Northeastern University

CS 2100: Program Design and Implementation 1

Practice Quiz 1

Instructions

- Do not begin the quiz until instructed to do so.
- You may use both sides of a sheet of paper up to 8.5"x11" for reference, but no other resources, including phones, computers, AI, headphones, and ear pods.
- You have until the end of the class period to complete the quiz.
- Students may not leave the classroom during the first 10 minutes of the quiz (except in case of emergency).
- Hand your completed quiz to an instructor before leaving the room.
- Talk to an instructor if you need to leave the room and reenter.

Git

You're working on a Python calculator project with a teammate. You've just finished implementing a `divide()` function in `calculator.py`, and your tests pass locally. What commands are necessary to incorporate any changes that your teammate made until now, and then push your changes to the GitHub repo?

Functions and Documentation

Please write appropriate documentation for this function:

```
def find_median(numbers: Optional[list[Optional[float]]]) -> float:
    """









    """
    if numbers is None:
        raise ValueError("Cannot find median of None")
    if len(numbers) == 0:
        raise ValueError("Cannot find median of empty list")
    if None in numbers:
        raise ValueError("List contains None value")

    non_none_numbers: list[float] = [float(num) for num in numbers if num
    sorted_nums = sorted(non_none_numbers) # built-in sort function
    n = len(sorted_nums)

    if n % 2 == 1: # odd length
        return sorted_nums[n // 2]
    else: # even length
        mid1 = sorted_nums[n // 2 - 1]
        mid2 = sorted_nums[n // 2]
        return (mid1 + mid2) / 2.0
```

Unit Testing

Please write unit tests for `find_median()`. Make sure to test all three `ValueError`s and both length options (test function signatures are provided).

```
class TestFindMedian(unittest.TestCase):
    """Unit tests for find_median function"""

    def test_odd_length(self) -> None:




    def test_even_length(self) -> None:




    def test_empty_list(self) -> None:




    def test_none_input(self) -> None:




    def test_list_with_none(self) -> None:
```