CS 1102 Unit 5 – Programming Assignment

In this assignment, you will again modify your Quiz program from the previous assignment. You will create an abstract class called "Question", modify "MultipleChoiceQuestion" to inherit from it, and add a new subclass of "Question" called "TrueFalseQuestion".

This assignment will again involve cutting and pasting from existing classes. Because you are learning new features each week, you are retroactively applying those new features. In a typical programming project, you would start with the full suite of features, enabling you to re-use code in ways natural to Java and avoid cutting and pasting.

First create the new "Question" abstract class.

- Open your "CS1102" project in Eclipse.
- Select "New" -> "Class" from the File menu.
- Enter the Name "Question".
- Check the box for the "abstract" modifier.
- Click "Finish".

The new file "Question.java" should appear in the editor pane. Make sure it is also listed under "(default package)" in the "Package Explorer" pane, along with "Quiz.java" and "MultipleChoiceQuestion.java".

- If you forgot to check the "abstract" box, add the modifier "abstract" to the "Question" class.
  ```
  public abstract class Question {
  ```

Add variables and methods to the "Question" class that will be inherited by the subclasses. Copy the following from the "MultipleChoiceQuestion" class and paste them into the "Question" class.

- The class variables "nQuestions" and "nCorrect".
- The instance variables "question" and "correctAnswer".
- The instance method "check".
- The class method "showResults".

Do not copy the constructor for "MultipleChoiceQuestion" or the instance method "ask".

The editor pane for "Question" should now show one error at the statement in "check" that calls the "ask" method.

- Add an abstract declaration for the "ask" method. This should be inside the "Question" class but outside all the method definitions.
  ```
  abstract String ask();
  ```

Note that this is a method declaration only, with no "{…}", because the method is abstract. It must be defined (implemented) in any concrete (non-abstract) subclasses of "Question".

The error warning in the editor pane should disappear. The "ask" call in "check" will use the methods defined in the subclasses of "Question".

Now modify the "MultipleChoiceQuestion" class to be a subclass of "Question".

- Delete from "MultipleChoiceQuestion" all the variables and methods that you pasted into "Question": "nQuestions", "nCorrect", "question", "correctAnswer", "check", and "showResults".
- Do not delete the constructor or the "ask" method.

The editor pane should show many errors, particularly in the constructor.

- Make "MultipleChoiceQuestion" a subclass of "Question" using the "extends" keyword.
  ```
  public class MultipleChoiceQuestion extends Question {
  ```

All the error warnings should disappear.

Convert "Quiz" to use "Question" variables with "MultipleChoiceQuestion" objects.

- Change the type of any "MultipleChoiceQuestion" variables in the main method of "Quiz" to "Question".
- But do not change the constructor calls used to initialize these variables. They should still be "MultipleChoiceQuestion".

Your program should still work using "Question" variables to reference "MultipleChoiceQuestion" objects. Test it to make sure.

Next add a new subclass of "Question" for true/false questions.

- Select "New" -> "Class" from the File menu.
- Enter the Name "TrueFalseQuestion".
- Enter the Superclass "Question".
- Click "Finish".

The new file "TrueFalseQuestion.java" should appear in the editor pane.

- If you did not add the Superclass name, you will need to add "extends Question" to the class declaration.
  ```
  public class TrueFalseQuestion extends Question {
  ```

The IDE may have already added an empty "ask" method because of the abstract method it found in "Question". It may have also added the annotation "@Override", which is optional and instructs the compiler to make sure the method actually does override a method in a parent class.

Add an implementation of the "ask" method that is specific to true/false questions.

- Import the "JOptionPane" package.
- Like in "ask" for "MultipleChoiceQuestion", have the new "ask" pose the "question" String repeatedly until the user provides a valid answer.

- In this case, valid answers are: "f", "false", "False", "FALSE", "n", "no", "No", "NO", "t", "true", "T", "True", "TRUE", "y", "yes", "Y", "Yes", "YES".
- When users provide an invalid answer, display the message, "Invalid answer. Please enter TRUE or FALSE."
- Convert any valid answer representing true or yes to "TRUE" and any valid answer representing false or no to "FALSE".
- Hint: Convert all answers to upper case before checking their validity.

Add a "TrueFalseQuestion" constructor to initialize the "question" and "correctAnswer" Strings inherited from "Question".

- Give the constructor two String parameters, "question" and "answer".
- Use "this" to set the instance variable "question" using the parameter "question". Add the text, "TRUE or FALSE: " to the beginning of the question.
  ```
  this.question = "TRUE or FALSE: "+question;
  ```
  (You could also use "super.question", since the instance variable is in "Question". Either works because "TrueFalseQuestion" does not hide the instance variable in "Question" with its own instance variable named "question".)
- Set the instance variable "correctAnswer" to only "TRUE" or "FALSE" based on "answer". Allow the parameter "answer" to be any String that is considered valid by "ask".

Now add a true/false question to your quiz!

- Initialize a "Question" variable using a "TrueFalseQuestion" constructor.
  ```
  Question question = new TrueFalseQuestion(…);
  ```

Run your program and test that the following are true.

- It accepts only valid answers.
- It responds appropriately to correct and incorrect answers.
- It provides accurate counts of correct answers and total questions.

Finally, add at least four more true/false questions, for a total of at least five multiple-choice questions and five true/false questions. Test your program again.