**CODING EXERCISE 3**

Your line manager asks you to produce an interactive Quiz game for an upcoming event. He doesn’t want anything too fancy, but he wants something neat and user-friendly.

**THE MANAGER’S INSTRUCTIONS**:

1. The beginning of your code must: display the author’s name (i.e., code author), clearly state the purpose of the game, outline the rules of the game, display the name of the game (choose an appropriate name) and version of the release, and show the copyright details, i.e., © [author name].
2. When the user launches the game, he or she must be greeted with a message in the form “Welcome to the world of [Name of game], [Name of player]!!”. Next, what appears on-screen must be the rules of the game and instructions on what he or she must do to play the game pleasantly. From there, the game must ask the user the following question:
   1. **Are you happy to play the game with those rules?** His or her answer must be a form of “yes” or “no” (your pick). If the user chooses “yes”, then *start the game* immediately. On the other hand, if the user’s answer is “no”, then *end the game* and display the message: “Sad to see you go [Name of player]. We hope to see you next time!!”
   2. If the user enters anything other than yes or no or derivatives thereof, capture that as an error within your code and display it on-screen in the form of a neat message that INSTRUCTS the user they must enter yes or no to proceed. Give them the option to enter yes or no again.

**REQUIREMENTS AND RULES**:

1. The game is a series of multiple-choice questions, meaning a question must be posed, and the user must choose the correct answer from a few given choices.
2. The user needs a way to enter their name before gameplay. Their name can be anything, so there shouldn’t be any restrictions.
3. The user should have an option to inform the game that they want a certain number of questions to answer in a given round. To that end, limit their options to either a 3 or 5 question round. Catch any errors, make said errors user-friendly, and give the user the option to enter their data correctly again.
4. You must create 8 to 10 questions in total (up to you what they’re), each with 3 choices: a- [possible answer1], b- [possible answer 2] and c- [possible answer 3]. One of them must be the correct answer. The user must also have an option to enter their answer which, can only be a, b or c. Catch any potential errors that the user might make, display them into nice, helpful messages, then re-give them the option to enter their answer again.
5. From your pool of 8 to 10 questions, the computer must display them randomly, one at a time, every time the user answers a question, whether correct or incorrect. However, questions must not be repeated.
6. If the question is answered correctly, the user should be awarded 10 points with the following message: “Congrats [name of player]. You got 10 points!!”. If the question is answered incorrectly, the user gets 0 points with the following message: “Aww, what a shame. That wasn’t the correct answer. You got 0 points”.
7. At the end of the round, sum their points. For a round of 3 questions, they must get two questions correct to win, i.e., 20 points; and, for a 5-question round, four correct answers are needed, i.e., 40 points, to be declared the victor. If the user is the victor, display the message “Bravo, bravo [Name of player]! You’ve won the round!”. On the other hand, if the user loses the game, display the message “Oh dear. So close and yet so far. Good luck next time [Name of player]!”
8. At the end of a game, regardless of the outcome, the game must ask the user the following question: **Would you like to play again?** If he or she answers yes, relaunch the game with the last entered round. If their response is no, the message should say: “Thank you for playing [Name of game]. See you next time!”. *End the game*. And, of course, catch any errors and package them into something digestible for the user. Give the user the option to enter their answer again.

**ADDITIONAL REQUIREMENTS**:

Build the program in Python and add your source code file to your Github repository. Do not forget to use comments to show your steps.