```java
package il.ac.hit.quizzy;

public class Main {
    public static void main(String[] args) throws QuizException{
        //creating question
        QuizFactory factory = new QuizFactory();
        IQuiz quiz = factory.createQuiz(QuizType.GUI);
        quiz.setName("Quiz Demo");
        //creating 1st question
        IQuizQuestionBuilder builder1 = new QuizQuestion.Builder();
        builder1.setTitle("We Love Canada");
        builder1.setQuestion("Canada starts with…?");
        builder1.addAnswer("Canada starts with the letter ‘A’.",false);
        builder1.addAnswer("Canada starts with the letter ‘B’.",false);
        builder1.addAnswer("Canada starts with the letter ‘C’.",true);
        builder1.addAnswer("Canada starts with the letter ‘D’.",false);
        builder1.addAnswer("Canada starts with the letter ‘E’.",false);
        IQuizQuestion question1 = builder1.create();
        //creating 2nd question
        IQuizQuestionBuilder builder2 = new QuizQuestion.Builder();
        builder2.setTitle("We Love Australia");
        builder2.setQuestion("Australia starts with …?");
        builder2.addAnswer("Australia starts with the letter ‘A’.",true);
        builder2.addAnswer("Australia starts with the letter ‘B’.",false);
        builder2.addAnswer("Australia starts with the letter ‘C’.",false);
        builder2.addAnswer("Australia starts with the letter ‘D’.",false);
        builder2.addAnswer("Australia starts with the
```

```
27  letter 'E'.",false);
28          IQuizQuestion question2 = builder2.create();
29          //adding questions to quiz
30          quiz.addQuestion(question1);
31          quiz.addQuestion(question2);
32          //saving quiz to file and read it back
33          IQuizFilesDAO dao = SimpleCSVQuizFilesDAO.
   getInstance();
34          dao.saveQuizToFile(quiz,"quiz1.data");
35          IQuiz loadedQuiz = dao.loadQuizFromFile("
   quiz1.data");
36          loadedQuiz.start();
37      }
38  }
39
```

```java
package il.ac.hit.quizzy;

import java.util.List;

/**
 * The IQuiz interface defines the abstract methods for a quiz.
 */
public interface IQuiz {
    /**
     * Starts the quiz.
     */
    void start();

    /**
     * Sets the name of the quiz.
     *
     * @param text the name of the quiz
     */
    void setName(String text);

    /**
     * Gets the name of the quiz.
     *
     * @return the name of the quiz
     */
    String getName();

    /**
     * Adds a question to the quiz.
     *
     * @param question the question to be added
     */
    void addQuestion(IQuizQuestion question);

    /**
     * Gets the list of questions in the quiz.
     *
     * @return the list of questions
     */
    List<IQuizQuestion> getQuestions();
```

```
41 }
42
```

```java
package il.ac.hit.quizzy;

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

/**
 * The GUIQuiz class implements the IQuiz interface and runs the quiz in a GUI.
 */
public class GUIQuiz implements IQuiz {
    private String name;
    private List<IQuizQuestion> questions = new ArrayList<>();

    /**
     * Starts the quiz in the GUI.
     */
    @Override
    public void start() {
        // Implement the GUI-based quiz functionality
    }

    /**
     * Sets the name of the quiz.
     *
     * @param text the name of the quiz
     */
    @Override
    public void setName(String text) {
        this.name = text;
    }

    /**
     * Gets the name of the quiz.
     *
     * @return the name of the quiz
     */
    @Override
    public String getName() {
        return name;
    }
```

```
40
41      /**
42       * Adds a question to the quiz.
43       *
44       * @param question the question to be added
45       */
46      @Override
47      public void addQuestion(IQuizQuestion question) {
48          questions.add(question);
49      }
50
51      /**
52       * Gets the list of questions in the quiz.
53       *
54       * @return the list of questions
55       */
56      @Override
57      public List<IQuizQuestion> getQuestions() {
58          return questions;
59      }
60  }
61
```

```java
package il.ac.hit.quizzy;

/**
 * The QuizType enum represents the different types of quizzes.
 */
public enum QuizType {
    TERMINAL, GUI
}

```

```java
package il.ac.hit.quizzy;

/**
 * The QuizFactory class follows the Factory design pattern and creates IQuiz instances.
 */
public class QuizFactory {
    /**
     * Creates an IQuiz instance based on the specified quiz type.
     *
     * @param type the type of quiz to create
     * @return the created IQuiz instance
     */
    public IQuiz createQuiz(QuizType type) {
        switch (type) {
            case TERMINAL:
                return new TerminalQuiz();
            case GUI:
                return new GUIQuiz();
            default:
                throw new IllegalArgumentException("Invalid quiz type");
        }
    }
}

```

```java
package il.ac.hit.quizzy;

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

/**
 * The QuizQuestion class implements the IQuizQuestion interface and follows the Builder design pattern.
 */
public class QuizQuestion implements IQuizQuestion {
    private String title;
    private String question;
    private List<String> answers;
    private List<Boolean> correctAnswers;

    /**
     * Constructs a new QuizQuestion instance using the provided Builder.
     *
     * @param builder the QuizQuestion.Builder instance
     */
    protected QuizQuestion(Builder builder) {
        this.title = builder.title;
        this.question = builder.question;
        this.answers = new ArrayList<>(builder.answers);
        this.correctAnswers = new ArrayList<>(builder.correctAnswers);
    }

    /**
     * The Builder class for the QuizQuestion class, following the Builder design pattern.
     */
    public static class Builder implements IQuizQuestionBuilder {
        private String title;
        private String question;
        private List<String> answers = new ArrayList
```

```
33 <>();
34         private List<Boolean> correctAnswers = new
   ArrayList<>();
35 
36         /**
37          * Sets the title of the quiz question.
38          *
39          * @param text the title of the quiz question
40          * @return the Builder instance
41          */
42         @Override
43         public IQuizQuestionBuilder setTitle(String
   text) {
44             this.title = text;
45             return this;
46         }
47 
48         /**
49          * Sets the text of the quiz question.
50          *
51          * @param text the text of the quiz question
52          * @return the Builder instance
53          */
54         @Override
55         public IQuizQuestionBuilder setQuestion(
   String text) {
56             this.question = text;
57             return this;
58         }
59 
60         /**
61          * Adds an answer to the quiz question.
62          *
63          * @param text    the text of the answer
64          * @param correct whether the answer is
   correct or not
65          * @return the Builder instance
66          */
67         @Override
68         public IQuizQuestionBuilder addAnswer(String
   text, boolean correct) {
```

```
 69              this.answers.add(text);
 70              this.correctAnswers.add(correct);
 71              return this;
 72          }
 73
 74          /**
 75           * Creates a new QuizQuestion instance using
    the information provided in the Builder.
 76           *
 77           * @return the created QuizQuestion instance
 78           */
 79          @Override
 80          public IQuizQuestion create() {
 81              return new QuizQuestion(this);
 82          }
 83      }
 84
 85      /**
 86       * Gets the title of the quiz question.
 87       *
 88       * @return the title of the quiz question
 89       */
 90      @Override
 91      public String getTitle() {
 92          return title;
 93      }
 94
 95      /**
 96       * Gets the text of the quiz question.
 97       *
 98       * @return the text of the quiz question
 99       */
100      @Override
101      public String getQuestion() {
102          return question;
103      }
104
105      /**
106       * Gets the list of answers for the quiz
    question.
107       *
```

```
108      * @return the list of answers
109      */
110     @Override
111     public List<String> getAnswers() {
112         return answers;
113     }
114 
115     /**
116      * Gets the list of correct answers for the quiz question.
117      *
118      * @return the list of correct answers
119      */
120     @Override
121     public List<Boolean> getCorrectAnswers() {
122         return correctAnswers;
123     }
124 }
125 
```

```java
package il.ac.hit.quizzy;

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

/**
 * The TerminalQuiz class implements the IQuiz interface and runs the quiz in the terminal.
 */
public class TerminalQuiz implements IQuiz {
    private String name;
    private List<IQuizQuestion> questions = new ArrayList<>();

    /**
     * Starts the quiz in the terminal.
     */
    @Override
    public void start() {
        Scanner scanner = new Scanner(System.in);
        int score = 0;
        for (IQuizQuestion question : questions) {
            System.out.println(question.getQuestion());
            for (int i = 0; i < question.getAnswers().size(); i++) {
                System.out.println((i + 1) + ". " + question.getAnswers().get(i));
            }
            int userAnswer = scanner.nextInt();
            if (question.getCorrectAnswers().get(userAnswer - 1)) {
                score++;
            }
        }
        System.out.println("Final score: " + score + "/" + questions.size());
    }

    /**
```

```
35       * Sets the name of the quiz.
36       *
37       * @param text the name of the quiz
38       */
39      @Override
40      public void setName(String text) {
41          this.name = text;
42      }
43
44      /**
45       * Gets the name of the quiz.
46       *
47       * @return the name of the quiz
48       */
49      @Override
50      public String getName() {
51          return name;
52      }
53
54      /**
55       * Adds a question to the quiz.
56       *
57       * @param question the question to be added
58       */
59      @Override
60      public void addQuestion(IQuizQuestion question) {
61          questions.add(question);
62      }
63
64      /**
65       * Gets the list of questions in the quiz.
66       *
67       * @return the list of questions
68       */
69      @Override
70      public List<IQuizQuestion> getQuestions() {
71          return questions;
72      }
73  }
74
```

```java
package il.ac.hit.quizzy;

/**
 * The IQuizFilesDAO interface defines the abstract methods for a quiz file data access object.
 */
public interface IQuizFilesDAO {
    /**
     * Saves a quiz to a file.
     *
     * @param quiz      the quiz to be saved
     * @param fileName  the name of the file to save the quiz to
     * @throws QuizException if there is an error saving the quiz to the file
     */
    void saveQuizToFile(IQuiz quiz, String fileName) throws QuizException;

    /**
     * Loads a quiz from a file.
     *
     * @param fileName  the name of the file to load the quiz from
     * @return the loaded quiz
     * @throws QuizException if there is an error loading the quiz from the file
     */
    IQuiz loadQuizFromFile(String fileName) throws QuizException;
}

```

```java
package il.ac.hit.quizzy;

import java.util.List;

/**
 * The IQuizQuestion interface defines the methods for a quiz question.
 */
public interface IQuizQuestion {
    /**
     * Gets the title of the question.
     *
     * @return the title of the question
     */
    String getTitle();

    /**
     * Gets the text of the question.
     *
     * @return the text of the question
     */
    String getQuestion();

    /**
     * Gets the list of answers for the question.
     *
     * @return the list of answers
     */
    List<String> getAnswers();

    /**
     * Gets the list of correct answers for the question.
     *
     * @return the list of correct answers
     */
    List<Boolean> getCorrectAnswers();
}

```

```java
package il.ac.hit.quizzy;

/**
 * The QuizException class represents an exception that occurs when working with the quiz.
 */
public class QuizException extends Exception {
    /**
     * Constructs a new QuizException with the specified message.
     *
     * @param message the message that describes the exception
     */
    public QuizException(String message) {
        super(message);
    }

    /**
     * Constructs a new QuizException with the specified message and root cause.
     *
     * @param message    the message that describes the exception
     * @param rootCause  the root cause of the exception
     */
    public QuizException(String message, Throwable rootCause) {
        super(message, rootCause);
    }
}

```

```java
package il.ac.hit.quizzy;

/**
 * The IQuizQuestionBuilder interface defines the methods for building a quiz question.
 */
public interface IQuizQuestionBuilder {
    /**
     * Sets the title of the quiz question.
     *
     * @param text the title of the quiz question
     * @return the IQuizQuestionBuilder instance
     */
    IQuizQuestionBuilder setTitle(String text);

    /**
     * Sets the text of the quiz question.
     *
     * @param text the text of the quiz question
     * @return the IQuizQuestionBuilder instance
     */
    IQuizQuestionBuilder setQuestion(String text);

    /**
     * Adds an answer to the quiz question.
     *
     * @param text    the text of the answer
     * @param correct whether the answer is correct or not
     * @return the IQuizQuestionBuilder instance
     */
    IQuizQuestionBuilder addAnswer(String text, boolean correct);

    /**
     * Creates a new QuizQuestion instance using the information provided in the Builder.
     *
     * @return the created QuizQuestion instance
     */
    IQuizQuestion create();
```

```
38 }
39
```

```java
package il.ac.hit.quizzy;

import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

/**
 * The SimpleCSVQuizFilesDAO class implements the IQuizFilesDAO interface and follows the Singleton and DAO design patterns.
 */
public class SimpleCSVQuizFilesDAO implements IQuizFilesDAO {
    private static IQuizFilesDAO instance;

    private SimpleCSVQuizFilesDAO() {
    }

    /**
     * Gets the singleton instance of the SimpleCSVQuizFilesDAO class.
     *
     * @return the singleton instance
     */
    public static IQuizFilesDAO getInstance() {
        if (instance == null) {
            instance = new SimpleCSVQuizFilesDAO();
        }
        return instance;
    }

    /**
     * Saves a quiz to a CSV file.
     *
     * @param quiz      the quiz to be saved
     * @param fileName  the name of the file to save the quiz to
     * @throws QuizException if there is an error saving the quiz to the file
     */
```

```
36      @Override
37      public void saveQuizToFile(IQuiz quiz, String
    fileName) throws QuizException {
38          try (FileWriter writer = new FileWriter(
    fileName)) {
39              writer.write(quiz.getName() + "\n");
40              for (IQuizQuestion question : quiz.
    getQuestions()) {
41                  writer.write(question.getTitle() +
    "," + question.getQuestion() + ",");
42                  for (int i = 0; i < question.
    getAnswers().size(); i++) {
43                      writer.write(question.getAnswers
    ().get(i) + "," + question.getCorrectAnswers().get(i
    ) + ",");
44                  }
45                  writer.write("\n");
46              }
47          } catch (IOException e) {
48              throw new QuizException("Error saving
    quiz to file: " + e.getMessage(), e);
49          }
50      }
51
52      /**
53       * Loads a quiz from a CSV file.
54       *
55       * @param fileName  the name of the file to load
    the quiz from
56       * @return the loaded quiz
57       * @throws QuizException if there is an error
    loading the quiz from the file
58       */
59      @Override
60      public IQuiz loadQuizFromFile(String fileName)
    throws QuizException {
61          try (Scanner scanner = new Scanner(new File(
    fileName))) {
62              String quizName = scanner.nextLine();
63              IQuiz quiz = new TerminalQuiz();
64              quiz.setName(quizName);
```

```
65
66              while (scanner.hasNextLine()) {
67                  String line = scanner.nextLine();
68                  String[] parts = line.split(",");
69                  IQuizQuestionBuilder builder = new
    QuizQuestion.Builder();
70                  builder.setTitle(parts[0]);
71                  builder.setQuestion(parts[1]);
72                  for (int i = 2; i < parts.length; i
     += 2) {
73                      builder.addAnswer(parts[i],
    Boolean.parseBoolean(parts[i + 1]));
74                  }
75                  quiz.addQuestion(builder.create());
76              }
77
78              return quiz;
79          } catch (IOException e) {
80              throw new QuizException("Error loading
    quiz from file: " + e.getMessage(), e);
81          }
82      }
83  }
84
```