Lab 3 Math Quiz

Brandon Gonzalez, Ryan Demboski

9/29/19

Section 001

1. **Problem statement**

This problem asked you to make a math quiz, the math quiz had 3 levels beginner, intermediate, and advanced. Each level has its own set of requirements, beginner has addition and subtraction questions that are between one and ten, intermediate had addition, multiplication, subtraction, and division questions between 1 and 25, lastly advanced included multi-step problems that were between 1 and 25. The user should be able to choose the level that they want and how many questions they want. At the end of the quiz it must give the user a score, 2/3=Well done! Between 1/3 and 2/3 is You need more practice, and less than 1/3 is you need more practice.

- Random number of questions picked by user
- Difficulty picked by user
- Beginner 1-10
- Intermediate 1-25
- Advanced 1-25
- Quiz scores
- All choices work regardless of case
- Use if-elif-else statements

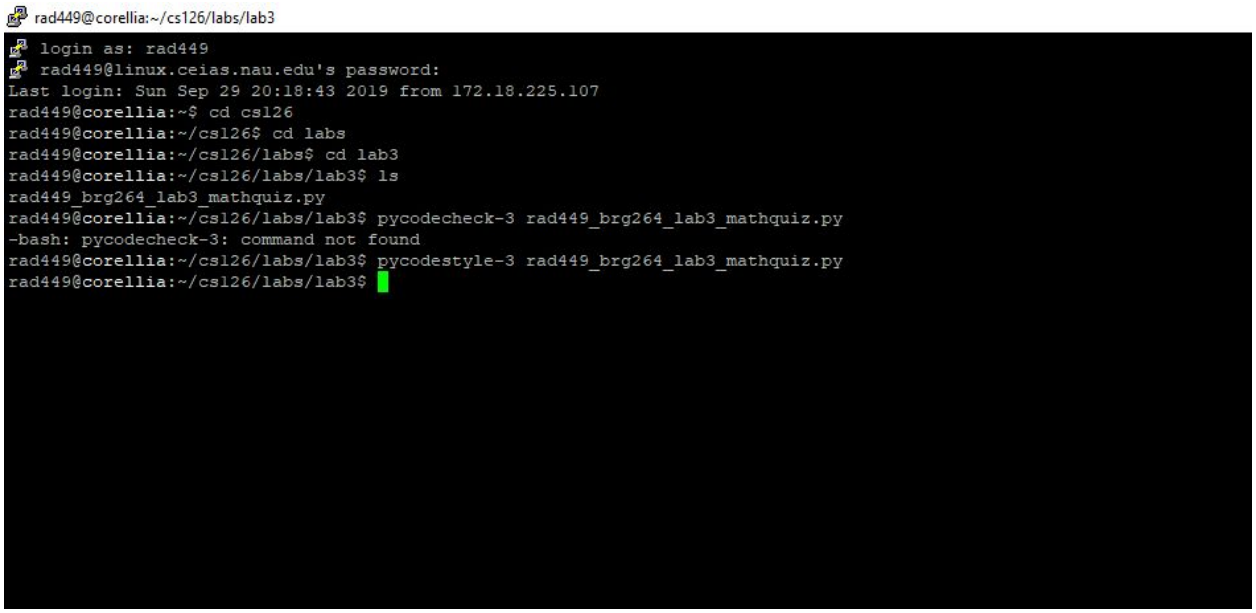
2. Planning

We started our planning by splitting our problem into parts. Our first part was how to make a function that lets the chooser input a certain amount of questions and that amount is returned. Our second part was creating a counter that counted every question right or wrong and added it up. Our 3rd part was creating our 3 different levels of the quiz. Lastly was the quiz score and returning the proper score to the user.

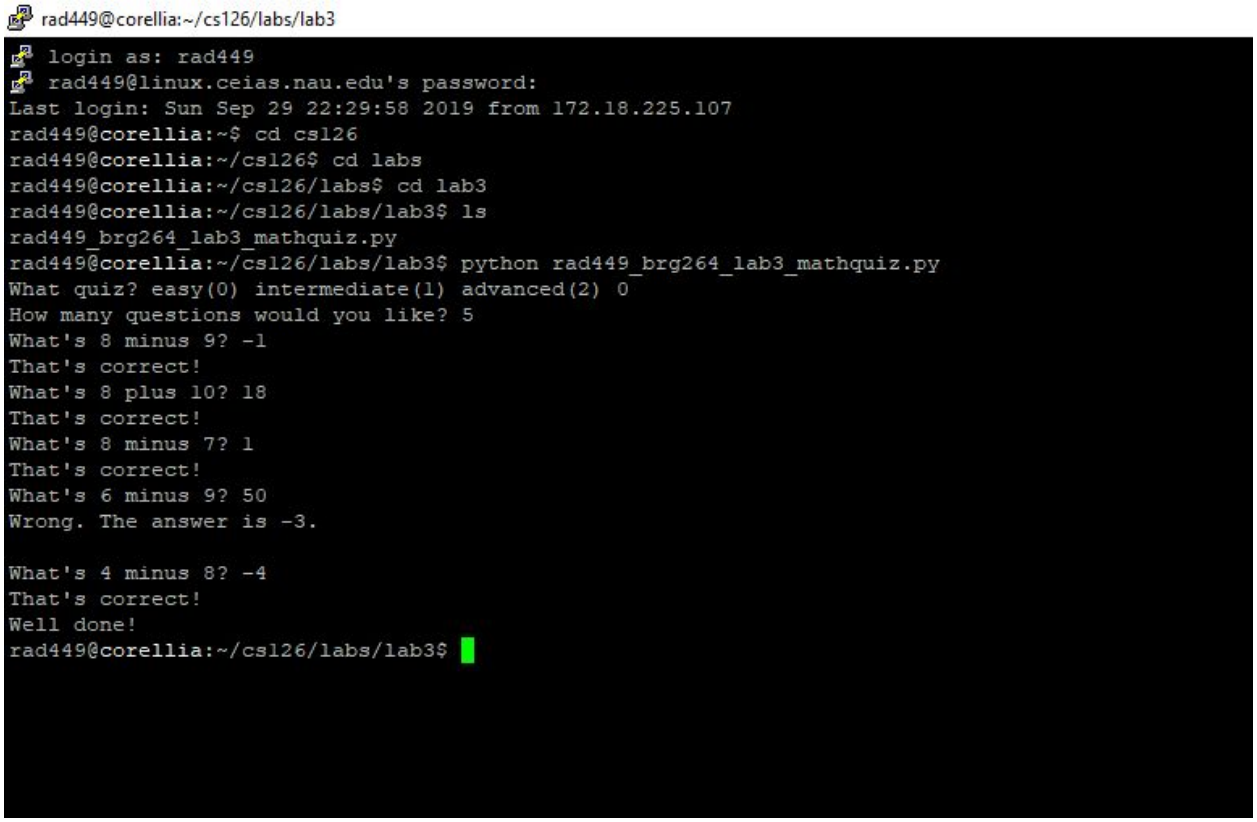
3. Implementation and Testing

We implemented our plan by going step by step. We first created our function that allowed the user to input any amount of questions. Next, we used the counter that was given to us in the prelab to count the amount correct. After that we started by working on the beginner level after that was complete, we moved on to the intermediate and advanced. Lastly, we established the correct return statements for quiz scores and tested our code. Overall our plan worked great, we ran into a few issues but after some troubleshooting, we were able to create a working math quiz.

Proof of PEP8 Compliance:



Example of the script running:



Another example:

```
rad449@corellia:~/cs126/labs/lab3$ python rad449_brg264_lab3_mathquiz.py
What quiz? easy(0) intermediate(1) advanced(2) 2
How many questions would you like? 5
1 divided by 12 minus 3? -2.92
That's correct!
What's 10 plus 4 times 3? 22
That's correct!
What's 14 times 6 plus 2? 1000000
Wrong. The answer is 86.

What's 23 times 6 plus 2? 140
That's correct!
6 minus 13 divided by 2? -0.5
That's correct!
Well done!
rad449@corellia:~/cs126/labs/lab3$
```

Last example:

```
rad449@corellia:~/cs126/labs/lab3$ python rad449_brg264_lab3_mathquiz.py
What quiz? easy(0) intermediate(1) advanced(2) 1
How many questions would you like? 4
What's 9 plus 8? 5
Wrong. The answer is 17.

What's 17 plus 3? 5
Wrong. The answer is 20.

What's 20 divided by 19? 5
Wrong. The answer is 1.05.

What's 6 minus 5? 5
Wrong. The answer is 1.

Please ask your math teacher for help!
rad449@corellia:~/cs126/labs/lab3$
```

4. Reflection and Refactoring

Overall our approach worked great. Working with input statements we were able to ask the user the proper questions and allow them to input integers. We made our level choice into numbers so 1= beginner, 2=intermediate, and 3=advanced. We used if and elif statements for most of our statements that asked the actual math questions. Some answers required us to make float statements. We also used print statements to print the proper terms that should be stated depending on the scores. To improve we could've put some if statements together and make it look a bit cleaner, overall our plan worked great and the code worked great. An alternative technique might've been to leave the level choice as variables and not change it, this would've made the code a bit easier to understand for whoever was evaluating it.