CS 1110, LAB 5: SEQUENCES AND ASSERTS

Name: _____ **Net-ID**: _____

There is an online version of these instructions at

http://www.cs.cornell.edu/courses/cs1110/2012fa/labs/lab5

You may wish to use that version of the instructions.

Just when you had become an expert at string slicing, you discovered another sliceable data type: lists. However, lists are different from strings in that they are *mutable*. Not only can we slice a list, but we can also change its contents. The purpose of the lab is to introduce you to these new features, and demonstrate just how powerful the list type can be.

As with last week, we realize that you have a lot on your plate. You have the first stage of the color models assignment coming up, as well as the first exam. So our goal in this lab is to make it as short as possible while still making sure that you are adequately prepared for the exam.

Requirements For This Lab. There are no files to download for this lab. For the most of the time you will be playing with the Python interactive prompt. We do ask you to implement some functions below. However, just want you to write the implementation on a piece of paper (which you will show to the instructor). You do not need to submit any modules, and you do not need to write any unit tests.

As always, you should try to finish the lab during your section. However, if you do not finish during section, you have **until the beginning of lab next week to finish it**. You should always do your best to finish during lab hours; remember that labs are graded on effort, not correctness.

---

1. List Expressions and Commands

For this part of the lab, you will work in the Python interactive prompt. Create the following list:

```
lablist = ['H','e','l','l','o',' ','W','o','r','l','d','!']
```

Like a string, this is a list of individual characters. Unlike a string, however, the contents of this list can be changed.

Enter the following statements and/or commands **in the order they are presented**. When you type in expressions, Python will immediately display the value; the commands below are all followed by a print statement showing the new contents of the list. Each case, describe what you see and *explain the result*

Command or Expression	Result/Explanation
`lablist.remove('o')` `print lablist`	
`lablist.remove('x')`	
`lablist.index('W')`	
`lablist.index('B')`	
`lablist[0] = 'J'` `print lablist`	
`lablist.insert(5,'o')` `print lablist`	
`s = lablist[:]` `print s`	

`s[0] = 'C'`
`print s`
`print lablist`	

2. List Functions

On the nest page are two function specifications; implement them. You will probably want to test them out on the computer. However, to turn them in you just need to write the final version on a piece of paper and show it.

For each of these functions, you might find the following list methods useful.

Method	Result When Called
l.index(c)	**Returns**: the first position of c in list l; error if not there
l.count(c)	**Returns**: the number of times that c appears in the list l.
l.sort()	Rearrange the elements of the list l in ascending order. This alters the list; it does not make a new list

Function `swap(thelist,a,b)`.

```
def swap(thelist,a,b)
    """Swap the elements at positions a and b in thelist.

    Precondition:  thelist is a list; a, b are valid positions in
    thelist."""
```

Function `lesser_than(thelist,value)`.

The function below **should not alter** `thelist`. If you need to call a method that might alter the contents of thelist, you should make a copy of it first.

```
def lesser_than(thelist,value)
    """Returns number of elements in thelist strictly lesser
    than value, without altering thelist.

    Example:  lesser_than([5, 9, 1, 7], 7) evaluates to 2

    Precondition:  thelist is a list of ints; value is an int, and
    value is contained in thelist"""
```

3. Asserts

The function `swap(thelist,a,b)` requires that `a` and `b` be valid positions in the list. Write an assert statement that verifies this is indeed the case (and produces an error message if not).

Suppose that you have added this assert statement to your function `swap(thelist,a,b)`. Now consider the following procedure:

```
def try_swap(thelist)
    """Attempt to swap the first two elements of thelist.

    Precondition:  thelist is a list."""
    try:
        swap(thelist,0,1) print 'The list is altered:  '+`thelist`
    except:
        print 'The list is unchanged:  '+`thelist`
    print 'The procedure try_swap is done'
```

Suppose you perform the function call `try_swap([0,1,2])`. What is the printed output of this procedure?

On the other hand, suppose you perform the function call `try_swap([0])`. What is the printed output of this procedure?

When you have answered this question, show your work to your instructor. You are done.