```
// Authour: Samuele Joshi, ID: 180318461, Last updated: 08/12/2019 |

/* Description: Quiz game which allows the player to enter an answer they believe is correct based on the question.

  If correct, wrong or impossible, a score is assigned and stored in an array. At the end, they can view their highest mark for each question.*/


// Packages which my java source uses in order to refer from one class to another directly by its name.

import java.io.File;  // Imports the File for my program to use.

import java.io.FileWriter;   // Imports the FileWriter for my program to enter data on the file.

import java.io.IOException;  // Imports the IOException to handle errors.

import java.io.FileNotFoundException; // Imports the FileNotFoundException. Used if the program cannot find the file (shows error).

import java.util.ArrayList; // Imports Arraylist to use arraylists in the program.

import java.util.Scanner; // Imports Scanner for users to make an input in the program.


public class Main {

  /* Main method - User is presented with a welcome screen of the game and the main menu.

  *  Here the user can select the three options presented by the main menu.

  *  Depending on what they select, they will execute that piece of code.

  * */

  public static void main(String [] args){

    Scanner userInput = new Scanner(System.in);

    welcomeScreen(); // Prints on the console the title of the game.

    try {

      // Specifies the file which we want to add data in.

      FileWriter playerInfo = new FileWriter("C:\\Users\\Samuele Joshi\\Documents\\Computer Science - Year 1\\File test\\filename.txt");

      // Array of objects - Stores the Quiz question, correct answer, wrong answer & impossible answer.

      final int maxQ = 5; // Max amount of questions.

      Quiz[] qc = new Quiz[maxQ];
```

```
        qc[0] = quizConstructor("Q1. Who made the song 'Straight Outta Compton'?", "NWA", "G-Unit", "Mobb Deep");

        qc[1] = quizConstructor("Q2. What year did the Vietnam war start?", "1955", "1964", "1976");

        qc[2] = quizConstructor("Q3. Who painted the Mona Lisa?", "Leonardo da Vinci", "Michelangelo", "Shostakovich");

        qc[3] = quizConstructor("Q4. Which ice giant planet is part of the Jovian planets?", "Neptune", "Saturn", "Venus");

        qc[4] = quizConstructor("Q5. According to the Babylonians, how many days are in a year?", "360", "365", "270");


        /* While loop controls the main menu. If the user does not select a valid option,

          the menu is presented again unless they enter a valid option or they exit the program. */

        int option = 0; // Option is set to 0 for it to be neutral. If 3 is entered, the while loop ends and the program ends.

        while (option != 3) {

            System.out.print("Select an option:\n1. Start Quiz\n2. View Report\n3. Quit\n----------\nOption entered: ");

            option = userInput.nextInt();

            // If 1 is entered, the quiz starts.

            if (option == 1) {

                // Passes the Quiz array of objects in the method for them to be used.

                //

                Integer[] storeArray = askQuestions(qc, playerInfo);

                sortScore(storeArray);

                printScore(storeArray);

            }

            // If 2 is entered, the results of the previous game is displayed.

            else if (option == 2) {

                viewReport();

            }

            // If 3 is entered, the code returns a 3 to the loop condition which the while loop exits the loop causing the program to terminate.

            else if (option == 3) {
```

```
            System.out.println("Thank you for playing.");
            option = 3;
          }
          // If anything which is invalid is entered, the while loop will loop the main menu again.
          else {
            System.out.println(option + " is an invalid input. Try again.");
          }
        }
        System.exit(0); // Terminates the program.
      } // If an error occurs, the user is informed in the console.
      catch (IOException e) {
        System.out.println("Error");
        e.printStackTrace();
      }
    }
    // welcomeScreen method - Prints out the game title.
    private static void welcomeScreen(){
      System.out.println("-----------------------------------------------------");
      System.out.println("|              Impossible Quiz Game              |");
      System.out.println("-----------------------------------------------------");
    }
    /* askQuestions method - User is asked a set of questions controlled by a for loop in which the user has to
      enter in an answer that they believe is the correct answer. Whatever the user enters in, a record is made of it
      for each question asked in a text file. If statements decide whether the answer is correct, wrong or impossible.
      Within the specific if statement block, a score is assigned and then once all questions have been answered, these scores are kept in the array.
     */
    private static Integer[] askQuestions(Quiz [] qc, FileWriter playerInfo){
      Scanner userInput = new Scanner(System.in); // Allows user to make an input.
```

```
ArrayList<Integer> scoreKeeper = new ArrayList<Integer>(); // Stores score in this array.

try{ // For loop controls the question. Each loop completed, a new question is printed to the user. Which the user then enters their answer.

  for(int q = 0; q < qc.length; q++) {

    // I use the get (read only) to print out the quiz question, correct, wrong and impossible.

    System.out.println(getQuestion(qc[q]) + " " + getCorrect(qc[q]) + " " + getWrong(qc[q]) + " " + getImpossible(qc[q]));

    System.out.print("Enter Answer: ");

    String answerEntered = userInput.nextLine();

    System.out.println(" ");

    // Enters the following data in the text file as a report for user. The question, answer entered and the correct answer.

    playerInfo.write(getQuestion(qc[q]) + "\nAnswer entered: " + answerEntered + "\nCorrect answer: " + getCorrect(qc[q]) + "\r\n");


    // If the answer entered is correct, a random score between 1 to 6 is made. Then the score is entered in the array.

    if (answerEntered.equalsIgnoreCase(getCorrect(qc[q]))) {

      int score = (int) (Math.random() * 6 + 1); // Random number generated here.

      if (score == 1) {

        System.out.println("You got " + score + " mark.");

      } else {

        System.out.println("You got " + score + " marks.");

      }

      scoreKeeper.add(score); // Random number gets added to the array here.

    }

    // If the answer entered is wrong, 2 marks are deducted. Then the score is entered in the array.

    else if (answerEntered.equalsIgnoreCase(getWrong(qc[q]))) {

      scoreKeeper.add(-2);

      System.out.println("2 marks deducted.");

    }

    // If the answer entered is impossible, all marks are removed from the array.
```

```
            else if (answerEntered.equalsIgnoreCase(getImpossible(qc[q]))) {
                scoreKeeper.clear();
                System.out.println("All marks lost.");
            }
            // If the answer entered is not valid, this message is printed on the console.
            else {
                System.out.println("Input invalid.");
            }
        }
        playerInfo.close(); // The data entered ends here.
    } // If an error is presented, a message is printed informing that there is an error.
    catch (IOException e) {
        System.out.println("Error");
        e.printStackTrace();
    }
    // Turns an arraylist into an array.
    Integer[] myArray = new Integer[scoreKeeper.size()];
    for (int i = 0; i < scoreKeeper.size(); i++) {
        myArray[i] = scoreKeeper.get(i);
    }
    return myArray; /* Returns all score elements to another array called "store Array"
    which that data is then used for later purposes. */
}
// viewResults method - User is able to see where they went wrong as the program stores all
records of what the user has entered in.
public static void viewReport(){
    try { // The file which the program needs to read.
        File readResults = new File("C:\\Users\\Samuele Joshi\\Documents\\Computer Science - Year
1\\File test\\filename.txt");
        Scanner resultReader = new Scanner(readResults); // Reads the text.
        while (resultReader.hasNextLine()) {
            String in_text = resultReader.nextLine(); // Stores text in a string variable.
```

```
            System.out.println(in_text); // Prints out the text in the file.
        }
        resultReader.close(); // Program stops showing anything else in the file.
    } // If the file is missing, the program will report this error.
    catch (FileNotFoundException e) {
        System.out.println("Error - File Missing");
        e.printStackTrace();
    }
}
// sortScore - Using bubblesort, all score elements in storeArray are arranged from the highest to lowest score.
public static void sortScore(Integer [] storeArray) {
    int p = storeArray.length; // Determines how many elements are in the array. This is done to stop the loop at the specific amount.
    for (int i = 0; i < p - 1; i++) { // Moves the elements around.
        for (int j = 0; j < p - i - 1; j++) {
            if (storeArray[j] < storeArray[j + 1]) {
                int temp = storeArray[j];
                storeArray[j] = storeArray[j + 1];
                storeArray[j + 1] = temp;
            }
        }
    }
}
// printScore - Once "storeArray" is sorted from high to low, it is then printed for the user to see what marks they got.
public static void printScore(Integer [] storeArray) {
    System.out.println("Highest Score to Lowest Score:");
    for (int i = 0; i < storeArray.length; i++) {
        System.out.println(storeArray[i] + " ");
    }
}
```

```
   // quizConstructor method - Allows me to store all important data of a quiz in one record. This is used at the start for the array of objects.
   public static Quiz quizConstructor (String quizQ, String quizC, String quizW, String quizImp){
      Quiz qsUniversal = new Quiz();
      qsUniversal = setQuestion(qsUniversal, quizQ);
      qsUniversal = setCorrect(qsUniversal, quizC);
      qsUniversal = setWrong(qsUniversal, quizW);
      qsUniversal = setImpossible(qsUniversal, quizImp);
      return qsUniversal;
   }

   //Set (Write) allows the user to enter in data. Get (Read) allows the user to make the program read the data entered.
   public static String getQuestion (Quiz qc){
      return qc.question;
   }
   public static Quiz setQuestion (Quiz qc, String new_question){
      qc.question = new_question;
      return qc;
   }

   public static String getCorrect (Quiz qc){
      return qc.correct;
   }
   public static Quiz setCorrect (Quiz qc, String new_correct){
      qc.correct = new_correct;
      return qc;
   }

   public static String getWrong (Quiz qc){
      return qc.wrong;
   }
```

```
    public static Quiz setWrong (Quiz qc, String new_wrong){
        qc.wrong = new_wrong;
        return qc;
    }

    public static String getImpossible (Quiz qc){
        return qc.impossible;
    }
    public static Quiz setImpossible (Quiz qc, String new_impossible){
        qc.impossible = new_impossible;
        return qc;
    }
}
// Variables used for the records I use. Class is called Quiz as it is the body of the game.
// Rather than having multiple variable to store multiple data, I can refer it and data is stored.
class Quiz{
    String question;
    String correct;
    String wrong;
    String impossible;
}
```