Project 4

Corrections

Here are some corrections and clarifications. They are now integrated in the original instructions below.

1. For any question, instead of the text "Try again! You have 2 more tries" , use the text "You have this many remaining tries: " followed by the number of tries left for that question.
2. When the user gets the answer right, we do not need feature 1 text "You answered ..."
3. Feature 4 text "Sorry your answer is incorrect" is not expected from the tests, so you can omit it from your print statements.
4. If the user gets the answer right once, you need not ask for more inputs for the same question from the user even if some number of tries are left.
5. We made some tweaks to the tests to give more informative errors regarding why a test fails (so consider re-downloading them)
6. Use the following hint texts for the questions

Question 2: notice the quotes!

Question 3: Calcuate the right side first. Don't forget != means not equal to.
(Notice the spelling mistake! We missed it in editing.)

We have released a new `test2.py` file with our corrections, so you should probably use that for testing. However, you may still rely on `test.py` if you prefer (we don't want to make people who have already figured out how to pass the original tests redo any work). When grading, we'll run both `test.py` and `test2.py`, taking the better score. You can run the new tests like this:

```
python test2.py
```

If you want to run just one test when you're debugging, you can run `python -i test2.py`. This will drop you into interactive mode after all the tests run. This is convenient, because running each test just requires calling a function. For example:

```
>>> result = test_3()
... lots of output ...
>>> result
PASS
>>> result = test_4()
...
```

Description

The CS 301 midterm is coming up, so now is a good time to begin to study. A great resource for preparing is to look at the old midterm exam which is available here. Since we've been learning about conditionals this week, for this project we'll build an automated study tool program to quiz you on the questions. You'll get practice with conditionals and while loops.

Start by downloading `test.py`. Double check that these files don't get renamed by your browser (by running `ls` in the terminal from your `p4` project directory). You'll do all your work in a new `main.py` file that you'll create and hand in when you're done. You'll test as usual by running `python test.py`.

The lab this week is designed to give you practice with conditionals and while loops, so be sure to do the lab from home (if you missed it) before starting the project.

The project consists of writing code to create an interactive quiz program. We've broken the program down into 10 features you can add one at a time.

Note: This project does not provide you a project.py (you won't be needing it) or a main.py to start from (you can start from scratch). You should hand in main.py file when you are done.

Requirements

Questions and Functions

Feature 1: Ask a question, get an answer

Ask the user:

```
What is the type of the following? "1.0" + "2.0"
a) int
b) float
c) str
d) bool
e) NoneType
```

For all questions, print the question and then use

```
input("Your answer: ")
```

to have the user type in an answer.

After they have entered input, print out this line:

```
You answered 'a'. The correct answer is 'b'.
```

Where 'a' is what they entered and 'b' is whatever letter is the right answer.

Feature 2: Check the answer for correctness

Use an `if` statement to check to see if the answer is correct. If it is, print:

```
Congratulations! You got it right.
```

When the user gets the answer right, we do not need feature 1 text "You answered ..."

Feature 3: Clean the input on text

You may have noticed that if the user inputs a correct answer, but it's capitalization isn't right or it has extra spaces, then Feature 2 will think it's a wrong answer.

Create a new variable and use it to store a 'clean' version of the input by removing any spaces or capitalization using the str.strip() and str.lower() built in functions. Use the user's original input for the Feature 1 text.

Feature 4: Tell user when answer is wrong

Use an `else` statement to tell a user that their answer is wrong:

```
Sorry your answer was incorrect.
```

This should be printed on the line before the text from Feature 2.

Feature 5: Ask a fill in the blank question

Add this as question 2:

```
What is the type of the following?
"1" * 2
```

Notice there are no multiple choices here!

Feature 6: Make a function to ask the Questions

Notice how you have had to repeat code to ask question 1 and 2. Let's consolidate this into a function named askQuestion which has 2 parameters, one for the question and one for the answer. Call this function to ask both question 1 and question 2.

Feature 7: Add another question

Add this as question 3:

```
What does this expression evaluate to?
True != (3 < 2)
```

All three questions should use the askQuestion function which is called from the global block. Note that we are changing all user input to lower case so all answer variables should all be in all lower case.

Feature 8: Let them try again

Before asking any of the quiz questions, first ask the user

```
"How many tries do you want for each question: "
```

and save that in a variable. Use that variable in a `while` loop to give the user that many tries to get the answer right.

If they get it right, they should still see the Feature 2 ("Congratulations!") text. If the user gets the answer right once, you need not ask for more inputs for the same question from the user even if some number of tries are left.

If they get it wrong, instead of the Feature 4 ("Sorry your answer was incorrect") text, print

```
"You have this many remaining tries: " followed by the number of tries left for
that question.
```

Where the number is the correct number of tries left. They should see the Feature 1 text once they either get it right or use up all their tries.

Feature 9: Give a Hint

Use an `elif` to check to see if they are on their last try. If so, give them a hint. The hint text will have to be added as another parameter for the AskQuestion function.

Set the default value of the hint parameter to be "Check the textbook".

The hint for Question 2 should be: "notice the quotes!"

The hint for Question 3 should be: "Calcuate the right side first. Don't forget != means not equal to."
(Notice the spelling mistake! We missed it in editing.)

Feature 10: Keep Score

Use global variables to track the number of questions they have gotten right and how many they have gotten wrong.

After all questions have been asked, print out:

```
You tried 3 questions and got 2 right.
```

to display their score. Be sure the formatting and spaces are exactly the same to pass the tests.

Final result

If you've implmented all 10 features correctly, your code should look like this when it runs:

```
How many tries do you want for each question: 1

What is the type of the following? "1.0" + "2.0"
 a) int
 b) float
 c) str
 d) bool
 e) NoneType

Your answer: c



Congratulations! You got it right.

What is the type of the following? "1" * 2
Your answer: str

Congratulations! You got it right.

What does this expression evaluate to?
 True != (3 < 2)
Your answer: True
```

```
Congratulations! You got it right.
You tried 3 questions and got 3 right.
```

Congratulations! You built a cool midterm study program!