CSCI 101 Final Exam Review

The final exam will be cumulative but emphasize material from the second half of the course. All course topics are fair game; expect in-depth questions on topics since the midterm – number representation (two's-complement), circuits, architecture, lists, loops, dictionaries, sets, objects, complexity and algorithms – that build on topics from the first half of the course.

1. Review all posted notes, slides, examples, and sample solutions on the course web site. Review your own homework assignments and midterm as well as the sample midterm questions.

2. *Lists*

   a) Write a Python function named getValues() that repeatedly prompts the user for input, adding each value entered to a list. The loop should exit and return the list when the user enters 'q' (and the value 'q' should not be included the list).

   b) What output is produced by the following code?

```
def mystery(t):
    for i in range(len(t)):
        m = i
        for j in range(i+1, len(t)):
            if t[j] > t[m]:
                m = j
        t[i], t[m] = t[m], t[i]
    return t

mystery([45, 0, -7, 8, 15, 2])
```

3. *Loops and nested loops*

   a) The "Hailstone" sequence starts with any positive integer n, and defines the next value in the sequence as follows: if n is even, the next value is n/2; if n is odd, the next value is 3n+1. (The *Collatz conjecture* states that for any starting value of n, the sequence will always converge to 1. Although no counterexample is known, the conjecture has not been proven.) For example, for a starting value of n=10, the sequence is 10, 5, 16, 8, 4, 2, 1. Write a non-recursive function hailstone(n) that implements the Hailstone function and prints the sequence of values for an input n until the sequence reaches 1. For example, hailstone(10) should produce 10 5 16 8 4 2 1.

   b) Consider the following function:
   Note: recall that end='' in the print statement means that it won't automatically go to the next line after printing the symbol. Also, print() with no arguments just prints a newline character so that the next thing to be printed will be on the next line.

```
def printSquare(count, symbol):
   for i in range(count):
        for j in range(count):
            print(symbol, end='')
        print()
```

1. What would be printed out by printSquare(5,'#')?

2. Write a new version of the function called printTriangle(count, symbol), that prints out a right triangle of the same dimensions as the square. E.g., a sample run would be:

```
>>> printTriangle(5,'#')
#
##
###
####
#####
```

4. *Dictionaries*
   What output is produced by the following code?

```
d = dict()
d['summer'] = 'Sommer'
d['fall'] = 'Herbst'
for x in ['spring', 'summer', 'fall', 'winter']:
    if x in d:
        print(d[x])
    else:
        print(x)
```

5. *Objects*
   a) When writing object-oriented Python code, what does *self* refer to?
   b) What is the difference between objects and classes? What is the relationship between them?
   c) Create a class called Ball that has instance variables x, y, radius, color. Create an __init__ method as well as move() method that changes x and y by adding parameters dx and dy, respectively.

6. *Algorithms and Complexity*
   a) Give a brief description of the insertion sort algorithm. Express in Big-O notation the worst-case running time of insertion sort on n integers. Which sorting algorithm has a significantly better run-time?
   b) Suppose you have a sorted list of 2000 items. How many comparisons would you need to make (in the worst case) in order to find out whether a particular given value appears in the list or not?
   c) Suppose that an algorithm that is O(*n*) takes 15 seconds on your current computer to solve a problem in which *n* is 1,000. About how long will it take your computer to solve the problem when *n* is 3,000? Explain briefly.
   d) What is the worst case running time (expressed using Big-O notation) to sort a list of *n* integers using merge sort?

7. (a) Add the two 8-bit two's-complement binary numbers 11100111 and 00011111 together in binary, showing all your work. (b) Convert 1111111111010111 from two's complement to decimal.

8. Draw a circuit diagram that implements the XOR function (for two inputs) using only AND, OR, and NOT gates. [Recall that XOR is true when exactly one of its inputs are true, but not both.]