CSCI 1133, Spring 2015
Programming Assignment 6

Assigned: Friday March 6, 2015
Due: 11:55pm Thursday March 12, 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: `hw6`*prob*`.py` where *prob* is replaced by the corresponding problem letter: e.g., `hw6A.py` , `hw6B.py` Submit each source code file separately using the appropriate homework submission link on the Moodle website.

The total point value for programming assignments will be awarded for solutions that are *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*) **Digital Roots**

The *digital root* of an integer can be employed as a simple *checksum-value* to verify the integrity of transmitted information. The *digital root* of an integer is found by repeatedly taking the sum of its digits until a single digit is obtained. For example, the *digital root* of 3126 is 3:

3 + 1 + 2 + 6 = 12
1 + 2 = 3

Write the definitions for 2 separate *recursive* functions as follows:

1. `digitsum(n):` that takes a single integer argument and returns the *positive* sum of its digits, irrespective of the sign of the argument.
2. `digitalroot(n):` that takes a single integer argument and returns its *digital root*. This function should call the `digitsum` function to obtain the sum of digits at each iteration.

Constraints:

- *Both* functions must be implemented using recursion. Do not use any loop constructs: e.g., `while`, `for`, etc.
- Do not import any modules, use only simple arithmetic operations.

B. (*20 points*) **Stacking Balls**[1]

Spherical objects such as balls can be stacked to form a square pyramid with a single ball sitting on top of a square of 4 balls which, in turn, sits atop of a square of 9 balls and so on. Write a recursive function named `pyramid(h)` that will take the height of a pyramid as an integer argument and return the total number of balls required to construct it. e.g., a pyramid with a height of 1 requires a single ball. If the value is negative, your function should treat it as a positive height.

Constraints:

- The pyramid function must be implemented using *recursion*. Do not use any loop constructs: e.g., `while`, `for`, etc.
- Do not import any modules, use only simple arithmetic operations.

Adapted from Eric Roberts, Chap 5: "Introduction to Recursion"