NATIONAL SENIOR CERTIFICATE EXAMINATION
NOVEMBER 2020

INFORMATION TECHNOLOGY: PAPER II

Time: 3 hours 120 marks

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This question paper consists of 14 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 catch up on work that was not 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 code. Printing must be done after the examination. You will be given half an hour to print after the examination is 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:
    HopeDB_Access.mdb
    HopeDB_JavaDB.txt
    HopeDB_MySQL.txt
    SQLAnswerSheet.rtf
    SQLBrowser.exe

    Section B:
    clients.txt
    counsellors.txt

SCENARIO

A community-based counselling service called HOPE provides counselling for clients at a reduced fee called a rate. Clients can make multiple appointments with a counsellor at a particular office called the location.

SECTION A SQL

QUESTION 1

A database is used to keep track of clients and the appointments they make with counsellors. This database consists of three tables:

CLIENT table contains the details of all clients that can potentially book appointments.

FIELDS	**DATA TYPE**	**DESCRIPTION**
ClientID	INTEGER	A unique auto-numbered identification number for each client
ClientName	TEXT	The name and surname of the client
Age	INTEGER	The age of the client when they first joined HOPE
PostCode	TEXT	The postcode of the client's residential address

COUNSELLOR table contains the detail of all counsellors working for HOPE.

FIELDS	**DATA TYPE**	**DESCRIPTION**
CounsellorID	INTEGER	A unique auto-numbered identification number for each counsellor
CounsellorName	TEXT	The name of the counsellor
Rate	DOUBLE	The rate charged by this counsellor

APPOINTMENT table contains details of appointments made by clients with a counsellor for a particular date and location.

FIELDS	**DATA TYPE**	**DESCRIPTION**
AppointmentID	INTEGER	A unique auto-numbered identification number for each appointment
ClientID	INTEGER	The ClientID of the client who made this appointment. This is a foreign key to the Client table.
CounsellorID	INTEGER	The CounsellorID of the counsellor with whom this appointment was booked. This is a foreign key to the Counsellor table.
AppointmentDate	DATE	The date of the appointment
Location	TEXT	The name of the office for this appointment.

1.1 Display all information about Clients whose ages range from 18 to 30 years, inclusive. Order this list in ascending order of age. The correct output is shown below. Note that the secondary ordering may be different depending on the database package used.

ClientID	ClientName	Age	PostCode
28	Cynthia Fourie	18	1500
8	Lindewe Khoza	18	1500
16	Maria Nkosi	21	0152
3	Steve Jacobs	21	0125
33	Bianca Abrahams	22	0160
26	Zandile Methembu	22	1600
2	Leo Sithole	22	1240
31	Al Naidoo	23	0127
23	Willem du Plessis	23	1600
4	Pat Khumalo	23	1251
30	Bongiwe Mokoena	25	0160
7	Sibongile Ngcobo	25	1240
37	Hendrick van Zyl	27	0180
15	Peter Zwane	30	1240

(3)

1.2 Display the clients who live in the region whose postcode starts with `012`. The correct output is shown below. Note that the ordering may be different depending on the database package used.

ClientID	ClientName	Age	PostCode
6	Musi Mahlangu	38	0120
18	Patricia Williams	17	0120
35	Moses Sibisi	31	0121
38	Lucky Shabangu	16	0123
3	Steve Jacobs	21	0125
31	Al Naidoo	23	0127

(3)

1.3 Client codes need to be generated. This can be done by taking the last two letters of the client name and combining it with a random number from 11 to 17 inclusive. Write an SQL statement to output the client codes for all clients. Display the new client code next to the client name. A sample of the output is shown below. Note that the code will differ depending on the random number generated.

Note that only the first five records are shown.

ClientName	***Code*** *Note that the last two digits are randomly generated*
John Dlamini	ni12
Leo Sithole	le15
Steve Jacobs	bs12
Pat Khumalo	lo17
Sipho Nkosi	si15
...	...

(5)

1.4 Display the ClientID of clients who has never had an appointment in Bergsig, Panorama or Highlands. A sample of the correct output is shown below.

Note that only the first five records are shown.

ClientID	**Location**
2	Greenside
5	Greenside
24	Greenside
29	Greenside
36	Greenside
...	...

(3)

1.5 New counsellors have a rate of zero while they wait for their rate to be allocated. Display the counsellor names with the lowest non-zero rate. The correct output is shown below.

CounsellorName	**Rate**
Siyanda Mabuza	120
Joshua Hendricks	120
Linda September	120

(5)

1.6 Display the locations where more than 15 future appointments have been made. Display each location and the number of appointments. Do not include past appointments. The correct output is shown below.

Note the output may change depending on today's date or the date setting of your computer.

Location	NumberAppointments
Bergsig	22
Panorama	17

(6)

1.7 All clients younger than 15 receive a discount of 25%. Display the new rate of all appointments involving clients younger than 15. Display the appointment ID, client ID, client name, counsellor name and discounted cost. A sample of the correct output is shown below.

Note that when the list is ordered by ClientID, these five correct records would be shown. Also note that data may be formatted differently on your computer.

AppointmentID	Client ID	ClientName	CounsellorName	Discounted Cost
6	5	Sipho Nkosi	Joshua Hendricks	90
35	14	Mpho Sithole	Thabo Matlala	112.5
69	14	Mpho Sithole	Thabo Matlala	112.5
24	25	Linda Gumede	Matthew Kunene	135
44	25	Linda Gumede	Matthew Kunene	135
...	...	...	...	...

(5)

1.8 The Bergsig office closed for renovations on 24 October 2020. All appointments at that location will need to be changed. They will be moved to the Middelburg office. Write an SQL statement to do this. (4)

1.9 Insert a new follow-up appointment for ClientID 27 for 20 October 2020. Copy other information (CounsellorID and Location) from the client's latest appointment (this is the one with AppointmentID 18). (6)

40 marks

SECTION B OBJECT-ORIENTED PROGRAMMING

A community-based counselling service called HOPE offers clients the ability to book a timeslot to see their counsellor. Each timeslot is one hour long. The clients' details are stored in a text file called **clients.txt**.

A booking schedule needs to be created for five of the counsellors on a particular day. The names of the counsellors are listed in a text file called **counsellors.txt**. All the clients need an appointment to be booked with their counsellor and at a time later than or equal to their preferred time.

The program requires the following base classes:

Client
A client can book a time slot with any counsellor and has the following information:
- **clientName** – the name of the client
- **preferredCounsellor** – the name of the counsellor that the client would prefer to see.
- **earliestHour** – the earliest time slot the client would wish to book. 8 (representing 8:00) is the earliest while 16 (representing 16:00) is the latest slot. NOTE: 24-hour notation is used with values from 0 to 23.

TimeSlot
A class must be created to represent each time slot with a counsellor. Each time slot is one hour long. A **TimeSlot** has the following information:
- **counsellor** – the name of the counsellor for the time slot
- **startHour** – the starting hour of the time slot in 24-hour clock format. For example, a TimeSlot starting at 9, will have this field set to 9, while a timeslot starting at 14:00, will have this field set to 14.
- **isAvailable** – whether his time slot is available for booking.

The text file called **clients.txt** stores the details of the time slots required by clients. The client's name, counsellor and the earliest time slot are listed each on a separate row. A sample of the first five preferred bookings are listed below:

```
John Dlamini
Vernon Booysen
10
Leo Sithole
Matthew Kunene
13
Steve Jacobs
Vernon Booysen
8
Pat Khumalo
Heather Modise
6
Sipho Nkosi
Vernon Booysen
18
```

QUESTION 2

Use the class diagram below to create a new class called **Client**. This class will be used to store the details of a client. The diagram below indicates the fields and methods that are required.

Client
- clientName : string - preferredCounsellor : string - earliestHour : integer
+ Constructor(inCN : string, inPCS : string, inEH : integer) + getClientName() : string + getPreferredCounsellor() : string + getEarliestHour() : integer + toString() : string

2.1 Create a new class named **Client** with **clientName**, **preferredCounsellor** and **earliestHour** fields as indicated above. (4)

2.2 Create a constructor method that accepts a string ***inCN*** as a parameter representing the **clientName** field, a string ***inPCS*** as a parameter representing the **preferredCounsellor** field and an integer ***inEH*** representing the **earliestHour**. Use these parameters to assign values to the fields. Note that for a value in ***inEH*** of greater than 16, **earliestHour** should be set to 16 to ensure the slots fall within the available time. There is no problem with numbers less than 8 since the earliest available slot is 8. (4)

2.3 Create accessor/get methods for the **clientName**, **preferredCounsellor** and **earliestHour** fields. (2)

2.4 Code a **toString()** method to return the fields of a **Client** as a string in the following format:

```
<clientName><tab><preferredCounsellor><tab>Earliest <earliestHour>:00
```

For example:

```
John Dlamini    Vernon Booysen     Earliest 10:00
```

(4)

[14]

QUESTION 3

Create a **TimeSlot** class represented by the following class diagram.

TimeSlot
- counsellor : string - startHour : integer - isAvailable : boolean
+ Constructor(inCS : string, inSH : integer) + setIsAvailable(inIA : boolean) + getCounsellorName() : string + getStartHour() : integer + getIsAvailable() : boolean - getEndHour() : integer + toString() : string

3.1 Create a new class named **TimeSlot** with **counsellor**, **startHour** and **isAvailable** fields as indicated above. (3)

3.2 Create a constructor method that accepts a string ***inCS*** as a parameter representing the **counsellor** field and an integer ***inSH*** representing the **startHour** field. Use these parameters to assign values to the fields. The field **isAvailable** should be set to *true* by default. (4)

3.3 Create a mutator/set method for the **isAvailable** field. (2)

3.4 Create accessor/get methods for the **counsellorName, startHour** and **isAvailable** fields. (1)

3.5 Code a private method called **getEndHour** that returns the hour on which the **TimeSlot** ends. Remember that each **TimeSlot** is one hour long. For example, a time slot starting at 13 will end at 14. (2)

3.6 Code a ***toString()*** method to return information about a **TimeSlot** in the format:

```
<Counsellor Name>: <StartHour>:00 - <EndHour>:00
```

For example:

```
John Dlamini: 10:00 – 11:00
```

Note that the method created in 3.5 must be used in order to get full marks for this question. (4)

[16]

QUESTION 4

A new class **SlotManager** needs to be created.

4.1 Create a new class named **SlotManager** with two private fields as follows:

- An array of **Client** objects called **cArr** with enough capacity to store 20 **Client** objects
- An array of **TimeSlot** objects called **tArr** with enough capacity to store 40 **TimeSlot** objects (4)

4.2 Write code for the constructor method of the **SlotManager** class that takes in no parameter. The constructor method will read the contents of a text file called **clients.txt**. Three lines are used to store information about each client – client name, preferred counsellor and preferred time slot in that order. There are exactly 60 lines in the file.

Do the following:

- Open the file for reading.
- Loop through the file to read 60 lines. For each three lines read from the text file, create a **Client** object, and add the object to the array called **cArr**. (8)

4.3 Create a typed method/function named **displayAllClients**() in the **SlotManager** class that will return a string containing all the **Client** objects in the **Client** array called **cArr**. The **Client** objects should be included in the format of the toString method of the **Client** class with each client's information on a separate line. (4)

[16]

QUESTION 5

5.1 Write code to create a text-based user interface called **CounsellingUI** that will allow simple output. (1)

5.2 Create a **SlotManager** object using the appropriate method. (1)

5.3 Write code to display all the clients' information.

A sample of the output is shown below:

```
John Dlamini      Vernon Booysen      Earliest 10:00
Leo Sithole       Matthew Kunene      Earliest 13:00
Steve Jacobs      Vernon Booysen      Earliest 8:00
Pat Khumalo       Heather Modise      Earliest 6:00
Sipho Nkosi       Vernon Booysen      Earliest 16:00
```

(1)

[3]

QUESTION 6

A booking list needs to be generated for some of the counsellors listed in a text file called **counsellors.txt**. Each counsellor needs a list of hours starting from 8:00 and ending at 16:00 with the exception of 12:00 which is reserved for a lunch break.

The counsellors' names are stored on a single line separated by commas as shown below:

```
Vernon Booysen,Matthew Kunene,Heather Modise,Siyanda Mabuza,Joshua Hendricks
```

6.1 Create a void method/procedure named **generateTimeSlots()** in the **SlotManager** class that reads from the text file **counsellors.txt**.

Write code to read one line of the text file, isolate each counsellor's name and do the following:

- Each counsellor will need eight time slots created with his/her name as the counsellor. The first **TimeSlot** for each consultant will start at 08:00 and the last **TimeSlot** will start at 16:00. There is a slot every hour except for the timeslot starting at 12:00, as the counsellors break for lunch.
- There will be eight **TimeSlot** objects created for each of the five counsellors resulting in 40 **TimeSlot** objects. Add all 40 **TimeSlot** objects into the **TimeSlot** array called **tArr**. The objects can be added in any order.

Marks are allocated for the efficient use of loops to reduce processing time. (11)

6.2 Create a typed method/function named **displayAvailableTimeSlots(..)** in the **SlotManager** class that will return a String containing all the available time slots for each of the five counsellors in the **TimeSlot** array called **tArr**. The **TimeSlot** objects should be included in the format of the toString method of the **TimeSlot** class with each timeslot's information on a separate line. (4)

6.3 In the **CounsellingUI** class, add code to:

- Call **generateTimeSlots()** in the **SlotManager** class created in Question 6.1.
- Call **displayAvailableTimeSlots()** in the **SlotManager** class in Question 6.2.

Possible output is shown below. Note that the list can be in any order:

```
Vernon Booysen: 8:00 - 9:00
Vernon Booysen: 9:00 - 10:00
Vernon Booysen: 10:00 - 11:00
Vernon Booysen: 11:00 - 12:00
Vernon Booysen: 13:00 - 14:00
Vernon Booysen: 14:00 - 15:00
Vernon Booysen: 15:00 - 16:00
Vernon Booysen: 16:00 - 17:00
Matthew Kunene: 8:00 - 9:00
Matthew Kunene: 9:00 - 10:00
Matthew Kunene: 10:00 - 11:00
Matthew Kunene: 11:00 - 12:00
Matthew Kunene: 13:00 - 14:00
Matthew Kunene: 14:00 - 15:00
Matthew Kunene: 15:00 - 16:00
Matthew Kunene: 16:00 - 17:00
Heather Modise: 8:00 - 9:00
Heather Modise: 9:00 - 10:00
Heather Modise: 10:00 - 11:00
Heather Modise: 11:00 - 12:00
Heather Modise: 13:00 - 14:00
Heather Modise: 14:00 - 15:00
Heather Modise: 15:00 - 16:00
Heather Modise: 16:00 - 17:00
Siyanda Mabuza: 8:00 - 9:00
Siyanda Mabuza: 9:00 - 10:00
Siyanda Mabuza: 10:00 - 11:00
Siyanda Mabuza: 11:00 - 12:00
Siyanda Mabuza: 13:00 - 14:00
Siyanda Mabuza: 14:00 - 15:00
Siyanda Mabuza: 15:00 - 16:00
Siyanda Mabuza: 16:00 - 17:00
Joshua Hendricks: 8:00 - 9:00
Joshua Hendricks: 9:00 - 10:00
Joshua Hendricks: 10:00 - 11:00
Joshua Hendricks: 11:00 - 12:00
Joshua Hendricks: 13:00 - 14:00
Joshua Hendricks: 14:00 - 15:00
Joshua Hendricks: 15:00 - 16:00
Joshua Hendricks: 16:00 - 17:00
```

(2)
[17]

QUESTION 7

7.1 In the **SlotManager** class, code a method named **createBookedSlotArray()**. The method will select a time slot for each client with the **Client**'s preferred counsellor (as indicated by the **Client**'s **preferredCounsellorName** field) and the earliest hour the client is able to attend (as indicated by the **Client**'s **earliestHour** field).

Note the following:

- Only one client can be booked in a counsellor's time slot.
- There are enough time slots in the **TimeSlot** array to be allocated according to all the **Client'**s preferences.
- If the client's preferred time is before 8:00 then the client must be awarded a time slot starting from 8:00.

The method should return a String with the following format

```
Appointments:
<clientName1> (<earliestHour>) to see <counsellorName>: <SH>:00 - <EH>:00
<clientName2> (<earliestHour>) to see <counsellorName>: <SH>:00 - <EH>:00
<clientName3> (<earliestHour>) to see <counsellorName>: <SH>:00 - <EH>:00
…
```

Where `<SH>` and `<EH>` are `<startHour>` and `<endHour>` respectively.

Below is sample output with only the first five clients' allocations shown.

```
Appointments:
John Dlamini (10) to see Vernon Booysen: 10:00 - 11:00
Leo Sithole (13) to see Matthew Kunene: 13:00 - 14:00
Steve Jacobs (8) to see Vernon Booysen: 8:00 - 9:00
Pat Khumalo (6) to see Heather Modise: 8:00 - 9:00
Sipho Nkosi (16) to see Vernon Booysen: 16:00 - 17:00
...
```

Note that your output may differ in order and time allocated to a client as long as a client is awarded the preferred counsellor and has a time slot that is not earlier than the **earliestHour** field of that **Client**. (13)

7.2 Add code to **CounsellingUI** to call and display the appointments using the **createBookedSlotArray()** method you created in Question 7.1. (1)

[14]

80 marks

Total: 120 marks