(PRO) Program Assignment Instructions

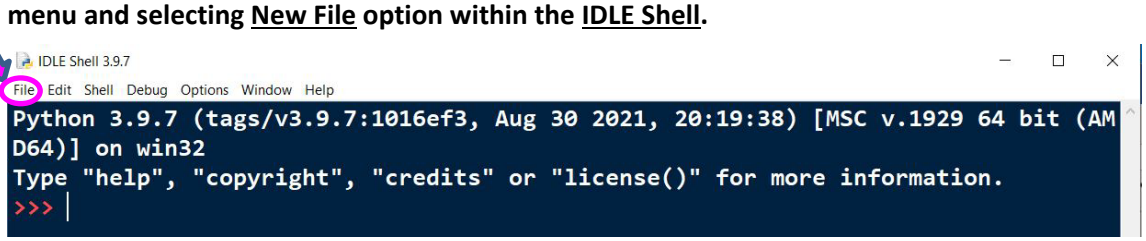
Last Changed: 4/26/2023 12:00 AM

Read and follow the directions below carefully and perform the steps in the order listed. You will be solving one program as instructed and turning in your work electronically via an uploaded file within Eagle Online/Canvas. Make sure and check your work prior to uploading the file.

*Note: Refer to **(SET) How to Download Install and Use Python IDLE (Windows User)** file (**Page 8**) and/or **Use Python IDLE Video** link within **Module 2** on how to **create**, **enter**, **save**, **run**, and **submit** a **script (source code/program) file**.*

Instructions

1. **Using Python IDLE, create a New Empty Script (Source Code/Program) File in your working drive by clicking the File menu and selecting New File option within the IDLE Shell.**



2. **Save the new Script (Source Code/Program) File with the name PROLastFirst.py by choosing File menu's Save option *(Note: Make sure to know where you saved the file in your working directory for future submission)***

   *(NOTE: Where **LastFirst** should be replaced with your actual **Lastname** and **Firstname**. For Example, if your name is Mary Smith then your file name should be named: **PROSmithMary.py.** Reminder: You don't need to add the **.py** extension, IDLE will automatically add the extension)*

3. ***You will develop ONE GAME of your choice from the list of games described below:***

   NOTE: Make sure to follow the requirements below the game you choose and the GUIDELINES to follow in step 4.

❖ (Easy) Hi-Lo Guessing game:

- Have the computer generate a random number* between 1 and 100. This will now be the secret number.
- Use a for loop to allow users to have 10 chances to guess the number (you can use a while loop if desired to give user unlimited tries, ex: while user has not guessed the number).
  - Prompt user for a number between 1 and 100.
  - Read in the number.
  - If the number is less than the secret number, print a message stating it's too low.
  - Else if the number is higher than the secret number, print a message stating it's too high.
  - Else if the number is == to the secret number, then print an appropriate message and exit

*For example, to generate a random number between 1 and 10, you will need the following code:

```
import random
secretNumber = random.randint(1,10)
```

❖ (Easy) Mad Libs:

- Create a story with blanks. For example: Last month, I went to _____ (*city*) with _____ (*name of person*). We traveled for _____(*number*) hours by _____(*vehicle*).
- The following website contains samples, but you can come up with your own or modify one you find on-line: http://www.teach-nology.com/worksheets/language_arts/madlibs/
- Make sure the story is appropriate for all ages!
- The story must be longer than one sentence! I only included the sentence above as an example, but your story should be at least one paragraph with at least 8 to 10 sentences.
- Prompt the user for the necessary words. For example, for our short example listed above, I would prompt the user for a name of a person (read it in as a string). Then I would prompt them for a number (read it in as a double) and then prompt them for a vehicle (read it in as a string).
- Then print out the story with the words that the user entered: Last month, I went to *Paris* with *Oprah*. We traveled for *1000* hours by *bicycle*.
- The story will vary depending on what the user entered. There are many examples of this online (http://www.redkid.net/madlibs/).
- (Optional) To make the program more robust, have more than one story available, and use random number to pick the story or have user pick the story. Make it as simple or complex as you like!

❖ (Easy) Magic 8-ball:

- Prompt the user to enter a yes or no question (Will it rain? Will I make an A? etc…).
- Read in the question: read in the line using input()
- Generate a random number between 0-7.
- Create an array of strings with 8 items. The 8 strings will all be yes/no type phrases (“Absolutely!!!”, “There is no way that will ever happen!”, etc.).
- Based on the random number, print out the position in the array.
- If you’re not sure what a magic 8-ball is, you can pick another project.

❖ (Easy) Funny Face:

- Prompt the user for one character to represent a nose. Read in the character.
- Prompt the user for one character to represent an eye. Read in the character.
- Prompt the user for one character to represent a mouth. Read in the character.
- Based on the characters entered, print out a funny face.
- (Optional, but recommended) To make the program more robust, use a random number to choose between a few different funny faces so they aren’t always the same. Here is an example:

```
Enter a nose: <
Enter an eye: x
Enter a mouth: -

     |||||
     /   \
      x x
     @ < @
      ---
```

❖ (Easy to Moderate) Palindrome:

- Prompt the user for a word or phrase.
- Read in the word or phrase (it’s up to you if you want to deal with phrases).

- Check to see if it's a palindrome. If you use a phrase, ignore spaces and punctuation.
- Print out an appropriate response.

❖ **(Moderate) Rock/Paper/Scissors:**

- Have user enter a rock (r), paper (p) or scissor (s) or you can have them choose from a menu. Have computer randomly generate its own guess (random number between 0-2 and then assign a value based on random number). Print out the computer's guess and user's choice and tell the user who won. Remember, Rock beats Scissors, Rock loses to Paper, Paper beats Rock, Paper loses to Scissors, Scissors beats Paper, Scissor loses to Rock. *(Optional: A more advanced version is using the version of the game from the Big Bang Theory).*

❖ **(Moderate) Tic/Tac/Toe:**

- Have two users play against each other, which is easier than playing against the computer. You will want to use a 2-dimensional array to create a board. For advanced users, this is a great game to try out.

❖ **(Hard Games) Hangman -OR- Memory -OR- Battleship**

❖ **(OPTIONAL)** If you would like to write more than one game, create a menu of games for the user to choose. For example:

1436 Games Menu
---
1. Hi/Lo Guess
2. MadLibs
3. Magic 8
4. Tic/Tac/Toe
0. Quit

4. ***Make sure to follow the GUIDELINES below:***

a. **YOU MUST MAKE YOUR PROGRAM MODULAR:** Write your program using at least 2 functions (apart from the main function), though 3 or more functions will be required for making the highest possible grade. For example, if you chose to create the Hi-Lo Guessing game, have a function to read in the number, have a function to create the random number perhaps, etc…

b. **MAKE SURE YOUR PROGRAM USES AT LEAST ONE LIST, REPITITION STRUCTURE (LOOP), AND SELECTION STRUCTURE (IF/ELSE OR IF/ELIF):** Write your program using at least one List, Loop, and If.

c. **YOU MUST USE DESCRIPTIVE/APPROPRIATE IDENTIFIERS:** Your program should have descriptive Variables (example: guessNumber, nounWord, verbWord, etc.) and CONSTANTS (example: GAME_TITLE, SECRET_NUMBER, etc.)

d. **YOU MUST INCLUDE DOCUMENTATION:** Have enough documentation for understandability of your program by including comment block at the top including your name, the name of the game, a brief description of the game, and prior to each function describe in your own words the purpose/logic of that function *(refer to example programs)*

e. **YOU MUST INCLUDE INDENTATION AND SPACING:** Your program should have proper indentation and line spacing for readability of your program *(refer to example programs)*

f. **YOUR PROGRAM SHOULD BE YOUR OWN DESIGN:** That is, If I have code that is similar to any of these games in you canvas course modules' example program, YOU MUST CHANGE MY CODE (do not copy it exactly). For example, I have included a MadLibs program in one of the modules. Make sure that you change the story if you want to write a MadLibs program, and of course, use functions (as discussed in step a above).

g. **YOUR PROGRAM SHOULD LOOK PROFESSIONAL:** That is, in the comments, make sure no typos! use capital letters at the beginning of a sentence or phrase, use punctuation when appropriate, make the output neat, etc.

HAVE FUN IN DEVELOPING THE GAME OF YOUR CHOICE! 😊

5 After completing your program as instructed, make sure to **Run** your program/script file to obtain the output/results as required.

6. You may now proceed to Program Assignment INSTRUCTIONS and UPLOAD link within this module and follow the steps in the link or follow the steps below to submit your work as a **File Upload** (an attached **.py** file):

    a. Choose the **Start Assignment** button,
    b. Choose **File Upload** tab,
    c. Choose **Browse** to locate your **script (source/program) file** to add,
    d. Choose **Submit Assignment** to complete file upload.

NOTE: ONE OF THE COMMON MISTAKES IS THAT STUDENTS ENTER PYTHON COMMANDS/STATEMENTS IN THE "IDLE SHELL" DIRECTLY AND SAVE THE RESULTS TO A FILE AND SUBMIT WHICH IS INCORRECT!!!

INSTEAD...

YOU SHOULD FOLLOW THE ABOVE STEPS TO CREATE A NEW SCRIPT (SOURCE CODE/PROGRAM) FILE FROM THE IDLE SHELL, SAVE THE FILE, ENTER PYTHON STATEMENTS (PROGRAM) INTO THE FILE, RUN YOUR PROGRAM, AND SUMBIT THAT SCRIPT (SOURCE CODE/PROGRAM) FILE AND NOT THE OUTPUT OF THE IDLE SHELL!!!