Computing and Information Systems/Creative Computing
University of London International Programmes
CO2220 Graphical Object-Oriented and Internet programming in Java
Coursework assignment 1 2016–17

Generic Quiz Classes

GenericQuestion.java

```
public class GenericQuestion<T> {
     private String question;
     private T answer;

     public GenericQuestion(String question, T answer) {
          this.question = question;
          this.answer = answer;
     }

     public String getQuestion() {
          return question;
     }

     public T getAnswer() {
          return answer;
     }
}
```

GenericFreeQuizQuestion.java

```
public class GenericFreeQuizQuestion extends GenericQuestion<String> {
     public GenericFreeQuizQuestion(String question, String answer) {
          super(question, answer);
     }
}
```

GenericBaseQuiz.java

```
import java.util.ArrayList;
import java.util.List;
import java.util.Random;
import java.util.Scanner;

public abstract class GenericBaseQuiz<T extends GenericQuestion<?>> {
     private Scanner in;

     public GenericBaseQuiz() {
          in = new Scanner(System.in);
     }

     public void quiz() {
          int numberOfQuestions = getQuizSize();
          List<T> quizQuestions = getRandomQuestions(getQuestions(), numberOfQuestions);

          printGreeting(numberOfQuestions);
          List<Boolean> answers = askQuestions(quizQuestions);

          printResults(answers);
     }

     protected void printGreeting(int numberOfQuestions) {
          System.out.format("This quiz has %d questions. Good luck.%n", numberOfQuestions);
     }

     //this method was static in the non-generic version of this class.
     private List<T> getRandomQuestions(List<T> questions, int amount) {
          if (amount > questions.size()) {
               throw new RuntimeException("A Quiz cannot have more random questions than there are
                    questions to choose from (" + amount + " > " + questions.size() + ")");
          }
          Random r = new Random();
```

```
     List<T> quizQuestions = new ArrayList<>();

     //get random questions, making sure there are no duplicates
     T q;
     int index;
     while (quizQuestions.size() != amount) {
          index = r.nextInt(questions.size());
          q = questions.get(index);
          if(!quizQuestions.contains(q)) {
               quizQuestions.add(q);
          }
     }

     return quizQuestions;
}

private List<Boolean> askQuestions(List<T> theQuestions) {
     List<Boolean> answers = new ArrayList<Boolean>();
     for (T q : theQuestions) {
          boolean correct = getAnswerFromUser(q);
          if (correct) {
               answers.add(true);
          } else {
               answers.add(false);
          }
     }

     return answers;
}
```

```
private boolean getAnswerFromUser(T question) {
     String prefix = getQuestionPrefix(question);
     String suffix = getQuestionSuffix(question);
     System.out.print(prefix + " " + question.getQuestion() + " " + suffix);

     //make sure we receive a valid answer
     String answer;
     boolean answered;
     do {
          answer = in.nextLine();
          answered = isValidAnswer(question, answer);
          if (!answered) {
               System.out.print(invalidInput(question) + " ");
          }
     } while (!answered);

     return isCorrectAnswer(question, answer);
}

private void printResults(List<Boolean> answers) {
     int correct = howManyCorrect(answers);
     int total = answers.size();
     double percentage = correct*(100f/total);
     String msg = encouragingMessage(percentage);

     String output = String.format("You scored %d/%d. That's %.0f%%. %s", correct, total,
         percentage, msg);
     System.out.println(output);
}
```

```
    protected String encouragingMessage(double percentage) {
         if (percentage < 0) return "How have you done that?!";
         if (percentage == 0) return "Well, it'll be hard to do much worse.";
         if (percentage > 0 && percentage <= 20) return "Don't give up.";
         if (percentage > 20 && percentage <= 40) return "You'll have better days.";
         if (percentage > 40 && percentage <= 60) return "Nice work.";
         if (percentage > 60 && percentage <= 80) return "Very good.";
         return "Excellent!";
    }

    private static int howManyCorrect(List<Boolean> answers) {
         int number = 0;
         for (Boolean b : answers) {
              if (b) {
                   number++;
              }
         }
         return number;
    }

    protected abstract boolean isValidAnswer(T question, String answer);
     //can the user's input be interpreted as an attempt to answer the question?
    protected abstract boolean isCorrectAnswer(T question, String answer);
     //did the user answer the question correctly?
    protected abstract String invalidInput(T question);
     //what to say to the user if you don't know what to do with their input
    protected abstract int getQuizSize(); //the number of questions to ask per quiz
    protected abstract List<T> getQuestions(); //all of the questions
    protected abstract String getQuestionPrefix(T question);
     //Text to print before the question is printed to the screen
    protected abstract String getQuestionSuffix(T question);
     //Text to print after the question is printed to the screen
}
```

GenericFreeQuiz.java

```
import java.util.ArrayList;
import java.util.List;

public class GenericFreeQuiz extends GenericBaseQuiz<GenericFreeQuizQuestion> {

     private List<GenericFreeQuizQuestion> allQuestions;


     public GenericFreeQuiz() {
          super();
          allQuestions = populateList();
     }

     //What to print if the user has given us a string that we can't use as an answer
     protected String invalidInput(GenericFreeQuizQuestion question) {
          return "Your answer cannot be blank. Try again.";
     }

     //The number of questions to ask in the quiz
     protected int getQuizSize() {
          return 5;
     }

     //What to show just before the question
     protected String getQuestionPrefix(GenericFreeQuizQuestion question) {
          return "QUESTION:";
     }

     //What to show after the question. We don't show anything as you can see.
     protected String getQuestionSuffix(GenericFreeQuizQuestion question) {
          return "";
     }
```

```
//How to detect an invalid answer
protected boolean isValidAnswer(GenericFreeQuizQuestion question, String answer) {
     return answer.length() > 0;
}

//Did the user answer the question correctly?
protected boolean isCorrectAnswer(GenericFreeQuizQuestion question, String answer) {
     return question.getAnswer().equalsIgnoreCase(answer.trim());
}

protected List<GenericFreeQuizQuestion> getQuestions() {
     return allQuestions;
}

//Return the full list of questions
protected List<GenericFreeQuizQuestion> populateList() {
     List<GenericFreeQuizQuestion> list = new ArrayList<>();

     GenericFreeQuizQuestion q1 = new GenericFreeQuizQuestion("Name a primitive data type that
       starts with 'i'", "int");
     GenericFreeQuizQuestion q2 = new GenericFreeQuizQuestion("Name a primitive data type that
       starts with 'c'", "char");
     GenericFreeQuizQuestion q3 = new GenericFreeQuizQuestion("What is the value of the
       expression \"hello\".charAt(0) ?", "h");
     GenericFreeQuizQuestion q4 = new GenericFreeQuizQuestion("What is the value of the
       expression \"James\".charAt(\"James\".length() - 1) ?", "s");
     GenericFreeQuizQuestion q5 = new GenericFreeQuizQuestion("Type in the number of stars
       (asterisks) output by for(int i=10;i>0;i=i-1) System.out.print(\"*\");", "**********");
     GenericFreeQuizQuestion q6 = new GenericFreeQuizQuestion("Type in the number of stars
       (asterisks) output by for(int i=0;i<10;i=i+3) System.out.print(\"*\");", "****");
     GenericFreeQuizQuestion q7 = new GenericFreeQuizQuestion("Name a primitive data type that
       starts with 'b'", "boolean");
```

```
        list.add(q1);
        list.add(q2);
        list.add(q3);
        list.add(q4);
        list.add(q5);
        list.add(q6);
        list.add(q7);

        return list;
    }

    public static void main(String[] args) {
        GenericFreeQuiz quiz = new GenericFreeQuiz();
        quiz.quiz();
    }
}
```