Week 5 Practice Assessment

Due Jun 27 at 10am **Points** 6
Questions 7 **Time Limit** 120 Minutes
Allowed Attempts 2

Instructions

Coding under pressure can be scary, but like everything else we do at DBC, it's a skill that improves with practice. (Are you sensing a theme?) This practice assessment is designed as a no-stakes way for you to explore that feeling and equip yourself to handle it in the future.

You may not ask your cohortmates for help, but you can research online (including documentation, this curriculum itself, Stack Overflow, etc.) and refer back to any code you've written. If you get stuck, it's better to implement an alternative solution than none at all.

Good luck!

Take the Quiz Again

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	105 minutes	3 out of 6 *

* Some questions not yet graded

Score for this attempt: **3** out of 6 *
Submitted Jul 1 at 5:01pm
This attempt took 105 minutes.

Question 1 — 1 / 1 pts

Create a local repository for this project. (If you prefer to create one on GitHub and clone it, that's fine -- just don't push your code back up.) Create a file called `calculator.rb`.

While you won't be pushing code to GitHub, you are expected to use good Git workflow in this and any other assessment (something we can review by checking the history of your zipped repo). This is the only reminder you'll have to use Git workflow, so don't forget!

I was able to complete this task:

● True

○ False

Question 2

0 / 1 pts

Write a `calculate` method that takes three parameters: an integer, an operator string (example: "+"), and another integer. The method should execute the operation and return the result, an integer. The method should be able to handle the +, -, *, and / operations. For example, `calculate(4, '+', 5)` should return 9.

Write driver code that tests all four operations and prints the results.

I was able to complete this task:

Correct Answer ○ True

● False

Question 3

1 / 1 pts

Comment out your driver code and create a user interface for your program: Prompt the user for a calculation that the user would like to perform, and use your method to give the user a result. For instance, if the user types "4 + 5", the program should print a result of 9 and then exit. You can assume correct input on the user's part.

I was able to complete this task:

● True

○ False

Question 4

1 / 1 pts

Update your program to allow the user to do as many calculations as they want (so the user might enter 3 + 4, receive the answer, and then enter 7 - 1 as the next calculation, and so on). When the user types "done" instead of a calculation, the program can exit.

I was able to complete this task:

● True

○ False

Question 5

0 / 1 pts

When the user has finished performing calculations, but just before the program exits, print a count of the calculations performed, and a history of all the calculations that have been performed. The printout might look something like this:

```
3 calculations performed:
4 + 5 = 9
3 - 1 = 2
8 / 2 = 4
```

There are lots of ways you might do this, but your implementation should involve a data structure.

I was able to complete this task:

Correct Answer

○ True

● False

Question 6

0 / 1 pts

So far, we've assumed that the user will faithfully enter the right information. Update your program to handle invalid input, performing the calculation when possible and printing an error when the calculation could not be performed. **This is a deliberately broad prompt, and a bonus feature.** What sorts of mistakes might your user make? How can you accommodate those mistakes?

I was able to complete this release:

Correct Answer

○ True

● False