SENIOR CERTIFICATE EXAMINATION
NOVEMBER 2007

COMPUTER STUDIES HIGHER GRADE: PAPER II (PRACTICAL)

Time: 3 hours 80 marks

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This paper consists of 10 pages. Please check that your question paper is complete.

2. The question paper is to be answered using object oriented programming principles. All data is to be read and written to a database.

3. This paper is divided into two questions. Question one is to be answered by candidates programming in Java and Question two is to be answered by candidates programming in Delphi.

4. Only answer the relevant question which will ask detailed questions pertaining to your language and allocate marks according to how the task will be completed.

5. Make sure that you answer the questions in the manner described as marks will be awarded for your solution according to the specifications given.

6. Only answer the question that is stated. For example, if the question does not ask for data validation, then there are no marks awarded for it, and therefore no code needs to be written.

7. If you cannot get a section of code to work, comment it out so it will not be executed. However, it will still be marked. If possible try to explain the error to aid the marker.

8. Your program MUST compile. In order to ensure this, comment out the sections that do not compile.

9. When accessing a file, DO NOT give the full path names of the file, as this will create problems when the program is marked on a computer other than the one you are writing on. Merely refer to your file using its name and extension. Your data files MUST be in the same directory as program file(s).

10. Your programs must be coded in such a way that they work with any data and not just the sample data supplied.

11. Make sure that routines such as searches, sorts and selections are developed from first principles, and that you do not use the built-in features of a programming language for any of the above, unless you are accessing a table from a database using SQL statements.

12. In visual languages, ensure that you do not use the interface or built in components as your data structure. The data structure must be defined and declared by you the programmer.

13. Read the whole section before you choose a data structure. You will find that there may be an alternative method of representing the data which will be more efficient considering what questions are asked in the paper.

14. You must save all your work regularly on the disk you have been given, or the disk space allocated to you for this examination. If there is a power failure, you will only be given extra time equal to the length of the power failure. No extra time will given to make up for lost work.

15. Make sure that your name and examination number appears in every program that you code, as well as on every page of hard copy that you hand in.

16. Print a listing of all the programs/classes that you code. Print hard copy output, if possible. Printing is to be done after the examination. You will be given a half an hour to print after the examination has finished. Your teacher will tell you what arrangements have been made for the printing of your work.

17. Your program must make sensible use of subprograms and parameters. Marks will be deducted for lengthy subprograms, duplicate code and incorrect or insufficient use of parameters.

18. *If you are unable to access the database, then you can export the data to a text file and use a text file to input the data. You will be penalised for any missing SQL statements, however some credit will be awarded for a good working solution.*

QUESTION ONE
This section to be answered by candidates using JAVA

QUIZZ

Quinton Quizzicle has designed a quiz for his Computer Studies students. He has stored the questions, answers and the category of the question (hardware, system software and programming) in a table called **tblQuestions** in a database called **Quiz**. He wants to write a program to allow teachers to add and delete questions and also allow students to be tested using the questions he has stored in the database. A student can only run the test if he/she enters the correct password. Once a student has completed a test, a score will be displayed.

Clarification: The Access file is called **Quiz**. It contains a table called **tblQuestions** with the fields **Question, Answer** and **Category**. You need to know this for your SQL statements.

If you are unable to access the database, then you can export the data to a text file and use a text file to input the data. You will be penalised for any missing SQL statements, however some credit will be awarded for a good working solution.

*The table **tblQuestions** has been added as an Appendix on page 10. (You can type its contents into a text file if you are unable to export the data from the database. This is NOT recommended as you penalise yourself extensively with time).*

1.1 Create a class called **Question** with three fields.
- One field is used to store the question, another to store the answer and a third for the category.
- Declare these field variables so that they can be accessed by classes that inherit from this class.
- Use suitable types for each of these field variables. [3]

1.2 Code a constructor method to assign values to these variables based on the parameters passed to it. [2]

1.3 Code accessor methods for each field. [2]

1.4 Add a toString method to display the question, answer and category each on a separate line (for GUI environments this should generate a single string including an end of paragraph symbol (\n) to separate the lines). [3]

2.1 Create a class called **Student**
- that will have fields to store the student's name, the student's password and the student's test score.
- Use suitable types for each of these fields.
- Declare these fields so that they cannot be accessed outside the class. [3]

2.2 Code a constructor method which accepts parameters for the student's name and password. The score field must be initialised to 0. [3]

2.3 Add an accessor method for the score field – this must be displayed as a percentage, i.e. score out of 5. [2]

2.4 Add a method called **addToScore ()** to add an integer value to the score. The

integer value is sent as a parameter. [2]

3. **Java text based interface**

If you are using an IDE that produces a text based solution such as jGrasp or Ready then load the application called **QuizApp** that has been supplied to you and complete the code so that the following menu is displayed:

TEACHER MENU:
A. Display Questions
B. Insert Questions
C. Delete Questions

STUDENT MENU:
D. Run Quiz
E. Show Results
Q. Quit

Your code must be written so that the program redisplays the menu after an option has been selected and completed. The program must terminate if the user selects menu option Q (Quit).

GUI interface

If you are using a GUI interface such as Netbeans and can produce a graphical user interface then create an application called **QuizApp** with a main menu component that has the options TEACHER MENU and STUDENT MENU. The TEACHER MENU and the STUDENT MENU must appear as separate menu options on the menu bar. The TEACHER MENU must have sub-options Display Questions, Insert Questions and Delete Questions. The STUDENT MENU must have sub-options Run Quiz and Show Results:

TEACHER MENU	STUDENT MENU
Display Questions	Run Quiz
Insert Questions	Show Results
Delete Questions	

[3]

4. Load the class called **DBQuiz** that has been provided for you to manage the database and run the quiz. The **DBQuiz** class contains the code in its constructor to connect to the database. You will need to add code to this class to perform the menu options listed in question 3.

(Ideally managing the data and running the quiz should be performed in separate classes, but they have been combined for the sake of simplicity for this examination. You may choose to separate these classes if you feel that it better suits your solution).

4.1 Instantiate an object of the **DBQuiz** in the **QuizApp** class to establish a connection with the database called **Quiz** that has been provided for you.

[3]

A Menu Option Display Questions

TEXT BASED INTERFACE
Code a method called **displayQuestions** in the **DBQuiz** class to display all the questions, answers and categories in the table sorted alphabetically by category. Provide a suitable heading for the display.

GUI INTERFACE
Code a typed method called **displayQuestions** in the **DBQuiz** class to return a string listing all the questions, answers and categories in the table sorted alphabetically by category. Provide a suitable heading for the display. This should be displayed in an appropriate component on the form. [5]

B Menu Option Insert Questions

TEXT BASED INTERFACE
Code a method called **insertQuestions** in the **DBQuiz** class to repeatedly insert a record consisting of a question, answer and category into the table called **tblQuestions**. Text based interface can terminate this entry process when the user enters a space.

GUI INTERFACE
Code a method called **insertQuestions** in the **DBQuiz** class which accepts 3 string parameters (question, answer, category) and inserts a record into the database. You will need to have 3 text fields and a button on the form and an event handler for the button that calls this method and passes the contents of the text fields as the parameters. [6]

C Menu Option Delete Questions

Code a method called **deleteQuestion** in the **DBQuiz** class to delete a record in the table called **tblQuestion**. The user will input a question or part of the question that is to be deleted and the corresponding record must be removed from the table **tblQuestions**. [4]

D Menu Option Run Quiz

D.1 In the **DBQuiz** class create an array of 20 elements of the **Question** class to store the questions and answers from the database for a particular category. This array is a field of the **DBQuiz** Class. Add a field to record the number of questions and answers of a category in the array. [2]

D.2 Code a method called **importQuestions** in the **DBQuiz** class to input the category (either ***Hardware***, ***System Software*** or ***Programming*** – NB: case is important) that the user wishes to use as a test. GUI interfaces may use an Input Dialog box for this. Using this category, load only the questions that are in that category into the array you created in D.1 from the table called **tblQuestions** in the database. [8]

D.3 Create a method called **selectQuestions** in the **DBQuiz** class that, using the data in the array, limits the test to 5 questions in a random order. You may do this any way you choose, including creating a separate array for the test questions. Make sure that each question has the possibility of being a question in the test and that no question appears twice. Marks will be awarded for the elegance of your solution. [8]

D.4 In the main form of the GUI/**QuizApp** class create a void method to allow the user to enter their name and a password. In a GUI this may be done by using 2 input boxes. The password must then be validated using the following rules:

- The password must contain an even number of characters.
- The letters in the password must be alternate between vowel and consonant. The first character must be a vowel, the second a consonant, the third a vowel and the fourth a consonant etc. For example "otibax" would be a valid password.
- The password may contain upper or lower case characters.

The user must be repeatedly prompted to re-enter the password if it is incorrect.

If the password is correct, instantiate a **Student** object using the name and password as parameters. The program must ensure that a valid password is entered before a **Student** object is created. [12]

D.5 The user must be given the opportunity to run the test by having each question displayed and the user prompted for the answer. Once the user has entered the answer the user's response must be checked against the answer stored in the array. If the answer is correct, the score field in the Student object must be incremented by 1. Once five questions have been asked, the test is completed. [6]

E Menu Option Show Results

Add code to display the student's result for their test by converting the number of correct answers out of 5 to a percentage. [3]

80 marks

QUESTION TWO
This section to be answered by candidates using DELPHI

QUIZZ

Quinton Quizzicle has designed a quiz for his Computer Studies students. He has stored the questions, answers and the category of the question (hardware, system software and programming) in a table called **tblQuestions** in a database called **Quiz**. He wants to write a program to allow teachers to add and delete questions and also allow students to be tested using the questions he has stored in the database. A student can only run the test if he/she enters the correct password. Once a student has completed a test, a score will be displayed.

Clarification: The Access file is called **Quiz**. It contains a table called **tblQuestions** with the fields **Question, Answer** and **Category**. You need to know this for your SQL statements.

If you are unable to access the database, then you can export the data to a text file and use a text file to input the data. You will be penalised for any missing SQL statements, however some credit will be awarded for a good working solution.

*The table **tblQuestions** has been added as an Appendix on page 10. (You can type its contents into a text file if you are unable to export the data from the database. This is NOT recommended as you penalise yourself extensively with time.)*

1.1 Create a class called **TQuestionClass** with three fields.
- One field is used to store the question, another to store the answer and a third for the category.
- Declare these field variables so that they can be accessed by classes that inherit from this class.
- Use suitable types for each of these field variables. [3]

1.2 Code a constructor subprogram to assign values to these variables based on the parameters passed to it. [2]

1.3 Code accessor subprograms for each field. [2]

1.4 Add a toString subprogram to display the question, answer and category each on a separate line (this should generate a single string including an end of paragraph symbol (#10 / #13) to separate the lines). [3]

2.1 Create a class called **TStudentClass**
- that will have fields to store the student's name, the student's password and the student's test score.
- Use suitable types for each of these fields.
- Declare these fields so that they cannot be accessed outside the class. [3]

2.2 Code a constructor subprogram which accepts parameters for the student's name and password. The score field must be initialised to 0. [3]

2.3 Add an accessor subprogram for the score field – this must be displayed as a percentage, i.e. score out of 5. [2]

2.4 Add a subprogram called **addToScore ()** to add an integer value to the score. The integer value is sent as a parameter. [2]

3. Create an application called **QuizApp** with a main menu component that has the options TEACHER MENU and STUDENT MENU. The TEACHER MENU and the STUDENT MENU must appear as separate menu options on the menu bar. The TEACHER MENU must have sub-options Display Questions, Insert Questions and Delete Questions. The STUDENT MENU must have sub-options Run Quiz and Show Results:

TEACHER MENU	STUDENT MENU
Display Questions	Run Quiz
Insert Questions	Show Results
Delete Questions	

[3]

4. Add the unit udbQuiz.pas, i.e. add it to the uses clause of the form unit so that you can access the class called **TDBQuiz** (in the file udbQuiz.pas) that has been provided for you to manage the database and run the quiz.

NB: The **TDBQuiz** class contains the code in its constructor to connect to the database. When you instantiate this class the database will be connected and ready for use. The connection supplied is a query component called **DBQz**. All you need to do to work with the database is change and execute the SQL for this component.

You will need to add code to this class to perform the menu options listed in question 3.

(Ideally managing the data and running the quiz should be performed in separate classes, but they have been combined for the sake of simplicity for this examination. You may choose to separate these classes if you feel that it better suits your solution).

4.1 Instantiate an object of the **TDBQuiz** class in your main form unit. This will establish a connection with the database called **Quiz** that has been provided for you. The ADO query component you work with is called **DBQz** (this means your instantiated object of TBDQuiz cannot be called DBQz). [3]

A Menu Option Display Questions

Code a function called **displayQuestions** in the **TDBQuiz** class to return a string listing all the questions, answers and categories in the table sorted alphabetically by category. Provide a suitable heading for the display. This should be displayed in an appropriate component on the form. [5]

B Menu Option Insert Questions

Code a subprogram called **insertQuestions** in the **TDBQuiz** class which accepts 3 string parameters (question, answer, category) and inserts a record into the database. You will need to have 3 edits and a button on the form and an event handler for the button that calls this subprogram and passes the contents of the edits as the parameters. [6]

C Menu Option Delete Questions

Code a subprogram called **deleteQuestion** in the **TDBQuiz** class to delete a record in the table called **tblQuestion**. The subprogram receives a string parameter. The user will input a question or part of the question that is to be deleted and the corresponding record must be removed from the table **tblQuestions**. [4]

D Menu Option Run Quiz

D.1 In the **TDBQuiz** class create an array of a 20 elements of the class called the **TQuestionClass** to store the questions and answers from the database for a particular category. This array is a field of the **TDBQuiz** Class. Add a field to record the number of questions and answers of a category in the array. [2]

D.2 Code a subprogram called **ImportQuestions** in the **TDBQuiz** class to input the category (either ***Hardware***, ***System Software*** or ***Programming*** – NB: case is important) that the user wishes to use as a test. Use an Input Dialog box for this. Using this category, load only the questions that are in that category into the array you created in D.1 from the table called **tblQuestions** in the database. [8]

D.3 Create a subprogram called **SelectQuestions** in the **TDBQuiz** class that, using the data in the array, limits the test to 5 questions in a random order. You may do this any way you choose, including creating a separate array for the test questions. Make sure that each question has the possibility of being a question in the test and that no question appears twice. Marks will be awarded for the elegance of your solution. [8]

D.4 In the main form create a procedure to allow the user to enter their name and a password. In a GUI this may be done by using 2 input boxes. The password must then be validated using the following rules:

- The password must contain an even number of characters.
- The letters in the password must be alternate between vowel and consonant. The first character must be a vowel, the second a consonant, the third a vowel and the fourth a consonant, etc. For example "otibax" would be a valid password.
- The password may contain upper or lower case characters.

The user must be repeatedly prompted to re-enter the password if it is incorrect. If the password is correct, instantiate a **TStudentClass** object using the name and password as parameters. The program must ensure that a valid password is entered before a **TStudentClass** object is created. [12]

D.5 The user must be given the opportunity to run the test by having each question displayed and the user prompted for the answer. Once the user has entered the answer the user's response must be checked against the answer stored in the array. If the answer is correct, the score field in the **TStudentClass** object must be incremented by 1. Once five questions have been asked, the test is completed. [6]

E Menu Option Show Results

Add code to display the student's result for their test by converting the number of correct answers out of 5 to a percentage. [3]

80 marks

APPENDIX – Table **tblQuestions** in the database Quiz

Question	Answer	Category
Name the component that is primary memory but is powered by a small battery.	CMOS	Hardware
What is the name of the data structure that can store elements of the same type.	Array	Programming
Give the name of the programming structure to repeat an instruction a fixed number of times.	For loop	Programming
True or False. A while loop can never be infinite.	False	Programming
True or False. A global variable can be accessed throughout a program.	True	Programming
Name the data structure that can store many characters.	String	Programming
True or False. An argument is another name for a formal parameter.	False	Programming
A(n) ---------- is a variable that never changes.	Constant	Programming
True or False. An alias is the cute girl on TV who does amazing karate moves.	False	Programming
Another word for creating an object is ----------.	Instantiation	Programming
A common name for a programming error is a --------.	Bug	Programming
A(n) -------- translates source code line by line.	Interpreter	System Software
Give the common name for malicious code that can damage your computer.	Virus	System Software
Give the name of system software that manages and controls the computer.	Operating System	System Software
A(n) ------- is a signal sent from a device to the CPU to indicate that the device needs servicing.	Interrupt	System Software
----- memory is used to speed up access to slower RAM.	Cache	Hardware
The process of making extra copies of all your important files is called -------.	Backup	System Software
Data ------ is where files are made smaller to take up less space.	Compression	System Software
----- is where an instruction is divided into stages so that the CPU executes an instruction on each clock tick.	Pipelining	Hardware
A device to convert analog signals to digital signals and visa versa.	Modem	Hardware
A type of memory that is used when the computers hard drive is used to simulate RAM.	Virtual	System Software
A user friendly interface that consists of windows, icons mouse and a pointer. (Give the three letter acronym).	GUI	System Software
A type of printer that uses toner.	Laser	Hardware
An input device that can represent a picture or text on paper digitally.	Scanner	Hardware
A hard disk is divided into ----- and sectors.	Tracks	Hardware
A type of bus that is exclusively used for graphics. (Give the acronym)	AGP	Hardware
The ------- bus transfers data and instructions from the RAM to the CPU.	Data	Hardware
An optical medium that can store about 700 MB.	CD	Hardware
The program counter on the CPU is an example of a ------.	Register	Hardware
Games are the only types of application packages	False	System Software