Northeastern University

CS 2100: Program Design and Implementation 1

Practice Quiz 1

Instructions

- Please put all of your answers on the answer sheet. Only the answer sheet will be graded.
- 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 answer sheet to an instructor before leaving the room.
- Talk to an instructor if you need to leave the room and reenter.

Using objects

Here is a class that holds the data for an assignment, such as Homework 1:

```
class Assignment:
     """Stores data for an assignment that multiple students
     will do"""

     def init(self, name: str, due_time: str,
               starter_code: str, autograder: str):
          self.name = name
          self.due_time = due_time
          self.starter_code = starter_code
          self.autograder = autograder
          self.submissions: _____________ = []  # 1

def receive_submission(self, submission: Submission) -> None:
    """Store the submission in the list of submissions. If it
```

```
    is before the due time, run it through the autograder and
    store the score in the submission"""

    if submission.submission_time < self.due_time:
        score: float = self.run_autograder(submission.content)
        _____________ = score  # 2
    _____________.append(submission)  # 3
```

Here is a class that holds the data for a single student's submission for an assignment:

```
class Submission:
    """Stores data for a single assignment submission by a
    student"""

    def init(self, student: str, submission_time: str,
             content: str):
        self.student = student
        self.submission_time = submission_time
        self.content = content
        self.score: ____________ = None  # 4
```

Classes: constructors, methods, and attributes

Here is a `__str__()` method that goes inside the Submission class:

```
def __str__(self) -> str:
        """Return the name of the student, plus the
        submission time, to identify the submission.

        Example: "Joseph Aoun at 2025-01-20 14:34"
        """
        return ___________  # 5
```

Here is a method that goes inside the Assignment class:

```
def get_most_recent_graded_student_submission(self,
          student: str) -> Optional[Submission]:
    """Returns the most recent graded submission made by the
     given student, if there is one.
     If the student does not have any graded submissions,
     returns None"""


    to_return: Optional[Submission] = None
    for submission in self.submissions:
        if submission.student == student and __________:  # 6
            if ________ submission.submission_time > to_return.submission_time: # 7
                to_return = submission
    ____________  # 8
```

Unit testing

Let's test the two Assignment methods. Here are two test methods testing the get_most_recent_graded_student_submission() function in two cases: where there is a submission before and after the due date, and where all submissions are after the due date.

```
class TestAssignment(unittest.TestCase):

     def setUp(self) -> None:
          self.hw1 = Assignment('Homework 1',
               '2026-03-10 17:00', 'starter', 'autograder')

     def test_one_before_one_after_due_date(self) -> None:
          self.hw1.receive_submission(
               Submission('Stu', '2000-01-01 06:30', 'code'))
          self.hw1.receive_submission(
               Submission('Stu', '2030-01-01 06:30', 'code'))
          self.assertEqual('Stu at 2000-01-01 06:30', str(________))  # 9

     def test_all_after_due_date(self) -> None:
          self.hw1.receive_submission(
               Submission('Stu', '2040-01-01 06:30', 'code'))

          self.hw1.receive_submission(
               Submission('Stu', '2030-01-01 06:30', 'code'))
```

```
        self.assertEqual(_____________). # 10
```

11. Select the statement which is true:

    a. The submissions that were added to self.hw1 in the test function test_one_before_one_after_due_date() are still there in test_all_after_due_date().
    b. self.hw1 gets re-assigned a new Assignment value before each test function.
    c. If test_one_before_one_after_due_date() fails, then test_all_after_due_date() will not run.
    d. It is not possible to have two different Submission objects with the same student and submission time (if the student makes two submissions within the same minute).

12. Select the statement which is true:

    a. The two tests cover all possible edge cases for get_most_recent_graded_student_submission()
    b. The two tests cover all possible negative cases for get_most_recent_graded_student_submission()
    c. There are no edge cases tested.
    d. There are no negative cases tested.