CS 1110, LAB 8: RECURSION EXERCISES

http://www.cs.cornell.edu/courses/cs1110/2014sp/labs/lab08.pdf

First Name: _____ **Last Name**: _____ **NetID**: _____

This lab gives you experience with writing recursive functions. All of the functions in this lab will either be recursive functions on sequences (e.g. strings or lists), or recursive functions on integers, just as we saw in class. This is a fairly important lab; please finish it as soon as you can.

We have created several Python files for this lab. You can download all of the from the Labs section of the course web page.

http://www.cs.cornell.edu/courses/cs1110/2014sp/labs

When you are done, you should have the following two files.

- `lab08.py` (a module with stubs that must be completed)
- `test_lab08.py` (a **completed** test script).

Getting Credit for the Lab. The only file that you need to modify is `lab08.py`. The test script is provided as a convenience so that you do not have to write your own unit tests.

To successfully complete this lab, you should **implement the first four functions in the file** `lab08.py`. When you are done, show the results of executing the test module `test_lab08.py` and your implementation of `lab08.py` to your instructor.

What you don't finish during lab will be **homework** to be finished **until the beginning of lab next week**. Remember that labs are graded on effort, not correctness.

Lab Activities

In this lab, you are to implement the first four functions from the module `lab08.py`. These are the ones specified below. All implementations must be recursive (practicing recursion is the point of this lab).

```
def numberof(thelist,v):
    """Returns: number of times v occurs in thelist.
    Precondition: thelist is a list of ints, v is an int"""


def replace(thelist,a,b):
    """Returns: a COPY of thelist but with all occurrences of a replaced by b.
    Example: replace([1,2,3,1], 1, 4) = [4,2,3,4].
    Precondition: thelist is a list of ints, a and b are ints"""


def remove_dups(thelist):
```

Course authors: D. Gries, L. Lee, S. Marschner, D. Yoon, W. White

```
    """Returns: a COPY of thelist with adjacent duplicates removed.
    Example: for thelist = [1,2,2,3,3,3,4,5,1,1,1], the answer is [1,2,3,4,1]
    Precondition: thelist is a list of ints"""


def print_nested_list(input):
    """Prints out every single string in input, one per line.
    Example: print_nested_list(['this', ['is', 'a'], 'list', ['list', 'list' ]])
    should result in the following printout
      this
      is
      a
      list
      list
      list
    Precondition: input is a string, or a potentially nested non-empty list of
    strings.  no component list can be empty"""
```

Note that, to run the test module, we are using the assert functions that cannot tell what your functions are printing out, only what they return. So, to ensure that your `print_nested_list` function works well, you will want to check the printed results of the test module **visually** by comparing lines below `'> Your result :'` and `'> It should be :'`.

Even though we only ask you to work on the first four functions in module `lab08.py` in this lab, you will get greater fluency in recursion if you do them all. So, during the week, every once in a while write one of the remaining functions and test it. You should particularly try some of the recursive functions that appear later in `lab08.py`.