Assignment 8

Due April 22, 2022

April 14, 2022

Instructions

For this assignment, you should write a Python script `question.py` that defines three classes representing exam questions. The first line of `question.py` should contain a comment stating:

> The code for this project represents my own, independent work. I have neither given nor received help on this assignment from other students.

Add another comment with your name after this comment. The rest of the file should define the Question class, then FillInTheBlank, then MultipleChoice (described in the following sections).

Question class

Class representing a question on an exam. Your Question class should keep track of 3 data members:

- `question`: a string containing the question (e.g., 'The Python keyword `_____` allows us to define functions and methods.')
- `points`: a nonnegative integer indicating how many points the question is worth if answered correctly

Your Question class should have 7 methods:

- a constructor: accepts two arguments: the question, and the number of points (in that order), which it uses to initialize the data members. If the number of points is negative, it should throw a `ValueError` with an appropriate message.
- getters and setters for the question and points (`get_question`, `set_question`, `get_points`, `set_points`)

- Function to print out the question using `print()`: Questions should be printed with the question text first, followed by `' (PTS points)'`, where you should replace `PTS` with the number of points the question is worth (e.g., 'The Python keyword `_____` allows us to define functions and methods. (10 points)')
- `score`: accepts a string argument containing the student's answer (defaults to empty string), and does nothing (`pass`).

FillInTheBlank class

The FillInTheBlank class represents a fill-in-the-blank question. It stores one data member (in addition to the two stored by all Questions):

- `answer`: a string containing the answer to the question (e.g., 'def')

FillInTheBlank objects have a constructor, and they redefine the `score` method of Questions:

- Constructor: accepts three arguments: the question, the number of points, and the answer (in that order), which it uses to initialize all data members
- `score`: accepts a string argument containing the student's answer (defaults to empty string). Returns the maximum number of points if the student's answer equals the answer to the question, or 0 points if it does not (no partial credit)
- All other Question methods (same behavior)

MultipleChoice class

The MultipleChoice class represents a multiple choice question. It stores two data members (in addition to the two stored by all Questions):

- `choices`: a list of strings representing the options to select for this question
- `answer`: a string representing the answer to the question (e.g., 'c')

MultipleChoice objects have a constructor, and they redefine the `score` and print methods of Questions:

- Constructor: accepts four arguments: the question, the number of points, the list of choices, and the answer (in that order), which it uses to initialize all data members. When initializing the choices, be sure to make a *deep copy* of the list given.
- `score`: accepts a string argument containing the student's answer (defaults to empty string). Returns the maximum number of points if the student's answer equals the answer to the question, or 0 points if it does not (no partial credit)

- Print method: MultipleChoice questions should be printed with the question text, followed by the number of points in parenthesis (like other questions), then a blank line, then all of the question choices, one per line. For example,

  ```
  Which of the following is *not* a pillar of object-oriented programming? (10 points)

  a) Polymorphism
  b) Modularity
  c) Aliasing
  d) Encapsulation
  e) Inheritance
  ```

  The Python code below will print the letter and parenthesis for `num_choices` choices, though you will need to modify it to actually print the choices (we discussed the `chr` function much earlier in the semester):

  ```
  for i in range(num_choices):
    print(f'{chr(97 + i))')
  ```

  You may assume that MultipleChoice questions will never have more than 20 choices.

- All other Question methods (same behavior)

Sample output

You have been provided with `exam.py`, a Python script that defines an Exam class, and a sample exam `pyexam.txt`. If you run `exam.py` and enter `pyexam.txt` as the file (and it is located in the same directory), this script will use your Question, FillInTheBlank, and MultipleChoice classes to create and administer a short quiz. You see the answers to the questions if you open up `pyexam.txt`, but you should be able to answer the questions without looking them up.

Submission

You should submit your implementation of the Question, FillInTheBlank, and MultipleChoice classes in `question.py`. Your code should be compatible with the posted `exam.py` and print and score all questions correctly.

Hints

You will want to write your own test cases in `question.py` (using `if __name__ == '__main__':`) before trying to integrate your code with `exam.py`. Start by implementing Question and FillInTheBlank before MultipleChoice.