Educational Quiz

Computer Coursework

Neil Bugeja

2016-2017

Table of Contents

Problem Definition

For my Matsec Compting A-level project, an educational quiz will be created, suited for children between 6 and 7. This project was chosen because it can easily satisfy the given conditions and criteria while also having the potential to be very useful. The application will be created from scratch and it will do the following functions and tasks:

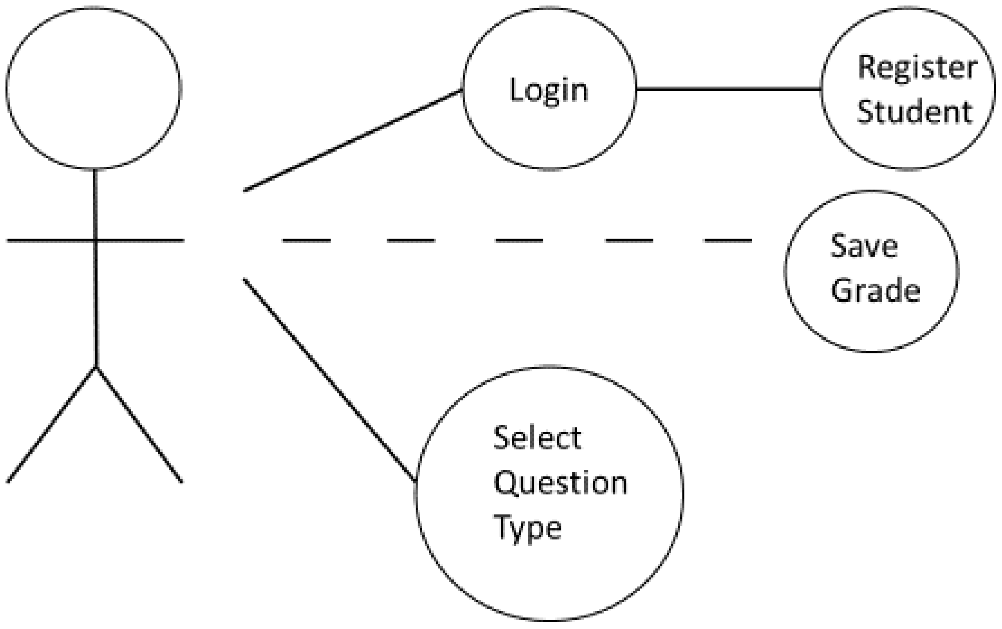
- Add and search pupils
- Store the pupil's scores
- View student's scores

The user will be asked a serious of questions carefully chosen for the age group that it is intended to be used on. As Java will be used as a programming language, this application will be able to run on any platform.

Furthuremore, the program will be customized such that it will provide a Graphical User Interface(GUI). This type of interface frees the user from learning the high-level programming language Java. Moreover, it will give the program a look and feel appearance. The program will consist of windows through the use of Jframe and it will also contain JLabels, Jtextfields and Jbuttons which will help the user navigate through it with ease.

Programming Elements

Use Case Diagram



Initial Sketches

Illustration and design of the created classes

The program is composed of nine classes. Each will be explained as follows:

DisplayPupilsTable

This class will allow the user to view a table of the student requested in the search function. Will include "**ID**", "**Name**", "**Surname**" as search functions and the student's "**ID**", "**Name**", "**Surname"** and "**Date of Birth**" will be given.

Pupil

This class stores the pupil's details.

PupilForm

This class is designed such that a Java Swing container(window) is outputted. The user must input his/her details in all of the text fields. Through the use of a presence check the user will be notified if he/she had not filled any field. Furthermore, it will also allow the user to edit any existing data.

Question

Abstract class used to retrieve the questions and answer from the text files. Contains three attributes and their respective getters and setters.

MultipleChoiceQuestion (inherits from Questions)

This class is designed to display a picture form, in the form of multiple choice answers. It is capable of displaying pictures during the quiz, while also being capable of shuffling the correct answer (further explained in the ***file handling*** section).

QuestionAndAnswer (inherits from Questions)

Contains three attributes and their corresponding getters and setters.

QuizMenu

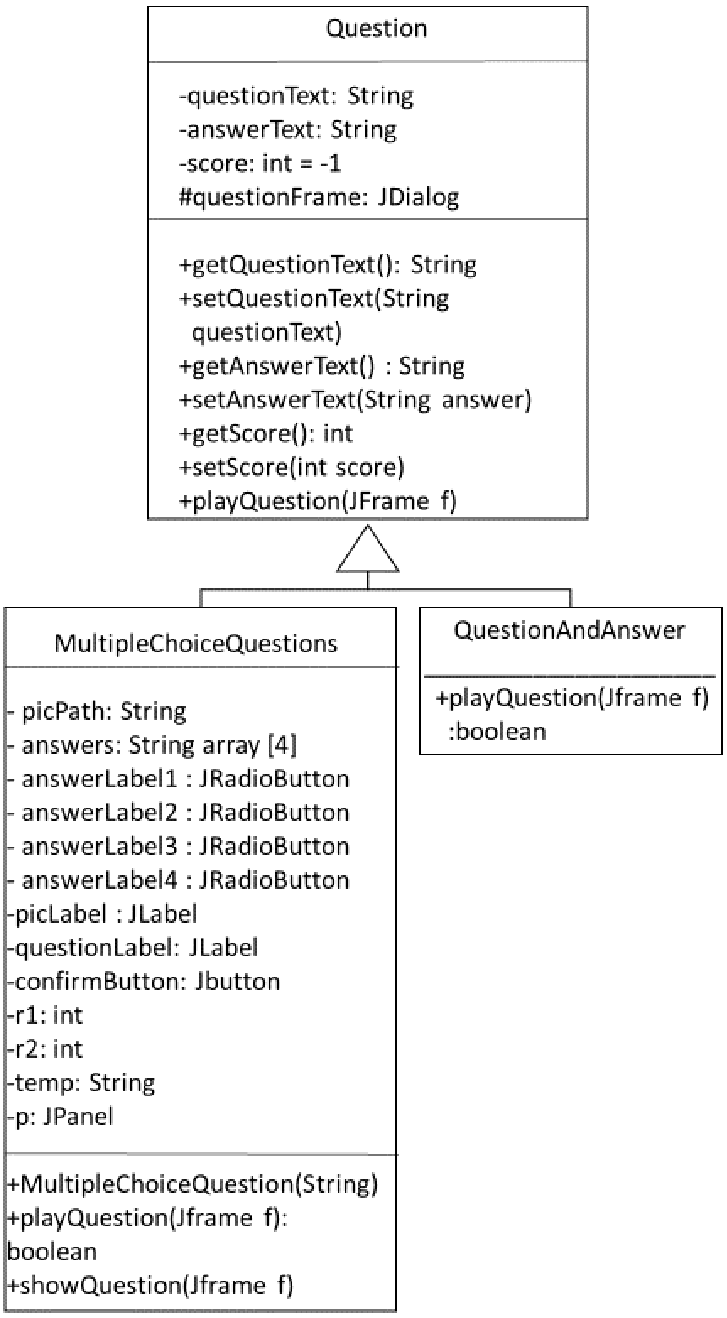
This class is built such that it will display a Java swing container. The user must choose between six options: "**Add Pupil", "Search Pupil", "Show Scores", "Start Questions Quiz", "Start Picture Quiz", "Start Mixed Questions Quiz".** The user will be redirected depending on the option chosen. Also contains the required coding to save and load a pupil.

Score

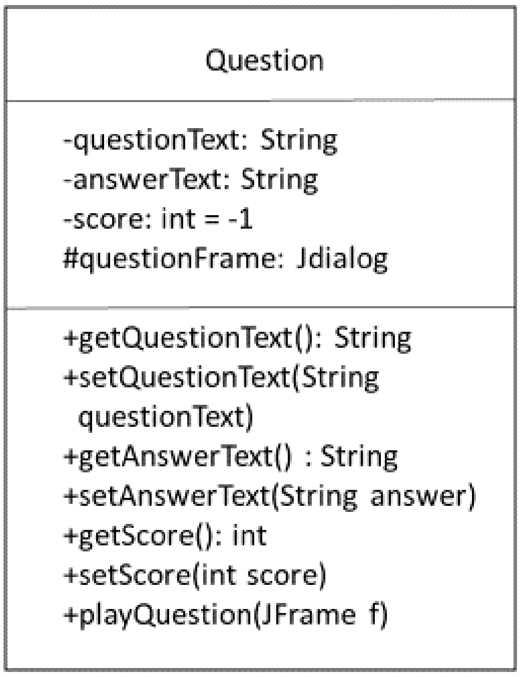
Used to calculate the score. Contains four attributes and their corresponding getters and setters.

Class Diagrams

The class diagram below involves 3 classes. The “MultipleChoiceQuestion” and “QuestionAndAnswer” classes are sub-classes of the super-class “Question”. The two subclasses are classes of object type Question

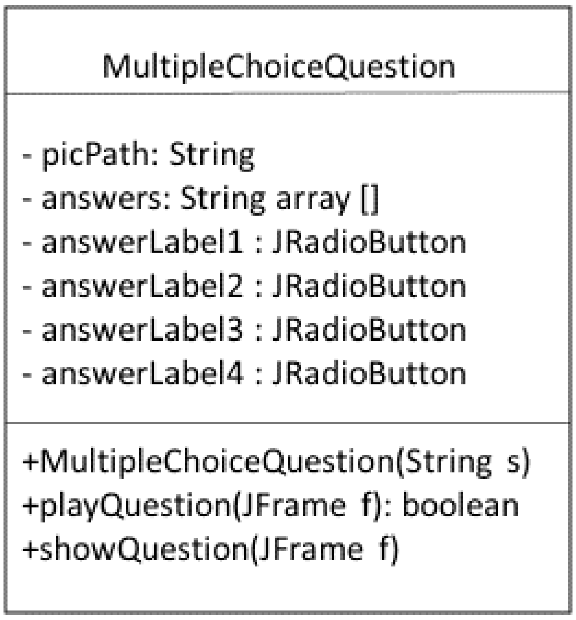

Question
-questionText: String
-answerText: String
-score: int = -1
#questionFrame: JDialog
+getQuestionText(): String
+setQuestionText(String questionText)
+getAnswerText() : String
+setAnswerText(String answer)
+getScore(): int
+setScore(int score)
+playQuestion(JFrame f)
MultipleChoiceQuestions
- picPath: String
- answers: String array [4]
- answerLabel1 : JRadioButton
- answerLabel2 : JRadioButton
- answerLabel3 : JRadioButton
- answerLabel4 : JRadioButton
-picLabel : JLabel
-questionLabel: JLabel
-confirmButton: Jbutton
-r1: int
-r2: int
-temp: String
-p: JPanel
+MultipleChoiceQuestion(String)
+playQuestion(Jframe f): boolean
+showQuestion(Jframe f)
QuestionAndAnswer
+playQuestion(Jframe f) :boolean

Question



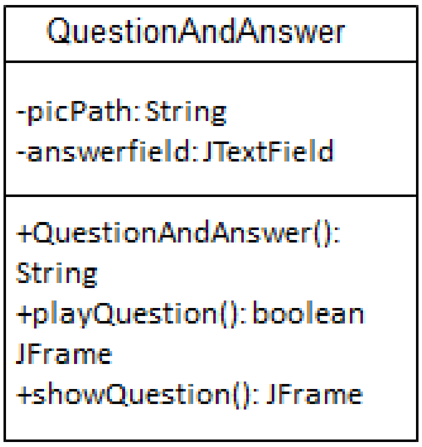
question	String used to store question
answer	String used to store answer
score	Int used to store the score
questionFrame	Used to create a dialog window
getQuestionText()	returns question
setQuestionText(String)	sets value of question to question
getAnswerText()	returns answer
setAnswerText(String anser)	sets value of answer to answer
getScore()	returns score
setScore(int score)	sets value of score to score
playQuestion(JFrame f)	

MultipleChoiceQuestion


MultipleChoiceQuestion
- picPath: String
- answers: String array []
- answerLabel1 : JRadioButton
- answerLabel2 : JRadioButton
- answerLabel3 : JRadioButton
- answerLabel4 : JRadioButton
+MultipleChoiceQuestion(String s)
+playQuestion(JFrame f): boolean
+showQuestion(JFrame f)

picPath	String used to hold the pictures' file location
answers	String array used to hold answer choices
answersLabel1	Holds the correct answer option
answersLabel2	Holds the incorrect answer option
answerLablel3	Holds incorrect answer option
answerLabel4	Holds incorrect answer option
MultipleChoiceQuestion(String s)	Hold the picture locations, the answers and the question.
showQuestion(JFrame f)	Used
playQuestion(JFrame f)	JFrame used to play quiz

QuestionAndAnswer

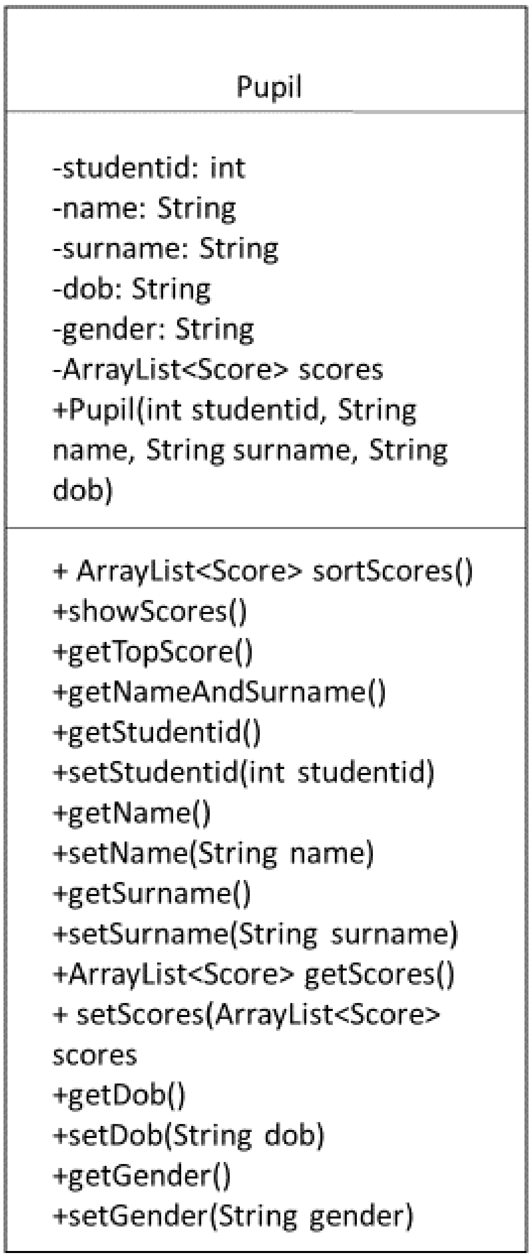

QuestionAndAnswer

-picPath: String
-answerfield: JTextField

+QuestionAndAnswer(): String
+playQuestion(): boolean JFrame
+showQuestion(): JFrame

picPath	String used to hold the picture's file location
answerfield	JTextField where pupil will write answer
QuestionAndAnswer()	Hold the picture locations, the answers and the question.
playQuestion()	JFrame used to play quiz
showQuestion()	JFrame used to display the questions

Pupil



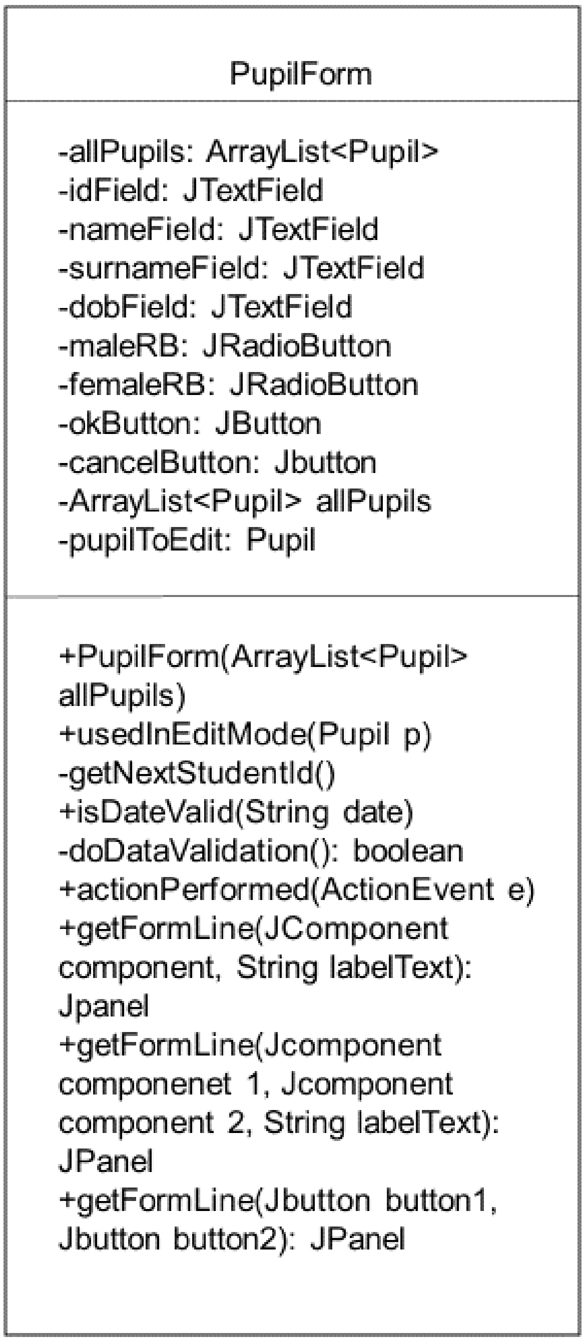
Pupil

-studentid: int
-name: String
-surname: String
-dob: String
-gender: String
-ArrayList<Score> scores
+Pupil(int studentid, String name, String surname, String dob)

+ ArrayList<Score> sortScores()
+showScores()
+getTopScore()
+getNameAndSurname()
+getStudentid()
+setStudentid(int studentid)
+getName()
+setName(String name)
+getSurname()
+setSurname(String surname)
+ArrayList<Score> getScores()
+ setScores(ArrayList<Score> scores
+getDob()
+setDob(String dob)
+getGender()
+setGender(String gender)

studentid	int used to sore student's id
name	String used to hold student's name
surname	String used to hold student's surname
dob	String used to hold student's D.O.B
gender	String used to holf student's gender
Pupil(int studentid, String name, String surname, String dob)	constructor with parameter
ArrayList<Scores> scores	ArrayList used to hold student's scores
showScores	shows student's scores
getTopScore	returns student's top score
getNameAndSurname	returns student's name and surname
getStudentid	returns student's ID
setStudentid(int studentid)	sets studentid to studentid
getName	returns name
setName(String name)	sets name to name
getSurname	returns surname
setSurname(string surname)	sets surname to surname
ArrayList<Score> getScores	returns scores
setScores(ArrayList<Score> scores)	sets scores to scores
getDob()	returns dob
setDOB(String dob)	sets dob to dob
getGender()	returns gender
setGender	sets gender to gender

PupilForm

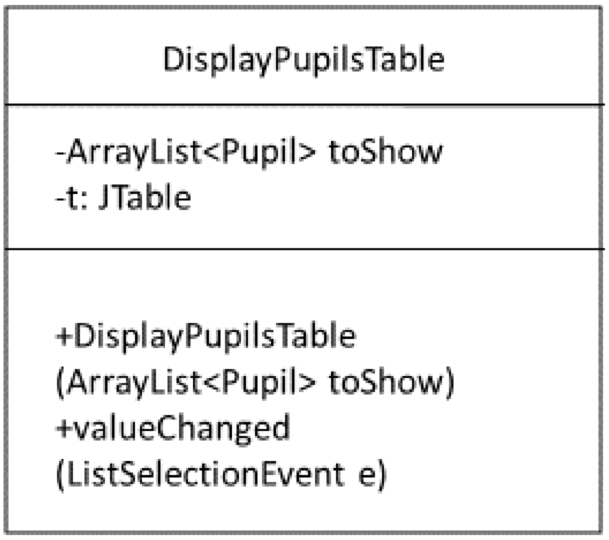


```
PupilForm
-allPupils: ArrayList<Pupil>
-idField: JTextField
-nameField: JTextField
-surnameField: JTextField
-dobField: JTextField
-maleRB: JRadioButton
-femaleRB: JRadioButton
-okButton: JButton
-cancelButton: Jbutton
-ArrayList<Pupil> allPupils
-pupilToEdit: Pupil

+PupilForm(ArrayList<Pupil>
allPupils)
+usedInEditMode(Pupil p)
-getNextStudentId()
+isDateValid(String date)
-doDataValidation(): boolean
+actionPerformed(ActionEvent e)
+getFormLine(JComponent
component, String labelText):
Jpanel
+getFormLine(Jcomponent
componenet 1, Jcomponent
component 2, String labelText):
JPanel
+getFormLine(Jbutton button1,
Jbutton button2): JPanel
```

allPupils	ArrayList to store pupils
idField:	JTextField where student is show
nameField	JTextField where student enters name
surnameField	JTextField where student enters surname

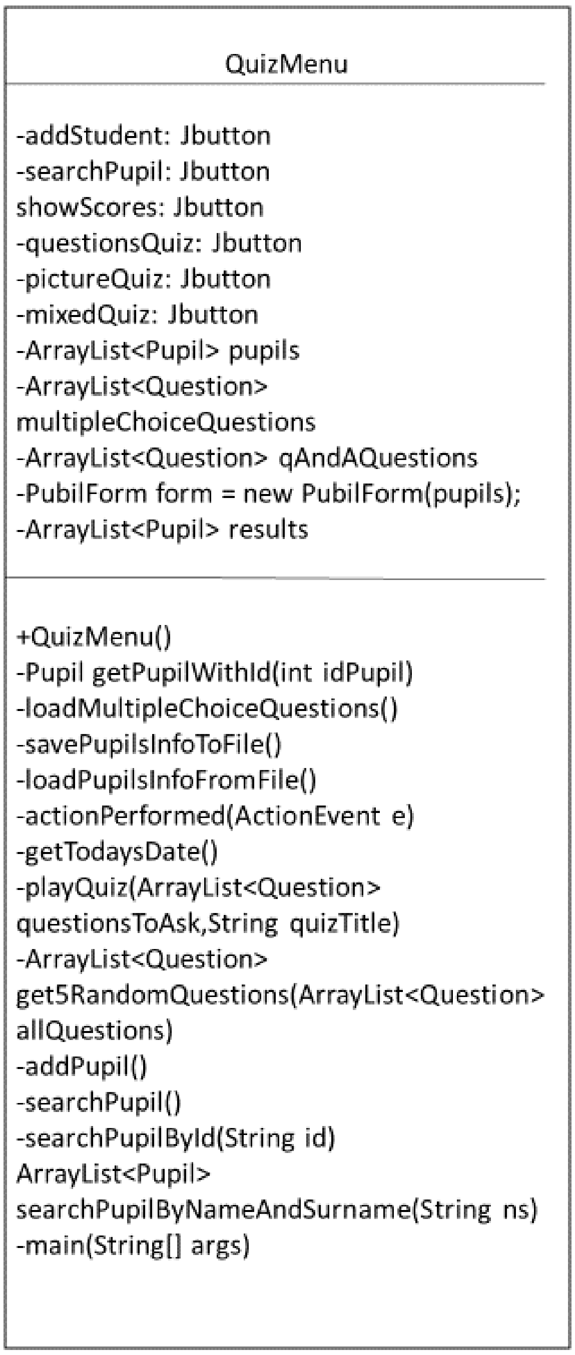
dobField	JTexteField where student enters dob
maleRB	JRadioButton where student picks male or female
femaleRB	JRadioButton where student picks male or female
okButton	JButton where student clicks ok
cancelButton	JButton where student clicks cancel
ArrayList<Pupil> allPupils	array lsit of type allPupils
pupilToEdit	pupil used to store pupilToEdit
PupilForm(ArrayList<Pupil> allPupils)	
usedInEditMode(Pupil P)	used to edit pupil's details
getNextStudenId()	return NextStudentId
isDateValid(String date)	used to check if date enters follow the format of "DD-MM-YYYY"
doDataValidation()	used to check if data entered by the student is valid
actionPerformed(ActionEvent e)	shows the form where the student enters the required data
getFormLine(JComponent component, String labelText)	
getFormLine(Jcomponent componenet 1, Jcomponent component 2, String labelText)	
getFormLine(Jbutton button1, Jbutton button2	

DisplayPupilsTable



ArrayList<Pupil> to show

QuizMenu



QuizMenu

-addStudent: Jbutton
-searchPupil: Jbutton
showScores: Jbutton
-questionsQuiz: Jbutton
-pictureQuiz: Jbutton
-mixedQuiz: Jbutton
-ArrayList<Pupil> pupils
-ArrayList<Question> multipleChoiceQuestions
-ArrayList<Question> qAndAQuestions
-PubilForm form = new PubilForm(pupils);
-ArrayList<Pupil> results

+QuizMenu()
-Pupil getPupilWithId(int idPupil)
-loadMultipleChoiceQuestions()
-savePupilsInfoToFile()
-loadPupilsInfoFromFile()
-actionPerformed(ActionEvent e)
-getTodaysDate()
-playQuiz(ArrayList<Question> questionsToAsk,String quizTitle)
-ArrayList<Question> get5RandomQuestions(ArrayList<Question> allQuestions)
-addPupil()
-searchPupil()
-searchPupilById(String id)
ArrayList<Pupil> searchPupilByNameAndSurname(String ns)
-main(String[] args)

addStudent	JButton where student clicks add student
searchPupil	JButton where student clicks search pupil

showScores	JButton where the student clicks show scores
questionsQuiz	JButton where the student clicks questions quiz
pictureQuiz	JButton where the student clicks picture quiz
mixedQuiz	JButton where the student click
ArrayList<Pupil> pupils	Array list of type pupils
ArrayList<Question> multipleChoiceQuestions	Array list of type multiplechoicequestions
ArrayList<Question> qAndAQuestions	Array lsit of type qAndAQuestions

Sub-programs Design

In this section, the explanation of methods will be done by means of pseudo-code. Flowcharts will be used to demonstrate the flow of application and to further explain the methods graphically in the Algorithm and Logic Section.

Pseudo-code

- **Double click Educational Quiz Jar file**
  - Once the program is launched the user will be greeted by the main menu.
  - The program will proceed depending on the user's option.

- **Add Pupil**
  - Input all the data required in the Pupil Form as instructed.
  - If the data is inputted correctly the user will be notified and will be redirected back to the main menu.
  - If the user enters incorrect data the user will be notified and will be redirected back to the Pupil Form

- **Search Pupil**
  - The user will be asked to enter the **"Pupil ID"** or the pupil's **Name and Surname.**
  - If the user enters the **"Pupil ID"** a Pupil Form with the pupil's details will be opened. The user can edit the **"ID", "Name", "Surname", "Date of Birth"** and **"Gender**". The user can also choose to delete the save file.
  - If the user enters the pupil's name and surname a Members Table will pop up, showing the pupils **"ID", "Name", "Surname", "Date of Birth"** and **"Gender**". To edit the user should take note of the **"ID"** and go to the above step

- **Show Scores**
  - Once the ***"show scores***" button from the main menu has been clicked a table should appear.
  - To view the scores simply press on the pupil of your desire and a history of the pupil's score will show up, along with their respective date.

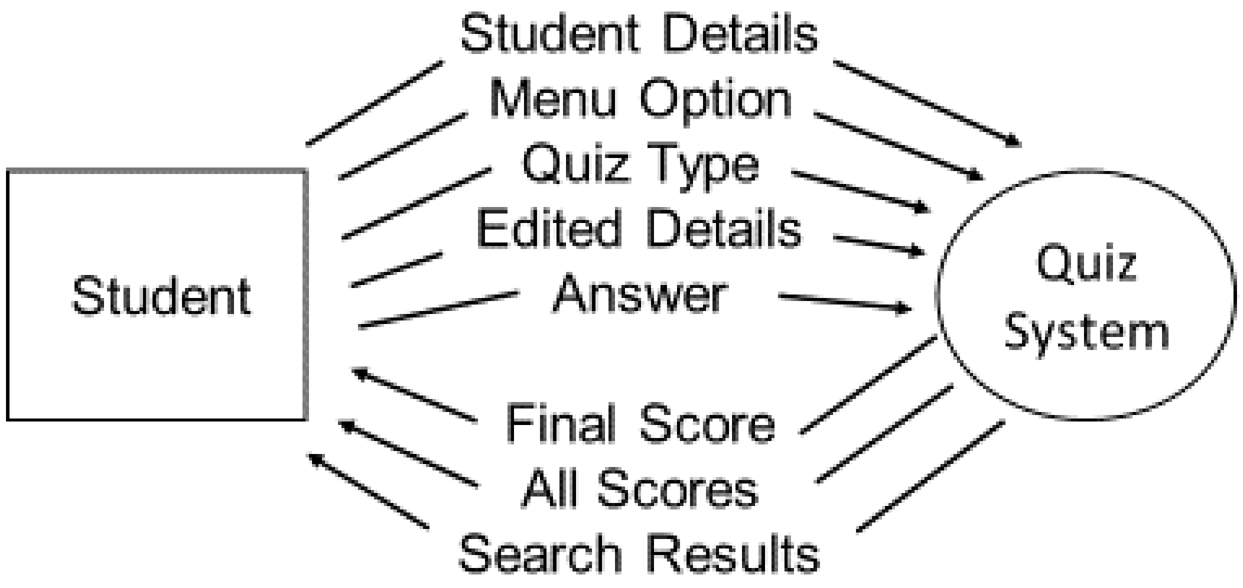
- **Start Question Quiz**
  - Random pictures will pop up along with their respective question and answer questions.

- The pupil should write an answer that they believe to be correct and proceed to the next question by clicking on the “Confirm” button
- At the end the pupil will be given his/her total score and asked if the pupil wishes to save this result.
- Should the pupil wish to save the final result, the program will ask the pupil to enter his/her student ID. The user will be notified if an invalid ID is entered, else the mark is saved. The pupil will be redirected to the main Menu
- Should the user not wish to save the mark, he/she should press no and will be redirected to the main menu.

- **Start Picture Quiz**
  - Random pictures will pop up along with their respective multiple choice question and answer.
  - The pupil should mark the answer that they think is correct and proceed to the next question by clicking on the “Confirm” button.
  - At the end the pupil will be given his/her total score and asked if the pupil wishes to save this result.
  - Should the pupil wish to save the final result, the program will ask the pupil to enter his/her student ID. The user will be notified if an invalid ID is entered, else the mark is saved. The pupil will be redirected to the main Menu
  - Should the user not wish to save the mark, he/she should press no and will be redirected to the main menu.

- **Mixed Quiz**
  - Random pictures will pop up along with their respective multiple choice type questions and questions and answer type questions .
  - The pupil should input or mark an answer that they believe to be correct and proceed to the next question by clicking on the “Confirm” button.
  - At the end the pupil will be given his/her total score and asked if the pupil wishes to save this result.
  - Should the pupil wish to save the final result, the program will ask the pupil to enter his/her student ID. The user will be notified if an invalid ID is entered, else the mark is saved. The pupil will be redirected to the main Menu
  - Should the user not wish to save the mark, he/she should press no and will be redirected to the main menu.

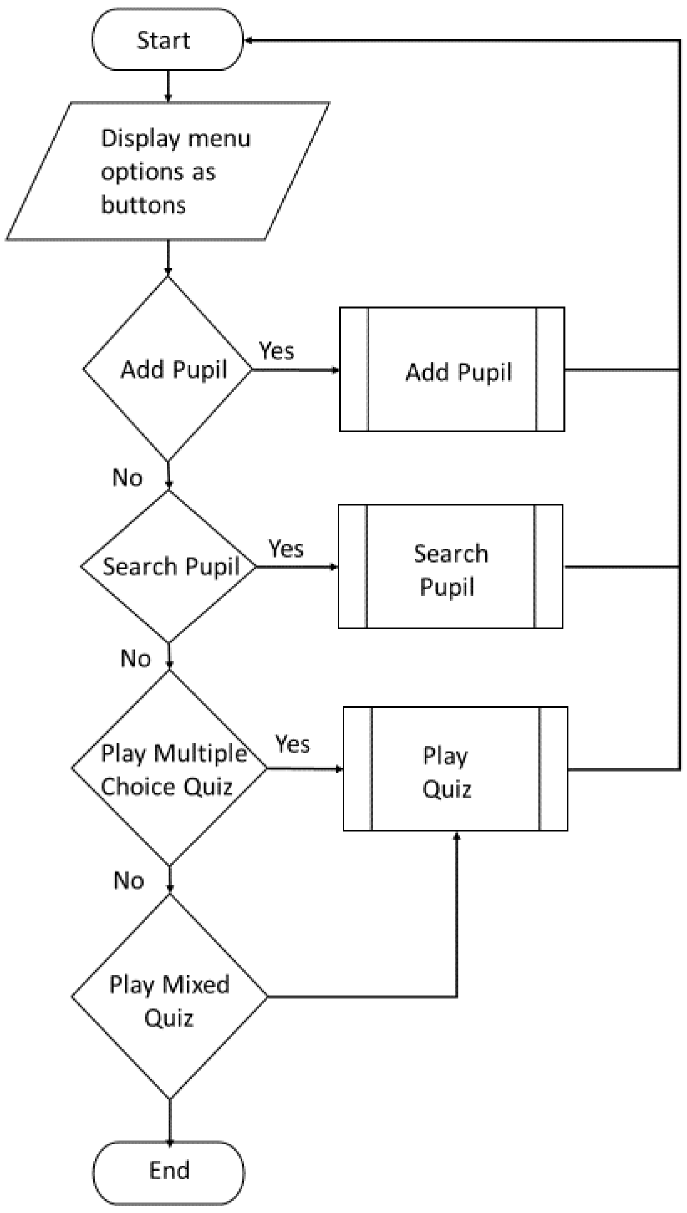
Data Flow Diagram Level 0



Algorithms and Logic

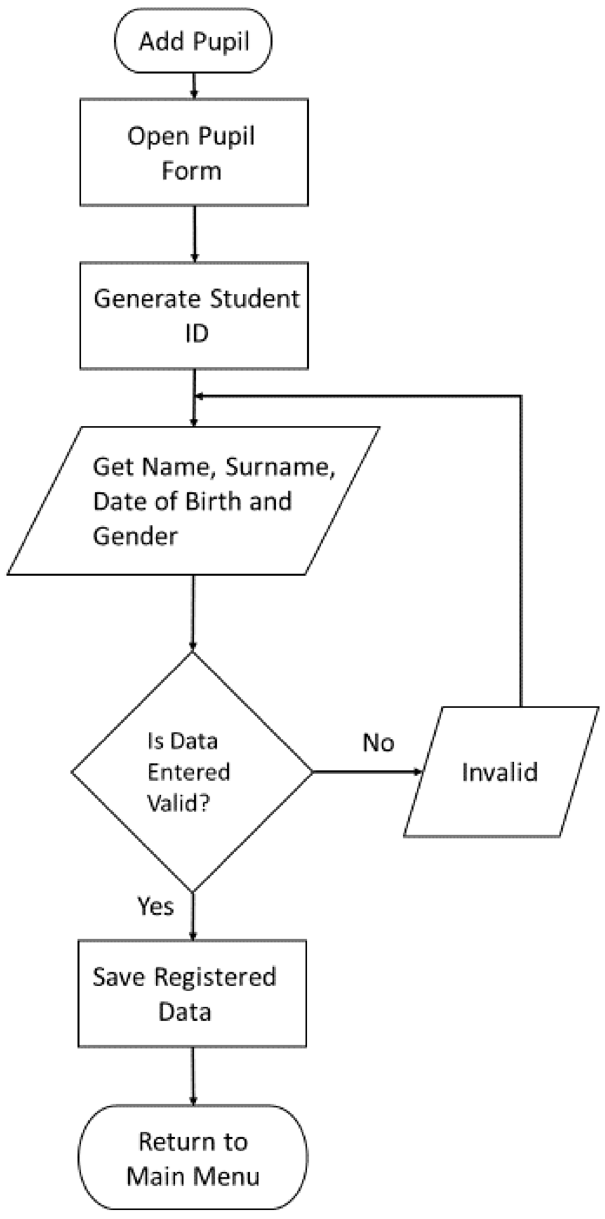
Flowcharts

Main Menu



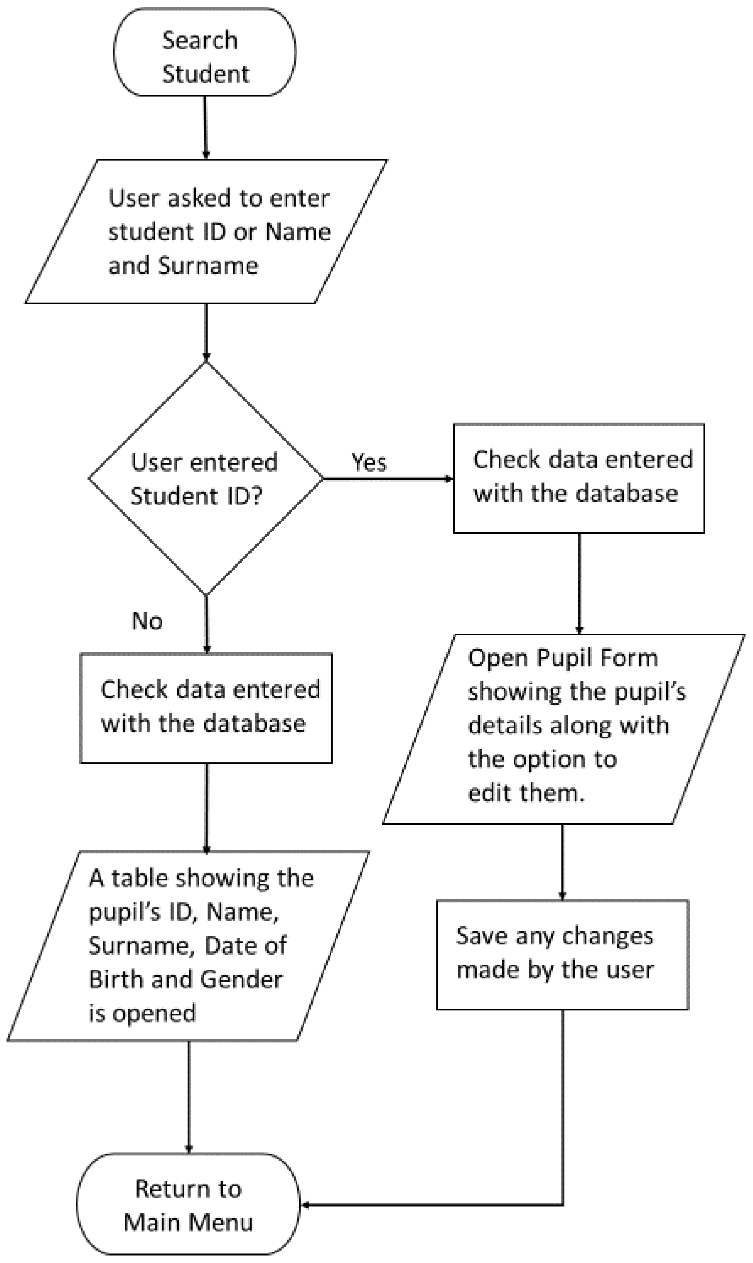
The flowchart above shows the process of the main menu. Once the program starts the main Menu will be opened. Here the user can decide what he/she wants to do with the program. The main menu will redirect the user to the respective choice chosen.

Add Pupil



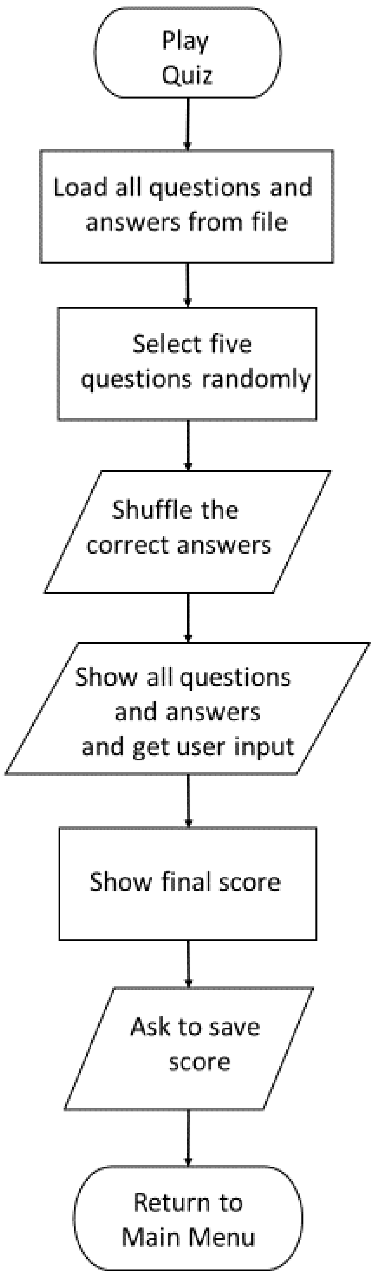
The flowchart above shows the process of registering a pupil into the system and the flow of data. The user is required to enter the required data in the Pupil Form as instructed. The system checks if the data entered if valid. If this is the case then the new pupil is registered into the system and the user will be redirected to the main menu. However, if this is not the case then the user will be required to re-enter his/her details correctly.

Search Pupil


Search Student
User asked to enter student ID or Name and Surname
User entered Student ID?
Yes
Check data entered with the database
No
Check data entered with the database
Open Pupil Form showing the pupil's details along with the option to edit them.
A table showing the pupil's ID, Name, Surname, Date of Birth and Gender is opened
Save any changes made by the user
Return to Main Menu

The flowchart above shows the process of searching and editing a pupil, and the flow of data. The user is asked to either enter a student ID or a name and surname. If the user enters the student's ID a pupil form opened, just like when adding a pupil. The difference is that the details are filled in and the user can view or edited them as pleased. Once the user is ready he/she will be directed to the main menu. If the user instead enters a name and surname, then a table containing the corresponding pupil's ID, name, surname, date of birth and gender is opened. Once done the user will be directed to the main menu.

Play Quiz


Play Quiz
Load all questions and answers from file
Select five questions randomly
Shuffle the correct answers
Show all questions and answers and get user input
Show final score
Ask to save score
Return to Main Menu

The flowchart above shows the process of the quiz. Firstly, the questions are loaded from a text file. Then five questions are selected randomly, along with their corresponding answers. The answers are shuffled. All questions are shown along with their answers. The user is required to pick the correct answer. When all questions are answered, the final score is final score is shows and the user is asked if he/she wishes to save the score. The user is then redirected to the main menu.

File Handling

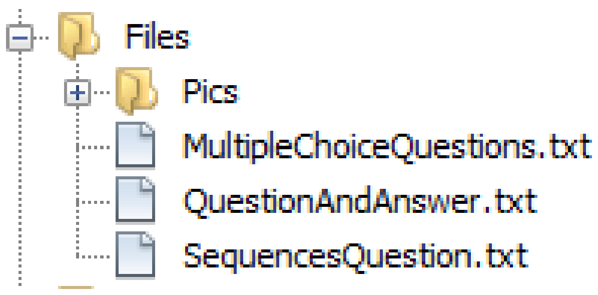
Text Files

What are text files?

Text files consist only of lines of text

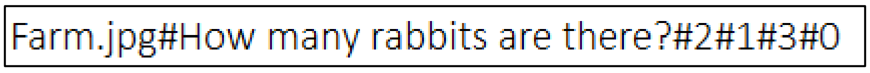
The use of text files in the application

The application makes use of three text files. Each created to suit a particular type of quiz:



Files
Pics
MultipleChoiceQuestions.txt
QuestionAndAnswer.txt
SequencesQuestion.txt

Inside each text file one can find various lines of text. The first thing indicated in these lines is which image the questions are being taken from. Secondly, the actual question is written. Lastly the correct answer is written, followed by any incorrect answers. Each on is distinguished from the other by the use of a # :



```
Farm.jpg#How many rabbits are there?#2#1#3#0
```

Code used to read text from a text file

```
private void loadMultipleChoiceQuestions() {
        try {
            BufferedReader in = new BufferedReader(new
FileReader("Files//MultipleChoiceQuestions.txt"));
            String str;
            while ((str = in.readLine()) != null) {
                multipleChoiceQuestions.add(new MultipleChoiceQuestion(str));
            }
            in.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
```

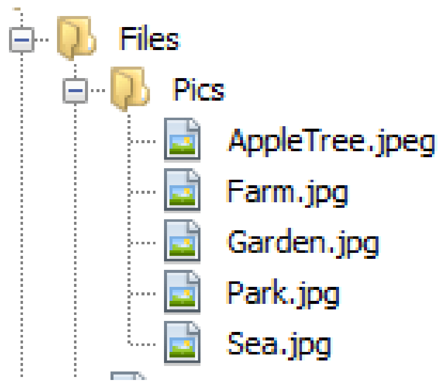
JPG Files

What are JPG files?

JPG is a file extension for a lossy graphics file.

The use of JPG files in the application

The JPG files are used to show pictures in the ***Picture Quiz***.



Code used to read a JPG file

```
public MultipleChoiceQuestion(String s) {
        String[] x = s.split("#");
        picPath = "Files//Pics//" + x[0];
        setQuestionText(x[1]);
        setAnswerText(x[2]);
        for (int i = 2; i <= 5; i++) {
            answers[i - 2] = x[i];
        }
    }
```

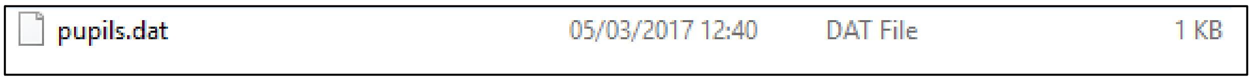
Serializable Files

What is serialization?

In computer science, serialization is the operation of translating data structures or objects into a format that can be stored.

The user of serializable files in the application

The program is written such that the pupils' information and scores are stored in a file so every time he/she wants to view their final score, the system will "remember" this using a serializable file. Thus, the pupil will be able to retrieve his/her final score.



```
pupils.dat                    05/03/2017 12:40      DAT File              1 KB
```

Code used to save to file

```
try {
            FileOutputStream fileOut = new FileOutputStream("pupils.dat");
            ObjectOutputStream out = new ObjectOutputStream(fileOut);
            out.writeObject(pupils);
            out.close();
            fileOut.close();
        } catch (Exception e) {

        }
```

<u>Code used to load from file</u>

```
try {
            FileInputStream fileIn = new FileInputStream("pupils.dat");
            ObjectInputStream in = new ObjectInputStream(fileIn);
            pupils = (ArrayList<Pupil>) in.readObject();
            in.close();
            fileIn.close();
        } catch (Exception e) {
            System.out.println("Files not loaded. Problem with file.");
        }
```

Object Oriented Principles

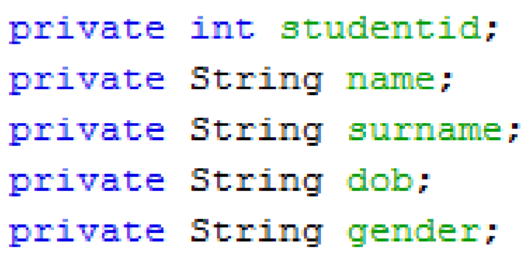
The class Question is a super-class and sub-classes QuestionAndAnswer and MultipleChoiceQuestions are its derived classes. The link between the super-class and the subclasses will be further explained in the ***Inheritance*** section. Encapsulation is used throughout the Question class so that the attributes whose access modifier is set to private can only be accessed by public methods.

Encapsulation

Encapsulation in Java is a mechanism of wrapping the data (variables) and code acting on the data (methods) together as a single unit. In encapsulation, the variables of a class will be hidden from other classes, and can be accessed only through the methods of their current class.

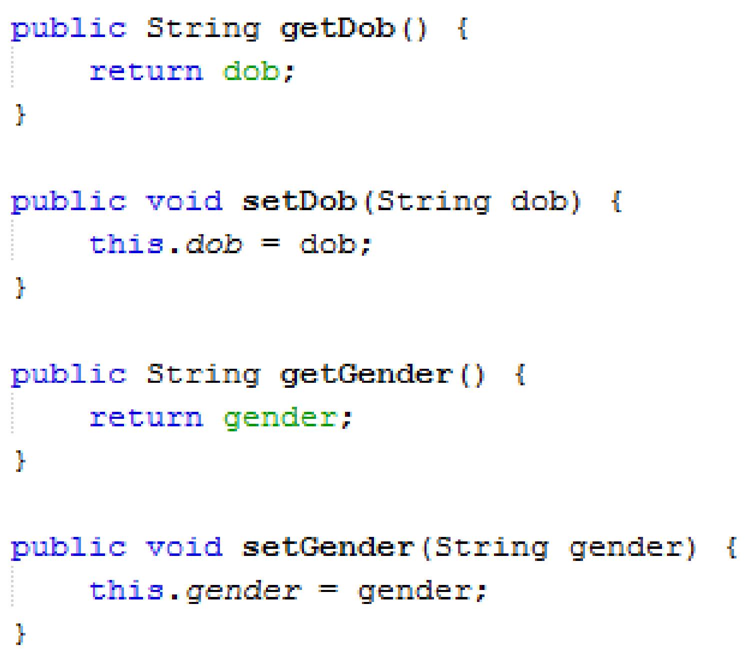
The following code, taken from the quiz program, demonstrates how to achieve Encapsulation in Java:

.



```
private int studentid;
private String name;
private String surname;
private String dob;
private String gender;
```

```java
public String getNameAndSurname(){
    return name + " " + surname;
}

public int getStudentid() {
    return studentid;
}

public void setStudentid(int studentid) {
    this.studentid = studentid;
}

public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public String getSurname() {
    return surname;
}

public void setSurname(String surname) {
    this.surname = surname;
}

public ArrayList<Score> getScores() {
    return scores;
}

public void setScores(ArrayList<Score> scores) {
    this.scores = scores;
}

public String getDob() {
    return dob;
}
```



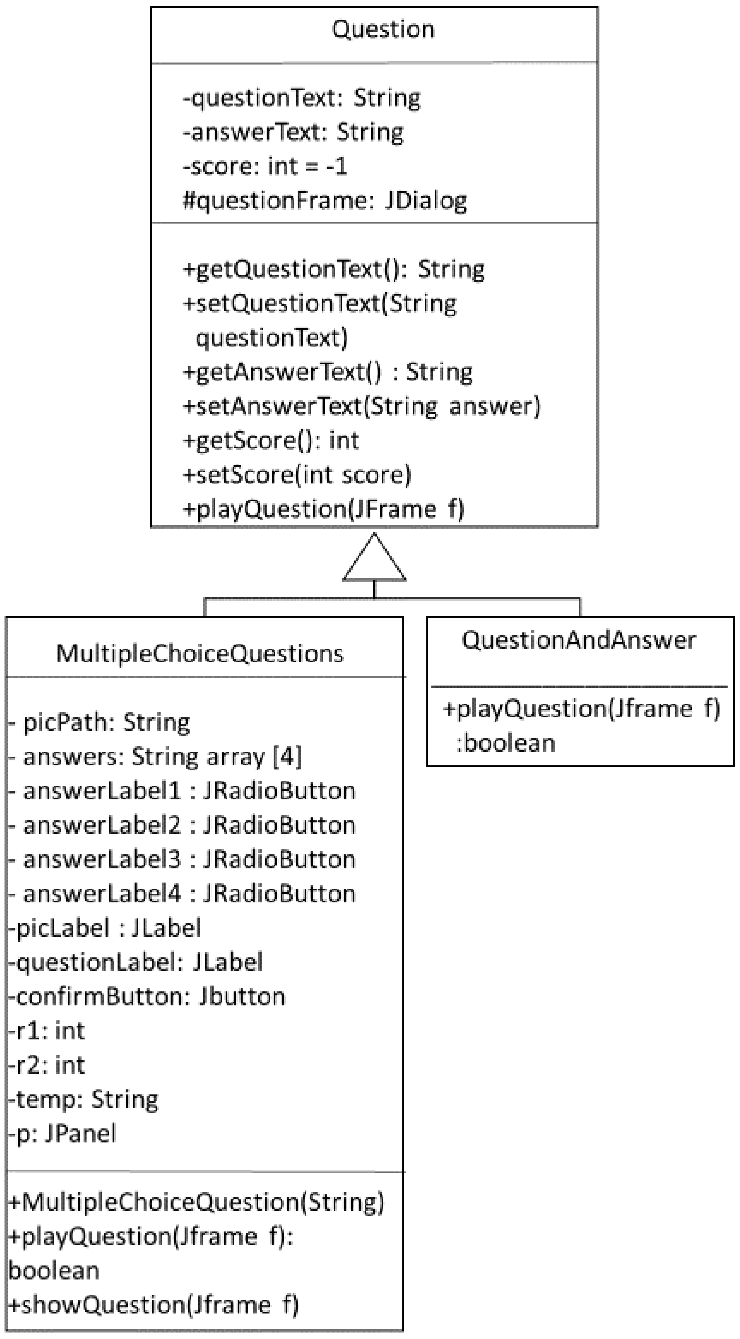
```java
public String getDob() {
    return dob;
}

public void setDob(String dob) {
    this.dob = dob;
}

public String getGender() {
    return gender;
}

public void setGender(String gender) {
    this.gender = gender;
}
```

Inheritance

<u>What is Inheritance?</u>

In broad words, inheritance is the capability of a class to use the properties and methods of another class. Inheritance encourages reusable code and proficiency in the structure of the program. This makes inheritance on of the pillars of object oriented programming. The keyword which denotes inheritance is ***extends***.

The use of Inheritance throughout the Quiz Program


Question
-questionText: String
-answerText: String
-score: int = -1
#questionFrame: JDialog
+getQuestionText(): String
+setQuestionText(String questionText)
+getAnswerText() : String
+setAnswerText(String answer)
+getScore(): int
+setScore(int score)
+playQuestion(JFrame f)
MultipleChoiceQuestions
- picPath: String
- answers: String array [4]
- answerLabel1 : JRadioButton
- answerLabel2 : JRadioButton
- answerLabel3 : JRadioButton
- answerLabel4 : JRadioButton
-picLabel : JLabel
-questionLabel: JLabel
-confirmButton: Jbutton
-r1: int
-r2: int
-temp: String
-p: JPanel
+MultipleChoiceQuestion(String)
+playQuestion(Jframe f): boolean
+showQuestion(Jframe f)
QuestionAndAnswer
+playQuestion(Jframe f) :boolean

This diagram shows how the QuestionAndAnswer and MultipleChoiceQuestions classes inherit from the Question class using the tree notation. The two sub-classes contain all the properties of the Question class and some extra properties which were added to them.

Polymorphism

What is Polymorphism?

Polymorphism is the ability of an object to take on different forms. This is when the sub-classes are instantiated using the sub-class name and declared using the super-class name.

```
Ex: Person p1 = new Teacher();
    Person p2 = new Student();
```

Class question is an abstract class. This means that this class cannot be instantiated because it contains abstract methods. By the use of polymorphism, the below method can be used to play any quiz:

```
private void playQuiz(ArrayList<Question> questionsToAsk,String quizTitle) {
        int score = 0;
        for (int i = 0; i < 5; i++) {
            questionsToAsk.get(i).playQuestion(this);
            score = score + questionsToAsk.get(i).getScore();
        }
        int n = JOptionPane.showConfirmDialog(this,
                "You've got a score of " + score + ". Do you want to save your score?",
"Total Score", JOptionPane.YES_NO_OPTION);
        if (n == 0) {
            String pupleId = JOptionPane.showInputDialog("Enter your pupil ID:");
            try {
                Pupil puple = getPupilWithId(Integer.parseInt(pupleId));
                if (puple == null) {
                    JOptionPane.showMessageDialog(null, "Pupil not registered", "Error",
JOptionPane.ERROR_MESSAGE);
                } else {
```

```
                Score scorePoints = new Score();
                scorePoints.setScore(score);
                scorePoints.setTotalPossableScore(questionsToAsk.size());
                scorePoints.setQuizType(quizTitle);
                scorePoints.setDate(getTodaysDate());
                puple.getScores().add(scorePoints);
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Invalid Pupil ID", "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }

}
```

Sorting and Searching

The program makes use of the searching and sorting algorithms. This application uses a **Linear Search** to perform its' search functions (Pupil ID or Name and Surname). The sorting technique used is the **Bubble Sort** which in this system is being used to sort the pupil's ID in ascending order.

The following section of code shows the ***Linear Search*** used to search for a Pupil by ID:

```
private Pupil searchPupilById(String id) {
        for (int i = 0; i < pupils.size(); i++) {
            if ((pupils.get(i).getStudentid() + "").equals(id)) {
                return pupils.get(i);
            }
        }
        return null;
    }
```

The following section of code shows the use of the ***Bubble Sort*** to sort the pupils according to their pupil ID

```
public ArrayList<Score> sortScores() {
        int n = scores.size();
        Score[] arr = scores.toArray(new Score[n]);
        for (int j = 0; j < arr.length; j++) {
            for (int i = j + 1; i < arr.length; i++) {
                if (arr[i].getScore() > arr[j].getScore()) {
                    Score t = arr[j];
                    arr[j] = arr[i];
                    arr[i] = t;
                }
            }
        }
        scores = new ArrayList<Score>(Arrays.asList(arr));
        return scores;
     }
```

Shuffle

The program also makes use of the shuffling algorithm. This is used to shuffle the correct answers, so that there is a very slight possibility for the same answer to appear in the same place as before

The following section of code shows the use of shuffling:

```
Random randomGenerator = new Random();
      for(int i = 0;i < 10;i++){
         int r1 = randomGenerator.nextInt(4);
         int r2 = randomGenerator.nextInt(4);
         String temp;
         temp = answers[r1];
         answers[r1] = answers[r2];
         answers[r2] = temp;
      }
```

Application of Java API's

As stated before, the application quiz program is based on GUI. Swing is an API for providing this kind of interface. These packages are used in classes which regard the design and interface of the program. All the classes which contain these components will be listed as follows. A coloured box will be assigned to each class and the boxes will be copied next to the features they made use of:

DisplayPupilsTable	🟫
MultipleChoiceQuestion	🟧
PupilForm	🟦
Pupil	🟥
Question	🟩
QuestionandAnswer	🟪
QuizMenu	⬛
Score	🟨

These are the APIs which were used in the classes above:

import java.awt.BorderLayout A border layout lays out a container, arranging and resizing its components to fit in five regions: north, south, east, west, and center.	🟫
import java.util.ArrayList Is used to store a list and manipulate the size of the array	🟫🟦🟥🟪
import javax.swing.Jframe provides a window with a border and a title	🟫🟧🟦🟩🟪
import javax.swing.Jtable provides a table with a border and a title	🟫

import javax.swing.ListSelectionModel 🟧
This interface represents the current state of the selection for any of the components that display a list of values with stable indices

import javax.swing.event.ListSelectionEvent 🟧
An event that characterizes a change in selection. The change is limited to a a single inclusive interval

import javax.swing.event.ListSelectionListener 🟧
The listener that's notified when a lists selection value changes.

import javax.swing.Jbutton 🟥🟦🟪
provides the functionality of a push button

import javax.swing.JDialog; 🟥🟩
Used to create a custom dialog

import javax.swing.JLabel; 🟥
component used to display text, pictures or both

import javax.swing.Jpanel 🟥🟦
is used to create a panel

import javax.swing.JRadioButton 🟥🟦
is used to create a group of radio buttons yet only a single button can be selected

import javax.swing.Jseparator 🟥🟦
provides a general purpose component for implementing driver lines

import java.awt.FlowLayout 🟦🟪
sets components in a line one after the other

import java.awt.event.ActionEvent 🟦🟪
indicates that a component-defined event has occurred (a button has been pressed)

import java.awt.event.ActionListener 🟦🟪
the event explained previously is passed on to the Action Listener (processes an action)

import java.text.SimpleDateFormat 🟦🟪

used for formatting and pasing dates in a locale-sensitive manner

import javax.swing.BoxLayout 🟦
sets components one after the other either in a vertical or in a horizontal orientation

import javax.swing.ButtonGroup 🟦
manages the selected/unselected state of a group of buttons

import javax.swing.Jbutton 🟦
provides the functionality of a push button

import javax.swing.Jcomponent 🟦

import javax.swing.JOptionPane 🟦🟥🟪
used to create and adjust different kinds of dialogs

import javax.swing.Jlabel 🟦
component used to display text, pictures or both

import javax.swing.JTextField 🟦
allows input of a single line of text

import javax.swing.UIManager 🟦🟪
manages the look and feel of the application

import java.io.Serializable 🟥🟨
to write text to a file

import java.util.Arrays 🟥
This class contains various methods for manipulating arrays

import java.awt.event.WindowAdapter 🟪
An abstract adapter class for receiving window events.

import java.awt.event.WindowEvent 🟪
A low-level event that indicates that a window has changed its status.

import java.io.BufferedReader 🟪

to read text from a file

import java.io.FileInputStream 🟪
obtains input bytes from a file in a file system.

import java.io.FileOutputStream 🟪
used for writing data to a File

import java.io.FileReader 🟪
to read streams of characters

import java.io.IOException 🟪
Signals that an I/O exception of some sort has occurred.

import java.io.ObjectInputStream 🟪
to read the ovjects written using an ObjectOutputStream

import java.io.ObjectOutputStream 🟪
writes primitive data types of java object

import java.util.Date 🟪
represents a specific instant in time

import java.util.Random 🟪
is used to generate a stream of random numbers

Testing

The test cases were first listed in a table and then tried out on the program. The table will list the test case number, field name, input, expected output, actual output and favourable outcome. The evidence of the testing, i.e the screen shots of the result from the testing is placed after the test cases table drawn below.

Test case table

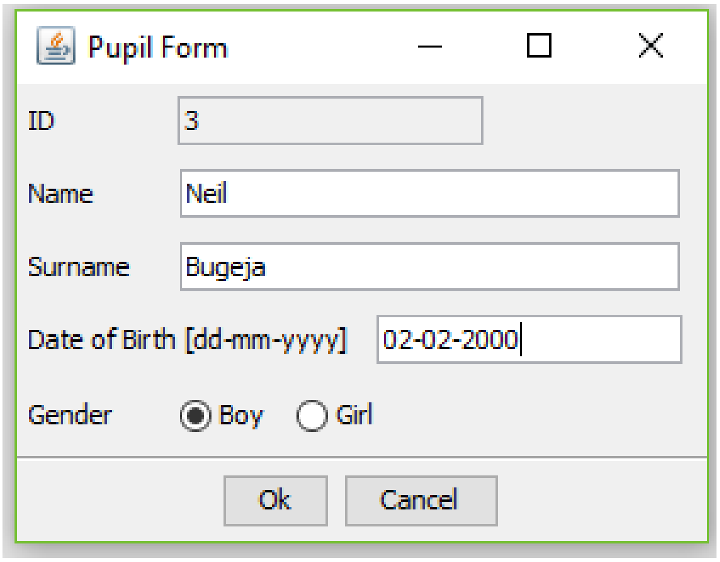
Test Case Number	**Field Name**	**Input**	**Expected Output**	**Actual Output**	**Favourable Outcome (Yes / No)**
1	Add Pupil	Correct data	Pupil added successfully	Prompt message box confirming that pupil is successfully added	Y
2	Add Pupil	Missing data	Prompt message box indicating where data is invalid	Prompt message box indicating where data is invalid	Y
3	Add Pupil	Symbols or numbers for Name and Surname	Prompt message box indicating where data is invalid	Prompt message box confirming that pupil is successfully added	N
4	Add Pupil	Letters or symbols for Date of Birth	Prompt message box indicating where data	Prompt message box indicating where data	Y

			is invalid	is invalid	
5	Add Pupil	Date of Birth not written in order requested	Prompt message box indicating where data is invalid	Prompt message box indicating where data is invalid	Y
6	Search Pupil	Registered Pupil ID	Pupil form enabling the editing of requested pupil	Pupil form enabling the editing of requested pupil	Y
7	Search Pupil	Registered Pupil Name and Surname	Table showing requested pupil's details	Table showing requested pupil's details	Y
8	Search Pupil	Pupil Name and Surname already registered twice	Pupil table contains all pupils with relevant Name and Surname	Pupil table contains all pupils with relevant Name and Surname	Y
9	Search Pupil	Successfully editing pupil	Details changed	Details changed	Y
10	Picture Quiz	All answers correct	Final score is 5	Final score is 5	Y
11	Picture Quiz	No answers correct	Final score is 0	Final score is 0	Y
12	Search Results	Show score	When clicked prompt box shows score	When clicked prompt box shows score	Y
13	Picture Quiz	Saving the score to an unregistered pupil ID	Pupil not registered prompt box	Pupil not registered prompt box	Y
14	Search Pupil	Pupil successfully deleted	Can't be found by search	Couldn't be found by search	Y
15	Questions Quiz	Answer is not case dependent	Final answer are correct irrelevant of case	Final answer were correct irrelevant of case	Y

16	Mixed Quiz	Shows both type of questions	Both type of questions will be shown	Both type of questions were shown	Y

Case 1

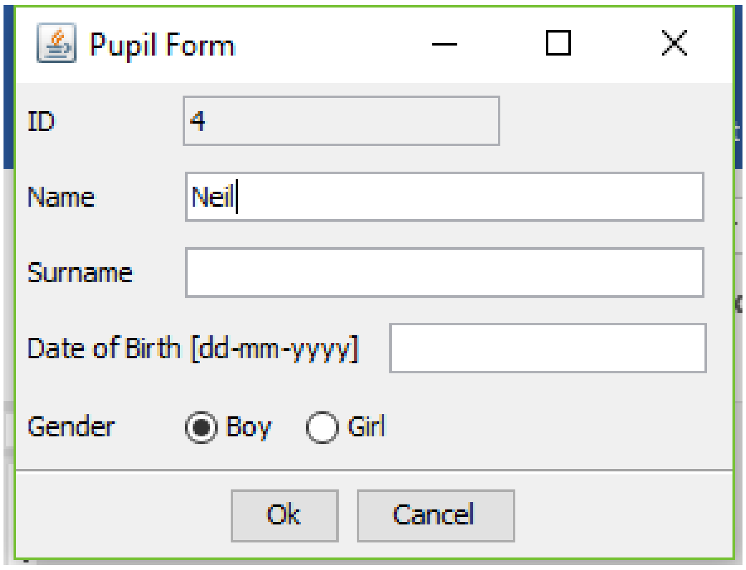
Adding pupil as expected


Pupil Form

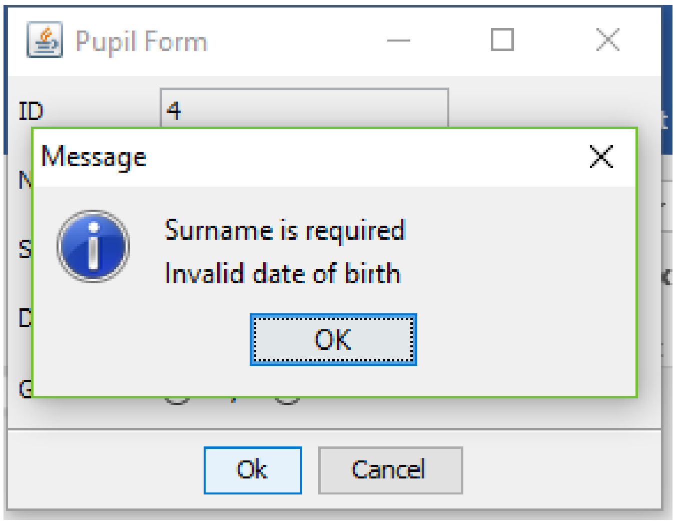
ID	3
Name	Neil
Surname	Bugeja
Date of Birth [dd-mm-yyyy]	02-02-2000
Gender	[✓ Boy] Girl

Case 2

Adding pupil with missing data


Pupil Form

ID	4
Name	Neil
Surname	
Date of Birth [dd-mm-yyyy]	
Gender	[✓ Boy] Girl


Pupil Form

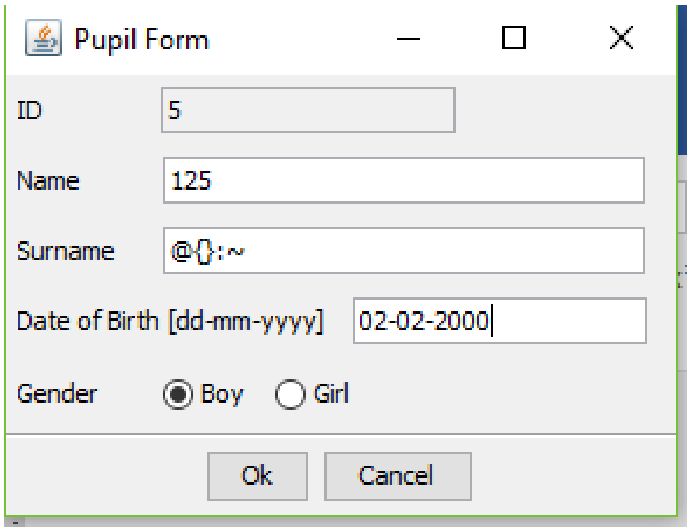
ID 4

Message

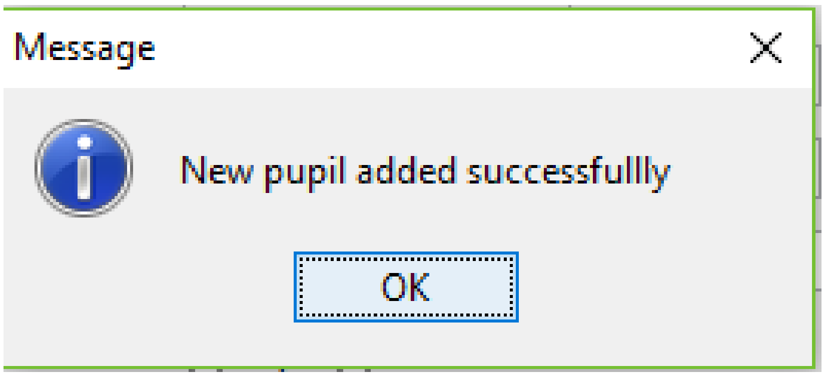
Surname is required
Invalid date of birth

Case 3

Adding Pupil with Symbols or Numbers as Name and Surname


Pupil Form

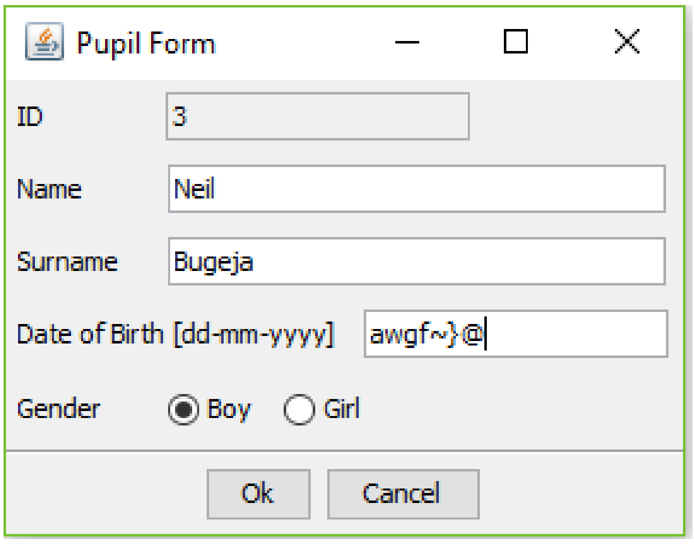
ID	5
Name	125
Surname	@{}:~
Date of Birth [dd-mm-yyyy]	02-02-2000
Gender	[✓ Boy] Girl


Message

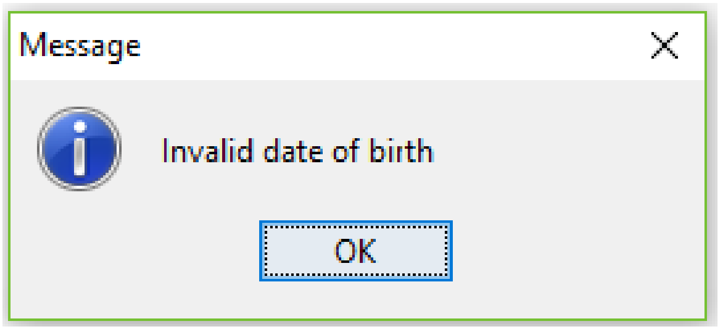
New pupil added successfullly

Case 4

Adding Pupil with Letters or Symbols as Date of Birth


Pupil Form

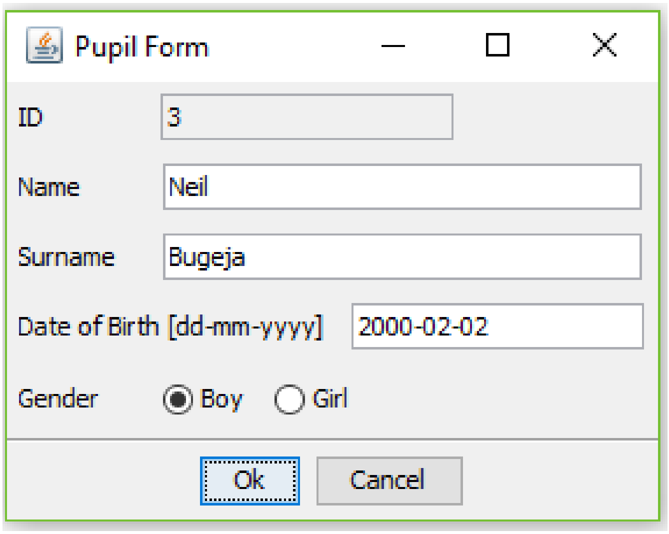
ID	3
Name	Neil
Surname	Bugeja
Date of Birth [dd-mm-yyyy]	awgf~}@
Gender	[✓ Boy] Girl


Message

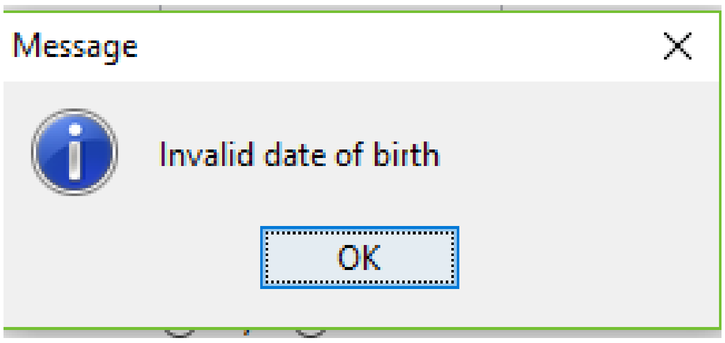
Invalid date of birth

Case 5

Adding Pupil with Date of Birth not as requested


Pupil Form

ID	3
Name	Neil
Surname	Bugeja
Date of Birth [dd-mm-yyyy]	2000-02-02
Gender	[✓ Boy] Girl

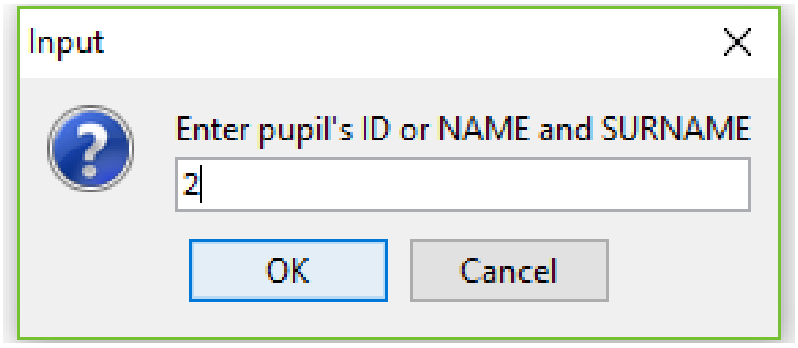


Message

Invalid date of birth

Case 6

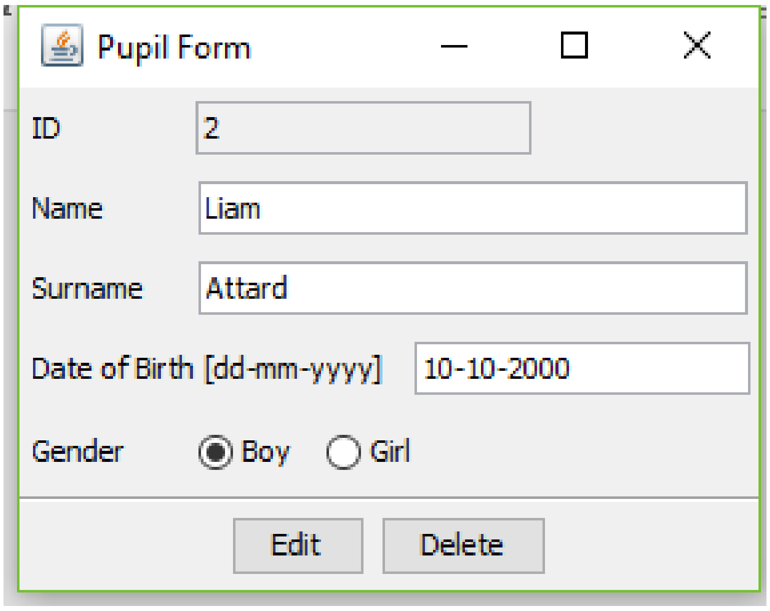
Search Pupil with ID



Input

Enter pupil's ID or NAME and SURNAME

2

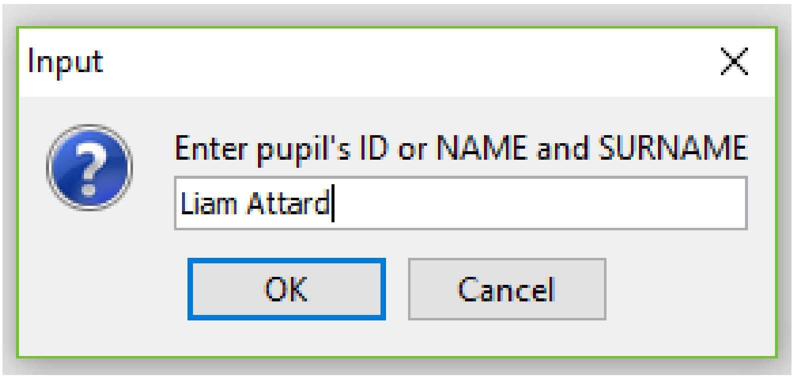


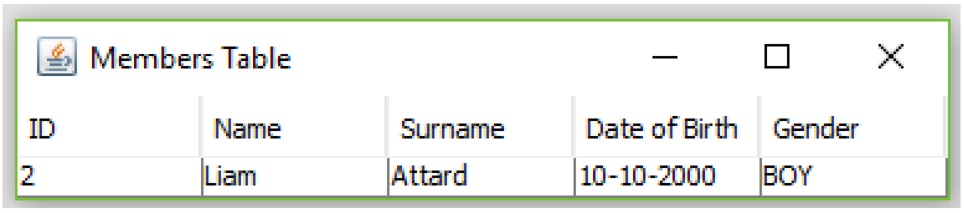
Pupil Form

ID	2
Name	Liam
Surname	Attard
Date of Birth [dd-mm-yyyy]	10-10-2000
Gender	[✓ Boy] Girl

Case 7

Search Pupil with Name and Surname

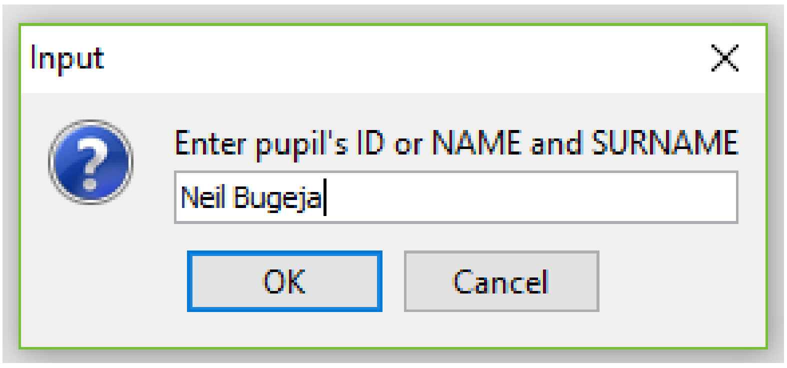


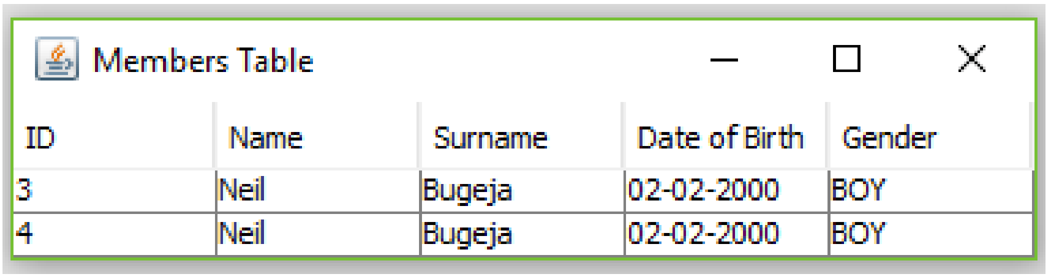

Members Table

ID	Name	Surname	Date of Birth	Gender
2	Liam	Attard	10-10-2000	BOY

Case 8

Search Pupil with Name and Surname registered twice

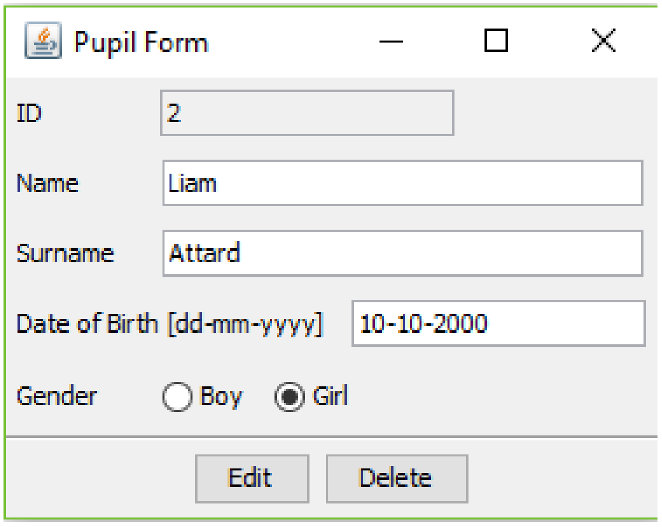



Members Table

ID	Name	Surname	Date of Birth	Gender
3	Neil	Bugeja	02-02-2000	BOY
4	Neil	Bugeja	02-02-2000	BOY

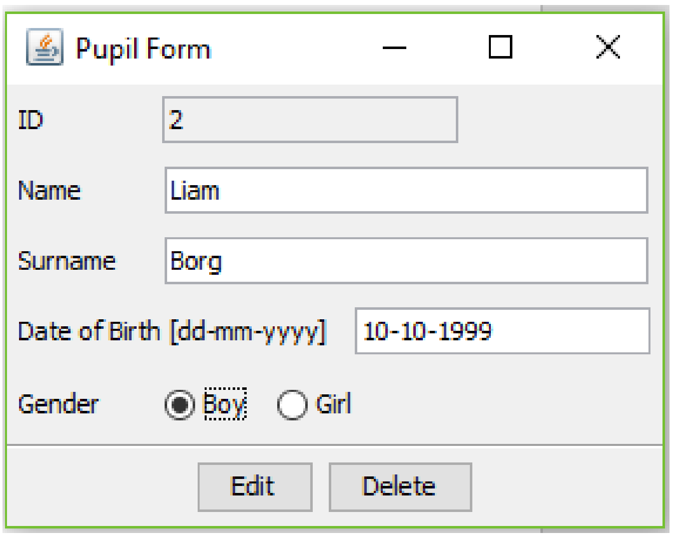
Case 9

Details successfully changed



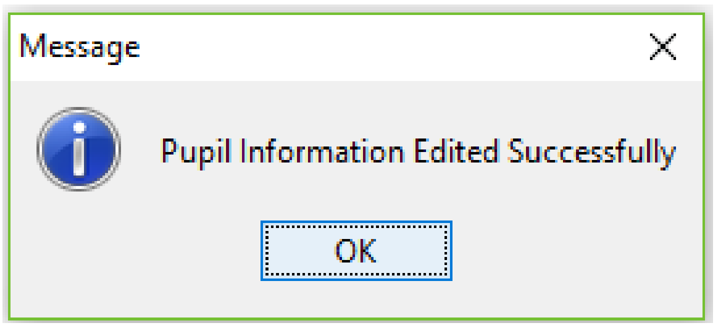
Pupil Form

ID	2
Name	Liam
Surname	Attard
Date of Birth [dd-mm-yyyy]	10-10-2000
Gender	○ Boy [✓ Girl]



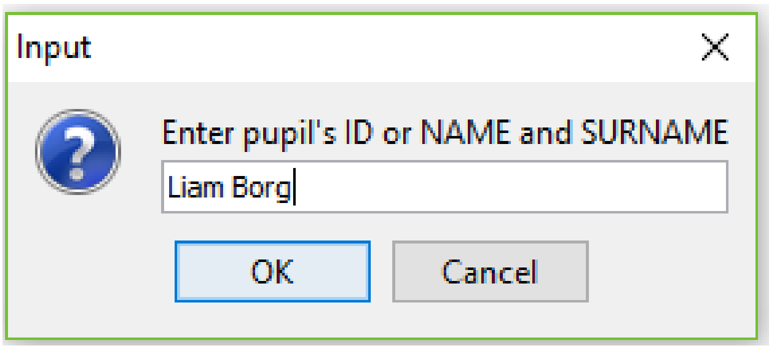
Pupil Form

ID	2
Name	Liam
Surname	Borg
Date of Birth [dd-mm-yyyy]	10-10-1999
Gender	[✓ Boy] ○ Girl



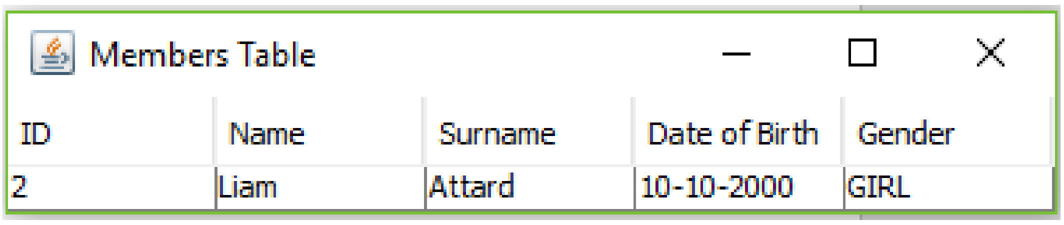
Message

Pupil Information Edited Successfully


Input

Enter pupil's ID or NAME and SURNAME

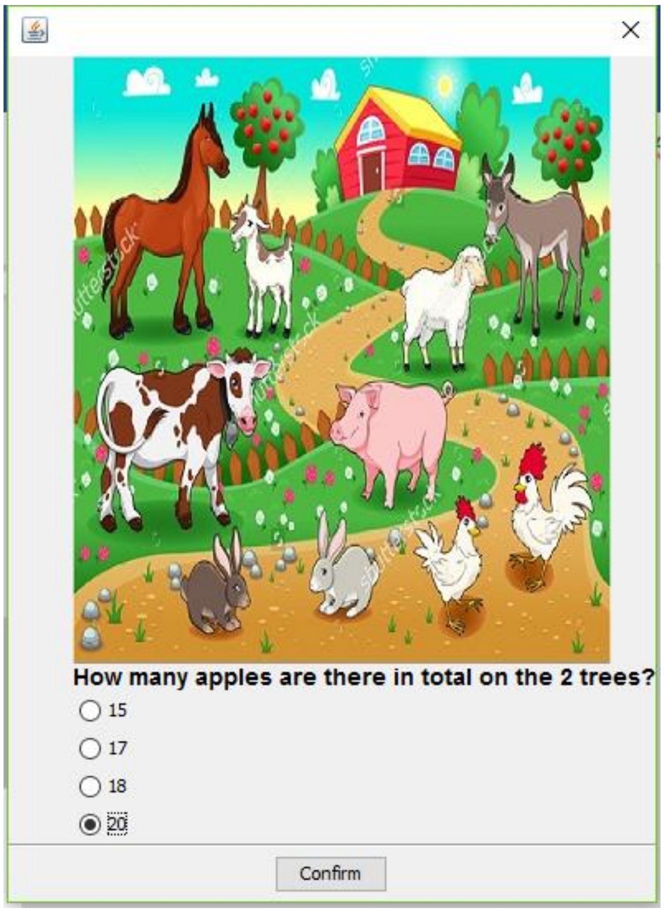
Liam Borg


Members Table

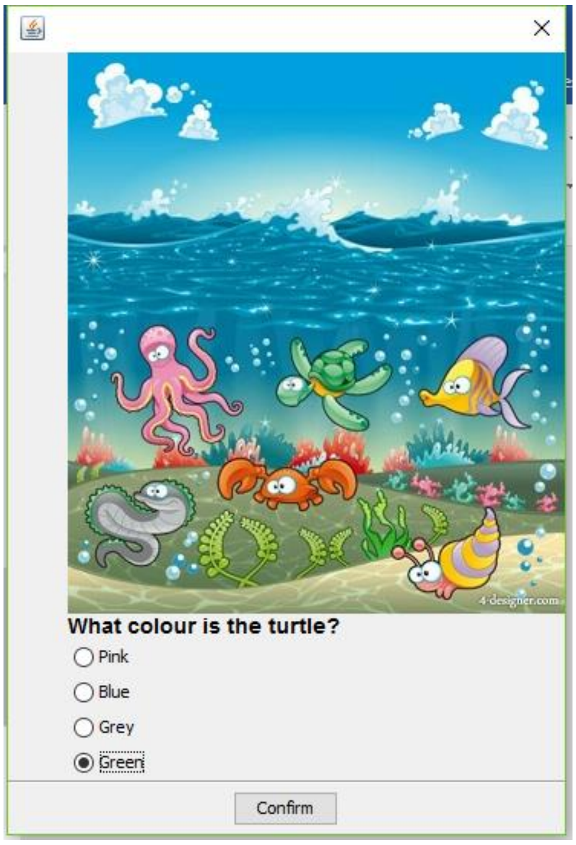
ID	Name	Surname	Date of Birth	Gender
2	Liam	Attard	10-10-2000	GIRL

Case 10

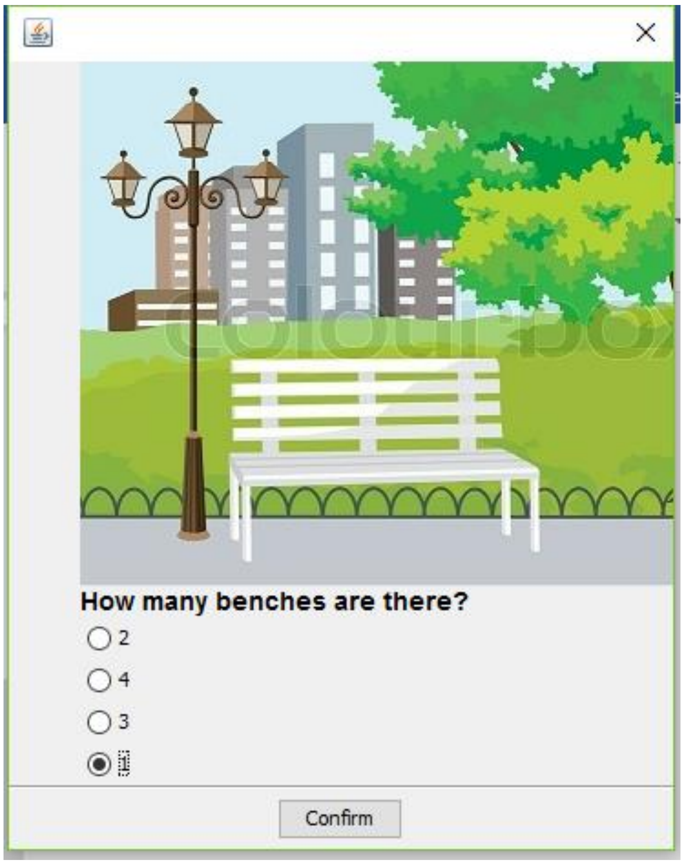
Picture Quiz all questions correct


How many apples are there in total on the 2 trees?

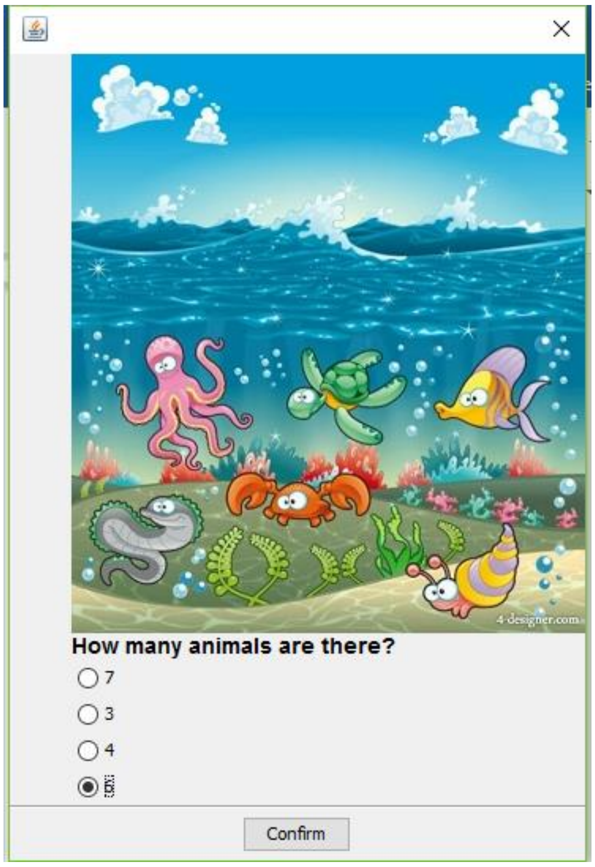
15 17 18 [✓ 20]


What colour is the turtle?

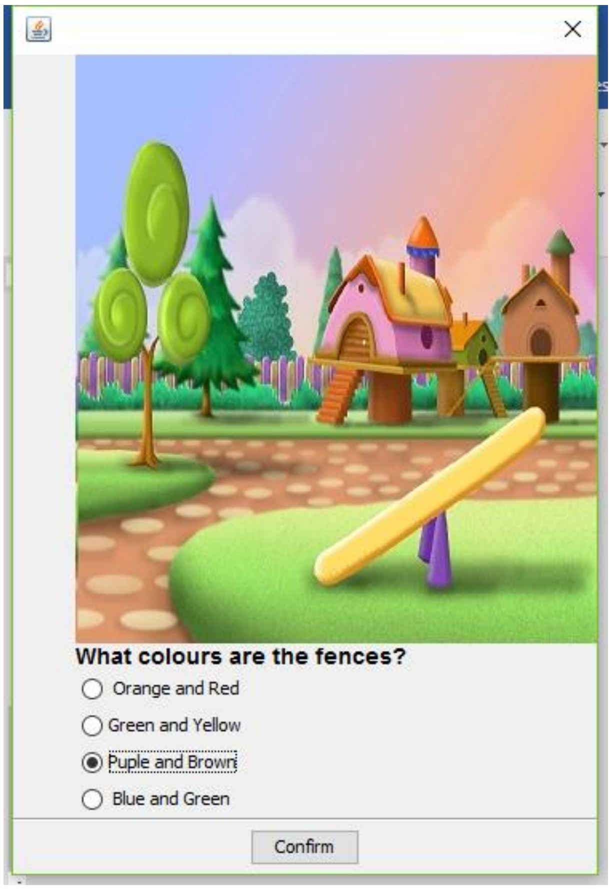
Pink Blue Grey [✓ Green]


How many benches are there?

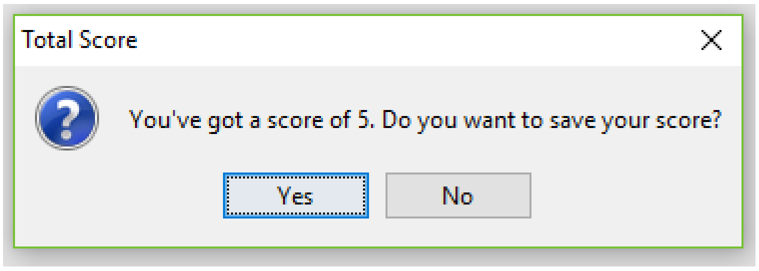
- ○ 2
- ○ 4
- ○ 3
- ● 1


How many animals are there?

- ○ 7
- ○ 3
- ○ 4
- ● 6


What colours are the fences?

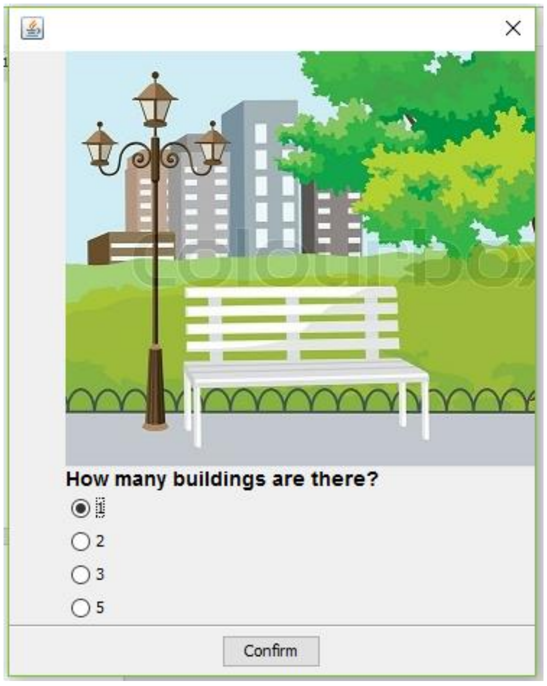
- ○ Orange and Red
- ○ Green and Yellow
- ● Puple and Brown
- ○ Blue and Green


Total Score

You've got a score of 5. Do you want to save your score?

Case 11

Picture Quiz with no questions correct



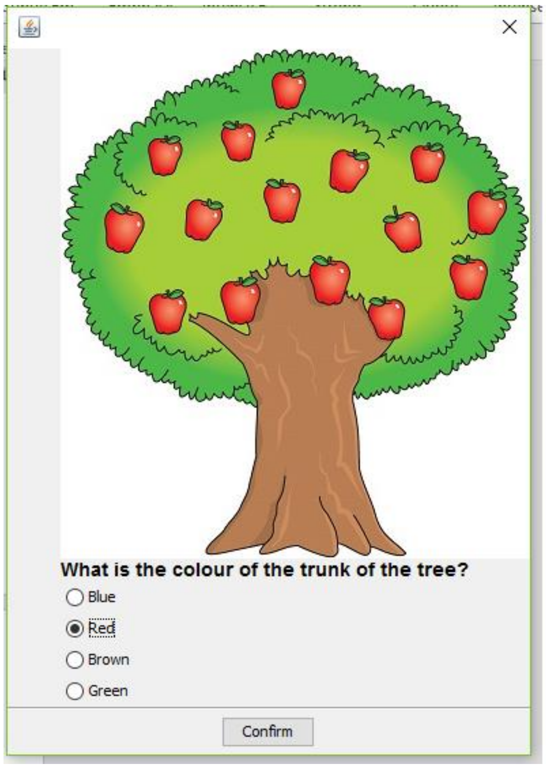
How many buildings are there?

[✓ 1]

2

3

5



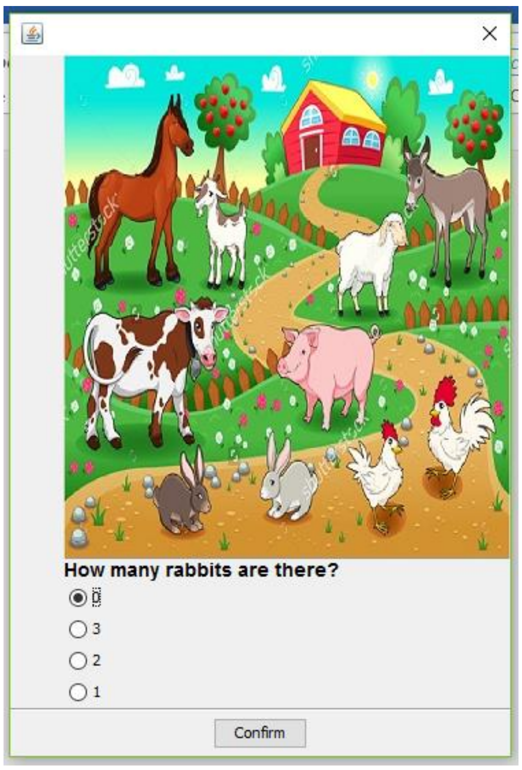
What is the colour of the trunk of the tree?

Blue

[✓ Red]

Brown

Green



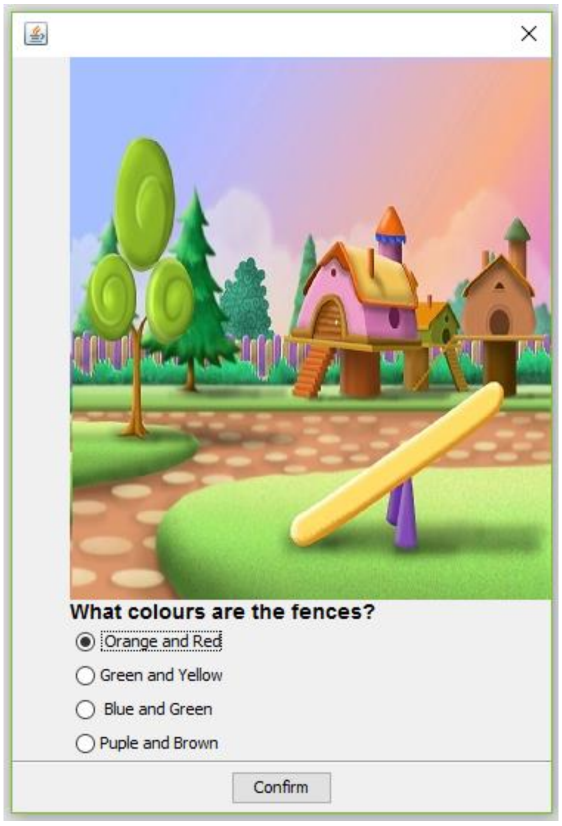
How many rabbits are there?

[✓ 0]

3

2

1



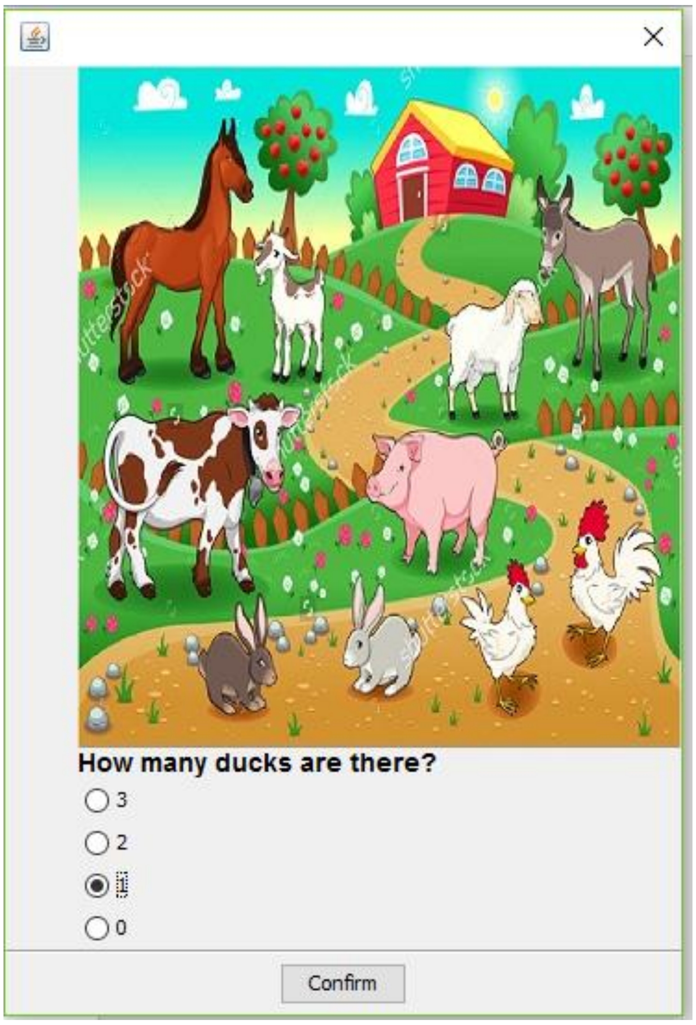
What colours are the fences?

[✓ Orange and Red]

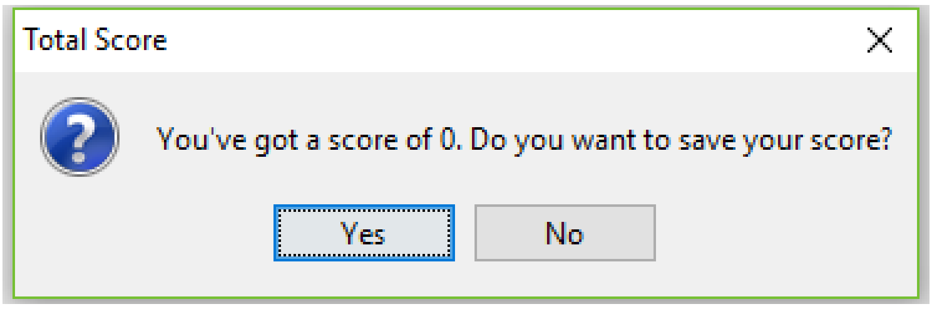
Green and Yellow

Blue and Green

Puple and Brown

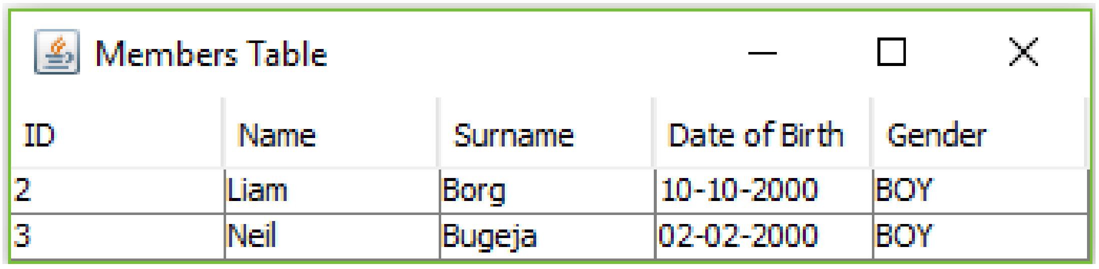

How many ducks are there?

- ○ 3
- ○ 2
- ● 1
- ○ 0

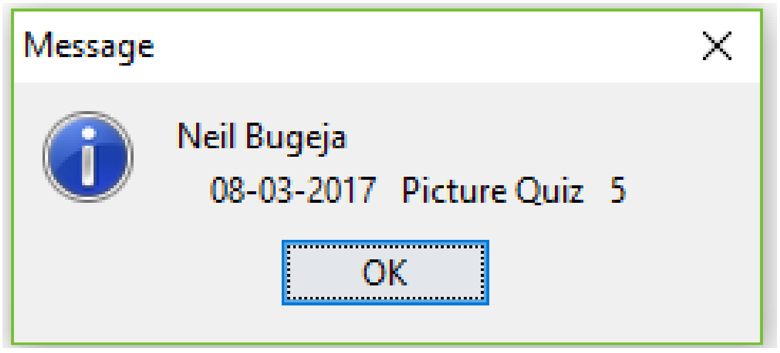

Total Score

You've got a score of 0. Do you want to save your score?

Case 12
Show score


Members Table

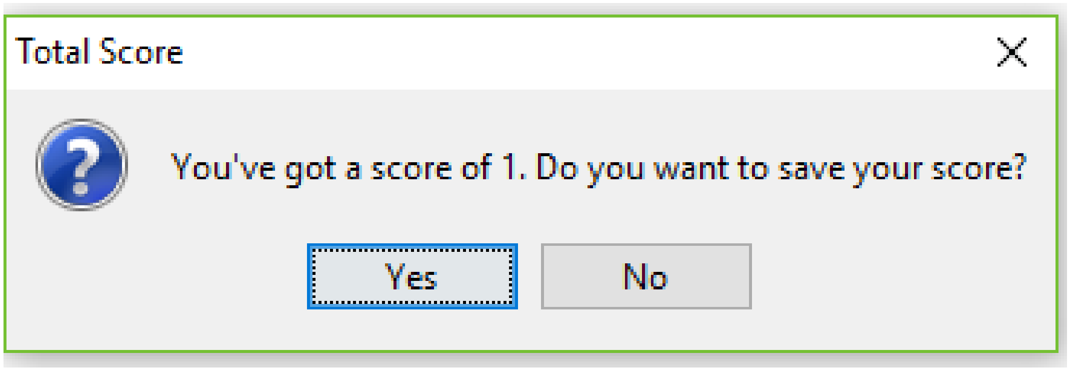
ID	Name	Surname	Date of Birth	Gender
2	Liam	Borg	10-10-2000	BOY
3	Neil	Bugeja	02-02-2000	BOY


Message

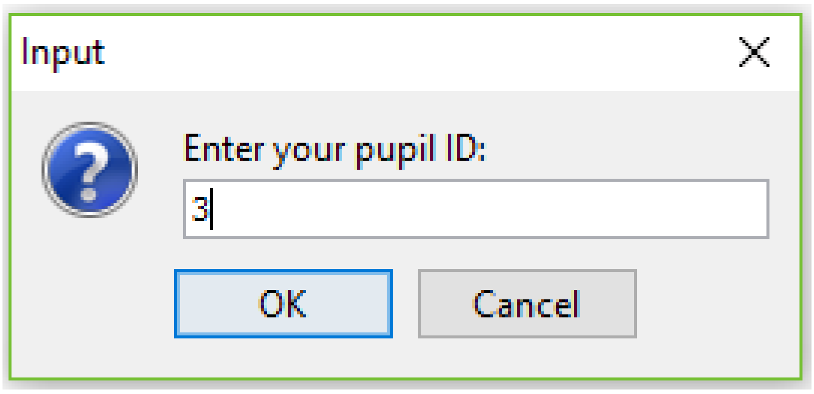
Neil Bugeja
08-03-2017 Picture Quiz 5

Case 13

Saving score to unregistered Pupil

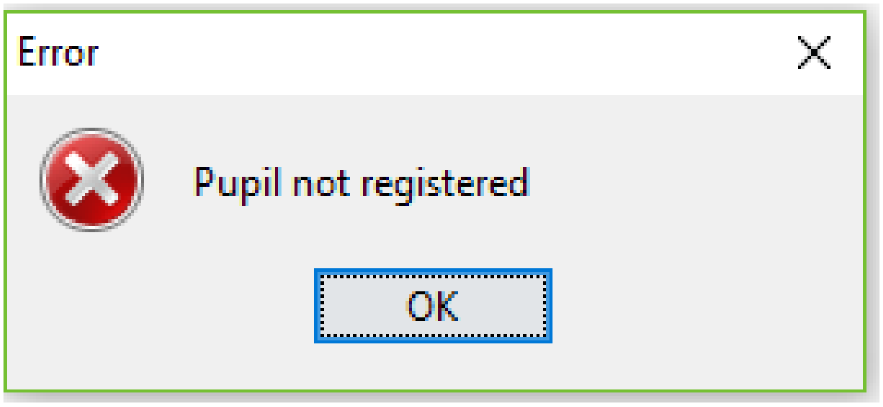

Total Score

You've got a score of 1. Do you want to save your score?


Input

Enter your pupil ID:

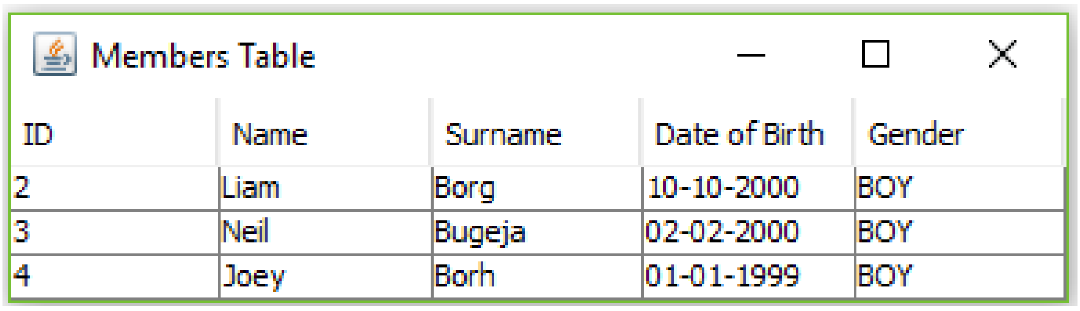
3


Error

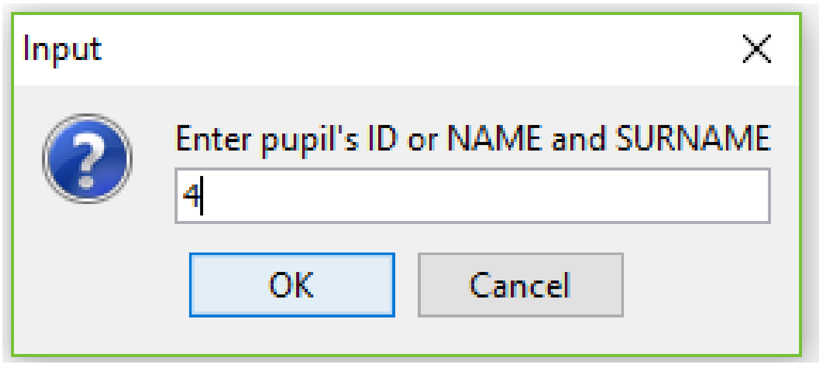
Pupil not registered

Case 14

Pupil successfully deleted

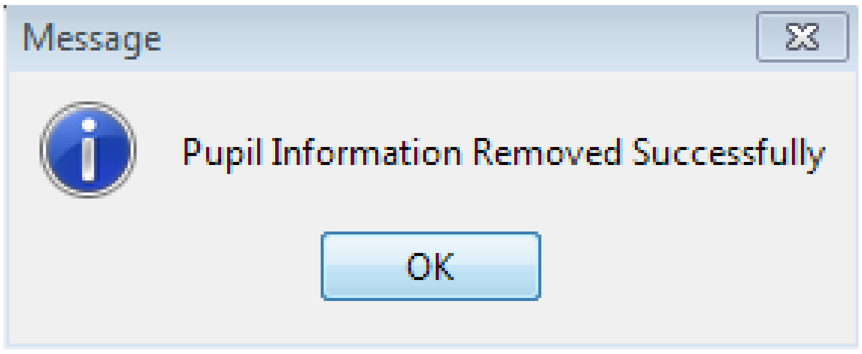

Members Table

ID	Name	Surname	Date of Birth	Gender
2	Liam	Borg	10-10-2000	BOY
3	Neil	Bugeja	02-02-2000	BOY
4	Joey	Borh	01-01-1999	BOY

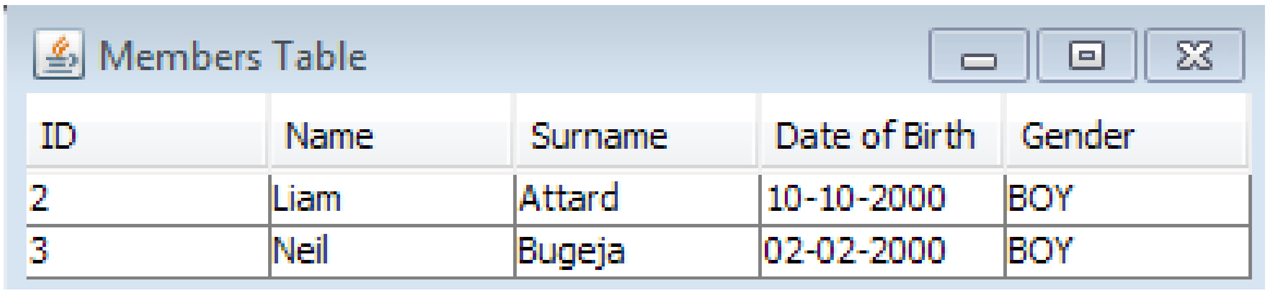

Input

Enter pupil's ID or NAME and SURNAME

4


Message

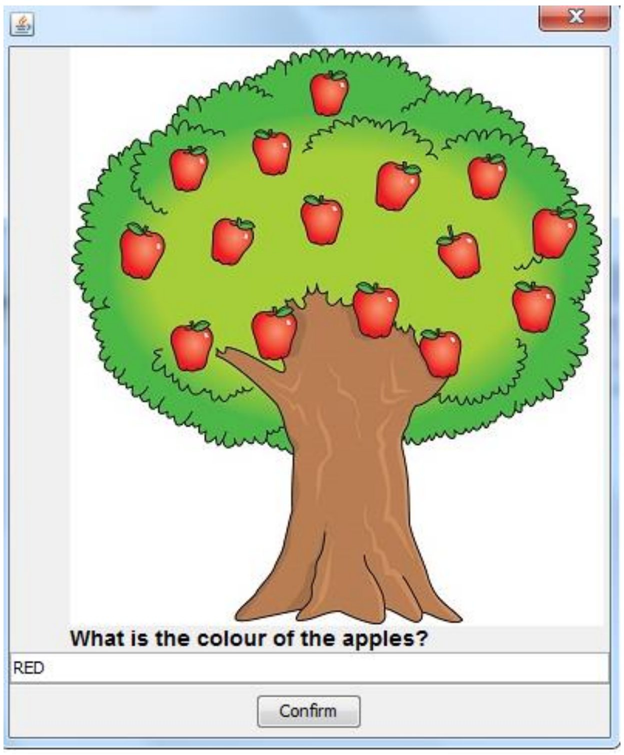
Pupil Information Removed Successfully


Members Table

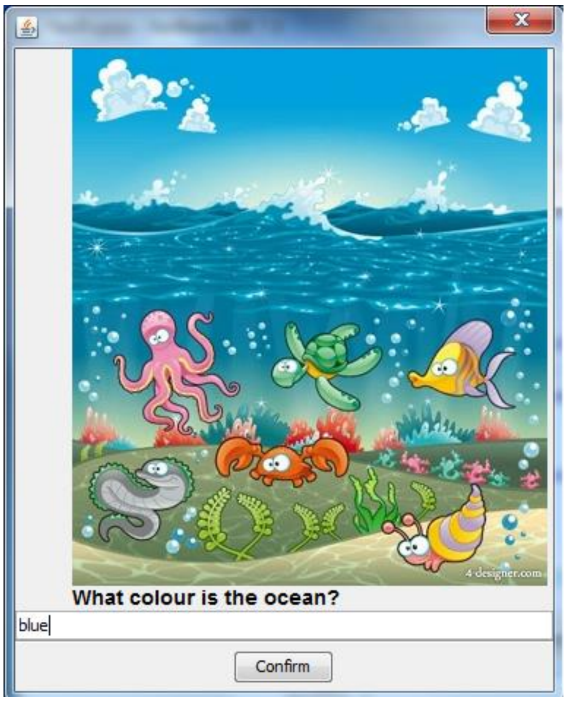
ID	Name	Surname	Date of Birth	Gender
2	Liam	Attard	10-10-2000	BOY
3	Neil	Bugeja	02-02-2000	BOY

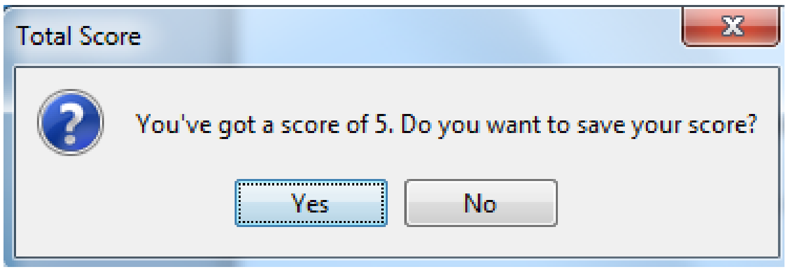
<u>Case 15</u>

<u>Questions Quiz not case sensitive</u>


What is the colour of the apples?

RED

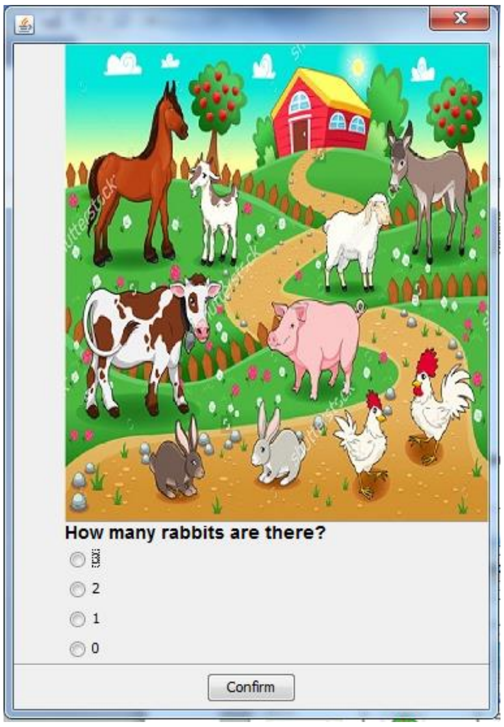



Total Score

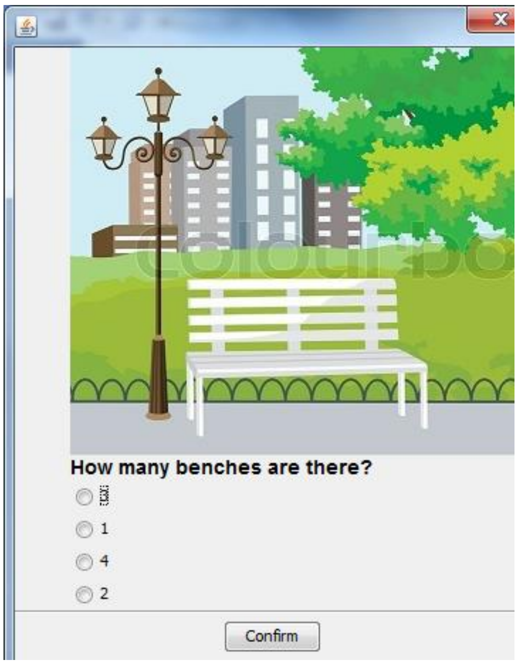
You've got a score of 5. Do you want to save your score?

Case 16

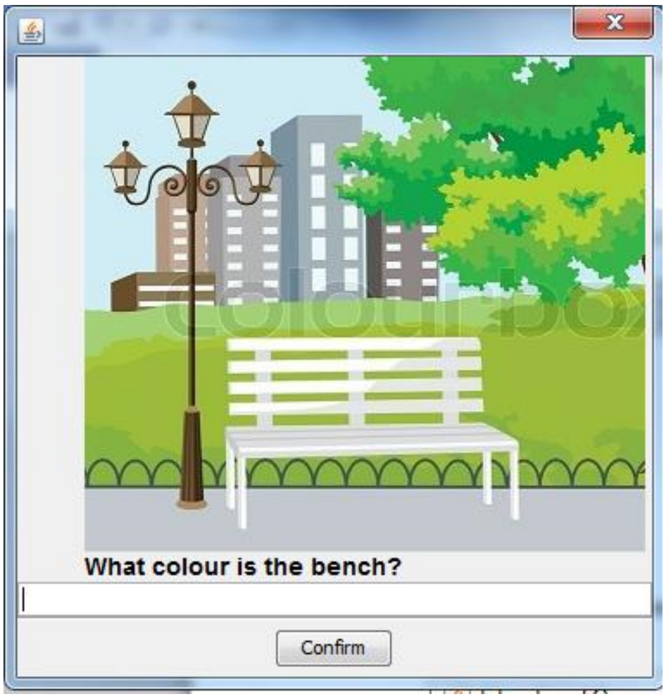
Mixed Questions

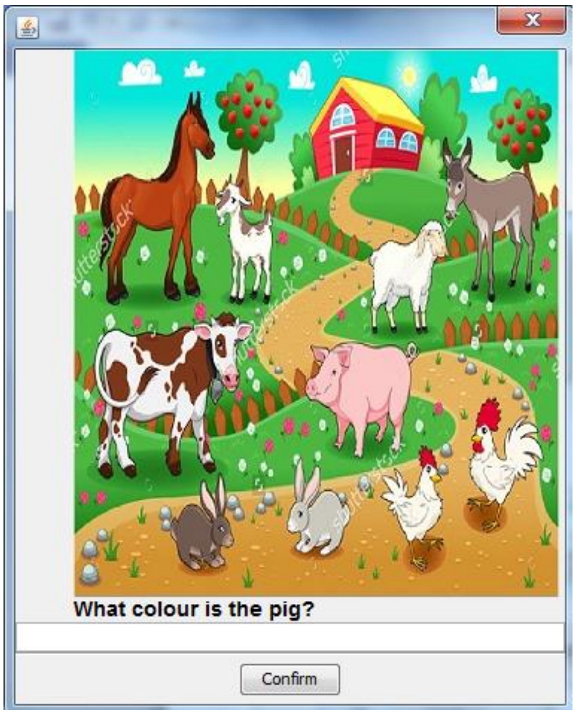

How many rabbits are there?

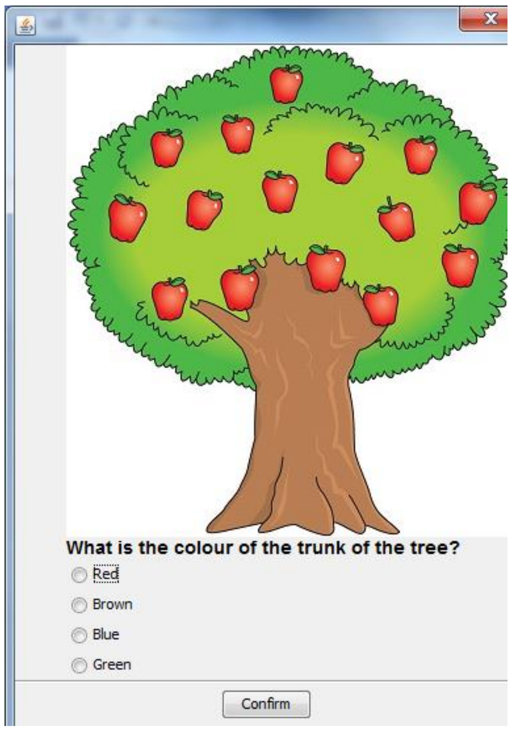
- ◯ ⍰
- ◯ 2
- ◯ 1
- ◯ 0


How many benches are there?

- ◯ ⍰
- ◯ 1
- ◯ 4
- ◯ 2


What colour is the bench?


What colour is the pig?



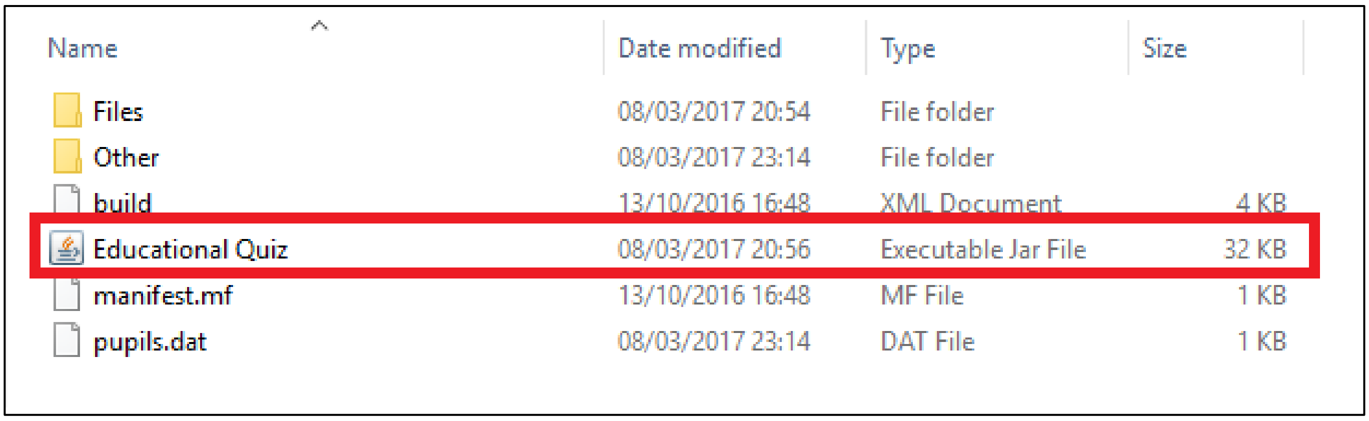
What is the colour of the trunk of the tree?

- ○ Red
- ○ Brown
- ○ Blue
- ○ Green

User Manual

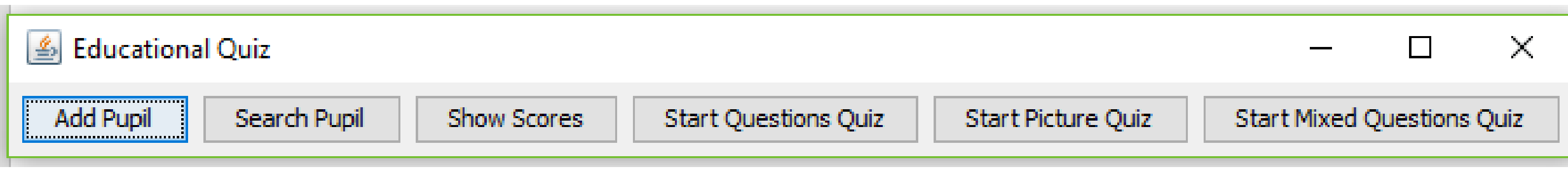
Running the program

Click on the Jar file outlined by the red box in order to access the quiz program



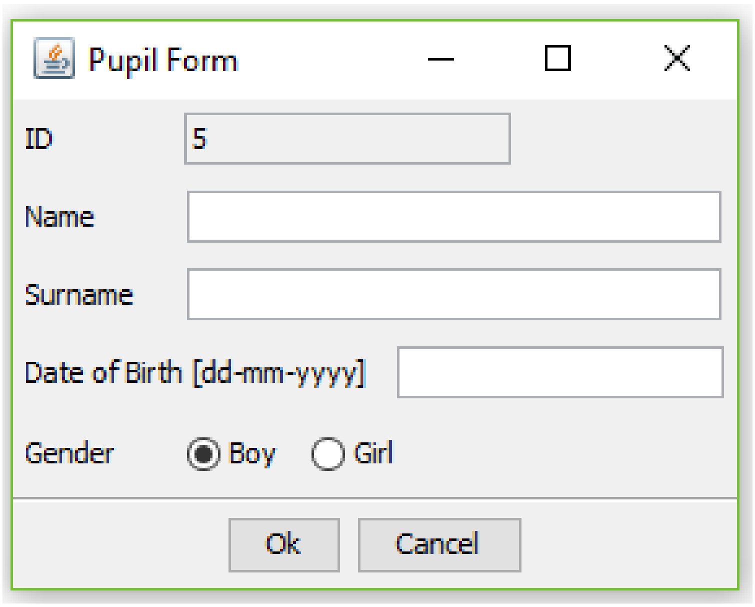
Name	Date modified	Type	Size
Files	08/03/2017 20:54	File folder	
Other	08/03/2017 23:14	File folder	
build	13/10/2016 16:48	XML Document	4 KB
Educational Quiz	08/03/2017 20:56	Executable Jar File	32 KB
manifest.mf	13/10/2016 16:48	MF File	1 KB
pupils.dat	08/03/2017 23:14	DAT File	1 KB

Main Menu



Once the program is running, the user shall be greeted by the ***Main Menu***. Here the user can decide what he/she wished to do with the program.

Add Pupil


Pupil Form

ID 5

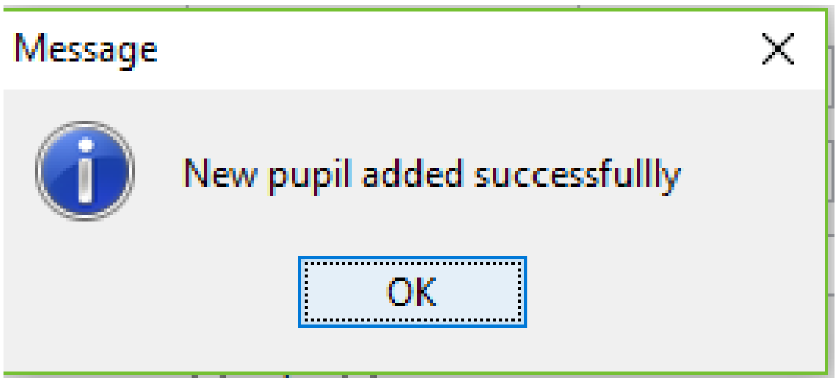
Name

Surname

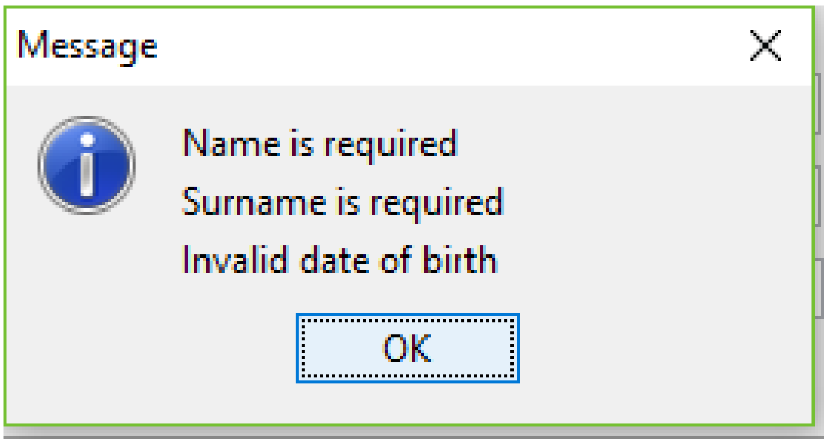
Date of Birth [dd-mm-yyyy]

Gender [● Boy] ○ Girl

All text fields must be filled. If not, then the pupil will not be registered correctly, if at all. Once all the details are filled in as instructed the user should click on the ok button. The program will notify the user weather or not the pupil was added successfully.


Message

New pupil added successfullly

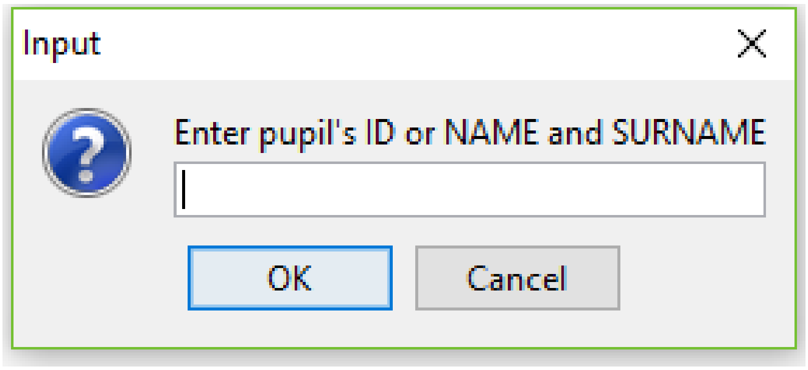

Message

Name is required

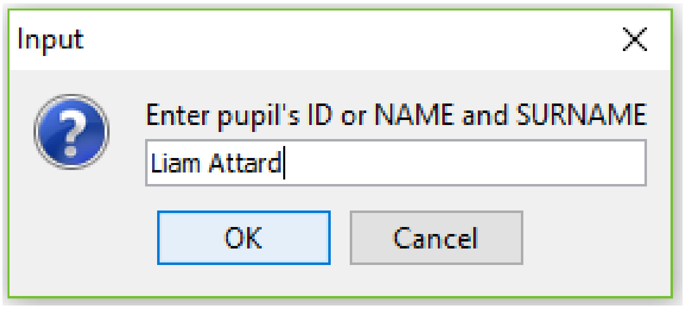
Surname is required

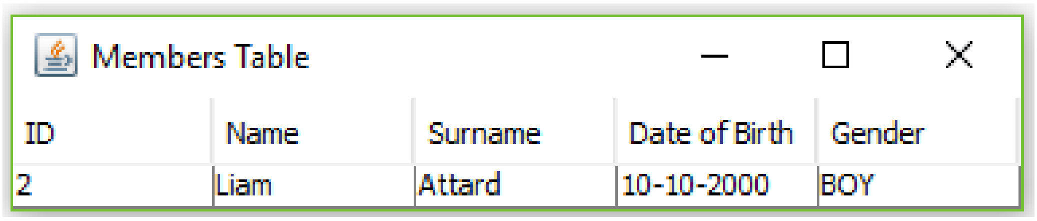
Invalid date of birth

Search Pupil



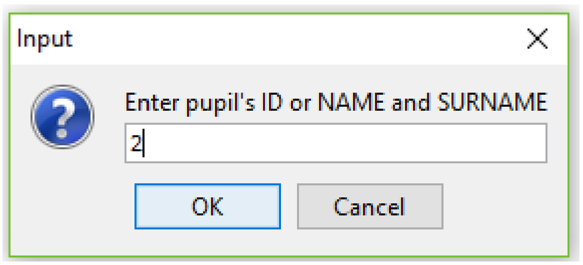
Here the user can search to either view the details of a pupil or to edit them. Should the user enter ***Name and Surname***, then a table following to the one below will appear:

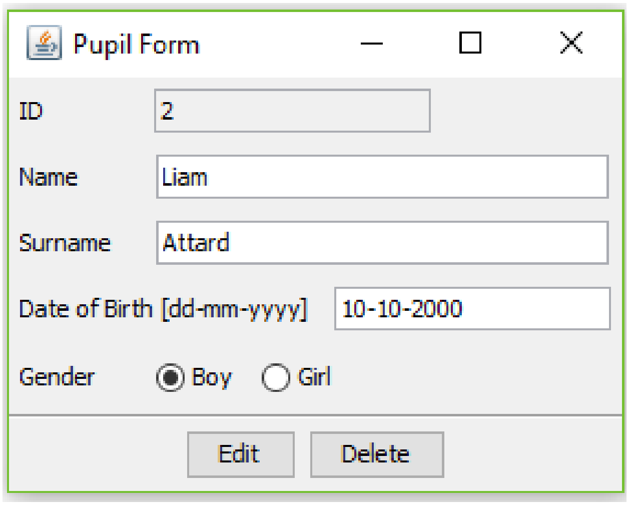



Members Table

ID	Name	Surname	Date of Birth	Gender
2	Liam	Attard	10-10-2000	BOY

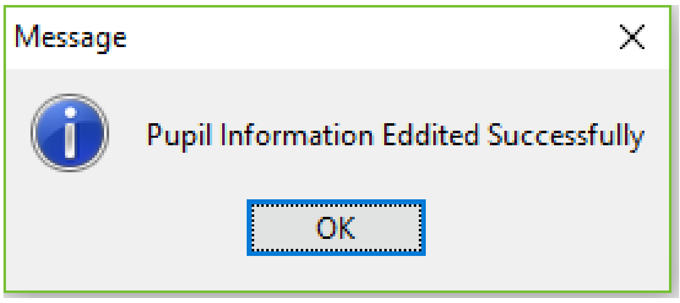
Here various details can be viewed, however none of them can be edited. To edit the user should instead take note of the ***ID*** and search with it.





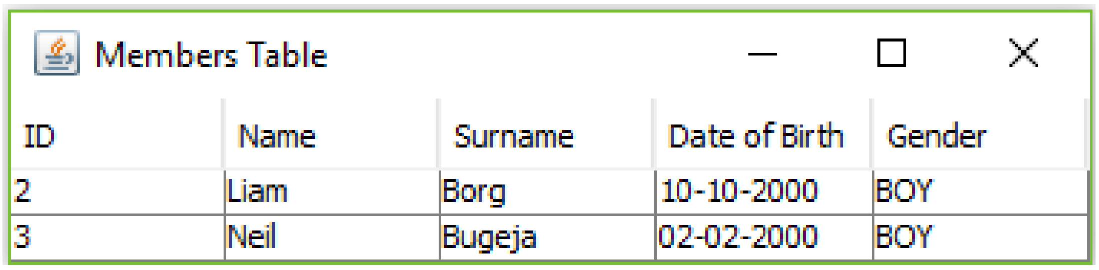
ID	2
Name	Liam
Surname	Attard
Date of Birth [dd-mm-yyyy]	10-10-2000
Gender	[✓ Boy] Girl

Here the user can either edit the details or entirely delete the user. Once the user clicks edit or delete, the program will notify the user that the details have been changed / deleted.



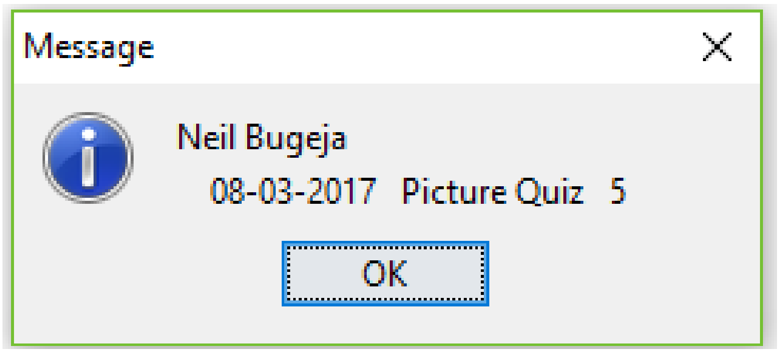
Pupil Information Eddited Successfully

Show Score



ID	Name	Surname	Date of Birth	Gender
2	Liam	Borg	10-10-2000	BOY
3	Neil	Bugeja	02-02-2000	BOY

Here the user can view all of the pupil’s scores. All one has to do is simply click on who he wishes to see the score of and the following pop up will display, containing the scores off all the pupil’s attempts along with their respective date.

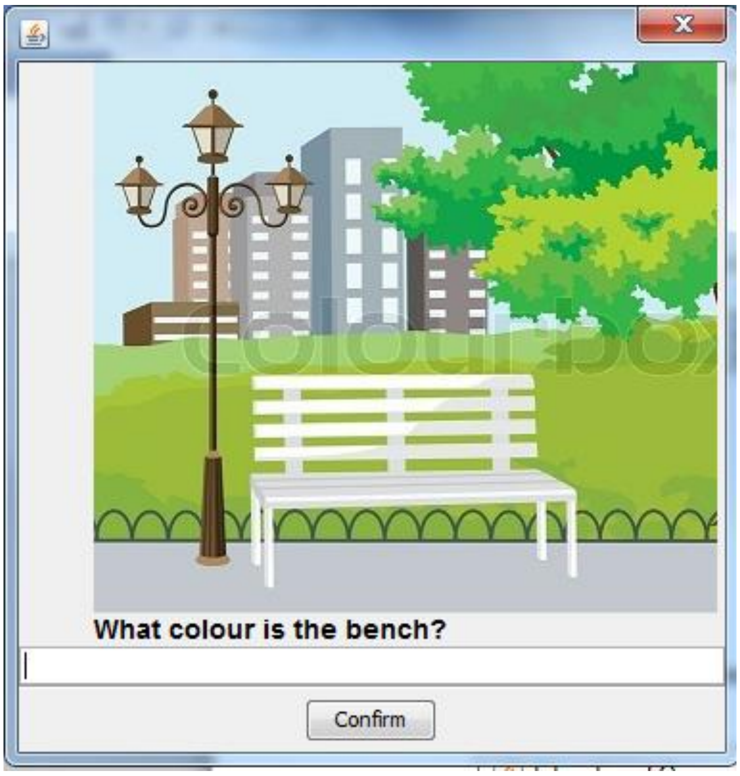


Message

Neil Bugeja

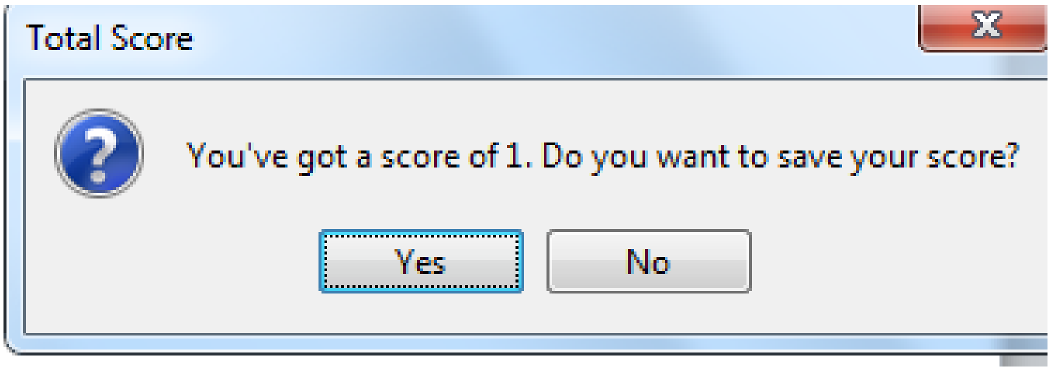
08-03-2017 Picture Quiz 5

Start Question Quiz



What colour is the bench?

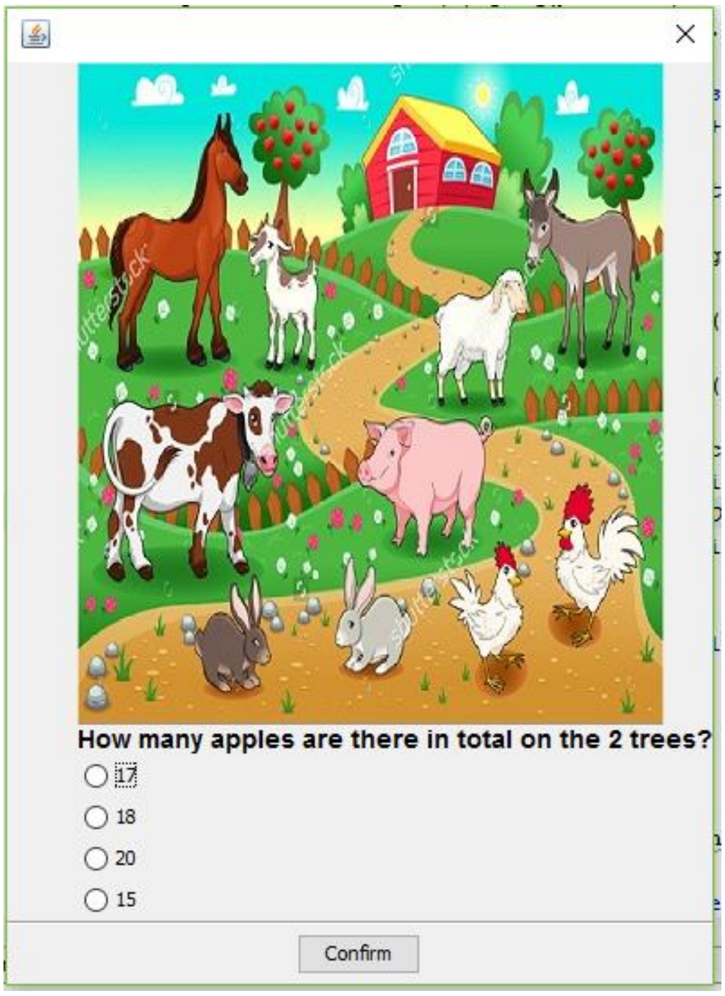
The pupil will be asked a serious of random questions. These questions are all picture based and of type question and answer. The pupil should input an answer which he/she believes is correct. At the end the pupil will be issued a final mark and asked if he/she wishes to save the mark.



Total Score

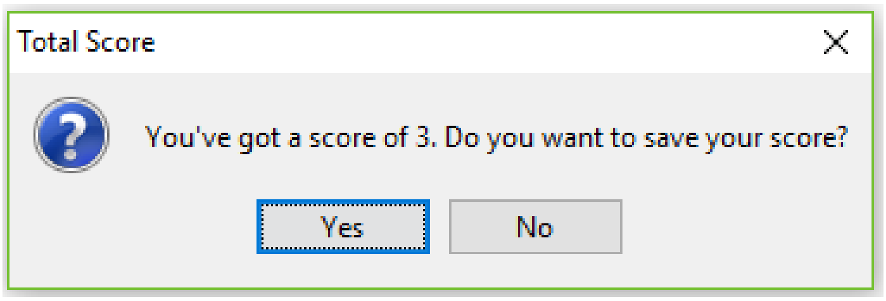
You've got a score of 1. Do you want to save your score?

Start Picture Quiz



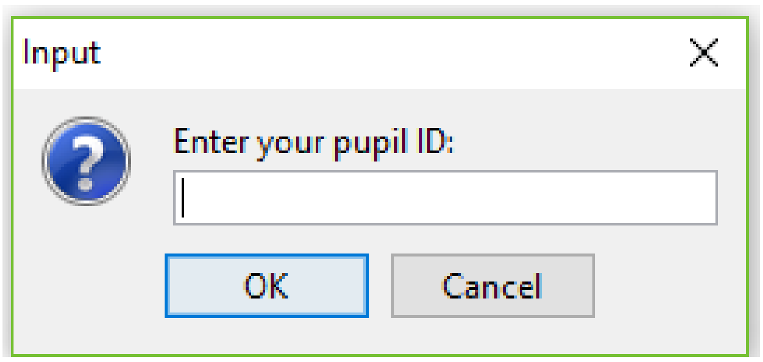
How many apples are there in total on the 2 trees?

The pupil will be asked a serious of random questions. These questions are all picture based and multiple choice. The pupil should mark which answer he/she believes is correct. At the end the pupil will be issued a final mark and asked if he/she wishes to save the mark.


Total Score

You've got a score of 3. Do you want to save your score?

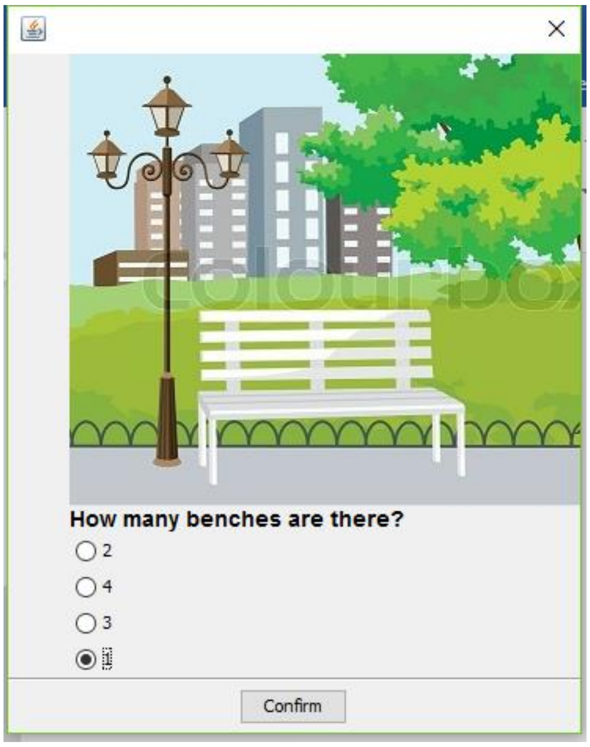
Should the user click on no then the main menu will appear. Else a pop up will show up, asking for the user's student ID to save the score.


Input

Enter your pupil ID:

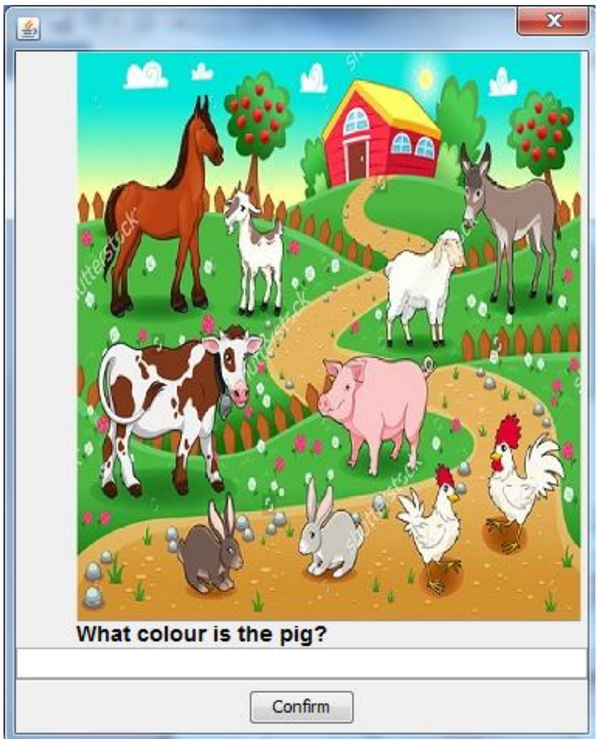
<u>Mixed Quiz</u>

The pupil will be asked a serious of random questions. These questions contain a mix between multiple choice type questions and question and answer type questions. The questions are randomly chosen. The pupil should mark which answer he/she believes is correct. At the end the pupil will be issued a final mark and asked if he/she wishes to save the mark.



How many benches are there?

- 2
- 4
- 3
- [✓ 1]



What colour is the pig?

Comments and Conclusions

In conclusion, after thoroughly assessing the program, it could be determined that the final product came out as planned. All the specifications of the computerized quiz have been met. Through the use of the Java Swing features which enable a GUI interface the user was more comfortable as the program was rather straight-forward and much easier to navigate through.

References

https://www.tutorialspoint.com/java/java_encapsulation.htm

https://docs.oracle.com/javase/7/docs/api/java/awt/BorderLayout.html

http://docs.oracle.com/javase/7/docs/api/javax/swing/ListSelectionModel.html

https://docs.oracle.com/javase/7/docs/api/javax/swing/event/ListSelectionEvent.html

https://docs.oracle.com/javase/7/docs/api/javax/swing/event/ListSelectionListener.html

https://docs.oracle.com/javase/7/docs/api/javax/swing/JDialog.html

https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html

https://docs.oracle.com/javase/7/docs/api/java/awt/event/WindowAdapter.html

https://docs.oracle.com/javase/7/docs/api/java/io/IOException.html

http://whatis.techtarget.com/fileformat/JPG-JPEG-bitmap

APPENDIX: Code Listing

DisplayPupilsTable

```
import java.awt.BorderLayout;

import java.util.ArrayList;

import javax.swing.JFrame;

import javax.swing.JTable;

import javax.swing.ListSelectionModel;

import javax.swing.event.ListSelectionEvent;

import javax.swing.event.ListSelectionListener;



public class DisplayPupilsTable extends JFrame  implements ListSelectionListener {



    private ArrayList<Pupil> toShow;

    private JTable t;
```

```
public DisplayPupilsTable(ArrayList<Pupil> toShow) {

    setTitle("Members Table");

    this.toShow = toShow;

    String[] headers = {"ID", "Name", "Surname", "Date of Birth", "Gender", };

    String[][] content = new String[toShow.size()][5];



    for (int i = 0; i < toShow.size(); i++) {

        content[i][0] = toShow.get(i).getStudentid() + "";

        content[i][1] = toShow.get(i).getName();

        content[i][2] = toShow.get(i).getSurname();

        content[i][3] = toShow.get(i).getDob();

        content[i][4] = toShow.get(i).getGender();

    }



    t = new JTable(content, headers);
```

```
        setLayout(new BorderLayout());

        add(t, BorderLayout.CENTER);

        add(t.getTableHeader(), BorderLayout.NORTH);

        pack();

        setVisible(true);



        ListSelectionModel cellSelectionModel = t.getSelectionModel();


cellSelectionModel.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);

        cellSelectionModel.addListSelectionListener(this);

    }



    public void valueChanged(ListSelectionEvent e) {

        if (e.getValueIsAdjusting()) {

            Pupil pupil = toShow.get(t.getSelectedRow());

            pupil.showScores();
        }

    }

}
```

MultipleChoiceQuestion

```
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.Random;
import javax.swing.BoxLayout;
import javax.swing.ButtonGroup;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JSeparator;

public class MultipleChoiceQuestion extends Question {

    private String picPath;
    private String[] answers = new String[4];

    private JRadioButton answerLabel1;
    private JRadioButton answerLabel2;
    private JRadioButton answerLabel3;
    private JRadioButton answerLabel4;

    public MultipleChoiceQuestion(String s) {
        String[] x = s.split("#");
        picPath = "Files//Pics//" + x[0];
        setQuestionText(x[1]);
        setAnswerText(x[2]);
        for (int i = 2; i <= 5; i++) {
            answers[i - 2] = x[i];
        }
    }

    public boolean playQuestion(JFrame f) {
        showQuestion(f);
        return true;
    }

    public void showQuestion(JFrame f) {
        JButton confirmButton = new JButton("Confirm");
        confirmButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                boolean answerCorrect = false;
                if (answerLabel1.isSelected()) {
                    if (answers[0].equals(getAnswerText())) {
                        answerCorrect = true;
                    }
                }

                if (answerLabel2.isSelected()) {
                    if (answers[1].equals(getAnswerText())) {
                        answerCorrect = true;
                    }
                }

                if (answerLabel3.isSelected()) {
                    if (answers[2].equals(getAnswerText())) {
                        answerCorrect = true;
                    }
                }

                if (answerLabel4.isSelected()) {
                    if (answers[3].equals(getAnswerText())) {
                        answerCorrect = true;
                    }
                }

                if (answerCorrect) {
                    setScore(1);
                } else {
                    setScore(0);
                }
                questionFrame.dispose();
            }
        });




        // To shuffle the possable answers
        Random randomGenerator = new Random();
        for(int i = 0;i < 10;i++){
            int r1 = randomGenerator.nextInt(4);
            int r2 = randomGenerator.nextInt(4);
            String temp;
```

```
            temp = answers[r1];
            answers[r1] = answers[r2];
            answers[r2] = temp;
        }
        //

        Font myFont = new Font("Arial", Font.BOLD, 14);
        JLabel picLabel = new JLabel(new ImageIcon(picPath));
        JLabel questionLabel = new JLabel(getQuestionText());
        questionLabel.setFont(myFont);
        answerLabel1 = new JRadioButton(answers[0]);
        answerLabel2 = new JRadioButton(answers[1]);
        answerLabel3 = new JRadioButton(answers[2]);
        answerLabel4 = new JRadioButton(answers[3]);

        ButtonGroup bg = new ButtonGroup();
        bg.add(answerLabel1);
        bg.add(answerLabel2);
        bg.add(answerLabel3);
        bg.add(answerLabel4);

        questionFrame = new JDialog(f, true);
        questionFrame.setLayout(new     BoxLayout(questionFrame.getContentPane(),
BoxLayout.Y_AXIS));
        questionFrame.add(picLabel);
        questionFrame.add(questionLabel);
        questionFrame.add(answerLabel1);
        questionFrame.add(answerLabel2);
        questionFrame.add(answerLabel3);
        questionFrame.add(answerLabel4);
        questionFrame.add(new JSeparator());
        JPanel p = new JPanel(new FlowLayout());
        p.add(confirmButton);
        questionFrame.add(p);

        questionFrame.pack();
        questionFrame.setVisible(true);
    }

}
```

PupilForm

```
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import javax.swing.BoxLayout;
import javax.swing.ButtonGroup;
import javax.swing.JButton;
import javax.swing.JComponent;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JRadioButton;
import javax.swing.JSeparator;
import javax.swing.JTextField;
import javax.swing.UIManager;

public class PupilForm extends JFrame implements ActionListener {
```

```
private JTextField idField = new JTextField(15);
private JTextField nameField = new JTextField(25);
private JTextField surnameField = new JTextField(25);
private JTextField dobField = new JTextField(15);
private JRadioButton maleRB = new JRadioButton("Boy");
private JRadioButton femaleRB = new JRadioButton("Girl");
private JButton okButton = new JButton("Ok");
private JButton cancelButton = new JButton("Cancel");

private ArrayList<Pupil> allPupils;
private Pupil pupilToEdit = null;

public PupilForm(ArrayList<Pupil> allPupils) {
    this.allPupils = allPupils;
    setTitle("Pupil Form");
    setLayout(new BoxLayout(getContentPane(), BoxLayout.Y_AXIS));
    add(getFormLine(idField, "ID              "));
    add(getFormLine(nameField, "Name          "));
    add(getFormLine(surnameField, "Surname     "));
    add(getFormLine(dobField, "Date of Birth [dd-mm-yyyy]  "));
    add(getFormLine(maleRB, femaleRB, "Gender      "));
    add(new JSeparator());
    add(getFormLine(okButton, cancelButton));

    maleRB.setSelected(true);
    ButtonGroup bg = new ButtonGroup();
    bg.add(maleRB);
    bg.add(femaleRB);
    okButton.addActionListener(this);
    cancelButton.addActionListener(this);
    idField.setText(getNextStudentId() + "");
    idField.setEditable(false);
```

```
        pack();
    }

    public void usedInEditMode(Pupil p) {
        pupilToEdit = p;
        idField.setText(pupilToEdit.getStudentid() + "");
        nameField.setText(pupilToEdit.getName());
        surnameField.setText(pupilToEdit.getSurname());
        dobField.setText(pupilToEdit.getDob());
        if (pupilToEdit.getGender().equalsIgnoreCase("BOY")) {
            maleRB.setSelected(true);
            femaleRB.setSelected(false);
        } else {
            maleRB.setSelected(false);
            femaleRB.setSelected(true);
        }
        okButton.setText("Edit");
        cancelButton.setText("Delete");
    }

    private int getNextStudentId() {
        if (allPupils.size() == 0) {
            return 1;
        } else {
            return allPupils.get(allPupils.size() - 1).getStudentid() + 1;
        }
    }

    public boolean isDateValid(String date) {
        try {
            SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
            sdf.setLenient(false);
            sdf.parse(date);
        } catch (Exception e) {
            return false;
        }
        return true;
    }

    private boolean doDataValidation() {
        String errorMsg = "";
        if (nameField.getText().trim().length() == 0) {
            errorMsg = errorMsg + "Name is required\n";
        }

        if (surnameField.getText().trim().length() == 0) {
            errorMsg = errorMsg + "Surname is required\n";
        }

        if (isDateValid(dobField.getText().trim()) == false) {
            errorMsg = errorMsg + "Invalid date of birth\n";
        }

        if (errorMsg.length() == 0) {
            return true;
        } else {
            JOptionPane.showMessageDialog(this, errorMsg);
            return false;
        }
    }

    public void actionPerformed(ActionEvent e) {
        if (e.getActionCommand().equals("Cancel")) {
            this.dispose();
        } else if (e.getActionCommand().equals("Edit")) {
            if (doDataValidation() == true) {
                pupilToEdit.setName(nameField.getText().trim());
                pupilToEdit.setSurname(surnameField.getText().trim());
                pupilToEdit.setDob(dobField.getText().trim());
                if (maleRB.isSelected()) {
                    pupilToEdit.setGender("BOY");
                } else {
                    pupilToEdit.setGender("GIRL");
                }
                JOptionPane.showMessageDialog(rootPane, "Pupil Information edited
Successfully");
                this.dispose();
            }
        } else if (e.getActionCommand().equals("Delete")) {
            allPupils.remove(pupilToEdit);
            JOptionPane.showMessageDialog(rootPane, "Pupil Information Removed
Successfully");
            this.dispose();
        } else { // Ok was pressed
            // Check that the data is valid
            if (doDataValidation() == true) {
                Pupil s = new Pupil(getNextStudentId(),
                        nameField.getText().trim(),
                        surnameField.getText().trim(),
                        dobField.getText().trim());
                if (maleRB.isSelected()) {
                    s.setGender("BOY");
                } else {
                    s.setGender("GIRL");
                }
                allPupils.add(s);
                JOptionPane.showMessageDialog(this, "New pupil added successfullly");
                this.dispose();
            }
        }
    }

    // The following getFormLine methods were taken from notes ////////////////////////////////////////////////
    public JPanel getFormLine(JComponent component, String labelText) {
        JPanel p = new JPanel(new FlowLayout(FlowLayout.LEFT));
        JLabel lab = new JLabel(labelText);
        p.add(lab);
        p.add(component);
        return p;
    }

    public JPanel getFormLine(JComponent component1, JComponent component2, String labelText) {
        JPanel p = new JPanel(new FlowLayout(FlowLayout.LEFT));
        JLabel lab = new JLabel(labelText);
        p.add(lab);
        p.add(component1);
        p.add(component2);
        return p;
    }

    public JPanel getFormLine(JButton button1, JButton button2) {
        JPanel p = new JPanel(new FlowLayout(FlowLayout.CENTER));
        p.add(button1);
        p.add(button2);
        return p;
    }
    ////////////////////////////////////////////////////////////////////////////////////////////////
}
```

Pupil

```
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Arrays;
import javax.swing.JOptionPane;


public class Pupil implements Serializable{

    private int studentid;
    private String name;
    private String surname;
    private String dob;
    private String gender;

    private ArrayList<Score> scores = new ArrayList<>();

    public Pupil(int studentid, String name, String surname, String dob) {
        this.studentid = studentid;
        this.name = name;
        this.surname = surname;
        this.dob = dob;
    }

     // Sort Scores
    public ArrayList<Score> sortScores() {
        int n = scores.size();
        Score[] arr = scores.toArray(new Score[n]);
        for (int j = 0; j < arr.length; j++) {
            for (int i = j + 1; i < arr.length; i++) {
                if (arr[i].getScore() > arr[j].getScore()) {
                    Score t = arr[j];
                    arr[j] = arr[i];
                    arr[i] = t;
                }
            }
        }
        scores = new ArrayList<Score>(Arrays.asList(arr));
        return scores;
    }

    public void showScores(){
        // First sort scors
        scores = sortScores();
        String toDisplay = "";
        for(int i = 0;i < scores.size();i++){
            toDisplay = toDisplay + "    " + scores.get(i).toString() + "\n";
        }
        JOptionPane.showMessageDialog(null,name + " " + surname + "\n" +
toDisplay);
    }

    public Score getTopScore(){
        if(scores.size() > 0){
            Score topScore = scores.get(0);
            for(int i = 1;i < scores.size();i++){
                if(scores.get(i).getScore() > topScore.getScore()){
                    topScore = scores.get(i);
                }
            }
            return topScore;
        }else{
            return null;
        }
    }

    public String getNameAndSurname(){
        return name + " " + surname;
    }

    public int getStudentid() {
        return studentid;
    }

    public void setStudentid(int studentid) {
        this.studentid = studentid;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public ArrayList<Score> getScores() {
        return scores;
    }

    public void setScores(ArrayList<Score> scores) {
        this.scores = scores;
    }

    public String getDob() {
        return dob;
    }

    public void setDob(String dob) {
        this.dob = dob;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }
}
```

Question

```
import javax.swing.JDialog;
import javax.swing.JFrame;

public abstract class Question {

    private String questionText;
    private String answerText;
    private int score = -1;

    protected JDialog questionFrame;

    public String getQuestionText() {
        return questionText;
    }

    public void setQuestionText(String questionText) {
        this.questionText = questionText;
    }

    public String getAnswerText() {
        return answerText;
    }

    public void setAnswerText(String answerText) {
        this.answerText = answerText;
    }

    public int getScore() {
        return score;
    }
    public void setScore(int score) {
        this.score = score;
    }

    public abstract boolean playQuestion(JFrame f);

}
```

QuestionAndAnswer

```
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.BoxLayout;
import javax.swing.ImageIcon;
import javax.swing.JButton;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JSeparator;
import javax.swing.JTextField;


public class QuestionAndAnswer extends Question {

    private String picPath;
    private JTextField answerField = new JTextField(10);

    public QuestionAndAnswer(String s){
        String[] x = s.split("#");
        picPath = "Files//Pics//" + x[0];
        setQuestionText(x[1]);
        setAnswerText(x[2]);
    }

    public boolean playQuestion(JFrame f) {
        showQuestion(f);
        return true;
    }

    public void showQuestion(JFrame f) {
        JButton confirmButton = new JButton("Confirm");
        confirmButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                boolean answerCorrect = false;
                if(answerField.getText().trim().equalsIgnoreCase(getAnswerText())){
                    answerCorrect = true;
                }else{
                    answerCorrect = false;
                }

                if (answerCorrect) {
                    setScore(1);
                } else {
                    setScore(0);
                }
                questionFrame.dispose();
            }
        });
        Font myFont = new Font("Arial", Font.BOLD, 14);
        JLabel picLabel = new JLabel(new ImageIcon(picPath));
        JLabel questionLabel = new JLabel(getQuestionText());
        questionLabel.setFont(myFont);

        questionFrame = new JDialog(f, true);
        questionFrame.setLayout(new BoxLayout(questionFrame.getContentPane(),
BoxLayout.Y_AXIS));
        questionFrame.add(picLabel);
        questionFrame.add(questionLabel);
        questionFrame.add(answerField);
        questionFrame.add(new JSeparator());

        JPanel p = new JPanel(new FlowLayout());
        p.add(confirmButton);
        questionFrame.add(p);

        questionFrame.pack();
        questionFrame.setVisible(true);
    }
}
```

QuizMenu

```
import java.awt.FlowLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.Random;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.UIManager;

public class QuizMenu extends JFrame implements ActionListener {

    private JButton addStudent = new JButton("Add Pupil");
    private JButton searchPupil = new JButton("Search Pupil");
    private JButton showScores = new JButton("Show Scores");
    private JButton questionsQuiz = new JButton("Start Questions Quiz");
    private JButton pictureQuiz = new JButton("Start Picture Quiz");
    private JButton mixedQuiz = new JButton("Start Mixed Questions Quiz");

    private ArrayList<Pupil> pupils = new ArrayList<>();
    private ArrayList<Question> multipleChoiceQuestions = new ArrayList<>();
    private ArrayList<Question> qAndAQuestions = new ArrayList<>();

    public QuizMenu() {
        // As soon as program starts, load all data from files
        loadQuestionAndAnswerQuestions();
        loadMultipleChoiceQuestions();
        loadPupilsInfoFromFile();
        setTitle("Educational Quiz");
        setLayout(new FlowLayout());
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        add(addStudent);
        add(searchPupil);
        add(showScores);
        add(questionsQuiz);
        add(pictureQuiz);
        add(mixedQuiz);

        // To continue from here
        addStudent.addActionListener(this);
        showScores.addActionListener(this);
        questionsQuiz.addActionListener(this);
        pictureQuiz.addActionListener(this);
        searchPupil.addActionListener(this);
        mixedQuiz.addActionListener(this);

        pack();
        setVisible(true);

        // When progam ends, save back all pupils data to file
        addWindowListener(new WindowAdapter() {
            public void windowClosing(WindowEvent e) {
                savePupilsInfoToFile();
            }
        });
    }

    // Linear Search for Pupil by ID
    private Pupil getPupilWithId(int idPupil) {
        for (int i = 0; i < pupils.size(); i++) {
```

```
        if (pupils.get(i).getStudentid() == idPupil) {
            return pupils.get(i);
        }
    }
    return null;
}

private void loadMultipleChoiceQuestions() {
    try {
        BufferedReader          in          =          new          BufferedReader(new
FileReader("Files//MultipleChoiceQuestions.txt"));
        String str;
        while ((str = in.readLine()) != null) {
            multipleChoiceQuestions.add(new MultipleChoiceQuestion(str));
        }
        in.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

private void loadQuestionAndAnswerQuestions() {
    try {
        BufferedReader          in          =          new          BufferedReader(new
FileReader("Files//QuestionAndAnswer.txt"));
        String str;
        while ((str = in.readLine()) != null) {
            qAndAQuestions.add(new QuestionAndAnswer(str));
        }
        in.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

// Taken from notes
private void savePupilsInfoToFile() {
    try {
        FileOutputStream fileOut = new FileOutputStream("pupils.dat");
        ObjectOutputStream out = new ObjectOutputStream(fileOut);
        out.writeObject(pupils);
        out.close();
        fileOut.close();
    } catch (Exception e) {

    }
}

// Taken from notes
private void loadPupilsInfoFromFile() {
    try {
        FileInputStream fileIn = new FileInputStream("pupils.dat");
        ObjectInputStream in = new ObjectInputStream(fileIn);
        pupils = (ArrayList<Pupil>) in.readObject();
        in.close();
        fileIn.close();
    } catch (Exception e) {
        System.out.println("Files not loaded. Problem with file.");
    }
}

@Override
public void actionPerformed(ActionEvent e) {
    switch (e.getActionCommand()) {
        case "Add Pupil":
            addPupil();
            break;
        case "Show Scores":
            DisplayPupilsTable x = new DisplayPupilsTable(pupils);
            x.setVisible(true);
            break;
        case "Start Questions Quiz":
            playQuiz(get5RandomQuestions(qAndAQuestions), "Question And Answer
Quiz");
            break;
        case "Start Picture Quiz":
            playQuiz(get5RandomQuestions(multipleChoiceQuestions),           "Picture
Quiz");
            break;
        case "Start Mixed Questions Quiz":
            ArrayList<Question> all = multipleChoiceQuestions;
            all.addAll(qAndAQuestions);
            playQuiz(get5RandomQuestions(all), "Mixed Questions Quiz");
            break;
        case "Search Pupil":
            searchPupil();
            break;
    }
}

// Taken from notes
public String getTodaysDate() {  // dd-MM-yyyy
    try {
        SimpleDateFormat sdf = new SimpleDateFormat("dd-MM-yyyy");
        return sdf.format(new Date());
    } catch (Exception e) {
        return "";
    }
}

private void playQuiz(ArrayList<Question> questionsToAsk, String quizTitle) {
    int score = 0;
    for (int i = 0; i < 5; i++) {
        questionsToAsk.get(i).playQuestion(this);
        score = score + questionsToAsk.get(i).getScore();
    }
    int n = JOptionPane.showConfirmDialog(this,
            "You've got a score of " + score + ". Do you want to save your score?",
"Total Score", JOptionPane.YES_NO_OPTION);
    if (n == 0) {
        String pupleId = JOptionPane.showInputDialog("Enter your pupil ID:");
        try {
            Pupil puple = getPupilWithId(Integer.parseInt(pupleId));
            if (puple == null) {
                JOptionPane.showMessageDialog(null, "Pupil not registered", "Error",
JOptionPane.ERROR_MESSAGE);
            } else {
                Score scorePoints = new Score();
                scorePoints.setScore(score);
                scorePoints.setTotalPossableScore(questionsToAsk.size());
                scorePoints.setQuizType(quizTitle);
                scorePoints.setDate(getTodaysDate());
                puple.getScores().add(scorePoints);
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "Invalid Pupil ID", "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
}

private ArrayList<Question> get5RandomQuestions(ArrayList<Question>
allQuestions) {
    // To get 5 different random questions
    ArrayList<Integer> allQuestionNumbers = new ArrayList<>();
    for (int i = 0; i < allQuestions.size(); i++) {
        allQuestionNumbers.add(i);
    }

    Random randomGenerator = new Random();

    ArrayList<Question> ans = new ArrayList<>();
    for (int i = 0; i < allQuestions.size(); i++) {
        int r = randomGenerator.nextInt(allQuestionNumbers.size());
        int pos = allQuestionNumbers.get(r).intValue();
        ans.add(allQuestions.get(pos));
        allQuestionNumbers.remove(r);
    }
    return ans;
}

private void addPupil() {
    PupilForm t = new PupilForm(pupils);
    t.setVisible(true);
}

private void searchPupil() {
    String searchKey = JOptionPane.showInputDialog("Enter pupil's ID or NAME
and SURNAME");
    // First try to search pupil by ID
    Pupil p = searchPupilById(searchKey);
    if (p != null) {
        PupilForm form = new PupilForm(pupils);
        form.usedInEditMode(p);
        form.setVisible(true);
    } else {
        // If pupil not found, then try to search pupil by NAME and SURNAME
        ArrayList<Pupil> results = searchPupilByNameAndSurname(searchKey);
        new DisplayPupilsTable(results);
    }
}

private Pupil searchPupilById(String id) {
    // Linear Search Implementation
    for (int i = 0; i < pupils.size(); i++) {
        if ((pupils.get(i).getStudentid() + "").equals(id)) {
            return pupils.get(i);
        }
    }
    return null;
}

private ArrayList<Pupil> searchPupilByNameAndSurname(String ns) {
    // Linear Search Implementation
    ArrayList<Pupil> searchResults = new ArrayList<>();
    for (int i = 0; i < pupils.size(); i++) {
        if (pupils.get(i).getNameAndSurname().equalsIgnoreCase(ns)) {
            searchResults.add(pupils.get(i));
        }
    }
    return searchResults;
}
public static void main(String[] args) {
    try {

UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());
    } catch (Exception e) {
        // Do nothing
    }

    new QuizMenu();
}

}
```

Score

```
import java.io.Serializable;



public class Score implements Serializable{
    private String date;
    private String quizType;
    private int score;
    private int totalPossableScore;

    public String getDate() {
        return date;
    }

    public void setDate(String date) {
        this.date = date;
    }
    public String getQuizType() {
        return quizType;
    }

    public void setQuizType(String quizType) {
        this.quizType = quizType;
    }

    public int getScore() {
        return score;
    }

    public void setScore(int score) {
        this.score = score;
    }

    public int getTotalPossableScore() {
        return totalPossableScore;
    }

    public void setTotalPossableScore(int totalPossableScore) {
        this.totalPossableScore = totalPossableScore;
    }

    public String toString(){
        return date + "   " + quizType + "   " + score;
    }
}
```