**Python Lab 8a Ball Motion**

A. At the top of your program, make variables x, y, and radius. Give them values. Draw a circle using x, y, and radius.

B. At the top of your program, make variables deltaX and deltaY, which stand for the change in X and the change in Y. Set deltaX to 1.5 and set deltaY to 3.5.

In the draw handler, add these lines:

x = x + deltaX

y = y + deltaY

You should see your circle move in a line, and eventually go off the screen.

C. In the draw handler, add an if statement that checks to see if the ball is touching the top or the bottom of your frame:

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_or \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ :

deltaY = -1 \* deltaY \*

Test to see if your ball bounces off the top and the bottom of your screen. It should bounce when the edge of the ball touches the edge of your frame.

D. Make your ball bounce off the left and right sides of your screen.

E. At the top of your program, set deltaX to be a random integer between 1 and 6.

Set deltaY to be a random integer between 1 and 6.

Set the radius to be a random number between 20 and 50.

Set your frame width to be a random integer between 400 and 500.

Set your frame height to be a random integer between 300 and 400.

Your ball should still bounce off all the walls.

F. Challenge. Make a rectangle that acts as a paddle on one side of your frame. Use key commands to move the paddle. If the ball touches the paddle, change the direction of the ball just like you did for frame walls.

G. Challenge. Add a second paddle in the middle of your frame, with different key commands, and make it move the ball going off in any direction.

**Python Lab 8b Lists**

A. Python Lists have a number of built-in functions. Take a look at the Code Skulptor documentation.

What list function allows you to:

Create an empty list mylist = \_\_\_\_\_\_\_\_\_\_

Put the items in an order in the list mylist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Get rid of an item at a certain number mylist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Add an item onto the end of the list mylist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Add an item at a certain place in the list mylist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Get rid of a certain item by name mylist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

B. Open up the CodeSkulptor program <http://www.codeskulptor.org/#examples-list_methods.py>

Run the program a few times to see how it works. Look at the code carefully to see what each part does.

Make the following changes:

1. Make a new BUTTON called reverse. It reverses the items in your list. Test this.

2. Make a new BUTTON called sort. It sorts the items in your list. Test this.

3. Make a new BUTTON called FirstToLast. It removes the first thing in your list and adds it to the end. Test this well.

4. Fix the code for the “new task” input box handler, so that it only adds a new task if that task is NOT currently in the list. (see the code in remove\_name for a hint)

5. Make a newTEXT INPUT BOX called Move Number to Front. It takes item # that you chose, and moves it up to the front. It does not duplicate the item. If the number is out of range, it does nothing.

C. Challenge Tasks

Make a second list, called do\_later. This list starts out empty. When your tasks option has five things in it, and you try to add a new thing, it gets added to do\_later. When the original list removes something, the first item in do\_later is removed from do\_later and added to tasks

Print do\_later to the console every time a change is made to do\_later. Use the same numbering system and formatting that was used to draw the tasks to the screen.

**Python Lab 8c List of Lists**

A. It is possible to have a list that stores lists. Enter the following code into a new program:

people = [ ]

people.append([‘Stevie’, 9])

people.append([‘Mia’, 11])

print people

Add three more people to your list. You must keep the format the same of [‘name’, age]

B. Enter in the following code. The variable p stands for one element of the list. The for loop iterates through each element of the list.

for p in people:

print (p)

Turn this code into its own function and call the function at the bottom of your program.

C. Make a function that prints out only the names. Call this function from the bottom of your program.

D. Make a function that prints out only the ages. Call this function from the bottom of your program.

E Make a function that adds up the ages and prints out the average. Call this function…...

F. Use the “raw\_input” command to ask the user to enter a name. Then use it again to ask the user to enter age and convert the age to an int. Put these two things into a list called newperson.

G. Make a button called “add a student.” Make text boxes for the name and the age. This button will make a new entry for the name and age in the text boxes, and will also print out the list of people.

Challenge Tasks:

H. Make a button called “remove student.” It takes the name that was entered in the text box, searches the list for that name, and then removes that student and all their information. Test this on students at the front, back, and middle of your list.

I. Make a button that sorts the list by age.

J. Make a button that sorts the list by name.

**Python Lab 8d A List of Circles**

A. Make a new program. Add this code near the top of your program:

**circles = []**

**position = [10,20]**

**color = ‘orange’**

**newcircle = [ position, color]**

**circles.append( newcircle )**

**print (circles)**

When you run your program, you should see the list print out to the console.

B. Change your draw handler so that it draws the circle in your list:

**def draw(canvas):**

**for c in circles:**

**pos = c [\_\_] # fill in the blank**

**color = c [\_\_] # fill in the blank**

**canvas.draw\_circle(pos, 10, 1, color, color)**

C. Look up in the Docs how to make a mouse\_click handler. In the code for the mouse click handler, add this:

color = ‘orange’

newcircle = [ position, color] # position is the parameter of your handler

circles.append( newcircle )

print (circles)

Now, every time you click, a new orange circle should appear.

D. Change the code for your button so that it gets rid of all the circles and leaves you with an empty list.

E. Add code to the mouse\_click handler to set the color to one of seven random color choices.

You can pick the seven choices, but it cannot be the same color as the background.

F. Make a button that keeps every third item in your list. (keep, remove, remove, keep, remove, remove,…...)

Challenge Tasks:

G. When a circle is made, give it a different random velocity (in both the x and y directions) and make them all move as you did in Day 12. Each circle will move in a linear fashion.

H. Make all the circles bounce off the walls.

I. Make all the circles bounce off each other.