CS 1110, LAB 4: ASSIGNMENT 1

http://www.cs.cornell.edu/courses/cs1110/2015fa/labs/lab04.pdf

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

Today's lab is an open office hour to work on Assignment 1. Take advantage of it to get whatever last minute help that you might need. If you have finished Assignment 1, we have an optional exercise for you to work on (see below). However, that **is not required**.

Since many students often wait to the last minute, the consultants will be fairly busy during this lab. If you feel that you are being ignored (e.g. you have held your hand up for several minutes with no one recognizing you), approach one of the course staff and ask about "getting a place in line".

Getting Credit for the Lab. Because you are working on the assignment, you will receive full credit for this lab if you turn in the assignment on time (e.g. Thursday before midnight). There is nothing else to show to you instructor. You do not even need to swipe your card this time.

If you do decide to complete the optional exercise, you may show that to your instructor.

Window Exercises (OPTIONAL)

As we said in class, a type is a collection of *values* together with its operations. Normally when we think of values, we think of numbers, or `True` and `False`. But these values could take on any form. A value could be a window on your screen. That is the motivation for the type `Window`.

Windows are not built-into Python. You need to import a module in order to use them. The module that provides window support is called `tkturtle` (the reason for this name will become clear in Assignment 4). Start Python and import this module now.

```
import tkturtle
```

The basic types all have literals to represent their values. We represent ints by whole numbers; we represent strings as characters inside double quotes. But there are no literals for a `Window`. For this type, we have to use a special functon – called a constructor – to create a new Window object.

The name of a constructor function is generally the same name as the type. The type `Window` is provided by the module `tkturtle`. Enter the following assignment statement into Python.

```
w = tkturtle.Window()
```

What happens? What is stored in the varible `w`?

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

Window Attributes. As we discussed in class, attributes are named variables that are stored inside an object. You can use attributes in expressions, or even assignment statements.

One of the interesting thing about GUI objects is that assigning new values to an attribute can have visible effects. In the table below we have a list of expressions and assignment statements. **Enter these into the Python shell in exactly the order presented**. If it is an expression, give (or guess) the value that Python returns. If it is an assignment statement explain (or guess) the result of the assignment.

Statement or Expression	Expected Result	Actual Result	Reason for Actual Result
`w.x`			
`w.x = 100`			
`w.y`			
`w.y = 100`			
`w.width = 10`			
`w.height`			
`w.title`			
`w.title = 'window'`			

Window Methods. Window objects also have methods. Unlike string methods, which are functions, these methods are procedures that do something to the object. Execute calls for the three methods shown in the table below. Explain what happens when you call them.

Method	Result When Called
`w.beep()`	
`w.iconify()`	
`w.deiconify()`	

Repositioning a Window. You have already seen the Window size and position is controlled by attributes. For the Window whose name is in `w`, look at the attributes for the x-coordinate and y-coordinate. Write their values here:

Next, create a second Window object, storing its name in another variable. This should pop up a new window. What are the x-coordinate and y-coordinate for this window?

Is there something unusual about how screen coordinates work? What do you notice about the difference in coordinates between the two windows?

Resizing a Window. Create a new Window, storing its name in variable `w`. Try resizing the Window with your mouse to make it bigger. Look at the attribute function `resizable` of `w` to see whether the Window is resizable. What is the value of this attribute?

Now execute the assignment

```
w.resizable = False
```

Try resizing the Window whose name is in `w` with your mouse. Is it resizeable now?

Assign the attribute `resizable` to True. Once you have done that, call the procedure

```
w.setMaxSize(50,100)
```

What happened to your Window?

What happens when you try to resize this Window?