**CPSC1012 Advanced Portfolio 3 – Files, List, and Data Class**

**Weight: 5% of your final mark**

**Trivia Game – Part 1 - Arrays**

Write a program named **AdvancedPortfolio03Part01-*YourFullName*** that plays a simple trivia game on a topic on your choice. The game will read questions from a CSV text file. Each question has a corresponding answer and point value between 1 and 3 based on the difficulty of the question. Implement the game using three lists. A list of type **String** should be used for the questions. Another list of type **String** should be used to store the answers. A list of type **int** should be used for the point values.

The index into the three lists can be used to tie the question, answer, and point value together. For example, the item at index 0 for each array would correspond to question 1, answer 1, and the point value for question 1. The item at index 1 for each array would correspond to question 2, answer 2, and the point value for question 2, and so forth. Manually create at least ten questions, answers, and point values and store it in a CSV text file. When the program starts read the questions from the CSV text file into your program.

Your program should ask the player each question one at a time and allow the player to enter an answer. The answer should not be case-sensitive. If the player’s answer matches the actual answer the player wins the number of points for that question. If the player’s answer is incorrect, the player wins no point for the question. Your program should show the correct answer if the player is incorrect. After the player has answered all questions, the game is over, and your program should display the player’s total score.

**Trivia Game – Part 2 – Data Class**

Write a program named **AdvancedPortfolio03Part02-*YourFullName*** that is a modification of the Trivia Game in Part 1 to use a single list instead of three lists. This can be accomplished by creating a **Trivia** object that encapsulates the question, answer, and the point value for a particular trivia question. Next, create a single list of **Trivia** objects instead of three separate lists for the question, answer, and point values. This change will make your program more scalable if there were ever additional properties to add to a **Trivia** object (you would not need to add another array for each property). Although the program has internally changed to a single list of objects, the execution of the program should be identical to before.

**Marking Guide**

|  |  |  |
| --- | --- | --- |
| **Description** | **Marks Possible** | **Marks Earned** |
| Correctness   * Displays all questions from CSV text file one at a time * Correct/incorrect answer are handled appropriately * Correct player total score * Part 2 Trivia class declaration includes properties for question, answer, and point value. * Part 2 uses a single list of Trivia objects | 5 |  |
| Structure   * Code to load CSV data into their corresponding lists * Code to iterate through list and display one question at a time * Code to track total points | 3 |  |
| Style and Readability   * Horizontal and vertical white space * Meaningful identifiers | 1 |  |
| Documentation   * Opening documentation * Source code comments | 1 |  |
| **Total:** | **10** |  |

**Coding Requirements**

The following coding standards must be followed when developing your program:

* Opening documentation at the beginning of the source file describing the **purpose**, **input**, **process**, **output, author, last modified date** of the program.
* Write only one statement per line.
* Write only one declaration per line.
* Use camelCase for local variable names and method parameter name.
* Use PascalCase for method names and constant variable names.
* If continuation lines are not indented automatically, indent them one tab stop (four spaces).
* Do NOT use the goto statement.
* There can only be one exit point for a loop, do not use the break statement inside a loop
* Do NOT use static variables.

**Demonstration and Submission Requirements**

* Demonstrate to your instructor your working program before submitting to Moodle. Be prepared to answer questions about your code after the demonstration. **No marks will be given** if you are unable to explain your code or if you submit your project without a demonstration of your working program to your instructor.