CSCI 1133, Spring 2015
Programming Assignment 3

Assigned: Friday Feb 13, 2015
Due: 11:55pm Thursday Feb 19, 2015

This is ***not*** a collaborative assignment; you must design, implement and test the solution(s) on your own. You may not consult or work with anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class. Obtaining or *providing* solutions to any homework problems for this class is considered academic misconduct. If you are not sure what this means, consult the class syllabus or discuss it with the course instructor.

This assignment requires writing two *separate* Python scripts that must be submitted online *prior* to the due date/time. Late submissions will not be accepted. Name each source code: `hw3`*prob*`.py` where *prob* is replaced by the corresponding problem letter: e.g., `hw3A.py` , `hw3B.py` Submit each source code file separately using the appropriate homework submission link on the Moodle website.

The total point value for each problem will be awarded for a solution that is *complete*, *correct*, and *well constructed*. A *"well constructed"* program entails good design, appropriate comments and general readability (descriptive names for variables and procedures, appropriate use of blank space, etc.). The following will result in a score reduction equal to a percentage of the total possible points:

- Incorrectly named/submitted source file (10%)
- Constraints not followed (40%)
- Failure to execute due to syntax errors (30%)

Note that your work will be graded using, and must function correctly with, the current version of Python 3 on CSE Labs UNIX machines. If you complete your programming assignment using a different system, it is your responsibility to ensure your programs work on CSELabs machines *prior* to submitting them.

A. (20 *points*) **Decimal to Binary Conversion**

Write a Python program that will input an integer value greater than or equal to zero and output its equivalent *binary* representation. e.g.:

$5_{10} = 101_2$

Note that the least significant (rightmost) binary digit is simply the remainder of division by 2. The equivalent binary representation can be obtained by repeatedly dividing by two and *constructing* the representation one digit at a time. Note that you will need to construct the binary representation using string concatenation, then output the resulting string. You also need to verify that the input is within the stated bounds.

Constraints:

- Use an indefinite `while` loop (no `for` loops, no user defined functions) to accomplish the conversion
- Do not import any modules; use only built-in functions and simple arithmetic operations
- Do not use any string methods other than simple concatenation

Example:

```
Enter an integer value: 5
Base-2 equivalent: 101

Enter an integer value: 0
Base-2 equivalent: 0

Enter an integer value: -3
Input must be greater or equal to zero!
```

B. (20 *points*) **Perfect Numbers**

An integer is said to be a *perfect number* if the sum of its divisors, including 1 (but not the number itself), is equal to the number. For example, 6 is a perfect number because 6 = 1 + 2 + 3. Write a Python program that will identify and print all the perfect numbers in some closed interval [ 1, n ], one per line. Hint: there are exactly four such numbers in the interval [1,10000].

Constraints:

- You must use a nested `while` loop (no `for` loops, no user defined functions)
- Do not import any modules
- Input the upper bound of the interval as an integer and verify that it is greater than 1

Example:

```
Enter the upper bound: 10
Perfect numbers in the interval [0,10]:
6

Enter the upper bound: 0
Invalid upper bound!
```