Python for Rapid Engineering Solutions, HW1

There are three parts to this homework. Make sure you do all three! The objective of this homework set is to get you started writing Python code.

Part 1

Newman, exercise 2.1 (20 points). This assignment is on page 30 in my edition. Note the following changes from what is in the text.

1. Use 9.81 m/s**2 as the acceleration due to gravity.

2. Your output should be EXACTLY in the following format where x.xx is a floating point number with two digits past the decimal place:
time to hit ground x.xx seconds

3. Rather than dropping the ball from a height of 100m, use the "input" command to read a value from the terminal. The initial height must be 10m $\leq v \leq$ 1000m. If a height outside that range is specified, an error message should be printed and the program should exit. The error message should state, EXACTLY, "Bad height specified. Please try again." Use EXACTLY this string within the input command:
"enter height: "

4. Rather than just dropping the ball, allow another "input" command to request an initial vertical velocity. A positive initial velocity is upward, a negative initial velocity is downward. Limit the initial velocity to -20m/s $\leq v \leq$ 20m/s. If the initial velocity is outside that range, print an error message and exit. The error message should be exactly: "Initial velocity is too large! Slow down!" To obtain the initial velocity, use EXACTLY this string within the input command:
"enter initial upward velocity: "

NOTE: only request the velocity if the height was entered successfully.

5. **Your program MUST be called hw1_1.py.**

Why are we specifying these formats? So we can automate checking and so you can run many different heights to verify you've got it right. To assist you, there are some files listed on the HW1 page that will help you verify that you have the format correct.

Run your program as you would normally. Then, when you are ready, execute this on a command line:

> python hw1_1.py < hw1_1_t0.in > hw1_1.out

Then, compare hw1_1.out with hw1_1_t0.out. If they are the same, you're good for that test. If they are different, you'll need to figure out why. Similarly, you can run:

> python hw1_1.py < hw1_1_t1.in > hw1_1.out

Then compare hw1_1.out with hw1_1_t1.out.

You only need to turn in hw1_1.py for this part of the assignment. We will test your code with various heights and velocities.

Why force you to use the input function instead of just using 100m? Because known answer tests make debugging a lot easier. So, we'll make this first, very simple program, one where you get to exercise that skill. Plus, you'll get to see how we'll be grading your work.

Part 2

The objective of this homework set is to get you started writing Python code. Newman, exercise 2.12 (30 points), on page 83 in my edition. For output, ONLY print the list of primes. That is, your output code should look something like:

print(my_prime_list)

No, you don't have to use that variable name, but do use one that has some meaning. The only output from your program should be the list, and it'll look something like:

[2, 3, 5, ...]

Where the ... represents all the rest of the list of primes that you've found.

Your program MUST be called hw1_12.py and only the program need be turned in for this part of HW1.

Part 3

The objective of this homework set is to get you started writing Python code. Newman, exercise 2.13 (50 points) in Newman, pages 83-84 in my edition. (Make sure you notice the problem continues onto page 84!)

We're going to modify the assignment a bit.

1. Include the factorial function shown in the first part of the problem so you confirm that you've got the recursion thing down.

2. Instead of fixed values, we'll again have you read values in for the 3 parts of the problem. Use the following strings to get data from the user in this order:

"Enter an integer for a factorial computation: "
"Enter an integer for a Catalan number computation: "

"Enter the first of two integers for a GCD calculation: "
"Enter the second integer for the GCD calculation: "

And format your output so it looks like this if you entered 5, 20, 32, and 40:

factorial of 5 is 120
catalan value of 20 is 6564120420.0
greatest common divisor of 32 and 40 is 8

Make sure you print a blank line, print(""), before you print the output.

As in HW1 Part 1, on the HW1 page are a pair of test files you can use to verify that things are working properly.

At the command prompt:

> python hw1_13.py < hw1_13_t0.in > hw1_13.out

and then compare hw1_13.out with hw1_13_t0.out. They should be the same.

You just need to turn in hw1_13.py for this part of HW1.

Your program must be called hw1_13.py.