|  |
| --- |
| **INFORMATION TECHNOLOGY**  **PAPER 1**  **3 hours**  **120 marks** |

|  |  |  |  |
| --- | --- | --- | --- |
| **INSTRUCTIONS AND INFORMATION** | | | |
|  |  | |  |
| 1. | This is a three-hour examination. Because of the nature of this examination it is important to note that you will NOT be permitted to leave the examination room before the end of the examination session. | |  |
|  |  | |  |
| 2. | You require the files listed below in order to answer the questions. They are EITHER on a stiffy disk OR CD issued to you, OR the invigilator/educator will tell you where to find them on the hard drive of the workstation you are using OR in which network folder it is. If the files are issued to you on a CD, you need to copy them onto your hard disk. | |  |
|  |  | |  |
|  | **QUESTION 1**  Question1\_u.pas  Question1\_p.dpr  Question1\_u.dfm  School.mdb  LearnersTb.txt  EducatorsTb.txt  **QUESTION 2**  Question2\_u.pas  Question2\_p.dpr  Question2\_u.dfm  Learner.txt  **QUESTION 3**  Question3\_u.pas  Question3\_p.dpr  Question3\_u.dfm | |  |
|  |  | |  |
|  | If a disk or CD containing the files was issued to you, write your surname on the label. | |  |
|  |  | |  |
| 3. | Save your work at regular intervals as a precaution against power failures. When saving please replace the ‘X’ with your surname. | |  |
|  |  | |  |
| 4. | Read ALL the questions carefully. Do only what is required by the question. | |  |
|  |  | |  |
| 5. | During the examination you may use the manuals originally supplied with the hardware and software. You may also use the HELP functions of the software. You may NOT refer to any other resource material. | |  |
|  |  | |  |
| 6. | At the end of this examination session you will be required to hand in the stiffy or CD given to you by the invigilator with your work saved on it, or you must make sure that all your work has been saved on the network as explained to you by the invigilator/educator. Ensure that all files can be read. | |  |
|  |  | |  |
| 7. | You also have to hand in printouts of the programming code for all the questions you have done. | |  |
|  |  | |  |
| 8. | All printing of programming questions will take place within an hour of the completion of the examination. | |  |
|  |  | |  |
| **SCENARIO**  **Schools need to determine staffing for the following year and it is all dependent on the subject choices of each grade. With the Grade 9s choosing their new subjects, they need to be advised on the different subjects as well as requirements of certain subjects. For example, to do Physical Science and Information Technology, Mathematics is advisable. Another concern is to ensure that learners that cannot cope with the demands of Mathematics, opt for Mathematical Literacy.** | | | |
|  |  | |  |
| **QUESTION 1: DATABASE AND DELPHI** | | | |
|  |  | |  |
| The database, **School.mdb**, which contains data related to this topic, has been supplied to you in a folder named **Question 1**.  Two text files have been supplied as well. If you cannot use the database provided, use the text files named **EducatorsTb.txt** and **LearnersTb.txt** to create your own database named **School** containing two tables named **LearnersTb** and **EducatorsTb**. Change the data types of the fields of the tables to the specifications given below.  The **EducatorsTb** table stores data on the educators. The fields in this table are defined as follows:    The following table is an example of the data contained in the table named **EducatorsTb** in the database named **School.mdb**.    The **LearnersTb** table stores data on the learners in Grade 9. The fields in this table are defined as follows:    The following table is an example of the data contained in the table named **LearnersTb** in the database named **School.mdb.**    You have also been supplied with an incomplete Delphi program with a unit named **Question1\_U** and a project named **Question1\_P** in the folder named **Question 1**. Open the incomplete program. | | | |
| * The program should be able to connect to the database named **School.mdb**. When you do QUESTION 1.1 and you find that the connectivity is not in place, use the following steps to establish connection with the database: * Click on the ADOQuery component named **qrySchool**. * Click on the Ellipse button (three dots) to the right of the ConnectionString property in the Object Inspector. * Click on the Build button which takes you to the Data Link Properties dialogue box. * Select Microsoft Jet 4.0 OLE DB Provider and click on Next. * The first option on the Connection tab sheet allows you to browse and find the **School.mdb** file. * Remove the user name Admin. * Click on the Test Connection button. * Click OK on each one of the open dialogue windows.   NOTE: If you cannot establish connectivity with the database at all when you execute the program you must still do and submit the programming code for marking.  **Marks will be awarded for the programming code that contains the SQL statements in the unit named Question1\_U as well as code that makes use of an inputbox as required by the question.** | | | |
|  |  | |  |
| 1.1 | Complete the code in the **All Educators** button by formulating an SQL statement to display all the fields in the EducatorsTb table in alphabetical order according to the surnames.  Example of output of the first few records: | | (3) |
|  |  | |  |
| 1.2 | Complete the code in the **Languages** button by formulating an SQL statement to display the Title, Surname and TeacherCode fields in the EducatorsTb table of all the teachers that teach a language (ENG/AFR/XHO).  Example of output: | | (5) |
|  |  | |  |
| 1.3 | Complete the code in the **Subject Numbers** button by formulating an SQL statement that will determine how many Grade9s have chosen to take a particular subject as entered via an inputbox. Display with a suitable heading.  Example of output if **IT** (which is in subject combination 5 – S5) was entered: | | (5) |
|  |  | |  |
| 1.4 | Complete the code in the **CAT May** button by formulating an SQL statement that will display the name and surname of all the learners who will be in Mrs May’s CAT class the following year. Mrs May’s code is AM.  Example of output of the first few records: | | (3) |
|  |  | |  |
| 1.5 | The Art teacher has resigned and the school has decided not to offer Art to the future Grade Tens as there is not enough interest, but to let them rather change to Life Science. Complete the code in the **Art** button by formulating an SQL statement to update the change from Art to Life Science (LSC) in the LearnersTb. Display all the fields in the LearnersTb table once the change has been made. | | (5) |
|  |  | |  |
| 1.6 | The Grade Head wants to ensure that all the classes are roughly the same size. Complete the code in the **Class Size** button by formulating an SQL statement that will display the class as well as the total number of learners in each class.  Example of output: | | (4) |
|  |  | |  |
| 1.7 | Complete the code in the **CONS or TOUR** button by formulating an SQL statement to display all the fields of the learners who want to do Consumer Studies (CONS) or Tourism (TOUR), which are both in Subject Combination 7 (S7), and are in the ‘G’ class.  Example of the output of the first few lines: | | (5) |
|  |  | |  |
| * Enter your name and surname as a comment line in the first line of the file named **Question1\_UX.pas** containing the SQL statements. * Save the unit **Question1\_UX** and the project **Question1\_PX** (File|Save All). * Make a printout of the code of the **Question1\_UX.pas** file. | | | **[30]** |
|  |  | |  |
| **QUESTION 2: DELPHI PROGRAMMING** | | | |
|  |  | |  |
| This question is intended to test object-oriented programming skills. You are required to produce a solution that includes all classes specified in the instructions. No marks will be allocated to any alternative solution such as one program not creating an object.  You have been supplied with a text file named **Learner.txt**, which contains information regarding Grade 9 learners.  The contents of the text file is shown below:    The name and surname is given first, then the English mark followed by the Maths mark. | | |  |
|  |  | |  |
| 2.1 | Create an object class (another unit) named **Subjects\_UX** and save this unit as **Subjects\_UX** in your **Question2\_X** folder (X should be replaced by your surname). Write the following code as part of this class: | |  |
|  |  | |  |
|  | 2.1.1 | Define a class named **TLearner**. This class must contain the following private fields:  fname  feng  fmaths  faverage  Ensure that you choose appropriate data types for these fields. | (4) |
|  |  | |  |
|  | 2.1.2 | Write a parameterised **constructor** method which accepts the name, English and Maths marks. All the fields must be initialised in the constructor. | (5) |
|  |  | |  |
|  | 2.1.3 | Write a method named **CalcAverage** which calculates the average of the Eng and Maths marks. | (3) |
|  |  | |  |
|  | 2.1.4 | Write a boolean method named **Qualify** which will determine the subjects which each learner may take in Grade Ten depending on their Grade Nine marks. In order to qualify their average for English and Maths (faverage) must be 60 or more.  The method must return true if the learner qualifies to do IT, Physical Science or Maths and false if they do not qualify. | (4) |
|  |  | |  |
|  | 2.1.5 | Write a method named **tostring** that builds and returns a string with information about each learner:  fname<tab>feng<tab>fmaths<tab>faverage | (3) |
|  |  | |  |
|  | 2.1.6 | Write an appropriately named ‘get’ method (accessor method) to return the name of the learner. | (2) |
|  |  | |  |
|  | 2.1.7 | Write an appropriately named ‘get’ method (accessor method) to return the average of the learner. | (2) |
|  |  | | **[23]** |
| 2.2 | Write code to do the following in the **Question2\_UX** unit in the given program. | |  |
|  |  | |  |
|  | 2.2.1 | Create an array named **arrLearners** that holds objects of **TLearner**. Write code in the OnActivate Event Handler of the form to read information from the text file **Learner.txt** according to the following steps:   1. Test if the file exists. Display a suitable message if the file does not exist and terminate the program. 2. Use a loop to ...  * Read a line of text from the text file * Separate the text into the name, English mark and Math mark * Use this information to create a new TLearner object and place the object in the array named arrLearners.  1. Use a counter variable to keep track of how many items there are in the array. | (13) |
|  |  | |  |
|  | 2.2.2 | **Menu Option: AVERAGE**  Display the name and average of the Maths and English marks for each learner. | (4) |
|  |  | |  |
|  | 2.2.3 | **Menu Option: IT, PHYSICAL SCIENCE, MATHS**  Call the tostring method to display the information of all the learners who qualify to do either IT, Maths or Physical Science. | (4) |
|  |  | |  |
|  | 2.2.4 | **Menu Option: SEARCH FOR A LEARNER**  Allow the user to enter the name of a learner. A suitable message must be displayed if the name was found or not. | (11) |
|  |  | |  |
| * Enter your name and surname as a comment line in the first line of the files named **Question2\_UX.pas** and **Subjects\_UX.pas**. * Save the units **Question2\_UX, Subjects\_UX** and the project **Question2\_PX** (File|Save All). * Make a printout of the code of the **Question2\_UX.pas** as well as the **Subjects\_UX.pas** file. | | | **[32]** |
|  |  | |  |
| **QUESTION 3: DELPHI PROGRAMMING** | | |  |
|  |  | |  |
|  | The Grade Ten’s have decided that they will be collecting donations for a charity organisation over a period of 4 weeks. The Grade Head has asked the Grade 12 IT class to write a program to manage the fundraising. | |  |
|  |  | |  |
| 3.1 | Write code for the **FormCreate** EventHandler that will place row headings (Class 1 – 7) and column headings (Week 1 – 4) in the StringGrid component.  This procedure must also generate random numbers between 100 and 600 (inclusive) which will represent the amount of money collected each week by the respective classes.  Example of output: | | (8) |
|  |  | |  |
| 3.2 | Write a procedure named **CalculateTotals** that will calculate the total amount of money collected by all the classes for each week. Display a heading as well as the totals for each week. | | (7) |
|  |  | |  |
| 3.3 | Write a procedure named **CalculateAvgWeek** that will calculate the average amount of money collected by all the classes for each week. Display a heading as well as the averages for each week. | | (9) |
|  |  | |  |
| 3.4 | Complete the code for the **Calculate Totals and Averages** EventHandler which will call the respective procedures.  Example of output: | | (2) |
|  |  | |  |
| 3.5 | Complete code for the **Class that raised the most money** EventHandler which will determine which class raised the most money. Use a ShowMessage component to display the winning class.  Example of output: | | (9) |
|  |  | |  |
| * Enter your name and surname as a comment line in the first line of the file named **Question3\_UX.pas.** * Save the unit **Question3\_UX** and the project **Question3\_PX** (File|Save All). * Make a printout of the code of the **Question3\_UX.pas** file. | | | **[35]** |