NATIONAL SENIOR CERTIFICATE EXAMINATION
MAY 2021

INFORMATION TECHNOLOGY: PAPER II

Time: 3 hours 120 marks

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This question paper consists of 12 pages. Please check that your question paper is complete.

2. This question paper is to be answered using object-oriented programming principles. Your program must make sensible use of methods and parameters.

3. This paper is divided into two sections. All candidates must answer both sections.

4. This paper is set in programming terms that are not specific to any particular programming language (Java/Delphi) or database (Access/MySQL/JavaDB).

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

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

7. If you cannot get a section of code to work, comment it out so that it will not be executed and so that you can continue with the examination. If possible, try to explain the error to aid the marker.

8. Your programs must be coded in such a way that they will work with any data and not just the sample data supplied or any data extracts that appear in the question paper. You are advised to look at the supplied data files carefully.

9. Make sure that routines such as searches, sorts and selections for arrays are developed from first principles, and that you do not use the built-in features of a programming language for any of these routines.

10. All data structures must be defined and declared by you, the programmer. You may not use components provided within the interface to store and later retrieve data.

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

12. You must save all your work regularly on the disk you have been given, or the disk space allocated to you for this examination. You should also create a backup of the original files before you start in case the original version is accidentally modified by your solution.

13. If your examination is interrupted by a technical problem such as a power failure, you will, when you resume writing, be given only the time that was remaining when the interruption began to complete your examination. No extra time will be given to redo work that had not been saved.

14. Make sure that your examination number appears as a comment in every program that you code as well as on every page of hard copy that you hand in.

15. Print a code listing of all the programs/classes that you have coded. Printing must be done after the examination. You will be given 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.

16. You should be provided with the following two folders (in bold) and files. These files are to be used as data for this examination. Note that the database files are provided in MS Access, JavaDB and MySQL format. Ensure that you are able to open the files with the packages that you will use to code your solutions to this examination.

Section A:
ExaminationsDB.mdb (Access)
ExaminationsDB.sql (MySQL import)
ExaminationsJavaDB.sql (JavaDB import)
SQL Answer Sheet.rtf
SQLBrowser.txt
Section B:
mysubjects.txt
timetable.txt

SCENARIO

The Examinations Board of South Africa (EBSA) offers supplementary examinations to students under certain circumstances. They are developing software to assist with the scheduling and registration of students for the supplementary examinations. They require your assistance with their database as well as some software that will help students organise their individual supplementary examinations. Data from 2017 is provided in order to develop and test the database.

SECTION A STRUCTURED QUERY LANGUAGE

EBSA offers certain subjects as supplementary examinations. These examinations are written on most days of the month of March every year. Every day there is a morning examination session starting at 09:00 and an afternoon examination session starting at 14:00. Every subject can have one or two papers depending on the subject in question. Students from various centres (schools) need to record their details with EBSA. In addition, they need to register for the supplementary examinations they wish to write. A student may write a number of supplementary examinations, but they do not need to write all papers in a given subject. For example, a student can register for Mathematics Paper 1 (MAT1) **OR** Paper 2 (MAT2) **OR** both. A student who registers to write a paper is called a candidate.

The database **ExaminationsDB** contains three tables. The fields in the database are described in the following tables together with some sample data. The first ten rows of data are shown, but the tables do contain more data.

tblStudents contains the details of all the students that are registered to write a supplementary examination.

Field Name	Data Type	Description
StudentID	Number	A unique ID for each student. This field is an autonumber field.
Firstname	Text	The first name of the student.
Surname	Text	The surname of the student.
DateOfBirth	Date	The date of birth of the student.
Gender	Text	The gender of the student as a single character "M" or "F".
Centre	Number	The centre number of the student.

tblStudents table – sample data (first 10 records)
Data might be formatted differently on your computer depending on its settings

StudentID	Firstname	Surname	DateOfBirth	Gender	Centre
1	Keiko	Abbott	1999-03-30	F	1003
2	Eagan	Alvarado	2000-04-06	M	1006
3	Cynthia	Andrews	1999-05-08	F	1000
4	Kelsie	Atkinson	2000-04-09	F	1007
5	Nichole	Avila	2000-03-29	F	1014
6	Colette	Baldwin	2000-05-24	F	1012
7	Tate	Bates	2000-01-21	M	1007
8	Fulton	Bauer	1999-02-25	M	1016
9	Hector	Beasley	1999-12-30	M	1019
10	Lynn	Becker	2000-05-29	F	1011

tblTimeTable contains information on all the papers that are available to write during the supplementary examinations.

Field Name	Data Type	Description
PaperID	Text	The unique ID of each examination paper.
ExamDate	Date	The date on which the paper is to be written.
Paper	Text	The name of the examination paper.
StartTime	Time	The starting time of the exam.
EndTime	Time	The ending time of the exam.

tblTimeTable table – sample data (first 10 records)
Data might be formatted differently on your computer depending on its settings

PaperID	ExamDate	Paper	StartTime	EndTime
ACC1	2017-03-27	Accounting PI	09:00	11:00
ACC2	2017-03-17	Accounting PII	14:00	16:00
AFRA1	2017-03-02	Afrikaans First Add Language PI (Reading)	14:00	16:30
AFRA2	2017-03-09	Afrikaans First Add Language PII (Writing)	14:00	16:30
AFRH1	2017-03-01	Afrikaans Home Language PI (Reading)	14:00	17:00
AFRH2	2017-03-08	Afrikaans Home Language PII (Writing)	14:00	17:00
BUS1	2017-03-17	Business Studies PI	09:00	11:00
BUS2	2017-03-15	Business Studies PII	14:00	16:00
CAT1	2017-03-21	Computer Applications Technology PI (Theory)	09:00	12:00
CAT2	2017-03-28	Computer Applications Technology PII (Practical)	09:00	12:00

tblCandidates contains details of all the students who have registered (i.e. the candidates) and for which examinations. A student may register for one or more supplementary examinations.

Field Name	Data Type	Description
StudentID	Number	The ID of the student who is writing the examination. This field is a foreign key to **tblStudents**.
PaperID	Text	The ID of the examination paper that the student is writing. This field is a foreign key to **tblTimeTable**.

tblCandidates table – sample data (first 10 records)

StudentID	PaperID
1	LIF2
2	ZUL1
2	TOU
2	ENGH1
2	ZUL2
2	ITN1
3	CON
3	HIS2
4	AFRH2
4	MAT1

QUESTION 1

1.1 Write a query that will list the **Paper**, **StartTime** and **EndTime** of all the examination papers in **tblTimeTable** that end at 16:30. The correct output is given below.

Note that data might be formatted and ordered differently on your computer.

Paper	StartTime	EndTime
Afrikaans First Add Language PII (Writing)	14:00	16:30
English First Add Language PI (Reading)	14:00	16:30
English First Add Language PII (Writing)	14:00	16:30
IsiZulu First Add Language PI (Reading)	14:00	16:30
IsiZulu First Add Language PII (Writing)	14:00	16:30

(3)

1.2 Write a query that will list all the details of all examination papers in **tblTimeTable** that are Paper II for that subject. Paper II subjects will contain "PII" in the paper name. Your results should be sorted by **ExamDate** in ascending order. The first 5 rows of the correct output are shown below.

Note that data might be formatted and ordered differently on your computer.

PaperID	ExamDate	Paper	StartTime	EndTime
PHY2	2017-03-03	Physical Sciences PII (Chemistry)	09:00	12:00
HIS2	2017-03-06	History PII	09:00	11:00
AFRH2	2017-03-08	Afrikaans Home Language PII (Writing)	14:00	17:00
AFRA2	2017-03-09	Afrikaans First Add Language PII (Writing)	14:00	16:30
ZUL2	2017-03-10	IsiZulu First Add Language PII (Writing)	14:00	16:30
...	...	...	...	...

(4)

1.3 Write a query that will list the **Firstname**, **Surname** and **DateOfBirth** of all male students in **tblStudents** who have a birthday in March. The first 5 rows of the correct output are shown below.

Note that data might be formatted and ordered differently on your computer.

Firstname	Surname	DateOfBirth
Ira	Bell	1999-07-03
Rigel	Blackburn	1999-03-20
Forrest	Bryan	1999-03-27
Chancellor	Buck	1999-03-29
Erich	Carey	2000-03-16
...	...	...

(4)

1.4 Write a query that will determine the number of male and female students from each centre. Your query should list all centre numbers and the total number of males and females in every centre. The new field should be named **StudentTotal**. The first 5 rows of the correct output are shown below.

Centre	Gender	StudentTotal
1000	F	4
1000	M	2
1001	F	2
1001	M	1
1002	M	1
...	...	...

(5)

1.5 Write a query that will add a new student to **tblStudents**. The student's **firstname** is "Sarah" and her **surname** is "Henderson"; she is a **female born on 1999-09-23** and is registered to write at **centre** 1021. (4)

1.6 Write a query that will list the **PaperID**s of all Mathematics or Mathematical Literacy examination papers that have more than seven students registered to write. Your query should list the **PaperID** and the total number of students registered to write that examination paper. The correct output is given below.

PaperID	StudentCount
MAT1	11
MATL2	9

(6)

1.7 Write a query that will display the **PaperID** of any examination paper that does not have a student registered to write. Your query should display only the **PaperID**. A sample of the correct output is given below.

PaperID
ENGA1

(6)

1.8 Write a query that will display the full name of each student and the date of their last examination. The student's full name is their **Firstname** and **Surname** separated by a space. The date of a student's last exam is the date of the latest exam for which that student is a candidate. The first 5 rows of the correct output are given below.

Note that data might be formatted differently on your computer.

FullName	LastExam
Abbot Kemp	2017-03-03
Adria Hartman	2017-03-14
Ahmed Elliott	2017-03-23
Aladdin Haley	2017-03-24
Alden Bell	2017-03-06
...	...

(8)

40 marks

SECTION B OBJECT-ORIENTED PROGRAMMING

EBSA wants to develop a computer program that will read in the timetable and one student's subject choices and display a variety of information about the supplementary examinations.

Supplementary examinations are written on weekdays – either in the morning (AM) or in the afternoon (PM). Morning examinations start at 09:00 (9 AM) and afternoon examinations start at 14:00 (2 PM). Each examination paper may have a different duration, but they all start at either 09:00 or 14:00. The end time of the examination paper can be calculated by adding the duration to the start time. If a subject has two papers, the number of the paper (PI or PII) will be included in the name of the paper.

Each **Exam** has the following information:

- Examination Date – the date on which the examination paper is to be written
- Paper – the name of the paper
- Start Time – the time at which the paper starts
- Duration – the duration of the paper in hours

Each day of March can be regarded as an **ExamDay**. An **ExamDay** stores the details for one particular day in March, including the examination papers for that day, if applicable. Data from 2017 is provided to help develop and test this program.

Each **ExamDay** contains the following information:

- Date – the full date of each day in the month. Possible values are any date from 2017-03-01 to 2017-03-31.
- Morning Exam – the details of the exam paper scheduled for the morning of that day, if any.
- Afternoon Exam – the details of the exam paper scheduled for the afternoon of that day, if any.

Some days there will only be a morning or an afternoon examination. There are no supplementary examinations scheduled on weekends. You have been given a file called *timetable.txt*, which contains the information of exactly 37 supplementary examination papers set to be written in March 2017. There will always be exactly 37 papers in the given file. The papers are listed in date order. There are no examination papers that are on the same date and time as another examination paper. Each line of this file contains the information of a single supplementary examination paper for a particular subject in the month of March. The line is formatted as follows:

```
examinationDate#paper#startTime#duration
```

The first 10 lines of the file are given below.

```
2017-03-01,Economics,09:00,3
2017-03-01,Afrikaans Home Language PI (Reading),14:00,3
2017-03-02,Physical Sciences PI (Physics),09:00,3
2017-03-02,Afrikaans First Add Language PI (Reading),14:00,2.5
2017-03-03,Physical Sciences PII (Chemistry),09:00,3
2017-03-03,IsiZulu First Add Language PI (Reading),14:00,2.5
2017-03-06,History PII,09:00,2
2017-03-06,English Home Language PI (Reading),14:00,3
2017-03-07,Mathematics PI,09:00,3
2017-03-07,English First Add Language PI (Reading),14:00,2.5
```


QUESTION 2

Use the class diagram below to create a new class called **Exam**. This class will be used to create objects that will store the details of one examination paper. The diagram below indicates the fields and methods that are required. Take careful note of the method and parameter names in the UML class diagram.

Exam
– examDate : string – paper : string – startTime : string – duration : real
+ Constructor(inExamDate : string, inPaper : string, inStartTime : string, inDuration : real) + getEndTime() : string + getAMorPM() : string + getExamDate() : string + getPaper() : string + toString() : string

2.1 Write code to create a new class called **Exam**. (1)

2.2 Write code to create the four fields for the **Exam** class as indicated in the class diagram above. (3)

2.3 Write code to create a constructor method that accepts three strings and one real number as parameters that represent the examination date, paper name, start time and duration, respectively. These parameters should be used to set the values of the four fields of the **Exam** class. (3)

2.4 Write code to create a method called ***getEndTime()*** that returns a string containing the end time of the examination. Your method should use the start time and duration fields to calculate the end time. (5)

2.5 Write code to create a method called ***getAMorPM()*** that will return a string containing "AM" or "PM" based on the starting time of the examination. All morning examinations start at 09:00 and should return "AM". All afternoon exams start at 14:00 and should return "PM". (3)

2.6 Write code to create accessor methods for the **examDate** and **paper** fields. (2)

2.7 Write code to create a method called ***toString()*** that will return the information about the **Exam**. The format of the returned string should be as follows:

```
startTime-endTime<space>paper
```

e.g.

```
09:00-12:00 Economics
```

(3)

[20]

QUESTION 3

Use the class diagram below to create a new class called **ExamDay**. This class will be used to store the details of the examination papers for a single day in a month. Two of the fields are **Exam** objects (the class created in the previous question). The diagram below indicates the fields and methods that are required.

ExamDay
– examDate : string – morning : **Exam** – afternoon : **Exam**
+ Constructor(inExamDate : string, inMorning : **Exam**, inAfternoon : **Exam**) + getExamDate() : string + setExam(exam : **Exam**) + toString() : string

3.1 Write code to create a new class called **ExamDay**. (1)

3.2 Write code to create three fields that will store the **examDate, morning** and **afternoon** examinations associated with an **ExamDay**. Note that the **morning** and **afternoon** fields are objects of the class **Exam**. These fields should not be visible from outside the class. (3)

3.3 Write code to create a constructor method that will initialise all the fields of the **ExamDay** class. Note that in addition to the examDate parameter, the second and third parameters are objects of the **Exam** class. Use the parameters to initialise the fields of the **ExamDay** class. (3)

3.4 Write code to create an accessor method for the **examDate** field. (2)

3.5 Write code to create a method called ***setExam(..)*** that accepts an **Exam** object as a parameter. This method should determine whether the **Exam** object passed in as a parameter is a morning or afternoon exam and then set the relevant field of the **ExamDay** class. (4)

3.6 Write code to create a ***toString()*** method that will return a string comprised of the information for the **ExamDay**. The string returned should be in the following format:

```
Examination Date<newline>
AM: <morning examination details><newline>
PM: <afternoon examination details><newline>
```

Use the ***toString()*** methods of the **morning** and **afternoon** fields respectively to display the information for the morning and afternoon examinations. For example:

```
2017-03-02
AM: 09:00-12:00 Physical Sciences PI (Physics)
PM: 14:00-16:30 Afrikaans First Add Language PI (Reading)
```

Note that on certain days no exams or only one exam might be written. An example of an **ExamDay** with no afternoon exam is as follows:

```
2017-03-07
AM: 09:00-12:00 Mathematics PI
PM: No Exam
```

(6)
[19]

QUESTION 4

4.1 Write code to create a new class called **ExamManager**. (1)

4.2 Write code to declare the following two arrays as instance variables:

- An array that can be used to store exactly 37 **Exam** objects.
- An array that can be used to store exactly 31 **ExamDay** objects (one object for each day in the month of March 2017). (4)

4.3 Write code to create a constructor method that will read the contents of a file of examinations. The file name should be given as a string parameter to the constructor method. Each line will result in one **Exam** object being added to the array. Do the following:

- Check if the file exists. If it does not exist, display an error message.
- Open the file for reading.
- Loop through the file 37 times. In each iteration of the loop:
  - Read in one line and split the **Exam** data contained in that line into the examination date, paper name, start time and duration.
  - Create an **Exam** object using the data.
  - Add the **Exam** object to the array.
  - Note that you do not need to create any **ExamDay** objects in this method. (10)

4.4 Write code that will create a method called ***displayAll***. This method should return a string that contains the information of all examinations. Use the object's ***toString*** methods that you created in the questions above. Each **Exam**'s details should be on a new line. (5)

4.5 Write code to create a method called ***getExamOnDay*** that will search the array of examination objects for the examination object on a particular day at a particular time and return the **Exam** object if found. This method should accept two parameters: a String representing the date and a String containing either "AM" or "PM". This method should loop through the array of examinations and return the **Exam** object that occurs on the day and in the session ("AM" or "PM") specified by the parameters. If no match is found, return **null/nil**.

There is only one examination per session ("AM" or "PM") on a given date. Any search is acceptable. (6)

[26]

QUESTION 5

5.1 Write code to create a simple user interface called **ExamUI** that will allow simple output. (1)

5.2 Declare and instantiate an **ExamManager** object at the appropriate place in the code. Call the constructor using *timetable.txt* as the filename argument. (1)

5.3 Write code that will display a list of all examinations by calling the appropriate methods in the **ExamManager** class. (1)

[3]

QUESTION 6

You have been provided with a file called *mysubjects.txt* that contains a list of subjects that a particular student is registered for. You are required to use the code written in the previous questions to display a full examination timetable for that student, including days with no examinations. The file contains the following data:

```
English Home Language
Afrikaans First Add Language
Mathematics
Information Technology
Geography
Physical Sciences
```

You are required to extract the data from the file and generate an **ExamDay** object for each day from 1 March 2017 to 31 March 2017 (including weekends). You should only include examinations that match the student's subjects in the file, *mysubjects.txt.*

6.1 Write code that will create a method called ***populateTimeTable*** in the **ExamManager** class. This method should accept the filename that contains the subject list as a parameter. This method should then populate the array of **ExamDay** objects based on the subjects in the file. The method should return a string of all the **ExamDay** objects.

Note that this particular student will write every paper associated with the subjects in the file, e.g. Mathematics is listed as one of the student's subjects and the student will write both PI and PII for this subject. (11)

6.2 Write code in **ExamUI** that will call the ***populateTimeTable*** method and display the results. The first 12 lines of the correct output are shown below:

```
2017-03-01
AM: No Exam
PM: No Exam

2017-03-02
AM: 09:00-12:00 Physical Sciences PI (Physics)
PM: 14:00-16:30 Afrikaans First Add Language PI (Reading)

2017-03-03
AM: 09:00-12:00 Physical Sciences PII (Chemistry)
PM: No Exam

2017-03-04
AM: No Exam
PM: No Exam
```

(1)
[12]

80 marks

Total: 120 marks