# OOP Revision: QUIZ

Schools in Pretoria are taking part in a General knowledge quiz and each school needs to send two participants to the competition.

The file called 'Participants.txt' contains the data of the participants:

```
Pecanwood#Harties#9809265026084#125
Woodhill#Pretoria#9910138473021#131
Pecanwood#Harties#9804038674737#141
Woodhill#Pretoria#9906847389610#137
```


Each line in the text file contains the following data:
***School* # *Area the school is in* # *RSA ID Number (of the learner representing that school)* # *IQ***

IQ (intelligence quotient) is a number meant to measure people cognitive abilities (intelligence) in relation to their age group.

The following table gives information about the average IQ per age group:

| **Age Group** | **Average IQ** |
|---|---|
| 18-19 | 134 |
| 17 | 136 |
| 16 | 133 |
| 15 | 123 |
| 14 | 125 |

## QUESTION 1

**1.1** Create a new class called **Student** that will contain two fields - the ID number of a student and his/her IQ.
*(Here a student's South African ID number and IQ will be stored)*

**1.2** Create a constructor method that will assign values to the fields.
*(This will be used to create a new object for a student and place values (his/her ID & IQ) in the fields)*

**1.3** Create a getter method that will return the IQ of student.
*(This will be used to get the IQ of a student)*

**1.4** Create a method called **getAverageIQ** that will look at the first 6 digits of the ID number and return the average IQ (use the above table). So you are returning the average IQ for the age group that the students falls under, e.g. Peter Parker has a IQ of 125 and his age group's average IQ is 134, so the method is returning 134.

**1.5** Create a toString method that will return the ID number and IQ of a student in the following format:

ID: *<id number>* (IQ) ,e.g. ID: 9809265026084 (125)

*(This will be used to display the data of one student)*

**1.6** Create a method called **isOnPar** that will return a Boolean true or false indicating whether the student's IQ is above (return true) or below (return false) the average IQ per age group.

**QUESTION 2**

**2.1** Use the following class diagram to create a new class called **School**:

| **School** |
|---|
| **Properties/Fields:**<br>- String school<br>- String area<br>- student : Array of 2 Student objects |
| **Methods:**<br>+ Constructor (String s, String a)<br>+ String getSchool()<br>+ String getArea()<br>+ String toString()<br>+ insertStudent_ID_IQ (Student learner)<br>+ int maxIQ()<br>+ int countAboveAvg() |

**2.1.1** Declare the fields of the class as indicated in the class diagram.

**2.1.2** The **constructor method** must receive the name and school from the text file and put null values in the array of objects.
*(Here we are not yet going to store the students' data in the array. It only happens in question 2.1.5. We are only going to store the school's data in the fields and make the object null (meaning it is empty). Always look at the class diagram and see what parameters it should have. A school will only be saved once with its students saved in the array.)*

**2.1.3** The **getSchool() and getArea() methods** must return the school's name and the area it is in respectively *(in two different methods)*.

**2.1.4** The **toString method** must return the name and area of the school and its participants in the following format:

*Pecanwood (Harties)*
*9809265026084 (125)*
*9604038674737 (141)*

**2.1.5** The **insertStudent_ID_IQ method** must receive a Student object as a parameter and store it in the array of objects.

**2.1.6** The **maxIQ() method** that must return the highest IQ of a student in a specific school. *(So here we will return the highest IQ between the two students in a school)*

**2.1.7** The **countAboveAvg() method** that must count and return the number of students in a school whose IQ is more than the average IQ of their age group.

**QUESTION 3**

**3.1** Create a new class called **SchoolManager.**

**3.2** Create an array called **schoolArr** that will store 30 **School** objects and a size variable to store the number of schools stored in the array.

**3.3** Create a method called **fetchArea** that will receive the name of a school and then return the area of the school.

**3.4** Create a method called **fetchSchool** that will receive the name of a school and then return all the information of the school (so return the school's object that is stored in the *schoolArr* array).
*(So here we have to search for the school in the array and return the object)*

**3.5** Create a constructor method that will read the data from the text file and store it in the *school* array. Every school must have just have one object (in the array). Display an error message when the file could not be found.

**3.6** Create a method called **displaySchools** that will display all the information of the schools as follow:

Pecanwood (Harties)
9809265026084 (125)
9604038674737 (141)

Woodhill(Pretoria)
9710138473021(131)
9806847389610(137)

**3.7** Create a method called **getMaxIQ** that will receive the name of a school and then returns it highest IQ from that school.

**3.8** Create a method called **schoolsAboveAverage** that will return all the school's names followed by how many students in that school have an IQ that is above the average IQ for their age group.

**QUESTION 4**

**4.1** Create a user interface class called **UIQuiz**.

**4.2** Declare and instantiate a SchoolManager objects.

**4.3** Display all the school's information.

**4.4** Display the area of Woodhill.

**4.5** Display the highest IQ of Pecanwood.

**4.7** Display all the schools' names followed by how many of the students of that school have an IQ that is above the average IQ for their age group.

**SAMPLE OUTPUT:**

*Pecanwood (Harties)*
*ID:9809265026084(125)*
*ID:9804038674737(141)*
*Woodhill (Pretoria)*
*ID:9910138473021(131)*
*ID:9906847389610(137)*

*The area of woodhill: Pretoria*
*The highest IQ at Pecanwood: 141*

*The number of students per school whose IQ is higher than their group average's IQ:*
*Pecanwood: 1*
*Woodhill: 1*

## QUESTION 1: Student class

```
package uiquiz;
import javax.swing.JOptionPane;
```

**public class Student** {

***//1.1***

```
private String idNumber;
 private int studentIQ;
```

***//1.2***

```
public Student(String id, int iq) {
   idNumber = id;
   studentIQ = iq;
}
```

***//1.3***

```
public int getStudentIQ() {
   return studentIQ;
}
```

***//1.4***

```
public int getAverageIQ() {
   int avgIQ = 0;

   String firstTwo = idNumber.substring(0, 2);                    // using the substring method
   firstTwo = "" + idNumber.charAt(0) + idNumber.charAt(1);  //or like this using charAt()

   int year = 1900 + Integer.parseInt(firstTwo);

   int age = 2016 - year;

   if (age == 18 || age == 19) {
      avgIQ = 134;
   } else if (age == 17) {
      avgIQ = 136;
   } else if (age == 16) {
      avgIQ = 133;
   } else if (age == 15) {
      avgIQ = 123;
```

```
        } else if (age == 14) {
            avgIQ = 125;
        }

        return avgIQ;
    }
```

***//1.5***

```
    public String toString() {
        return "ID:" + idNumber + "(" + studentIQ + ")";
    }
```

***//1.6***

```
    public boolean isOnPar() {
        boolean par = false;
        if (studentIQ > getAverageIQ()) {
            par = true;
        }
        return par;
    }

}
```

## QUESTION 2: School class

```
package uiquiz;

import javax.swing.JOptionPane;
```

## *public class School* {

***//2.1.1***

```
    private String school;
    private String area;
    private Student[] student = new Student[2];
```

***//2.1.2***

```
public School(String s, String a) {
   school = s;
   area = a;                                        //OR done in a bad way:
   for (int i = 0; i < 2; i++) {                    // student[0] = null;
         student[i] = null;                         // student[1] = null;
   }
}
```

***//2.1.3***

```
public String getSchool() {
   return school;
}

public String getArea() {
   return area;
}
```

***//2.1.4***

```
public String toString() {
   String output = "";                          // OR done in a bad way:

   output = school + " (" + area + ")\n";       //   return school + " (" + area + ")\n"
                                                //         + student[0].toString + "\n"
   for (int i = 0; i < 2; i++) {                //         + student[1].toString + "\n";

      if (student[i] != null) {
         output = output + student[i].toString() + "\n";    // inserting the if statement here
      }                                                     // we will only add a student's data to
                                                            // output when it is not empty
   }
   return output;
}
```

***//2.1.5***

```
public void insertStudent_ID_IQ(Student learner) {

   for (int i = 0; i < 2; i++) {

      if (student[i] == null) {
         student[i] = learner;
         break;
      }
```

```
    }
  }
```

***//2.1.6***

```
  public int maxIQ() {
    int highestIQ = 0;
    for (int i = 0; i < 2; i++) {
      if (student[i].getStudentIQ() > highestIQ) {
        highestIQ = student[i].getStudentIQ();
      }
    }
    return highestIQ;
  }
```

***//2.1.7***

```
  public int countAboveAvg() {
    int count = 0;
    for (int i = 0; i < 2; i++) {
      if (student[i].isOnPar() == true) {          //OR better: if ( student[i].isOnPar() )
        count++;                                   // Whenever we want to say "== true" we can just
      }                                            // leave it out.
    }
    return count;
  }

}
```

## QUESTION 3: SchoolManager class

```
package uiquiz;

import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;
```

```
//3.1
```

```
public class SchoolManager {
```

```
    //3.2
    private School[] schoolArr = new School[30];
    private int size = 0;
```

```
    //3.5
    public SchoolManager() {

        try {
            Scanner scFile = new Scanner(new File("Participants.txt"));

            while (scFile.hasNext()) {
                String line = scFile.nextLine();

                Scanner scLine = new Scanner(line).useDelimiter("#");

                String schoolName = scLine.next();
                String schoolArea = scLine.next();
                String studID = scLine.next();
                int studIQ = scLine.nextInt();

                Student pupil = new Student(studID, studIQ);

                                        //we create a student object and store the student's data
                                    (id number & iq) in there (in its fields).

                if (fetchSchool(schoolName) == null) {

                                        //in other words: if the school is not yet stored in the array.

                    schoolArr[size] = new School(schoolName, schoolArea);

                                        //we then create a new object for the school and store the
                                    school name and area in its fields.

                    schoolArr[size].insertStudent_ID_IQ(pupil);          // we place the student's data in
                                                                        the student's array (for that school).
                    size++;
                } else {
                    fetchSchool(schoolName).insertStudent_ID_IQ(pupil);

                }
            }
```

```
    } catch (FileNotFoundException ex) {
        JOptionPane.showMessageDialog(null, "Error. The text file could not be found.");
    }

}
```

***//3.3***

```
public String fetchArea(String schoolName) {
    String areaOfSchool = "";

    boolean found = false;

    for (int i = 0; i < size; i++) {
        String institute = schoolArr[i].getSchool();
        if (institute.equals(schoolName)) {          //OR: if (schoolArr[i].getSchool().equals(schoolName)
            areaOfSchool = schoolArr[i].getArea();
        }
    }
    return areaOfSchool;

}
```

***//3.4***

```
public School fetchSchool(String schoolName) {
    School college = null;

    boolean found = false;

    for (int i = 0; i < size; i++) {
        String institute = schoolArr[i].getSchool();
        if (institute.equals(schoolName)) {          //OR: if (schoolArr[i].getSchool().equals(schoolName)

            college = schoolArr[i];
        }
    }
    return college;

}
```

```
//3.6
```

```
public String displaySchool() {
    String output = "";

    for (int i = 0; i < size; i++) {
        output = output + schoolArr[i].toString();
    }

    return output;
}
```

```
//3.7
```

```
public int getMaxIQ(String schoolName) {

    int highestIQ = 0;

    for (int i = 0; i < size; i++) {
        String institute = schoolArr[i].getSchool();
        if (institute.equals(schoolName)) {          //OR: if (schoolArr[i].getSchool().equals(schoolName)
            highestIQ = schoolArr[i].maxIQ();
        }
    }
    return highestIQ;
}
```

```
//3.8
```

```
public String schoolsAboveAverage() {

        //Like the displaySchool method but using other methods instead of toString.

    String output = "";

    for (int i = 0; i < size; i++) {
        output = output + schoolArr[i].getSchool() + " : " + schoolArr[i].countAboveAvg() + "\n";
    }

    return output;
}

}
```

## QUESTION 4: UIQuiz main class

```
package uiquiz;

public class UIQuiz {

  public static void main(String[] args) {

      SchoolManager sm = new SchoolManager();

      System.out.println(sm.displaySchool());

      System.out.println("The area of woodhill: " + sm.fetchArea("Woodhill"));

      System.out.println("The highest IQ at Pecanwood: " + sm.getMaxIQ("Pecanwood"));

      System.out.println("\nThe number of students per school whose IQ is higher than their group
                                                  average's IQ:");

      System.out.println(sm.schoolsAboveAverage());

   }

}
```