**ENGR 102**

**Lab # 3A**

**Activity #1: To do in lab – as a team [30 Points]**

This activity is meant to help illustrate the process of asking a user for input and then reading their input, performing processing, and outputting a result. Many programs will follow that basic format.

As a team, write the following short programs. For each one, you are to write a program that converts from one type of unit to another. You should ask the user for input in one unit, convert it to the other, and output the answer in the converted units. Do so for each of the following conversions. For each, write a separate program (labeled 1a – 1h) and turn in all files together.

1. Pounds to Newtons [3 Points]
2. BTUs to Joules [3 Points]
3. Pascals to Millimeters of Mercury [3 Points]
4. Seconds per revolution to Hertz [3 Points]
5. Miles per Hour to Meters per Second [3 Points]
6. Fahrenheit to Celsius [3 Points]
7. Voltage to voltage level (power gain) measured in Decibel Volts (dBV) – for a reference of 1 volt [6 Points]
8. Differences in two Richter scale values to the ratio of energy released in two earthquakes [6 Points]

This sort of activity (unit conversions) might seem very basic, but it’s also critically important. Conversion (a) above has a particularly costly history. Before your team leaves today, please search and read a little bit about the Mars Climate Orbiter, and read the two statements from NASA:

* <https://mars.nasa.gov/msp98/news/mco990930.html>
* <https://mars.nasa.gov/msp98/news/mco991110.html>

Notice from the statements both the root cause of the failure, and the contributing factors.

**Activity #2: To do in lab – as a team [35 Points]**

This activity is meant to give your team a chance to write a larger program, together. You are to write a program that will read in four people’s names and birthdays, and will output them in a formatted list. The final list should include people’s names and their birthdates in two formatted columns.

1. Begin by pairing up – you should have 2 teams of 2 people, and each pair should do the following tasks individually. Do not talk to the other pair until the point at which you are to work together.
2. First, in a separate document (a text file, Word file, etc.) make a list of the variables that your team will use in this program. Your list should include:
   1. The variable names
   2. The type of each variable
   3. A *very brief* description of what each variable is (you can write one description for multiple variables if it is clear what they all are)
3. Next, writing in the separate document, determine what instructions you want to give to the users. You should determine the text you want to tell the users
   1. Be sure to be specific about the way you want them to enter information!
4. Third, writing in the separate document, determine how you want the output to be formatted. Consider how you want to align the various columns of the document. Note that all information should be lined up in clear columns as much as possible
5. Fourth, **and only after completing the above**, write your code as a pair. The two of you should write a Python program that asks four users for the data described, fills the appropriate variables, and outputs the result in the format you described.
   1. Your pair can (and should) test your program to make sure it is working as expected.
6. Now, once both pairs have written your code, each pair should try using the other pair’s program:
   1. The first pair should run their program, letting the second pair enter their information (names and birthdates) first, then the first pair should enter their information, and see the result.
   2. Likewise, the second pair should run their program, letting the first pair enter their information first.
7. Finally, all four individuals should look at the code of both programs. The four of you should:
   1. Create a single merged document (merging your two individual documents made in parts (b)-(d), above).
   2. Create a short summary (around ½ page, and no more than 1 page) in that document discussing the differences and similarities of your programs. In particular, focus on the following:
      1. Were the names you used for the variables similar or different? Did both pairs have the same sets of variables (but with different names)?
      2. To represent birthdays, did you use the same variable types? If both pairs had the same format (same number and types of variables) for representing birthdays, then, as a team, list one other alternative that you might have used.
      3. Did both pairs expect data to be formatted the same way – for example, was the formatting for the date the same in both pairs, and were the results printed in columns the same way?
      4. Were the instructions for how to format the input data clear? Think about different ways you might have asked for the birthday information to be entered.
8. Turn in the document, as well as each team’s program (name the programs differently)

Note: the purpose in this assignment was, in addition to giving you practice for reading input and formatting output, to see that there can be different ways of constructing programs, and different ways of assuming input might be formatted. In most programming, there is not a single solution to a problem, and the programmer’s choices for variables, information provided to the user, and output can vary, and still be correct.

**Activity #3: To do in lab – as a team and as individuals [35 Points]**  
The purpose of this activity is to practice working with strings.

First, building on what we learned about escape characters in strings (the \’ and \” options), there are some other “escape sequences” in Python, all beginning with a \ character. Try experimenting by printing out strings containing the following escape characters, to see what they do:

\n, \t, \\, \b. If you cannot deduce what these do, feel free to look them up in an online Python guide.

Make sure that *each member* of your team understands how each of those characters work.

You are each going to write a short program to generate mad-libs. A mad-lib is a short story in which a person is asked for some general terms (e.g. “Number from 10 to 18” and “Name of a river”), and then that term is inserted into some story. For instance, if the person entered “15” and “Brazos”, then the story might read: “I was 15 years old when I decided to sail a raft down the Brazos River.”

1. As a team, decide on at least 5 and no more than 10 different inputs that you will ask a user for. Agree on these as a team (e.g. they can be things like “A person’s name”, “The name of a city”, “A color”, “A word ending in -ing”.
   1. At least one of these inputs should be a numerical value.
2. Then, each member should individually write a mad-lib program that forms a story using those inputs. The story should:
   1. Perform at least one computation on the numerical input. For example, you might ask for an age in years, and convert it to months.
   2. Generate several sentences in a story that incorporates all of the input the user typed in.
   3. Present the story in a nicely formatted manner. The lines of text should be of reasonable length and flow into each other, without having single words on a line (unless at the end of a paragraph), etc.
   4. Your text should make use of the escape characters: \n, \t, and either \’ or \”, somewhere within the text.
3. Try out other teammates’ programs. As a team, decide which program performs best and submit it as a team assignment.