CISC106 Summer 2012 Lab05

- This lab an all subsequent labs will be due Sunday at 11:55 PM EDT on Sakai.
- The preparation problems below are to develop your understanding without creating extra work for you or the TA; these problems will not be graded. Be sure to read and understand them - they will help with the problems you must submit for grading
- Review the code examples from your notes in class.
- You may work with one or two other people on your lab (max size is three!). These people must be in your same lab section. If you do, **one** of you should be designated to submit the assignment on Sakai. **All of your names** should appear on code that you develop together[1].
- Whom do you think deducts more points: a happy TA, or a frustrated TA? Make your work easy to read! It isn't just good software engineering, it is good for your grade!
- EVERY python program/function must include header, doc string that contains a human-readable desciption of what the function does, and must be followed by a good series of tests, as discussed in class. Always test boundaries. Do not test erroneous input (e.g. a factorial function does not need to correctly handle strings).
- EVERY .py file must have a comment line at the very top containing your name(s), lab section, and a brief description of what the file is.
- Write the tests first! Real software engineers do this for very good reasons - so should you!
- Create lab05_tests.py (for tests) and lab05.py (for code) files. Use your lab03_tests.py as a reference for how to start lab05_tests.py.

Preparation (do not submit for grading)

1. Run IDLE and in at the prompt in the shell window enter the following two commands:

```
>>> import Tkinter
>>> Tkinter._test()
```

   A small window should pop up with two test buttons. Clicking the 'Click me' button will add brackets to its description. Clicking quit will close the window. If the window does not appear, ask the TA for help.

Problems (to be graded)

1. We've seen that Python already has an exponent operator (the `**` operator.) That being said, the exponent function is a good example of a function with a recursive definition.[2] For any base n, n^0 is simply 1. For any positive exponent m, n^m is $n \cdot n^{m-1}$.

[1]If you would like to work with someone but don't know whom, your TA may be able to help connect you to other students looking for lab partners.

[2]For natural number exponents, anyway.

Write a recursive function which calculates `base**exponent`. If `exponent` is negative or if it's a float, the behavior of the function is undefined. Feel free to write stuff like `self.assertEqual(expt(2, 10), 2**10)` in your test.

2. Write a function that takes an integer and then returns an *asterisk triangle* consisting of that many lines. An asterisk triangle is perhaps best described graphically:

```
*
**
***
****
```

   The above is a 4-line asterisk triangle

3. Now write a function which takes an integer and returns a *backwards* asterisk triangle (one which slopes downward to the left rather than to the right.) Again, a 4-line backwards asterisk triangle looks like:

```
   *
  **
 ***
****
```

4. From a box full of discs, we would like to know the probability of pulling two blue discs in a row when all the discs in the box are either red or blue. Write a function which can calculate this probability for a box filled with any number of red discs and any number of blue discs. A test case you may want to use: if the box contains 15 blue discs and 6 red discs, you have a 50% chance of drawing two blue discs in a row.

5. Now write a function that calculates the probability of drawing n blue discs in a row for some n between 0 and the number of discs in the box. (You don't have to handle cases where n is larger than the number of discs in the box.)

6. Download the file *discalculator.py* from the course website. Make sure you save it in the same folder as the rest of your lab 05 stuff. Open and run it in IDLE. Note that it tells you you've got a 0% chance no matter what numbers you enter for blue discs, red discs, and draws. Fix it so that it instead uses your function from part 5 to get the probability.

You should submit your lab05.py, lab05_tests.py, discalculator.py and any other docs required by your TA on Sakai.