ENGR 102 - Lab Assignment #11

Spring 2019

General course learning outcomes:

- demonstrate the construction of computer programs, including techniques to declare and use functions to solve computing-related problems.
- apply programming techniques to solve problems in engineering.
- complete a team programming assignment that ties together concepts learned in the class.

Activity 1: Again - to do in lab (team)

☑ *Create and use functions in Python*

Write a function and have the main program 'test' the function. For example, you might include several function calls and display the results, or create a program where a user can enter values and see the results.

Write a function that takes in a list and returns the minimum, mean and maximum values from the list. You may use the min, mean, and/or max built-in functions to do so.

Activity 2: Keanu Reeves Starring in "Average Velocity"

☑ *Use basic engineering equations and write a Python program to perform the calculations.*
☑ *Create and use functions in Python*

Write a function and have the main program 'test' the function. For example, you might include several function calls and display the results, or create a program where a user can enter values and see the results.

Write a function that takes in two parallel lists: a list of times (in increasing order), and a list of distance traveled by that point in time. The function should return a new list giving the average velocity between consecutive time measurements. *(Note: the new list should have length one less than the original lists.)*

Activity 3: Bravery!, Humility!, Anger-y!- to do in lab (team)

☑ *Create and use functions in Python*

Our Angry Bird friends need your help—again. This time, mysterious random piggy points are appearing on various planets, and Red and friends are the only ones who can catapult themselves into heroic action.

The main program has been created already (provided on the following page). It's your team's job to create the functions that will make it work for Red, Chuck, Bomb and Terence.

- When users are asked to pick items, provide nicely formatted informational menus. Provide at least the following options:

  Birds: Red = red, small bird; Chuck = yellow, small bird; Bomb = black, large bird; Terence = red, large bird

  Planets: Earth = 9.807 m/s^2, Mars = 3.711 m/s^2, Moon = 1.625 m/s^2, Jupiter = 24.79 m/s^2

- When plotting the trajectory, the target should be a menacing bright green pig-like circle, and the trajectory should be a dotted line representing the color and size of the bird thrown. If the bird hits the target, a large red X should mark the successful encounter.
- Make your program smart enough to account for the location and size of the target when determining a 'hit'. Note that the porcine target may appear in many possible places—in the sky or on the ground—but it will stay in the same place until hit. The target size is its diameter.
- You must use all of the provided lines of code, exactly as they appear.

(continued, next page)

```
# Standard Header Here

# ---------------------------------------- IMPORTS STATEMENTS----------------------------------------
# (Put the necessary import statements first)

# ---------------------------------------- FUNCTION DEFINITIONS----------------------------------------
# (Then put any function definitions)

def get_basics():
   """Takes user selections for active bird and planet. Returns (bird, planet). 'Bird' includes name,
      color and size. 'Planet' includes name and gravity. """
   a = bird_picker()                                  # Runs fn to provide bird menu
   b = planet_picker()                                # Runs fn to provide planet menu
   return a, b


def trajectory_y(x, g, vo, angle):
   """Returns (y-value) of the trajectory for a given x-value, gravity, initial velocity, and angle."""
   angle = radians(angle)
   return (x*tan(angle))-(g*x**2)/(2*(vo**2)*cos(angle)**2)
```

Create the necessary functions here

```
# ---------------------------------------- MAIN PROGRAM ----------------------------------------
# (Then your Main Program)

# Sets up loop so user can repeat the game as many times as desired ('y' to continue, 'n' to quit)
pig_counter = 0
again = 'y'
while again == 'y':

   # Program will pick a random distance (x from 10-1000), height (y from 0-50) and size of a target
   target = (random.randint(10, 1000), random.randint(0, 50), random.randint(10, 50))

   # Takes initial guesses
   bird, g = get_basics()                             # Runs fn to get bird and planet information
   v_guess, theta_guess = get_guesses()               # Runs fn to get initial velocity and angle guesses

   # Loops guesses until bird hits target
   x, y = trajectory(g, v_guess, theta_guess)         # Create current x- and y- value lists
   while not hit(x, y, target):                       # Program cycles until throw hits the target
       birds_plot(x, y, target, bird)                 # Plots trajectory & target of miss
       v_guess, theta_guess = get_guesses()           # Gets updated guesses from user
       x, y = trajectory(g, v_guess, theta_guess)     # Creates updated lists of x- and y-values

   # Handles winning case and asks if user would like to play again
   print('Yay!')
   pig_counter += 1
   birds_plot(x, y, target, bird, True)
   again = input('Would you like to play again? (y/n)')
   while again not in {'y', 'n'}:
       again = input('Please type either y or n only. Would you like to play again? (y/n)')

# Exiting when user decides to quit
print('\nThanks for playing! You popped %d pig(s) today!' % pig_counter)
```