Final Project Requirements Document

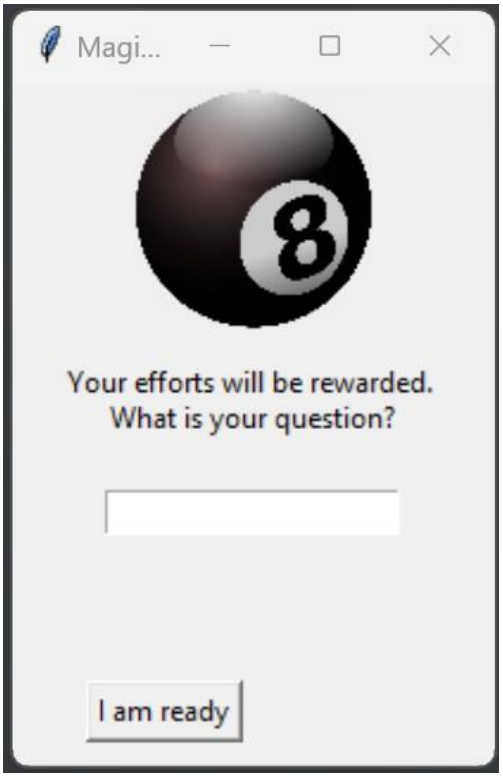
Anna Quigley *December 18, 2022*

- ☑ A working GUI tkinter application with at least two windows.
- ☑ Implementing a modular approach in your application.
- ☑ Consistent clear navigation throughout the GUI application.
- ☑ Use at least two images in your application
  - 8-ball
  - universe
- ☑ Include at least three labels.
  - image label
  - header label
  - answer line label
- ☑ Include at least three buttons.
  - I must learn
  - I am content
  - I am ready
  - I have more questions
  - I am satisfied
  - Exit program
- ☑ Include at least three call back function with each button, including exit button.
  - I must learn → get_question()
  - I am content → finished()
  - I am ready → return_answer()
  - I have more questions → get_question()
  - I am satisfied → finished()
  - Exit program → quit()
- ☑ Implement secure coding best practices, including input validation to check if the user entered the correct data type, make sure the entry box is not empty, etc.
  - Input validation works using check_question() to make sure that the question field is not blank.
  - I struggled with the program getting stuck on this module, but eventually worked out code without errors.
- ☐ Validation testing -. Develop an appropriate set of test data to fully validate the program against.
  - the data sets you tested against.
    - no input (blank question field)
    - long text field
  - a brief written explanation of the results of your tests and what you had to fix.

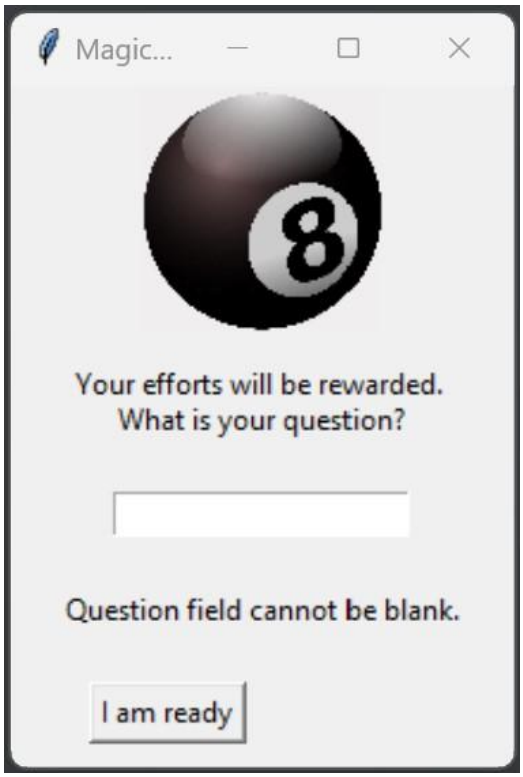
- First, I tried to write the program in one go. It opened a blank window. I separated the logic from the GUI and that worked. Then I worked on the GUI until I got a working frame. Finally, I put the two together.
- I had a problem with getting the randomized answer to display. I wanted it to replace the user's question in that field. I learned that the field was immutable, so I created a separate label field that was mutable for the answer line.
- I then struggled to get the result from the get_answer module to display as text on the answer_line. I solved this by moving the code from the get_answer module to inside of the return_answer module.

- screen shots of your good test data working. (execution of the program showing what the user would see)

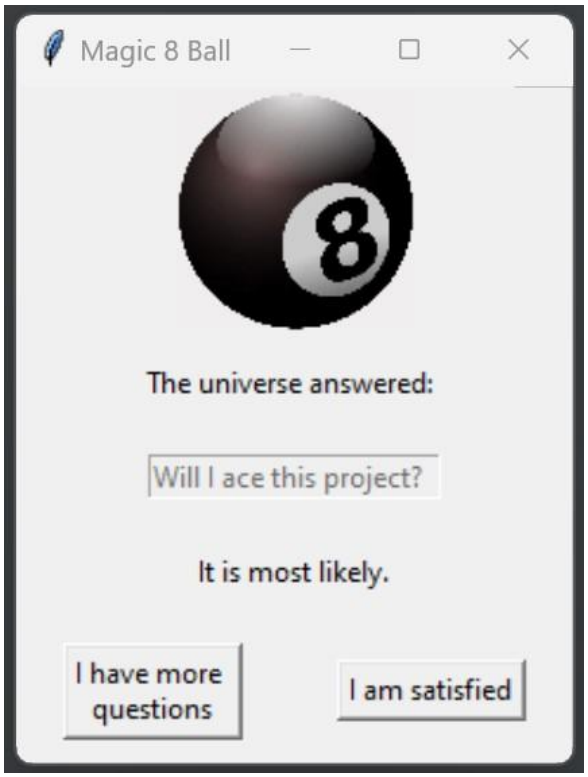
main()



get_question()



check_question()



return_answer()

(I did not program that answer, it was randomly selected!)

finished()

- ☑ User manual creation - You will write and submit a User's manual for your final project and submit it according to the instructions in the attached file.

- ☑ You will fully document (comment) the corrected Python tkinter source code with appropriate comments including:
  - ☑ A brief explanation of the purpose of each module (Sub) at the beginning of each Sub. (A header's comment)
  - ☑ Explanation of the purpose of each variable where it is declared. (An end line comment)
  - ☑ Line by line, or at least section by section comments within the code, explaining what the line/section does.
- ☑ The link of the GitHub repository for your final project.
  - https://github.com/aquigleyivy/SDEV140.git