CS 1110, LAB 7: LISTS, DICTIONARIES, AND FOR-LOOPS

http://www.cs.cornell.edu/courses/cs1110/2014fa/labs/lab07.pdf

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

The purpose of this lab is to get you used to writing functions with for-loops. The functions in this lab are prime exam questions (though we will not ask a 2D list question like `reshape` on the first prelim). This lab involves a bit of coding, so you may have trouble finishing it during lab time. Fortunately, you have two weeks this time to finish the lab.

Lab Materials. 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/cs1110/2014fa/labs

For today's lab you will notice two files.

- `lab07.py` (a module functions for you to implement)
- `test_lab07.py` (a unit test for your module)

Once again you should create a *new* directory on your hard drive and download all of the files into that directory. Alternatively, you can get all of the files bundled in a single ZIP file called `lab07.zip` from the Labs section of the course web page.

Everything in this lab will take place in the file `lab07.py`. There is nothing to write on this sheet. You also do not need to add any more test cases to `test_lab07.py`.

Getting Credit for the Lab. You must complete the for required functions in `lab07.py` . The fifth function – `reshape` – is optional; you can skip it if you wish. When you are done, show `lab07.py` to your instructor. You instructor will then swipe your ID card to record your success. You do not need to submit the computer file anywhere.

As with previous labs, you do not need to finish during your section. However, next week is Fall Break and **there are no labs next week**. Therefore, you have until your lab the following week (October 21) to finish. However, we strongly recommend that you finish this lab before then, as all of this material (except the optional function `reshape`) is fair game for the exam.

1. For-Loops and Lists

On the next page are two function specifications. The stubs for these functions are in the file `lab07.py`. You will need to use for-loops to implement them. You should not need any list methods other than `append`. Refer to last weeks lab for useful list methods.

When you are done implementing the functions, test them with the unit test `test_lab07.py`. The test cases in the unit test are sufficient; you should not need to add anymore.

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

```
def unique(thelist):
    """Returns: The number of unique elements in the list.

    Example: unique([5, 9, 5, 7]) evaluates to 3
    Example: unique([5, 5, 1, 'a', 5, 'a']) evaluates to 3

    Precondition: thelist is a list."""
```

```
def clamp(thelist,min,max):
    """Modifies the list so that every element is between min and max.

    Any number in the list less than min is replaced with min.  Any number in the
    list greater than max is replaced with max.  Any number between min and max is
    left unchanged.

    Example: if thelist is [-1, 1, 3, 5], then clamp(thelist,0,4) changes thelist
    to have [0,1,3,4] as its contents.

    Precondition: thelist is a list of numbers.  min and max are numbers with min
    <= max"""
```

2. For-Loops and Dictionaries

Below are two function specifications. The stubs for these functions are in the file `lab07.py`. Once again, you will need to use for-loops to implement them. Both of these functions ask that you create a dictionary from the inputs.

When you are done implementing the functions, test them with the unit test `test_lab07.py`. Once you have tested them, you are done with the lab.

```
def unique(thelist):
    """Returns: a dictionary that maps each element of thelist to its position.

    Example: invert(['a', 'b', 'c']) returns 'a':0, 'b':1, 'c':2
    Example: invert([]) returns

    Precondition: thelist is a list of primitive values (e.g.  not objects or
    lists)"""
```

```
def listify(thedict):
    """Return: a copy of thedict where each numeric value is replaced by a list of
    that number and the number plus 1.

    Example: listify('a':3, 'b':5', c:1) returns 'a':[3,4], 'b':[5,6], 'c':[1,2]

    Precondition: thedict is a dictionary whose values are integers (the keys are
    all unspecified)."""
```

3. Two-Dimensional Lists (OPTIONAL)

The last function is optional. While multidimensional lists might be on the first prelim (in the short-answer section), we will not ask you to write a function on them. However, this function is good practice for the later assignments. The specification is below. Once again, the function stub is in `lab07.py`, while the test cases are in `test_lab07.py`.

```
def reshape(thelist, rows, cols):
    """Returns: A rows*cols 2D list with the contents of thelist

    Hint: In a 2D list, the element at row x and col y is the x*cols+y element
    listed.

    Example: reshape([1,2,3,4],2,2) returns [[1,2], [3,4]]
    Example: reshape([1,2,3,4,5,6],3,2) returns [[1,2], [3,4], [5,6]]
    Example: reshape([1,2,3,4,5,6],2,3) returns [[1,2,3] ,[4,5,6]]

    Precondition: thelist is a list of numbers of size rows*cols.  rows and cols
    are positive integers."""
```