropriate).

Scope
This term refers to the region of the program to which an identifier applies. While it is not good practice, you can declare multiple variables within a program that use the same identifier *as long as* the identifiers have differing scopes; some exceptions to this are:

1. A constructor or method parameter will often have the same name as a class field it's intended to initialize or modify.
2. It is customary to use `i` as the condition variable in a for-loop (and, in cases of nested for-loops, to use `j` as the condition variable for the inner loop).

Note: When dealing with a class variable (field), it's always best to explicitly refer to it using the *this* keyword. For example:

```
class MyClass{
    private int myInt;

    public MyClass(int myInt){
        this.myInt = myInt;
    }
}
```

Even though there is a `myInt` field in the class, the constructor has a completely different `myInt` parameter. The field (`this.myInt`) is then assigned the value of the parameter, so any argument passed as that parameter will initialize the `myInt` field in the class.

Still confused? Take some time to review the *Scope* class below and understand why variables with the same name will be of different types (and produce different outputs) at various points in the program:

```
public class Scope{
    boolean b = true; // b1 has scope of entire class
    int x = 88; // x1 has scope of entire class

    Scope(){
        double d = 9.0;
        example(d);
        classVariable();
    }

    void example(double x){ // parameter x2 has scope of this method
        System.out.println("----- example(double x):\n"
                +       "Initial value of Local Variable `x`: " + x + "\n");

        x = 4.4; // reassign value of local variable x2

        System.out.println("New value of Local Variable `x`: " + x + "\n");

        for(int b = 0; b < 4; b++){ // b2 has scope of this loop
            int i = b + 4; // begin scope of int i
            System.out.println( "For Loop 1 in example(double x):\n"
                +       "Local Variable `b` (local to loop): " + b + "\n"
                +       "Local Variable `i` (local to loop): " + i + "\n"
                +       "Local Variable `x` (method parameter): " + x + "\n");
        } // end the scope of int b2; end scope of int i
```

```
        for(int b = 0; b < 4; b++){
            x = b;
            System.out.println( "For Loop 2 in example(double x):\n"
                +       "Local Variable `b` (local to loop): " + b + "\n"
                +       "Local Variable `x` (method parameter): " + x + "\n");
        } // end of the scope of this version of int b

        System.out.println("Local Variable `x` after Loop 2: " + x + "\n");

    } // end scope of double x2

    void classVariable(){
        System.out.println(     "----- classVariable():\n"
                +       "Instance Variable `b`: " + b + "\n"
                +       "Instance Variable `x`: " + x);
    }

    public static void main(String[] args){
        Scope s = new Scope();
    }
} // end of boolean b's scope; end of int x's scope
```

which produces this output:

```
----- example(double x):
Initial value of Local Variable `x`: 9.0

New value of Local Variable `x`: 4.4

For Loop 1 in example(double x):
Local Variable `b` (local to loop): 0
Local Variable `i` (local to loop): 4
Local Variable `x` (method parameter): 4.4

For Loop 1 in example(double x):
Local Variable `b` (local to loop): 1
Local Variable `i` (local to loop): 5
Local Variable `x` (method parameter): 4.4

For Loop 1 in example(double x):
Local Variable `b` (local to loop): 2
Local Variable `i` (local to loop): 6
Local Variable `x` (method parameter): 4.4

For Loop 1 in example(double x):
Local Variable `b` (local to loop): 3
Local Variable `i` (local to loop): 7
Local Variable `x` (method parameter): 4.4

For Loop 2 in example(double x):
Local Variable `b` (local to loop): 0
Local Variable `x` (method parameter): 0.0

For Loop 2 in example(double x):
Local Variable `b` (local to loop): 1
Local Variable `x` (method parameter): 1.0

For Loop 2 in example(double x):
Local Variable `b` (local to loop): 2
Local Variable `x` (method parameter): 2.0

For Loop 2 in example(double x):
Local Variable `b` (local to loop): 3
Local Variable `x` (method parameter): 3.0

Local Variable `x` after Loop 2: 3.0

----- classVariable():
Instance Variable `b`: true
```

```
Instance Variable `x`: 88
```

Solve Problem