Coding Exercise #1

OBJECTIVES

The most effective way to use programming is to start doing it! You may be familiar with the famous 'Hello World' exercise when first learning a new language. This exercise is a more interesting variation of this.

This first CE will be a little light and fluffy but will quickly catch you up (or teach you something new) before you can start to apply Python more usefully (and relevantly) on the class Assignments. Going forward, the remaining Coding Exercise will be much more course content specific.

View this CE as a bit of fun that I can use to gauge where you are in your coding literacy! This Coding Exercise pairs with Background Reading 1 (BR 1), available on D2L and the accompanying sample code for BR 1, available from GitHub (instructions available on next page).

The Background Readings may be quite easy for some of you and a little more challenging for others. Use these as a crutch, as required! If you are already comfortable with Python and programming fundamentals, you do not need to go through the BR in detail. This is for YOU to use as much or little as you need to complete this Coding Exercise. **There are no marks linked to going through the BR files** – these are meant to help you, as needed.

After completing this week's exercises,

You will know how to:

- Look at pre-made code in Eclipse after pulling files from GitHub
- Run a simple script

You will learn about:

- Commenting and indentation blocks
- Data types
- Basic commands
- Lists
- For loops
- Basic GitHub workflow

TASKS

- Open Eclipse and find the **sample-code-solutions** repository under the Project Package Explorer window (usually on the left-hand side). If you can't find this, please email me (opalazhc@unb.ca) or our TA, Ben (bloder@unb.ca).

  Following the instructions in the Background Reading 1 pdf, **pull** from this repository. This should make the file "background_reading_1_2022.py" appear under the repository. Use this as you go through the Background Reading pdf (as needed) to review the basics that are relevant to this Coding Exercise.

- Create a new .py file (the guides from Coding Exercise 0 may be helpful if you've forgotten how to do this) **in your last name-first name repository**. Name your file **coding_exercise_1.** The .py extension will automatically appear.

- This Coding Exercise asks you to make a mini Mad Libs generator. The instructions below walk you through how to do this in detail. Write a script (code) that does the following:

  1. Make a list of strings that contains several of your favourite past-tense verbs, i.e., each of the list items should be a string data type, 'walked', 'fell,', 'danced', etc. Your list can have as many entries as you like.

  2. Assign your list to a variable. Use a variable name that clearly tells the reader what the list contains.

  3. Make a second list of strings that contains several plural nouns. Your list can have as many entries as you like. Your list should be assigned to a variable, e.g., plural_nouns = [item1, item2]. Choose an appropriate name for your variable.

  4. Repeat the process above for two more lists: adjectives and singular nouns (i.e., person, place, or thing). Assign each of these lists to its own variable, selecting (unique) variable names that clearly tell the reader what that list contains.

  5. Make a fifth list that will contain the "story" part of the mad lib. Your list (assigned to a variable that is named something that makes sense and is clear) should have the following entries:

  ['We', 'for the first hour, reviewing the unit on', 'Ben commented that the Nuclear Engineering course was extremely', '. This was mainly because all semester he only watched TV shows on', 'and talked to his', '. I thought this was very', ', so I went home and']

  Note: the quotation mark placements above isolate each string in the list, and the comma placement between each string indicates the next element in the list is about to begin. These are placed intentionally and don't need to be moved around as you type these into the code editor.

  Your list should have 7 elements (7 strings).

6. To make the lib part of our Mad Lib, we'll write 7 additional strings that will later be inserted in between each of the strings in the list you made above in Step 5.

To start, assign a variable to the item in your **past tense verb** list that is at the location (or index) specified by a random number between 0 and the number of the last element in that list. Use the **random.randint(start, end)** function for this.

i.e., variable = list[**random.randint(start, end)]**

7. In total, the following 7 variables are needed, created just as above in Step 6, using the random.randint() function to select the list index each time:

- Two past tense verbs (created using your past tense verbs list and the random.randint(start, end) function)
- Two plural nouns (created using your plural nouns list and the random.randint(start, end) function)
- One singular noun (same as above, with the appropriate list)
- Two adjectives (same as above, with the appropriate list)

8. Combine all 7 variables from Steps 6 and 7 into one list in the following order:

Past tense verb, plural noun, adjective, plural noun, singular noun, adjective, and past tense verb.

9. Make an empty list called **fulllib**. Append the string, 'Ben and I were studying in the Head Rest for our Nuclear Engineering midterm.' to your empty **fulllib** list.

10. Using the **zip()** function, make a for loop that alternates appending each element from your list from Step 5 and your final list from Step 8 to the **fulllib** list.

11. Use the **join()** function to remove the quotation marks around each individual string element, i.e., fulllib = (" ".**join**(**fulllib**)). You can use single or double quotes in front of **join**; however, make sure to include a space between the marks. This is telling the code to add together all list items separated by quotes (all strings), and to include the amount of space between the two strings that you place between the " " marks in the " ".**join** command.

12. Use the **print()** function at the end of your program to print the **fulllib** list.

Your final Mad Lib print statement should have the form of:

Ben and I were studying in the Head Rest for our Nuclear Engineering midterm. We <u>past tense verb</u> for the first hour, reviewing the unit on <u>plural noun</u>. Ben commented that the Nuclear Engineering course was extremely <u>adjective</u>. This was mainly because all semester he only watched TV shows on <u>plural noun</u> and talked to his <u>singular noun</u> .
I thought this was very <u>adjective</u>, so I went home and <u>past tense verb</u>

13. Use comments and additional print commands, variables, etc. as you like throughout your code to clarify to the future user (or yourself) what is going on in your code.

Keep in mind, the above steps can be modified if you feel like doing something a little different in your code. There is no one true “right” way of writing code! The evaluation criteria (next page) are general to accommodate this. **The end! You’re done.**