**Quiz Evaluation**

My program fully meets the requirements of the task, allowing the user to play the online safety quiz. I feel that I have completed the task to the best of my ability, within the time allowed.

**How well my program meets the quiz requirements**

To meet the requirements of the quiz, I have used three functions (subprograms in Python) called ‘newPlayer’, ‘quiz’ and ‘result’. Using functions avoids repetition in the code, making it easier to read and allowing the program to run faster. The ‘newPlayer’ function allows input of a player name, validates it against a list of registered players and then displays the rules of the game. The ‘quiz’ function selects which questions to ask the player, displays these questions in a different order each time and allows the player to select an answer. This input is validated and a score is assigned to each correct answer. The ‘result’ function displays the results of the game, stating the player name, level number and total score.

I have used the list data structure to store the quiz questions and the player names. Lists allow the program to easily retrieve this data. I have used a third list, called ‘questionsAsked’ to store the questions that have been asked, so that the same question is not asked more than once during the quiz, which allows the quiz to run correctly. To do this, I used a WHILE loop to append the question if it has already been asked. Using a list data structure is beneficial because the lists can be appended to or changed if necessary. This means that the program could be easily developed to add more players or questions.

The subprogram ‘newPlayer’ uses the built-in input function to meet the requirement for the player to enter a player name. This subprogram uses a FOR loop to check through the list of player names and if the player name is stored in the ‘players’ list, the rules of the game are displayed, using the print function. Displaying the rules of the game will make it easier for the player to use the program. The program successfully uses the ‘players’ list to validate, using Boolean operators True and False, whether or not a player is registered, which prevents unregistered players from using the program.

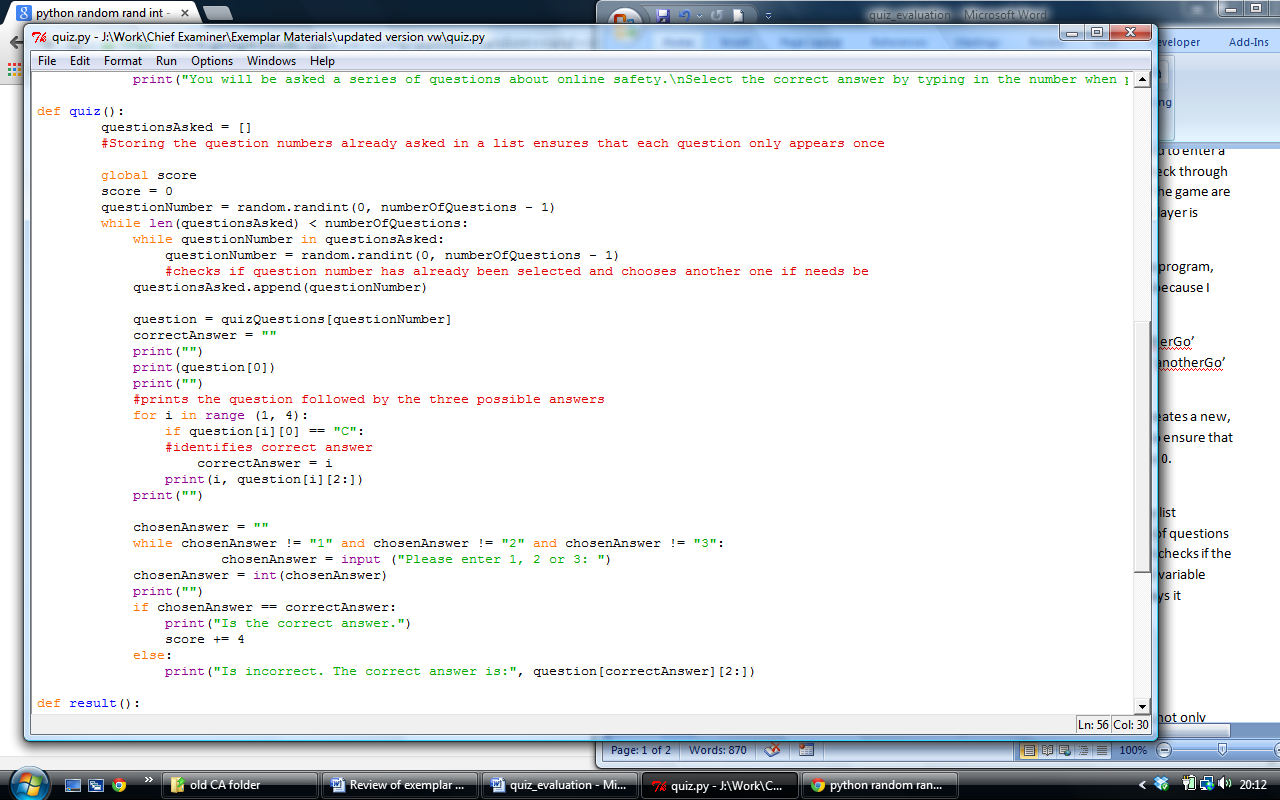
I also validated the second user input, which is when the player enters an answer for the question. I used a WHILE loop to display a message asking the player to enter either 1, 2 or 3 while the input was not “1”, “2” or “3”. I could have used an IF statement here to check if the input was correct but the program works better with a WHILE loop. Using validation in a WHILE loop means that if a player enters an invalid character, they will be prompted to enter a valid character and the program will not simply stop.

To make my program easy to read, I made sure I used variable names that are descriptive; I originally named the ‘anotherGo’ variable ‘playAgain’ but it was too similar to the other ‘play’ variables, so I chose to use ‘anotherGo’ to avoid any confusion. I also used correct indentation to structure the program and added detailed comments in my code to explain how the program works, making it easier to understand.

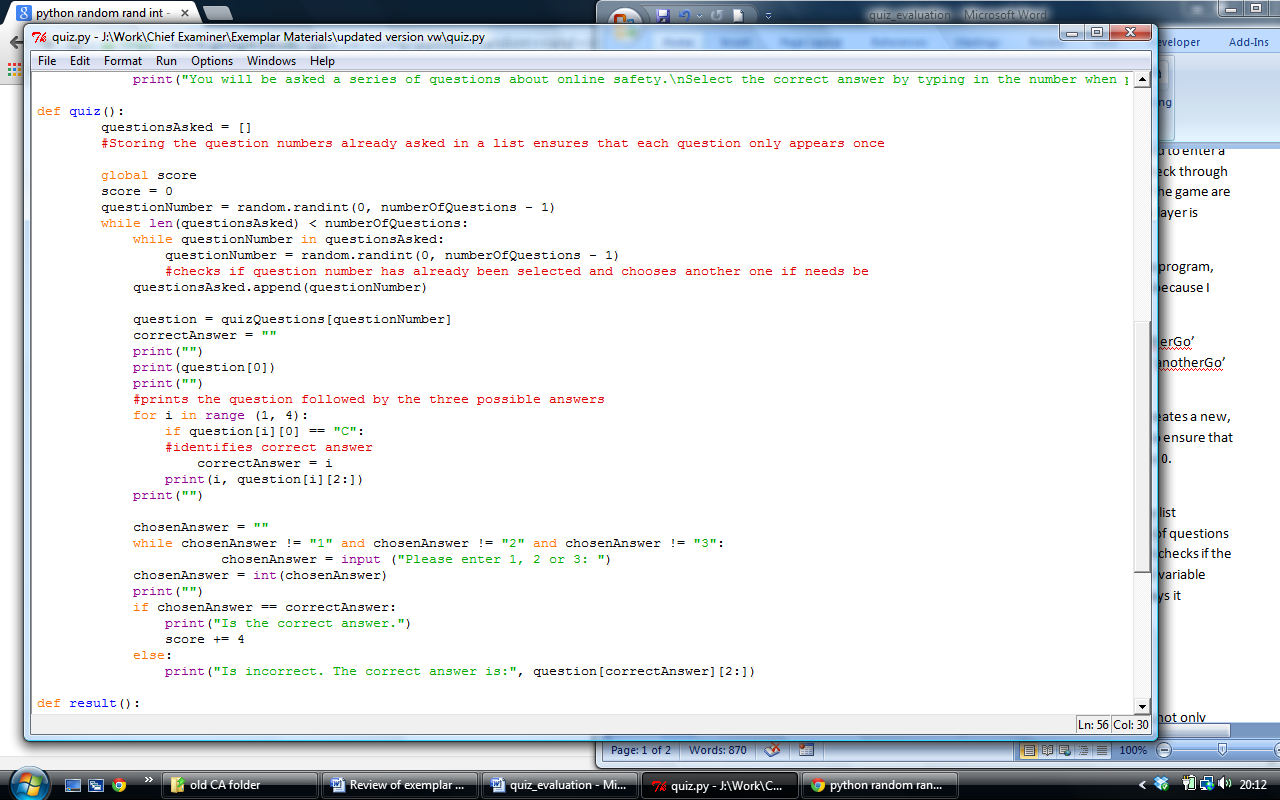
A strength of my program is that if a player answers a question incorrectly, the program not only tells them that the answer is incorrect; it also displays the correct answer. I did this by using a variable called ‘correctAnswer’ which identifies that correct answer in the ‘quizQuestions’ list. Another strength of my program is that it only allows registered players to take the quiz, through the use of the ‘players’ list. A weakness in this part is that the player has to type their name exactly as stated in the list, with a capital letter at the beginning of the name; otherwise the program will display the error message stating that the player is not registered. If I had more time, I could have added validation to allow upper or lower case player names, which would allow the name to be recognised regardless of capitalisation.

To improve my program, I could display the score each time the player answers a question, rather than waiting until the end of the quiz to display the final score. I could also add more questions or players, which could be done easily by appending to the lists. I could also include a GUI (graphical user interface) to improve the appearance of the program.

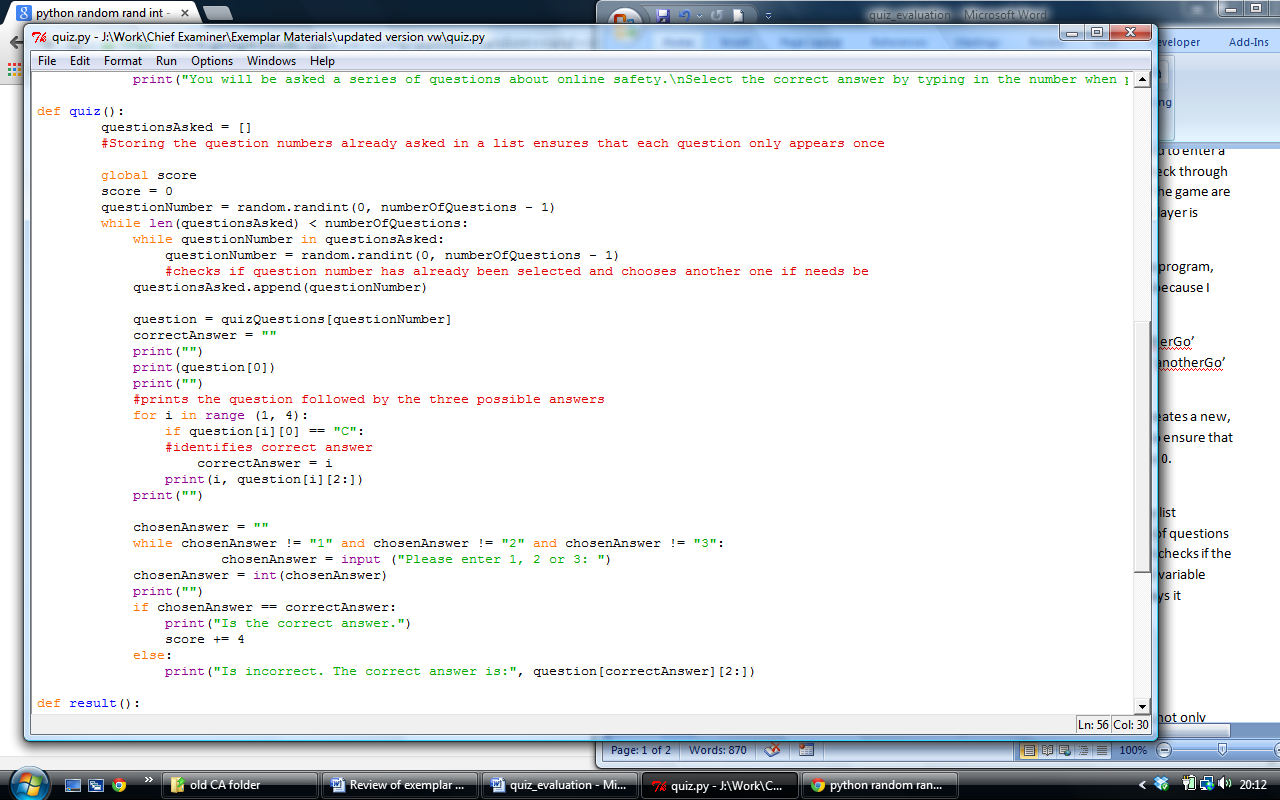
**How my program selects which questions to ask**

The program selects which questions to ask in the function ‘quiz’. This function firstly creates a new, empty list called ‘questionsAsked’, which will hold the questions that have been asked to ensure that the questions are not repeated. It declares a global variable called ‘score’ which is set to 0. This variable is global because it is also used in the ‘results’ function to display the total score.

The local variable ‘numberOfQuestions’ works out how many questions there are in the list ‘quizQuestions’. When a question has been asked, this question number is stored in the list ‘questionsAsked’. The program works out if this number is less than the number stored in the variable ‘numberOfQuestions’. It then checks if the question number has already been selected and chooses another one if it has.



The local variable ‘question’ is then created which takes a question from the ‘quizQuestions’ list and displays it along with the three possible answers. It then checks to see which possible answer begins with the letter “C”, this being the correct answer and stores it in the variable ‘correctAnswer’.



A local variable called ‘chosenAnswer’ is declared, which will store the input answer. A WHILE loop provides validation to check if the chosen answer is not 1, 2 or 3, while this is the case, the message “Please enter 1, 2 or 3:” will display. The program then checks if the value stored in the ‘chosenAnswer’ variable is the same as the value stored in the ‘correctAnswer’ variable. If it is, the message, “Is the correct answer”, is displayed and 4 points are added to the score. If it is not, a string stating that the answer is incorrect is displayed, along with the correct answer.

