CIT 590 Homework 1 – Lunar Lander

Purpose of this assignment

- Get started programming in Python
- Get started using IDLE (Integrated Development Environment)

General Idea of the Assignment

Lunar Lander is one of the earliest computer games. With a proper choice of initial values, it is fairly interesting to play, even though it is a text-only program.

You will take the role of an astronaut in the lunar module attempting to land on the moon's surface. Gravity pulls you towards the moon at an increasing rate of speed. In order to land safely, you must burn fuel to counter gravity's acceleration to land on the moon at a safe speed (below 10 m/s). However, be careful, you only can use so much fuel. And if you are too high when you run out of fuel, you'll inevitably crash at an unsafe speed.

The Math

This game is not timed. Rather, you will "simulate" the passage of time by one second after each number entered. The game will play out as follows:

1) The player will be given their current **altitude**, **velocity**, and **fuel**.
2) The player will specify how much fuel to burn in the next second.
3) The game will perform the calculations of how the player's actions change **altitude**, **velocity**, and **fuel** over the next "second".
4) If still above the moon's surface, go to step 1. Else, end the game (safely or otherwise…)

These steps will be inside of the loop that runs *while* your **altitude** is positive. Note that it is unlikely you will hit zero altitude exactly. Instead, use **altitude** being less than or equal to zero as your termination condition.

Note that the game should set an initial **altitude**, **velocity**, and **fuel** amount. **Altitude** = 1000.0 meters, velocity = 0.0 meters/second, and fuel = 1000.0 liters can lead to interesting games.

To break down step 3, assume the following

- Each turn, your **velocity** *increases* by 1.6 meters per second due to the force of gravity of the moon (assume altitude doesn't affect the force generated by gravity).
- Additionally, your **velocity** *decreases* by an amount proportional to the fuel you burn. This means you multiply **fuel** burnt times some constant to get the velocity change. Using a constant of 0.15 makes the game fairly easy to win.
- Your **altitude** decreases by your velocity (since you calculate every second, and your velocity is in meters per second, you don't have to do a multiplication step for units of time)

- Your **fuel** decreases by the amount specified by the programmer.

Note that it is possible that a player tries to “cheat” by specifying an illegal amount of fuel to burn. We want to reasonably prevent these errors as follows:

- If a player specifies burning zero fuel, this is fine (in fact, if they are out of fuel, this is their only choice).
- If a player tries to burn a negative amount of fuel, treat it as if they burnt zero fuel.
- If a player specifies to burn more fuel than they have, burn all their fuel.

The end game should specify:

- whether the landing was safe or not, (a safe landing is one where the final velocity is < 10m/s)
- at what velocity the landing occurred,
- how many seconds the landing took, and
- how much fuel is left.

Additionally, the program should ask if the player wants to play again. Any response that begins with ‘y’ (capital or lowercase) should play again. Any response that begins with ‘n’ (capital or lowercase) means the user wants to exit. Any response that begins with any other character should ask the player again.

Due Date + Submission.

This assignment will be due Tuesday, September 12, by 11:59 p.m. Submit the assignment through Canvas. Your submission should include:

1) lunar.py - the source code for your game
   This file must include a header that contains
   a) Your name
   b) Your Penn ID
   c) Either:
      i) A list of resources you used and/or people you got help from (including TAs/Instructor)
      ii) A statement you worked alone without help
2) readme.txt - a text file that explains
   a) How to run your code
   b) A list of inputs that leads to a “win”
   c) A list of inputs that leads to a “loss”

(Note, a new homework will come out Monday, September 11, which will be due the following Tuesday. You can expect to turn in an assignment once a week from there forward).