University of London

Computing and Information Systems/Creative Computing

CO2220 Graphical object-oriented and internet programming in Java

Coursework assignment 1 2020–2021

Introduction

This is coursework assignment 1 (of two coursework assignments in total) for 2020–2021. Part A asks that you demonstrate an understanding of simple graphics, inner classes, events and the *ActionListener* interface. Part B looks at the object-oriented programming paradigm plus getting our classes to work together to produce a desired result.

IMPORTANT NOTE: Please use at least Java 8 to compile and test your programs. Later versions of Java are also fine.

Electronic files you should have

Part A

Java file

- *RandomShapeGUI.java*

Part B

Java files

- *MultipleChoiceQuestionParser.java*
- *PopQuiz.java*
- *PopQuizFileManager.java*
- *PopQuizUserInterface.java*
- *Question.java*
- *QuestionParser.java*
- *TextQuestion.java*
- *TextQuestionParser.java*
- *TimeFormatter.java*
- *TimeKeeper.java*
- *TrueFalseQuestion.java*
- *TrueFalseQuestionParser.java*

Text files

- *MultipleChoiceQuestions.popquiz*
- *TextQuestions.popquiz*
- *TrueFalseQuestions.popquiz*

What you should hand in: very important

At the end of each section there is a list of files to be handed in. The examiners wish to read your Java code, so class files **will not** be marked. Any student giving in only a class file will not receive any marks for that part of the coursework assignment, so **please be careful about what you upload as you could fail if you submit incorrectly**.

- Please only hand in the files asked for, and not any additional files.
- Please put your name and student number as a comment at the top of each Java file that you hand in.

Marks for submitting files as requested
There is one mark allocated for handing in uncompressed files – that is, students who hand in zipped or .tar files or any other form of compressed files can only achieve 99/100 marks.

There is one mark allocated for handing in just the files asked for without putting them in a directory; students who upload their files in a directory can only achieve 99/100 marks.

There is one mark for naming any Java files and classes that you are asked to amend or write **exactly** as you have been asked to name them. This means that Java files should have the file name that the assignment requests and should have a class name that matches the file name. For example, a file called *RandomShapeGUI.java*, should contain a public class called *RandomShapeGUI.* As Java is case-sensitive, *Randomshapegui* **is not** the same name as *RandomShapeGUI;* please be exact.

Please also note that where students add identifying information to their Java files so that the class name and the file name differ, the file will not compile, for example:

- *JSmith_ RandomShapeGUI.java*
- *cwk1- RandomShapeGUI.java*
- *JSmith-CO2220-assignment1- RandomShapeGUI.java*

No marks for programs that will not compile
The examiners will compile and run your Java programs; for this reason, programs that do not compile will not receive any marks.

If your file does not compile **for any reason** (except file name/class name mismatch) you will receive no marks for that part of the coursework assignment. In particular, files that contain Java classes that cannot be compiled because they are the wrong type (e.g. PDFs), will not be given any marks for that part of the question.

Java version
Part B uses a class introduced in Java 8, so please use at least Java 8 to compile, run and test your work. Later versions of Java are also fine.

Coursework assignment 1

Readable code

This coursework assignment will highlight the advice given by Robert C. Martin in his book *Clean Code: A Handbook of Agile Software Craftsmanship.* (Stoughton, MA: Prentice Hall, 2008) [ISBN 9780132350884]. Martin (2008) describes a system for writing readable code. The key points are summarised below. Please note, there are other systems, but this assignment will focus on his recommendations.

Martin writes:

> *'One difference between a smart programmer and a professional programmer is that the professional understands that clarity is king. [...] We want to use the popular paperback model whereby the author is responsible for making himself clear and not the academic model where it is the scholar's job to dig the meaning out of the paper.'*

Martin writes that '*making your code readable is as important as making it executable*'. He believes that names of variables, methods and classes are a major part of what makes code readable:

> *'The name of a variable, function or class should answer all the big questions. It should tell you why it exists, what it does, and how it is used. If a name requires a comment, then the name does not reveal its intent.'*

Martin dislikes comments, noting that as code is updated, comments are rarely updated at the same time. Thus however helpful a comment may be at the start, once a class has been in use for a while, any comments are likely to be outdated and confusing. He believes that code should be written with names that make the intent clear, such that comments are redundant.

See the **Appendix 1** for an example of renaming a simple class to make it more readable.

When answering questions in this coursework assignment, please remember the following rules from Martin:

Formatting

> *'You should take care that your code is nicely formatted. You should choose a set of simple rules that govern the layout of your code, and then you should consistently apply those rules. […] It helps to have an automated tool that can apply those formatting rules for you.'*

Methods

- **Method should do one thing only**. If your method does more than one thing, break it up into separate methods.
- **Do not repeat yourself**. If you find yourself writing the same code more than once, put it into a method.
- **Too many arguments (parameters)**. '*No argument is best, followed by one, two and three. More than three is very questionable and should be avoided with prejudice'.*

Comments

If you do write a comment, make sure it is grammatical, short, does not state the obvious, and is really needed.

Names

- **Choose descriptive names**. '*Names in software are 90 per cent of what makes* *software readable*.'

- **Unambiguous names**. '*Choose names that make the workings of a function or variable unambiguous*.'
- **Names should describe side effects**. For example, a method getOos() that will make an `ObjectOuputStream` if one does not already exist should be called createOrReturnOos()

General

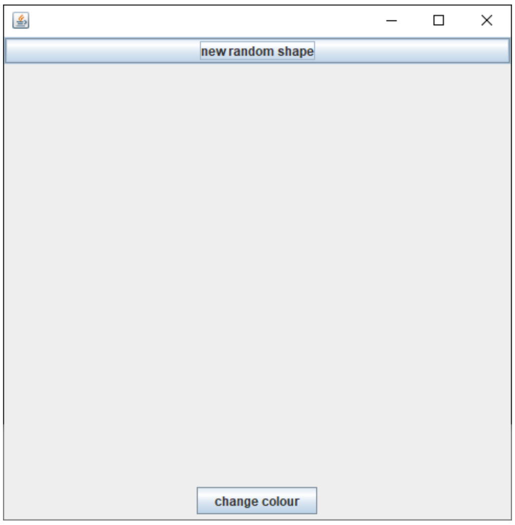
- **Obscured intent** – make the code as expressive as possible such that its intention is clear from a first reading.
- **Put conditional statements into a method to make their intention and effect clear**, for example:

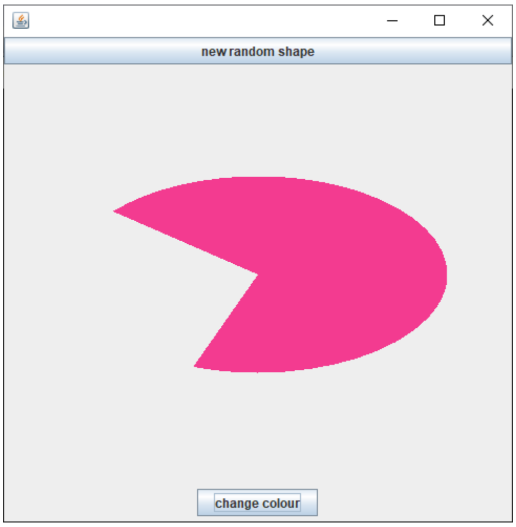
  BAD `if (guessedWord.length()< shortestLength)`
  GOOD `if(guessedWordIsTooShort(guessedWord))`

Part A

Consider the *RandomShapeGUI* class. Within the class a `JFrame` variable *frame* is created in the *go()* method. An inner class *RandomShapeDrawPanel* has extended `JPanel` in order to make a draw panel for graphics. This draw panel has been added to the frame by the statement `frame.getContentPane().add(BorderLayout.CENTER, drawPanel);` in the *go()* method. The *go()* method has also added buttons to the frame such that the user can interact with the graphics on the draw panel.

Compile and run the *RandomShapeGUI* class. You should see a `JFrame` with two buttons. If you press the top button ('new random shape') a shape should appear on the draw panel. The shape is randomly chosen from an oval, rectangle and arc. The colour and the dimensions of the shape are also generated randomly. If you press the bottom button ('change colour') the colour of the shape should change. You can press the *change colour* button as often as you wish, and with each press, the colour of the currently displayed shape should change. You can also press the new random shape button as often as you wish, and with each press a new shape with random colour and random dimensions is generated and shown on the draw panel. Note that the shape chosen may be the same shape as the one it replaces since there are only three choices. The colour and dimensions of the new shape should vary, although it is theoretically possible that exactly the same colour and dimensions are generated.





Note that in the *go()* method a `JPanel` instance (called *buttonPanel)* has been created to hold the *change colour* button using *FlowLayout*. *FlowLayout* is used because it will allow us to have two buttons side by side (see question 3). The *buttonPanel* has then been added to the bottom of the frame using *BorderLayout*.

Please complete the following tasks:

1. Add a new `JButton` to the *RandomShapeGUI* class. This button should have the text 'change dimensions' on it.

   [5 marks]

2. The *RandomShapeGUI* class already has three inner classes: *ShapeListener*; *ColorListener*; and *RandomShapeDrawPanel*. Write another inner class to implement the `ActionListener` interface and listen to the change dimensions button.

   [5 marks]

3. Add the *change dimensions* button to the *buttonPanel*.

   [5 marks]

4. Make any necessary changes to your new inner class and the *RandomShapeDrawPanel* inner class such that when the *change dimensions* button is pressed, the dimensions of the current shape on the draw panel change randomly. Ensure the user can repeatedly press the button for new random dimensions. The type of shape and the colour should remain the same, only the dimensions should change.

   [15 marks]

5. You may find that sometimes when the new random shape button is pressed, the new shape is difficult or impossible to see. Similarly, when testing your new *change dimensions* button the shape may disappear or be too small to make out clearly. This is because it is possible for the size of the shape to be too small for the user to be able to see it. Change the code that generates random dimensions so that the shape displayed on the draw panel will have minimum dimensions such that it will always be visible to the user.

   [5 marks]

6. In answering the above questions, you will have to give names to your new inner class, to any methods you write and to any new variables. Please follow the advice given by Martin (2008). See the section titled **Readable code** above. See **Appendix 1** for an example of renaming methods and variables to increase the readability of code.

   [5 marks]

Make sure that you read the *RandomRectangleGUI* class carefully before you start work.

Reading for Part A

- Sections 11.1–11.6 of Volume 1 of the subject guide (events).
- Section 12.1 of Volume 1 of the subject guide (graphics).
- The following pages from Chapter 12 of Sierra, K. and B. Bates *Head First Java* (Sebastopol, CA: O'Reilly, 2005) 2nd edition [ISBN 9780596009205]:
  - pp.353–356 simple GUIs
  - pp.357–362 events and listeners
  - pp.363–368 graphics
  - pp.369–374 listening for events
  - pp.375–381 listening for events with inner classes.

Deliverable for Part A

Remember to put your name and student number as a comment at the top of your Java file. You should submit only the file listed below.

- An electronic copy of your revised program: *RandomShapeGUI.java*

Part B

You have been given some Java files together with three text files that together make up a system that is intended to allow a user to generate quizzes with questions randomly chosen from the text files. Users are given a mark after taking a quiz and can choose to save the quiz just taken so that it can be reloaded and taken again, either in the current session, or in a later one.

The three text files each contain a different type of question:

MultipleChoiceQuestions.popquiz contains multiple choice questions, as the name suggests. Possible answers are numbered, and the user enters one of the numbers as their answer.

TextQuestions.popquiz contains questions where the answer is a `String` with no spaces, for example, the answer to *Name a primitive data type that starts with 'i'* would be `int`.

TrueFalseQuestions.popquiz contains statements with the user asked to state if they are *true* or *false*. The user answers the question by entering either *true* or *false*, in upper or lower case, or a mix.

The system you have been given is run through the *PopQuizUserInterface* class and is not complete, but if it were it would work as follows.

On first use, the user is presented with a menu:

```
******* POPQUIZ *******
Choose one of the following options:
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice:
```

If the user chooses option 2, they are presented with a new menu:

```
******* LOAD/SAVE *******
0. Back to main menu
```

```
1. Load quiz from disk
2. Show quizzes saved on disk
Enter your choice:
```

Choosing option 2 from the *LOAD/SAVE* menu gives:

```
Enter your choice: 2

No saved quizzes.
```

At this point the only sensible thing to do is to choose option 0 and return to the *POPQUIZ* menu.

From the *POPQUIZ* menu, choosing option 1 means that the user is asked the following:

```
There are 87 questions available. How many questions should
the quiz have?
```

Once the user enters a valid number, a quiz is generated by choosing randomly from the questions available. Once the user has given their answers, they are told how many questions they got right, and the *POPQUIZ* menu reappears with a new option: `0. Retake the previous quiz`

```
******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
```

If the user chooses option 2 then the *LOAD/SAVE* menu appears as follows:

```
******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice:
```

Note that the menu now has an extra option: `3. Save current quiz to disk`

Choosing option 3 from the *LOAD/SAVE* menu means that the user will be asked to give a name for the file in which the quiz they have just taken will be saved. Once the user has entered their file name, they are returned to the *POPQUIZ* menu. Assuming that the user chose the name *quiz1* the input and output would be as follows:

```
Enter your choice: 3

NOTE: The filename will end with .pqs whether you specify it
or not!
Enter filename: quiz1
Quiz saved.

******* POPQUIZ *******
Choose one of the following options:
```

```
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice:
```

If the user enters option 2, they will be returned to the *LOAD/SAVE* menu:

```
******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice:
```

Choosing option 2 from the *LOAD/SAVE* menu will give the following output:

```
quiz1.pqs
```

If the user then chooses option 1 from the *LOAD/SAVE* menu they are asked for a file name, an incorrect name is rejected, and a correct file name means that the quiz found in the file will be run as follows:

```
******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 1

Enter filename: zzz
Quiz not found!

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 1

Enter filename: quiz1.pqs
Retaking saved quiz quiz1.pqs
Hope it goes well!
//quiz follows
```

After the user has completed the quiz, the *POPQUIZ* menu reappears.

The test input / output discussed above has been somewhat shortened, see Appendix 2 for the entire input and output.

Answer the following questions in order to complete the system and run quizzes:

1. The system is missing the *MultipleChoiceQuestion* class, meaning that it cannot work as intended. Write the missing class.

   [9 marks]

2. Consider the *PopQuizUserInterface* class. The class contains methods called *method1(), method2(), method3(), method4(), method5()* and *method6()*. Read each method and rename it such that the name tells the reader about the intent of the method, without the need for a comment to explain the method any further.

   Where there are local variables in methods *method1(), method2(), method3(), method4(), method5()* and *method6()* that do not have descriptive names, rename them to something that tells you the purpose of the variable. For example in the statement `String x = getFilename();` renaming `x` to `filename` would tell you the purpose of the local `String` variable.

   When one of the six methods named above invokes another method to complete a task, and the method name does not explain what that task is, then please rename that method too.

   Please check you are following the advice given by Martin (2008) in the section above titled **Readable code**. See **Appendix 1** for an example of renaming methods and variables to increase the readability of code.

   [21 marks]

3. The *PopQuizUserInterface* class contains a method, *run()*, that runs the user interaction loop. The class will not work as it should, partly because one of the methods called by the *run()* method, *executeMainMenuChoice()*, has some missing statements. Complete the method. You will lose marks for this question if you write code within your *executeMainMenuChoice()* method to do a task that can be done by invoking a method that is already in the *PopQuizUserInterface* class.

   [9 marks]

4. The *PopQuizUserInterface* class will not work as it should, partly because one of the methods called by *method3(), executeLoadAndSaveMenuChoice()*, has missing statements. Complete the method. You will lose marks for this question if you write code within your *executeLoadAndSaveMenuChoice()* method to do a task that can be done by invoking a method that is already in the *PopQuizUserInterface* class.

   [9 marks]

5. Once you have completed questions 1–4, test the *PopQuizUserInterface* class to ensure that it works as intended. Test by making sure that your completed system can replicate the output given in **Appendix 2**, given the same input. This is excepting the quiz questions which should be different since they are randomly generated. All other input and output should be the same. Save your test run of the *PopQuizUserInterface* class into a PDF with a comment about the success of the testing. Your test run does not have to be perfect, provided that you identify any issues that you discovered in your testing you will receive full marks.

Please do not write more than 100 words for each issue found in your testing. If your testing demonstrates that the system works as expected you should write one comment of no more than 100 words.

[9 marks]

Reading for Part B
The following sections from Volume 1 of the subject guide:

- Chapter 2, sections 2.1, 2.2 and 2.3 (Inheritance and instance variables).
- Chapter 5 section 5.2 (Methods and instance variables) and section 5.4 (Local and instance variables).
- Chapter 8, sections 8.1–8.4.2 (Inheritance).
- Section 9.2 (Abstract Classes) and 9.3 (Abstract Methods). *Note that the Question class is abstract.*
- Sections 10.3, 10.4 and 10.5 (Constructors, including the *this* keyword and superclass constructors).

Deliverables for Part B
Remember to put your name and student number as a comment at the top of your Java file. You should submit only the files listed below:

- *MultipleChoiceQuestion.java*
- *PopQuizUserInterface.java*
- *A PDF with your answer to Question 5* (you must follow the generic University of London instructions for naming this file: YourName_SRN_COxxxxcw#.pdf).

Marks for CO2220 coursework assignment 1

The marks for each section of coursework assignment 1 are clearly displayed against each question and add up to 97. There are another 3 marks available for giving in uncompressed .java files, for giving in files that are not contained in a directory, and for giving in files with the correct names, and a matching class name. This amounts to 100 marks altogether. There are another 100 marks available from coursework assignment 2.

Total marks for Part A:	**[40 marks]**
Total marks for Part B:	**[57 marks]**
Mark for giving in uncompressed files:	**[1 mark]**
Mark for giving in standalone files; namely, files **not** enclosed in a directory:	**[1 mark]**
Mark for giving in Java files that have the file name that the assignment requests, and that have a class name that matches the file name. For example a file called *MyClass.java*, should contain a public class called *MyClass*:	**[1 mark]**
Total marks for coursework assignment 1:	**[100 marks]**

[END OF COURSEWORK ASSIGNMENT 1]

Appendix 1

A simple example of renaming methods and variables for greater readability.

Original

```
public class Calculator{

      public static void calc(int x){
            if (x >= 70){
                  System.out.println("grade = A");
                  return;
            }
            if (x >= 60){
                  System.out.println("grade = B");
                  return;
            }
            if (x >= 50){
                  System.out.println("grade = C");
                  return;
            }
            if (x >= 40){
                  System.out.println("grade = D");
                  return;
            }
            if (x<40) System.out.println("grade = F");
      }

      public static void main(String[] args) {
            calc(90);
            calc(53);
            calc(30);
      }
}
```

Renamed

```
public class GradeCalculator {

      public static void calculateAndPrintGrade(int finalMark){
            if (finalMark >= 70){
                  System.out.println("grade = A");
                  return;
            }
            if (finalMark >= 60){
                  System.out.println("grade = B");
                  return;
            }
            if (finalMark >= 50){
                  System.out.println("grade = C");
                  return;
            }
            if (finalMark >= 40){
                  System.out.println("grade = D");
                  return;
            }
            if (finalMark<40) System.out.println("grade = F");
      }

      public static void main(String[] args) {
            calculateAndPrintGrade(90);
            calculateAndPrintGrade(53);
            calculateAndPrintGrade(30);
      }
}
```

Appendix 2

```
****** POPQUIZ *******
Choose one of the following options:
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
Enter your choice: 4

I don't know how to do that.

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
Enter your choice: 3

You must take a quiz before you can save it!

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
Enter your choice: 2

No saved quizzes.

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
Enter your choice: 0

******* POPQUIZ *******
Choose one of the following options:
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 4

I don't know how to do that.

******* POPQUIZ *******
Choose one of the following options:
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 0
```

```
You must take a quiz before you can retake it!

******* POPQUIZ *******
Choose one of the following options:
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 1

There are 87 questions available. How many questions should the quiz
have? 3

You have selected a quiz with 3 questions. Good luck!

Question 1:
TRUE or FALSE: A class can have up to three constructors
Your answer: false

Question 2:
What will be the output of the following statement if t is a valid
int variable with the value 3: if (t==0 && t==3)
System.out.println("hello");
1 The statement will output hello
2 No output - the statement will cause a run time error
3 No output - the statement will cause a compilation error
4 No output but no errors
5 None of the above.
Your answer: 1

Question 3:
Why will this main method cause a compilation error: public static
void main(String[] args){int s = System.in.read();}
1 Because the int s = System.in.read(); statement can throw an
exception
2 Because the int s = System.in.read(); statement can throw an
exception, and this must be thrown or caught
3 because the words 'throws Exception' must always be included in
the main declaration (ie public static void main(String[] args)
throws Exception)
4 None of the above
Your answer: 1

You scored 1/3 and took 31 seconds to complete the quiz. That's
33.33%. You did OK!

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
```

```
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 3

NOTE: The filename will end with .pqs whether you specify it or not!
Enter filename: quiz1
Quiz saved.

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 2

quiz1.pqs

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 1

Enter filename: zzz
Quiz not found!

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 1

Enter filename: quiz1.pqs
Retaking saved quiz quiz1.pqs
Hope it goes well!
Question 1:
TRUE or FALSE: A class can have up to three constructors
Your answer: false

Question 2:
What will be the output of the following statement if t is a valid
int variable with the value 3: if (t==0 && t==3)
System.out.println("hello");
1 The statement will output hello
2 No output - the statement will cause a run time error
3 No output - the statement will cause a compilation error
```

```
4 No output but no errors
5 None of the above.
Your answer: 1

Question 3:
Why will this main method cause a compilation error: public static
void main(String[] args){int s = System.in.read();}
1 Because the int s = System.in.read(); statement can throw an
exception
2 Because the int s = System.in.read(); statement can throw an
exception, and this must be thrown or caught
3 because the words 'throws Exception' must always be included in
the main declaration (ie public static void main(String[] args)
throws Exception)
4 None of the above
Your answer: 2

You scored 2/3 and took 23 seconds to complete the quiz. That's
66.67%. Very good!

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 0

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 1

There are 87 questions available. How many questions should the quiz
have? 2

You have selected a quiz with 2 questions. Good luck!

Question 1:
The following statement will not compile: if 3/1.2 < 0
System.out.print("error");
1 Because the boolean conditional is invalid as an int is being
divided by a double
2 Because the boolean conditional is invalid as an int is being
divided by a float
3 Because the boolean conditional should be in round brackets
```

```
4 All of the above
Your answer: 2

Question 2:
Which of the following are primitive variables
1 Boolean
2 String
3 Integer
4 None of the above
Your answer: 4

You scored 1/2 and took 21 seconds to complete the quiz. That's 50%.
Nice job!

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 0


You have selected your previously taken quiz. Best of luck!

Question 1:
The following statement will not compile: if 3/1.2 < 0
System.out.print("error");
1 Because the boolean conditional is invalid as an int is being
divided by a double
2 Because the boolean conditional is invalid as an int is being
divided by a float
3 Because the boolean conditional should be in round brackets
4 All of the above
Your answer: 3

Question 2:
Which of the following are primitive variables
1 Boolean
2 String
3 Integer
4 None of the above
Your answer: 4

You scored 2/2 and took 10 seconds to complete the quiz. That's
100%. Perfect!

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
```

```
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 3

NOTE: The filename will end with .pqs whether you specify it or not!
Enter filename: quiz2
Quiz saved.

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 2

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 2

quiz1.pqs
quiz2.pqs

******* LOAD/SAVE *******
0. Back to main menu
1. Load quiz from disk
2. Show quizzes saved on disk
3. Save current quiz to disk
Enter your choice: 0

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 1

There are 87 questions available. How many questions should the quiz
have? 90
The number you have chosen is more than the total number of
questions!
Please enter a number no higher than 87: -5
The number must be greater than 0
Please enter a number no higher than 87: 1

You have selected a quiz with 1 question. Good luck!

Question 1:
TRUE or FALSE: A class can have arbitrarily many constructors
Your answer: true
```

```
You scored 1/1 and took 6 seconds to complete the quiz. That's 100%.
Perfect!

******* POPQUIZ *******
Choose one of the following options:
0. Retake the previous quiz
1. Take a new quiz
2. Load or save a quiz
3. Quit
Enter your choice: 3

Bye!
```

Summary of input

Input	Menu	Purpose
2	POPQUIZ (main menu)	Menu choice
4	LOAD/SAVE	Menu choice
3	LOAD/SAVE	Menu choice
2	LOAD/SAVE	Menu choice
0	LOAD/SAVE	Menu choice
4	POPQUIZ (main menu)	Menu choice
0	POPQUIZ (main menu)	Menu choice
1	POPQUIZ (main menu)	Menu choice
3		In answer to: *How many questions should the quiz have?*
false		Answer given to a quiz question
1		Answer given to a quiz question
1		Answer given to a quiz question
2	POPQUIZ (main menu)	Menu choice
3	LOAD/SAVE	Menu choice
quiz1		In answer to: *Enter filename:*
2	POPQUIZ (main menu)	Menu choice
2	LOAD/SAVE	Menu choice
1	LOAD/SAVE	Menu choice
zzz		In answer to: *Enter filename:*
1	LOAD/SAVE	Menu choice
quiz1.pqs		In answer to: *Enter filename:*
false		Answer given to a quiz question
1		Answer given to a quiz question
2	POPQUIZ (main menu)	Answer given to a quiz question
2		Menu choice
0	LOAD/SAVE	Menu choice
1	POPQUIZ (main menu)	Menu choice
2		In answer to: *How many questions should the quiz have?*
2		Answer given to a quiz question
4		Answer given to a quiz question
0	POPQUIZ (main menu)	Menu choice
3		Answer given to a quiz question
4		Answer given to a quiz question
2	POPQUIZ (main menu)	Menu choice
3	LOAD/SAVE	Menu choice

Input	**Menu**	**Purpose**
quiz2		In answer to: *Enter filename:*
2	POPQUIZ (main menu)	Menu choice
2	LOAD/SAVE	Menu choice
0	LOAD/SAVE	Menu choice
1	POPQUIZ (main menu)	Menu choice
90		In answer to: *How many questions should the quiz have?*
-5		In answer to: *How many questions should the quiz have?*
1		In answer to: *How many questions should the quiz have?*
true		Answer given to a quiz question
3	POPQUIZ (main menu)	Menu choice