By: Robbie Selwyn
ALL CODE is in *chapter4.py*
Each problem is done in a function named “problem<problem #>()”
2. *The problem was to draw an archery target using circle objects from the graphics library.* There were no inputs for this problem. To solve this problem, I drew each circle from the largest one first, to the smallest one last, all with a center at the exact center of the window. I made it such that each concentric circle had a radius that decreased 20px from the next larger circle. The output was a fully drawn target on a graphics window.
3. *Draw a face in the graphics window.* There are no inputs for this problem. To solve this problem, I thought of what made up a face. I know that a face has eyes, a nose, and a mouth. With what I wanted to draw in my mind, I wrote code using the graphics library to display what I wanted. The output was a face in the graphics window.
5. *Draw a set of 5 dice depicting a straight (give values 1,2,3,4,5 in a row on the dice faces).* There are no inputs to this problem. I broke this problem up into 2 parts to get to the solution. Part 1 was drawing all the dice outlines (no dots yet). I came up with a for loop that draw squares that are spaced apart. I wrote the code to space apart each of the dice so they aren’t conjoined. The second part was creating the dots on the dice. I really couldn’t think of a better, more efficient, way of doing this without just drawing each dice out. I did, however, calculate the appropriate positions of the dots so they are spaced out. The output was a set of five dice depicting a straight, all drawn on a graphics screen.
6. *Modify the graphics future value program to get the principal and apr from Entry graphics objects.* There are 2 inputs, both through the graphics window. The first input is the principal and the second is the apr. Solving this problem wasn’t very hard, especially because most of the code was given to us from the book. To get the principal and apr, I created graphical entry objects and a submit rectangle object. The submit button will work if you click anywhere on the screen. After testing my code, I realized if someone submitted a blank entry, my program would throw an **“Unexpected EOF when parsing”** error. To solve this, I added a try-except statement to my program. This statement will automatically close the program if it is going to encounter the error, and prompts the user to restart the program. After someone submits the values, I pass the values to the code from the book which displays the graph. The output is the graph of the future value of an input.
8 (extra credit). *Create a program that draws a line with mouse clicks and draws the midpoint. Your program should output the length of the line and slope.* The two inputs are mouse clicks. To solve this, I converted the mouse clicks to point objects and then drew the line through the points. To calculate the midpoint, slope, and distance, I plugged the click values into formulas. The output was a line with a marked midpoint, and text showing the length of the line and slope.
9. *Create a program that takes in two clicks and draws a rectangle with the corners at the clicks. Output the area and perimeter of the rectangle.* The inputs were two mouse clicks. To complete this problem, I first converted the mouse clicks to Point objects. Having them as point objects lets me easily pass them on to the Rectangle object to create the rectangle. One I created the rectangle, I ran the points through a series of formulas to calculate the area and perimeter of the rectangle. Then I output the information to the page. The full output is a

rectangle drawn with corners at the mouse click points, and information on the perimeter and area of the rectangle in text.