**COMBINED PRACTICAL EXAM**

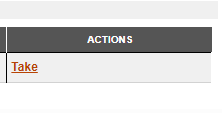
|  |  |  |  |
| --- | --- | --- | --- |
| **SUBJECT** | **Information Technology**  **Practical Exam** | **DATE** | **Nov 2017** |
| **GRADE** | **11** | **MARKS** | **120** |
| **EXAMINER** | **Mrs Kench** | **MODERATOR** | **Mr Blieden and Mrs Nocton-Smith** |
| **NAME** | **Dylan Beaumont** | **DURATION** | **3 hours** |
| **USER NAME** | **hsexam24** | **WEB NAME** | **IEB025** |
| **PASSWORD** | **Tv788106** | **WEB PASSWORD** | **Beaumont.24** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **COGNITIVE LEVELS** | | | | | |
| **LOW ORDER** | 30 % | **MIDDLE ORDER** | 40 % | **HIGH ORDER** | 29 % |

**INSTRUCTIONS**

**BEFORE THE EXAM:**

* Log onto the computer using your Hsexam account and password.
* Open Google Chrome internet browser.
* Type in the following web address **http://itemsonweb.ed-software.com.ar**
* You will then be prompted with a Login Screen
* Log onto the web site using the web name that starts with IEB and the web password
* Click Login
* Click on the link “Take” under Actions on the right hand side.



* Click Continue on the bottom right hand side.
* Download the files in both section A and section B. Copy these files into the folder that is assigned to you for this exam.
* Open your programming (e.g. Netbeans) and database (e.g. MS Acess or My SQL) applications.
* Open the files that you downloaded in your applications.

**DURING THE EXAM**

* SQL (Section A):
* Type in your answer for SQL into your DATABASE application as you normally do.
* When your answer is correct, you need to do TWO things: copy the SQL statement into
  + the SQL answer sheet that you downloaded in section A folder. Click the Next button to go to the next question. You can return to a question by clicking Previous.
  + the place on the web site for the SQL question. Click the Next button to go to the next question. You can return to a question by clicking Previous.
* Programming (Section B):
* Code your solution in your programming application such as Netbeans as you would normally do. Make sure you have saved your code in the folder space provided by the school.
* Once you have completed Question 2, 3 and 4, copy each class into the space provided on the web site. You will need to copy the **Leader** class into the space for Question 2, the **LeaderArray** class into the space for Question 3 and the **LeaderUI** in the space for Question 4.

**AFTER THE EXAM**

Print the following files:

* SQL Answer file
* Learner Class
* LearnerArray Class
* LearnerUI Class

Using the web site:

* Upload the your SQL Answer File to Section A and click Next
* Upload your entire programming project to Section B and click Next.

**SECTION A: SQL**

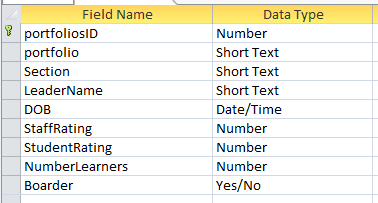
*The files required for this question can be found in the folder named* ***Section A.*** *Please complete your work in this folder****.***

A school has chosen its leaders and has assigned leaders to a particular portfolio within the school. Not all leaders have been assigned a portfolio. At the end of the first term, the leader’s ability to fulfil the requirements of their portfolio have been rated by staff and learners. In each portfolio, the number of learners who are involved in a portfolio are listed. Most portfolios fall under a larger section such as sport or culture. Some of the leaders are boarders.

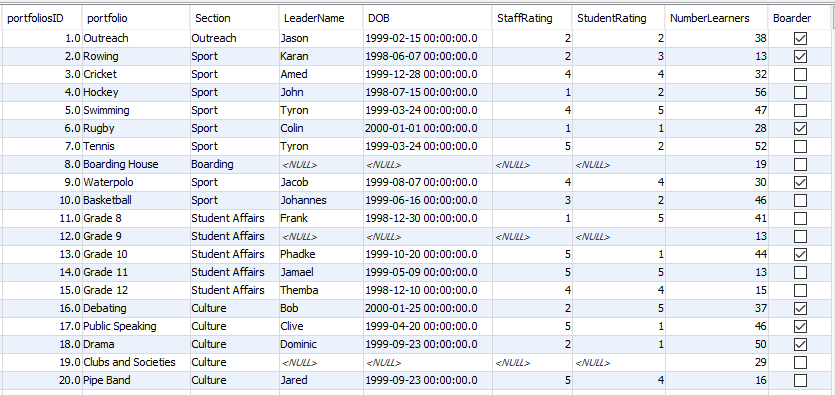
The data is stored in a table called **tblPortfolio** in a database called **LeaderDB**.

**Table design:**

**tblPortfolio**



The data is the table is show below:



**QUESTION 1 40 MARKS**

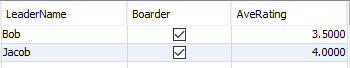
1.1 Display the portfolios and sections fields sorted alphabetically by section and then by portfolio. (3)

1.2 List the leaders whose staff rating or student rating is above 3 and who have more than 20 learners in their portfolio. (5)

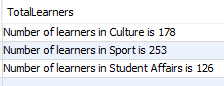
1.3 Determine the age of each leader in years. Ensure that the age is accurate, if a person has not had a birthday their age should be one less. For example. Amed has not has his birthday and should be 17 even though he is born in 1999. Name this new field **Age**. Display only leader’s name, date of birth and **Age** (in years). Do not display an age if there is no leader for a portfolio. (7)

1.4 Count the number of portfolios in each section. (4)

1.5 A new leader for boarding needs to be considered. This leader will be chosen from the existing leaders. The criteria for the leader of boarding is to have a rating average of 3 or more and to be a boarder themselves. The average rating is calculated using the staff and student rating fields. Code a SQL statement to list all the possible candidates using the criteria. Sort the result according the average in descending order. Your SQL statement should produce the following result:

 (8)

1.6 List the sections in the school that have more than 50 learners in total. Display your answer as **Number of learners in *Section* is *xxx***. Your output should appear as follows with the new field called **TotalLearners**:

 (8)

1.7 The sport portfolios are becoming more popular and are going to increase their number of learners by 20%. Code a query to increase the Number of learners in the sport portfolios by 20% rounded to 0 decimal places. (5)

**SECTION B OOP 80 MARKS**

After the potential leaders attended their leadership camp, the ratings from the staff, Grade 10s, Grade 11s and the camp organisers were recorded. Each potential leader has a single integer value for staff, Grade 10, Grade 11 and the camp staff. This data has been recorded in a text file called **Leaders.txt**. A sample of the text file is shown below.

You have been provided with the file **Leaders.txt** in the **Section B** folder. The file contains the leader’s name, staff rating, Grade 10 rating, Grade 11 rating and camp staff rating separated by a comma.

**Leaders.txt**

|  |
| --- |
| Jason,2,2,2,5  Karan,2,3,5,5  Amed,4,4,1,5  John,1,2,4,3  Tyron,4,5,2,1  Colin,1,1,1,3  Tyron,5,2,2,3  Jacob,4,4,1,2  Johan,3,2,2,3  Frank,1,5,5,2  Phadke,5,1,1,2  Jamael,5,5,5,3  Themba,4,4,2,1  Bob,2,5,3,2  Clive,5,1,3,3  Dominic,2,1,5,4  Jared,5,4,5,5 |

**QUESTION 2 34 MARKS**

Consider the following UML diagram for the Leader class.

|  |
| --- |
| **Leader** |
| * Sting name * int staffRating * int gr11rating * int gr10rating * int campRating |
| + Leader(String n, int sr, int gr11, int gr10, int cr)  + getName():String  + getStaffRating():int  + getGr11rating():int  + getGr10rating():int  + getCampRating():int  + setName(String n)  + setStaffRating(int sr)  + setgr11rating(int gr11)  + setgr10rating(int gr10)  + setCampRating(int cr)  + getPoints() : double  + removeVowels()  + lessThan3() : boolean  + toString() : String |

2.1 Create a class called **Leader** with fields to store the name, staff rating, grade 11 rating, grade 10 rating and the camp staff rating. Use appropriate types for your fields. (4)

2.2 Code a parameterised constructor that will assign values to each of the fields. (4)

2.3 Code accessor and mutator methods for all the fields. (4)

2.4 Code a method called **getPoints** to calculate and return the points for each leader stored as a double. The points for each leader are determined by multiplying the staff rating by 3, the Grade 11 rating by 2 and the Camp staff rating by 4. These are added to the Grade 10 rating.

Points = 3xstaffRate + 2xg11Rate + g10Rate + 4xcampRate

For example, Jason is awarded 32 points. (3x2 + 2x2 + 2 + 4x5) (5)

2.5 Code a **toString** method that will return the name, ratings and points combined as a string separated by tabs in the following format:

Name<tab>staffrating<tab>g11rating<tab>g10rating<tab>campRating<tab>points

Jason 2 2 2 5 32 (4)

2.6 Code a method called **removeVowels** to remove the vowels from the leader’s name excluding the first letter. The method must change the **name** field so that it is permanently changed. For example, Jared will become Jrd and Amed will become Amd. (9)

2.7 Code a method called **lessThan3** to return true if a leader has a rating less than three (i.e. a 1 or a 2) for any of the staff, Grade 11s, Grade 10s or the Camp Staff. The method must return false if ALL the ratings are 3 or above. (4)

**QUESTION 3 40 MARKS**

Consider the following UML diagram for the **LeaderArray** class.

|  |
| --- |
| **LeaderArray** |
| * Leader [] leadArr; * int size; |
| + Leader()  + sort()  + vowels()  + findLowRatings()  + toString() : String |

3.1 Code a class called **LeaderArray** to store 50 leaders in an array called **leadArr** and a variable to record the number of **Leader** objects in the array. (4)

3.2 Code a constructor to instantiate the array of leader objects using the text file **Leader.txt**. Display an error message if you cannot find the file. (8)

3.3 Code a **toString** method to display each leader, in the format of the **toString** of the **Leader** class. Each leader entry must appear on a new line. (5)

3.4 Code a method called **sort** to sort the array of leader elements in descending order of their points. Use the method **getPoints** you coded in question 2.4. (9)

3.5 Code a method to called **vowels** change the names of all the leaders to their abbreviated form without vowels. Use the method **removeVowels** in question 2.6. (5)

3.6 Code a method called **findLowRatings** to remove any leaders who have any rating less than 3. Use the method **lessThan3** that you coded in question 2.7. (9)

**QUESTION 4 6 MARKS**

4.1 Code a class called **LeaderUI**. (1)

4.2 Add code to instantiate a **LeaderArray** object. (1)

4.3 Add code to display all the leaders. (1)

4.4 Add code to sort and then display all the leaders. (1)

4.5 Add code to remove the vowels from the leader’s names and display all the leaders. (1)

4.6 Add code to remove any leaders who have a rating less than 3. Display the remaining leaders. (1)