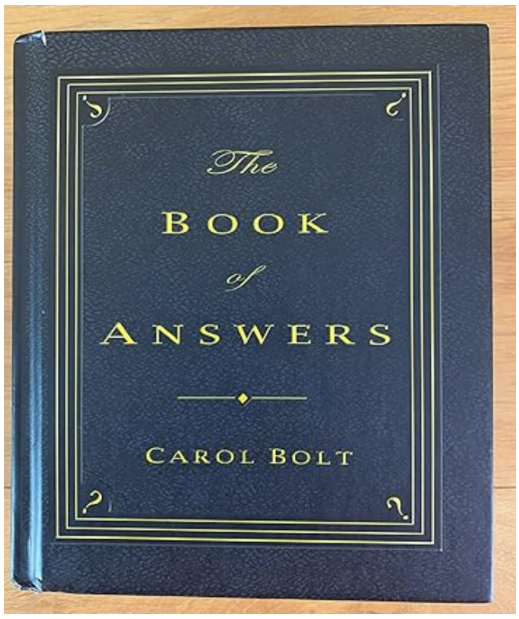
Homework #3: The Book of Answers - Digital Version

With this digital book of answers, the user can:

- ❖ Ask a question
- ❖ Open to a page randomly
- ❖ Receive an answer (You can specify your own answers in the book)



Example:

- The user opens the book
- The digital book of answers prompts the user to ask a question
- The user asks: “Will I change majors?”
- The digital book of answers gives one of the established answers
- The digital book of answers continues to ask for the next question until the user closes the book

(picture from Amazon https://www.amazon.com/Book-Answers-Carol-Bolt/dp/0786865660)

Instructions

In this assignment, you will be writing the ***DigitalBookofAnswers*** class with the following methods:

1. An **__init__(self, answers)** method: This will initialize a new ***DigitalBookofAnswers*** class
   - Set the attribute ***book_answer_list*** to the ***answers*** argument***.*** This is a list of the possible answers a user could receive from the book.
   - Set the attribute ***questions_asked_list*** to an empty list, this is used to store the asked questions.
   - Set the attribute ***answered_list*** to an empty list, this is used to store the indices of the picked answers.
2. A **__str__(self)** method**:**
   - Returns a string with all of the answers in ***book_answer_list,*** separated by dashes:

```
Follow Your Inner Voice - Stay Positive - Go For It - Believe in Yourself - Stay Open to the Future - Enjoy It
```

   - If ***book_answer_list*** is empty, return an empty string like “”.

3. A ***check_get_answer(self, question*)** method:

- First, check if the question has been asked before
  - If it has, this method returns a string "I've already answered this question. The answer is: <answer>" with the actual answer to that question. e.g. return "I've already answered this question. The answer is: Follow Your Inner Voice"
  - ***Note:*** When it is a repeated question, you should not add the index of this answer to the ***answered_list*** again.
- If the question has not been asked before, pick an answer at random from ***book_answer_list*** and return the answer in a string "<answer>".
  - Add the index of that answer at the end of ***answered_list***
  - ***Note:*** You need to add the index of the answer in the ***book_answer_list*** to the ***answered_list*** here.
  - ***Hint:*** Python has a built-in module that you can use to make random numbers: random module

4. An ***open_book(self)*** method: This method controls the book use for the ***DigitalBookofAnswers*** object

- If it is a new session, prompt the user to ask a question: "Turn 1 - Please enter your question: "
- If the question input is "Done" (case-sensitive) then print "Goodbye! See you soon." and stop the current loop of question-prompting.
- Otherwise, add the question to the ***questions_asked_list*** and use the **check_get_answer()** method to generate an answer. The steps are below:
  - Print out the answer
  - Add the question at the end of ***questions_asked_list***
  - Prompts the user to ask the next question. The turn number in this string should be updated: "Turn <turn_number> - Please enter your question: "
    - ***Hint:*** You can use the length of ***questions_asked_list*** to get what the next turn number should be

5. An ***answer_log(self)*** method**:** This method prints out the answers

- Using the ***answered_list*** to count how many times each answer is given to unique questions.
- Returns a list with frequency information for all the answers, each item in the list is a string, the string should look like "<number_of_times> - <answer>".
  - ***Note*****:** <answer> in this list should all be lowercase.

- The returned list should be sorted in descending order based on the number_of_times each answer is provided in response to unique questions. In other words, the most frequently given answers should appear first. If multiple answers have the same frequency, their order in the list doesn't matter.
    - ***Hint:*** You can use ***.sort()*** if you are more familiar with list
- If there are no answers in ***answered_list,*** it will print "Empty" (case-sensitive) and return an empty list.

6. A ***main()*** function:

- Create the ***DigitalBookofAnswers*** object and pass in a list of possible answers as ***book_answer_list***. For example:
    - Follow Your Inner Voice
    - Stay Positive
    - Go For It
    - Believe in Yourself
    - Stay Open to the Future
    - Enjoy It
- Initiate the book using the ***open_book()*** method
- Shows the output of the ***answer_log()*** method in the terminal screen

Given the example possible answers, here are two sample outputs from the main method: *Note: As the answers are picked randomly, your output might be different from the sample outputs.*

```
Turn 1 - Please enter your question: Done
Goodbye! See you soon.
Empty
[]
```

In this example,

- There are no actual questions asked, so the ***answered_list*** is empty, so it prints out "Empty"
- It also returns an empty list as the output of the ***answer_log(self)*** method.
- This output is shown in the terminal screen following the requirement in the ***main()*** function.

```
Turn 1 - Please enter your question: Should I have sushi now?
Follow Your Inner Voice
Turn 2 - Please enter your question: Should I have sushi now?
I've already answered this question. The answer is: Follow Your Inner Voice
Turn 3 - Please enter your question: Should I have sushi now?
I've already answered this question. The answer is: Follow Your Inner Voice
Turn 4 - Please enter your question: Should I go to park now?
Go For It
Turn 5 - Please enter your question: I am lost
Enjoy It
Turn 6 - Please enter your question: Should I sleep now?
Go For It
Turn 7 - Please enter your question: Done
Goodbye! See you soon.
['2 - go for it', '1 - follow your inner voice', '1 - enjoy it', '0 - stay positive',
 '0 - stay open to the future', '0 - believe in yourself']
```

In this example,

- The same question (Turn 1 to Turn 3) is asked three times, so the second two answers included “I've already answered this question. The answer is:” and only the first answer turn is included in ***answered_list,*** and used when counting the frequency of answers used for unique questions.
- Turn 4: a new question with a new answer
- Turn 5: a new question with a new answer
- Turn 6: a new question with an answer appeared for a different question before (Turn 4). The index of this answer in the ***book_answer_list*** needs to be added to ***answered_list*** as it is responding to a new unique question.
- The user entered Done at Turn 7, so it prints “Goodbye! See you soon.”
- It then shows the frequency information for each answer in a sorted list: starting with the one that is used twice to answer unique questions (Turn 4 and Turn 6), followed by the one that is used once to answer a unique question (Turn 1), and the remaining replies.

Grading Rubric - Total of 60 Points

- 5 points: the ***__init__*** method sets the object's ***book_answer_list,*** ***questions_asked_list,*** and ***answered_list*** correctly to the passed arguments, sets both the object's ***questions_asked_list*** and ***answered_list*** attributes to an empty list.
- 5 points: the ***__str__*** method
    - 3 points - When ***book_answer_list*** is not empty, return a string with all the possible answers in ***book_answer_list*** separated by dashes:
        - "Follow Your Inner Voice-Stay Positive"
    - 2 points - Otherwise, return an empty string like ""
- 5 points: the **check_get_answer** method returns "I've already answered this question. The answer is: <answer>" if the question has already been asked
- 5 points: the **check_get_answer** method adds the index of the answer to the ***answered_list***
- 5 points: If it is a new session, the ***open_book*** method prompts the user to ask a question: "Turn 1 - Please enter your question: "
- 5 points: the ***open_book*** method continually prompts the user for a question, using the prompt "Turn <turn_number> - Please enter your question: " as long as they don't input "Done"
- 5 points: the ***open_book*** method adds the questions to ***questions_asked_list***
- 5 points: the ***open_book*** method uses the **check_get_answer*()*** method to correctly get the answer
- 5 points: ***answer_log*** returns a formatted list with the information for each of the answers from ***answered_list,*** <answer> in this list should all be lowercase.
- 5 points: ***answer_log*** sorts the returned answer_log correctly
- 2 points: ***answer_log*** returns an empty list if there are no answers in ***answered_list.***
- 2 points: ***book_answer_list*** is properly defined and used in the ***main()*** function
- 2 points: the ***DigitalBookofAnswers*** object is properly defined and used in the ***main()*** function
- 2 points: the ***open_book*** method is used correctly in the ***main()*** function
- 2 points: the ***answer_log*** method is used and displayed correctly in the ***main()*** function

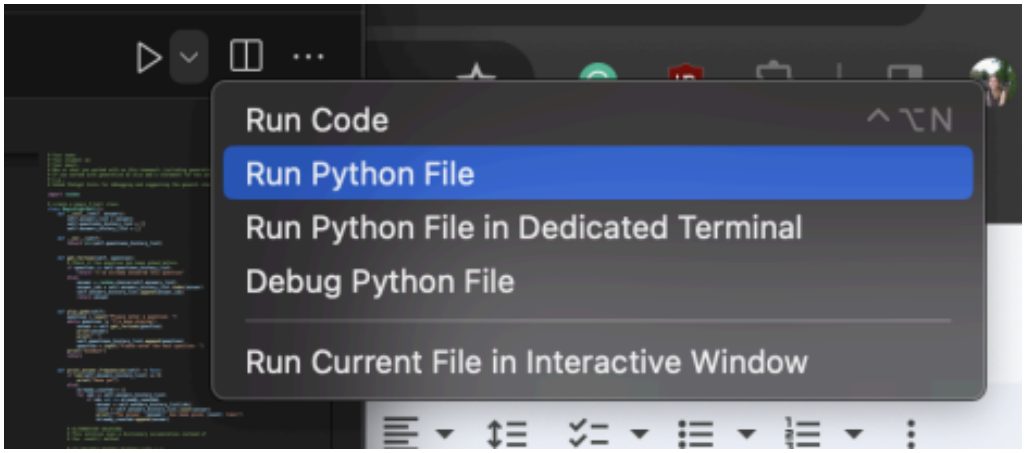
Extra Credit: 6 points

Create a **my_test()** function that creates a DigitalBookofAnswers object and tests each of the possible outcomes.

- **1 point:** Correct output from ***answer_log*** when no questions have been asked.
- **2 points:** Correct behavior from ***answer_log*** when ***answers_list*** is ['Stay Positive', 'Go For It', 'Enjoy It'] and ***answered_list*** (store the indices of the picked answers) is [2, 1, 2]
    - **Hint:** you can modify the value of attributes on a class that's already been created. For example, if your ***DigitalBookofAnswers*** object is called, you can make ***answered_list*** equal to an empty list by setting ***DigitalBookofAnswers***.answered_list = []
- **1 point:** Correct prompt from ***open_book*** to ask the first question "Turn 1 - Please enter your question: "
- **1 point:** Correct output from ***check_get_answer*** when the same question is asked twice.

Running Your Code:

If you are having trouble running your code / interacting with the program in VSCode, click the arrow in the top right corner of your VSCode window. Then, hit "Run Python File."



Submission instructions:

Follow the instructions on Canvas to submit your git repo link by the due date and time.