CS 121 – Lab #1 – Simple programs

Today, we will practice the problem-solving procedure with some simple exercises. Along the way, you will begin to use Spyder, the system that we will use in the course to compose and run Python programs.

A simple program consists of the following parts:

- A comment describing the purpose of the program.
- Statement(s) to perform input
- Statement(s) to perform calculations
- Statement(s) to perform output

Exercise #1

Let's start with an easy problem: finding the perimeter and area of a rectangle. We need a program that will ask the user to enter the length and width of a rectangle, and then print out its perimeter and area.

Step 1: Do we understand the requirements of the problem? Yes. But to clarify the problem, let's assume that the user will enter real numbers for the length and width, rather than integers. This means that all of the numbers in this program will be real numbers (i.e. floats).

Step 2: Next, let's design the program in English. We anticipate that we will eventually need variables for length, width, perimeter and area.

- The program will ask the user to enter the length.
- The program will ask the user to enter the width.
- We write the formula for perimeter, which is 2 times the sum of the length and width. The multiplication symbol is *, so we can write: perimeter = 2 * (length + width).
- We write the formula for area = length * width.
- Finally, we print the answers. We can do so using one or two print statements. It depends on whether we wish to print one or two sentences.

Step 3: Now that we know how to solve the problem, it's time to express it in the Python language. It should look something like this.

```
# rectangle.py – Calculate the perimeter and area of a rectangle.
# We ask the user for the length and width interactively.

# input
length = float(input("Please enter the rectangle's length:  "))
width = float(input("Please enter the width:  "))

# calculations
perimeter = 2 * (length + width)
area = length * width
```

```
# output
print ("The area of the rectangle is ", area)
print ("And the perimeter is ", perimeter)
```

Question: Why does the formula for perimeter require parentheses? In other words, why would it be wrong to say `perimeter = 2 * length + width`?

Now that you are ready to type in the program, we need to start the Spyder system. The precise way to invoke Spyder may vary depending on your computer. Here is how it can be done on a Mac: in the upper right corner of the screen is the "finder" magnifying glass icon. Click on this. A dialog box appears in the middle of the screen. Begin typing the word Anaconda… you will soon notice that the system will recognize the name of the Anaconda program. Ask the system to open Anaconda.

When Anaconda opens, you will see a short menu of programs. Select Spyder. When Spyder starts, allow it to take up the entire screen.

Most of the Spyder window is divided into two major parts. There is the *editor* on the left side, and the *shell* on the right side.

- As you might imagine, the purpose of the editor is for you to create and type in entire programs.

- The main purpose of the shell is to allow you to interact with a program when it runs. The shell is where you can see output. The shell is also where you type in your input while the program is running. The shell has another purpose – you can use the shell to experiment with individual Python statements. The shell can be used as a calculator. You can type in mathematical expressions or Python statements, and you can get immediate answers.

Using the editor (left side) of the Spyder window, create a new file called `rectangle.py`. Type in the program that you saw above. After typing the program, save the file onto the desktop or a USB drive.

Steps 4 and 5: The last two steps of the problem-solving procedure are to compile and run the program. However, in practice we can do these two steps at the same time when we ask Spyder to run your program. It turns out that Python is an *interpreted* language, so that it does not need a compiled form. The Spyder system can translate and execute your program at the same time.

There are a few ways to ask Spyder to run your program. The easiest way is the hit the F5 key. Watch the shell (right side of the window) for the output or any error messages that occur.

Does the program work as you expected? Feel free to ask the instructor or lab aide if you need help.

Exercise #2

Next, experiment with errors. Let's see how Spyder behaves when the program is incorrect. For example, remove a parenthesis symbol from one of your input statements. Or, delete an equals sign. What does Spyder say when you ask it to run the program?

Re-edit your program so that it works correctly. Now, we will experiment with another kind of error! Change your area calculation as follows:

```
area = length + width
```

Save and run the program. Does Spyder give you an error message? How does the mistake in the program manifest itself?

Exercise #3

Now that you have gone through the problem-solving procedure once, it is time to try it again on a new problem. For this exercise, we want to ask the user for the lengths of the 2 "legs" of a right triangle. The program will then determine and output the hypotenuse and area.

Hint: The formula for hypotenuse requires us to perform a square root. The easiest way to calculate a square root in Python is to raise a number to the 0.5 power. For example, `n ** 0.5` means the square root of n.

Write your English design here:

When you are ready to type in the program, create a source file called `triangle.py`.

Exercise #4

Practice makes perfect! Let's do another problem from scratch. We will calculate the cost of painting a room. Ask the user for the length, width and height of the room. Assume that the 4 walls are to be painted, but not the floor or ceiling. Also assume that the paint will cost us 10 cents per square foot. Tell the user how much it will cost to paint the room. Don't forget to include a dollar sign in your output.

Write your English design here:

When you are ready to type in the program, give it the name `room.py`.

When you are done with the triangle and room programs, please show them to the instructor or lab aide. We will ask you to run each program.

Before you go, you should save and keep a copy of all the programs you wrote today. For example, save them onto a USB drive that you can take home, or e-mail them to yourself.

You're done with your first lab!