NATIONAL SENIOR CERTIFICATE EXAMINATION
NOVEMBER 2022

INFORMATION TECHNOLOGY: PAPER I

Time: 3 hours 150 marks

PLEASE READ THE FOLLOWING INSTRUCTIONS CAREFULLY

1. This question paper consists of 16 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. When accessing files from within your code, DO NOT use full path names for the files, as this will create problems when the program is marked on a computer other than the one you are writing on. Merely refer to the files using their names and extensions, where necessary.

9. 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.

10. 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.

11. 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.

12. 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.

13. 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.

14. 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.

15. 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.

16. 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.

17. 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:
    XSpaceSystems.mdb
    XSpaceSystems_JavaDB.sql
    XSpaceSystems_MySQL.sql
    SQLAnswerSheet.rtf
    SQLBrowser.exe

    Section B:
    crewmembers.txt
    testResults.txt

SCENARIO

XSpaceSystem is a private space exploration company that aims to send manned missions to explore the neighbouring solar systems. They have employed many astronauts who will captain these missions and have built a variety of spaceships which will be used for the scheduled missions.

SECTION A STRUCTURED QUERY LANGUAGE

QUESTION 1

A database has been created called **XSpaceSystems** that contains the details of captains who will command a spaceship on specific missions to designated solar systems. The database contains the following tables:

tblCaptains contains the details of captains that command spaceships for particular missions.

FIELDS	DATA TYPE	DESCRIPTION
CaptainID	INTEGER	A unique autonumber identification number for each captain.
Fullname	TEXT	The name and surname of the captain.
DateHired	DATE	The date that the captain was hired by XSpaceSystems.
combatTraining	BOOLEAN	A Boolean value that will be true if the captain has done combat training.

The first 10 records of tblCaptains

CaptainID	Fullname	DateHired	CombatTraining
1	Neliswa Tlailane	2006/08/31	Yes
2	Themba Morena	2007/03/28	Yes
3	Mamello Mthandi	2009/03/15	No
4	Lyle Foster	2010/08/20	Yes
5	Bongeka Makhurubetshi	2006/10/20	Yes
6	Oratile Tshabalala	2010/05/17	No
7	Ben Motshwari	2009/02/12	Yes
8	Innocent Maela	2007/09/16	No
9	Thapelo Erasmus	2008/09/22	Yes
10	Gabriela Salgado	2005/02/11	No

tblSpaceShips contains the details of the spaceships that will be used by the captains on missions.

FIELDS	DATA TYPE	DESCRIPTION
SpaceshipID	INTEGER	A unique autonumber identification number for each spaceship.
Model	TEXT	The model of the spaceship.
Speed	DOUBLE	The speed of the spaceship as a multiple of lightspeed.
FuelConsumption	INTEGER	The amount of fuel used to travel 1 light year.
FuelCapacity	INTEGER	The maximum fuel capacity of the ship.

The first 10 records of tblSpaceShips

SpaceshipID	Model	Speed	FuelConsumption	FuelCapacity
1	T941	5.49	656	8659
2	M793	3.03	314	4540
3	M327	3.3	970	3321
4	S766	4.29	811	5070
5	T666	3.97	129	8016
6	S379	3.96	956	1933
7	T766	1.48	894	4565
8	T494	2.87	708	7654
9	M366	2.42	639	2174
10	S336	5.58	274	7064

tblMissions contains the details of the missions that have been assigned to captains together with their spaceships.

FIELDS	DATA TYPE	DESCRIPTION
MissionID	INTEGER	A unique autonumber identification number for each mission.
CaptainID	INTEGER	The **CaptainID** of the captain that has been assigned to the mission. This is a foreign key to the **tblCaptains**.
SpaceshipID	INTEGER	The **SpaceshipID** of the spaceship that has been assigned to the mission for the captain. This is a foreign key to the **tblCaptains**.
MainObjective	TEXT	The main objective of the mission.
DepartureDate	DATE	The scheduled departure date.
Distance	INTEGER	The distance to the destination solar system measured in light years.
SolarSystem	TEXT	The codename of the destination solar system.

The first 10 records of tblMissions

MissionID	CaptainID	SpaceShipID	MainObjective	DepartureDate	Distance	SolarSystem
1	8	28	Military Exercise	2024/04/19	6	X-645G
2	18	4	Mining	2021/11/07	3	F-331B
3	30	18	Military Exercise	2022/07/26	4	U-184A
4	16	8	Military Exercise	2024/04/20	9	F-306E
5	20	16	Mining	2025/05/01	7	F-479C
6	23	27	Exploration	2025/12/10	10	H-178D
7	19	2	Exploration	2025/12/12	2	Q-987C
8	15	3	Exploration	2021/09/20	7	E-615B
9	33	23	Military Exercise	2026/10/01	1	I-179G
10	2	5	Rescue	2024/10/13	13	W-695B

1.1 Display a list showing all the captain's details who have combat training. Order this list starting with the most recent date.

The first five records are shown below:

CaptainID	Fullname	DateHired	CombatTraining
17	Kermit Phete	2011/03/04	Yes
18	Sipho Hlanti	2010/12/24	Yes
42	Xiluva Jordaan	2010/09/08	Yes
4	Lyle Foster	2010/08/20	Yes
38	Karabo Dlamini	2009/08/16	Yes

(4)

1.2 Display all the ship's details whose model names start with the letter 'M' or 'T' and have a fuel consumption of less than 500.

The correct output is shown below:

SpaceshipID	Model	Speed	FuelConsumption	FuelCapacity
2	M793	3.03	314	4540
5	T666	3.97	129	8016
13	T288	2.51	466	6922
16	M256	2.87	446	5131
19	T204	2.81	300	2421
20	M763	1.71	362	6337
21	T367	2.33	457	6732
22	T257	2.05	239	6948
23	T313	1.32	282	4188
25	T382	1.56	226	7807

(4)

1.3 The Company would like to award long service awards to their employees. Display a list of all captains who will be with the company for 10 years or more this year.

The first ten records are shown below:

CaptainID	Fullname	DateHired	combatTraining
1	Neliswa Tlailane	2006-08-31	Yes
2	Themba Morena	2007-03-28	Yes
3	Mamello Mthandi	2009-03-15	No
4	Lyle Foster	2010-08-20	Yes
5	Bongeka Makhurubetshi	2006-10-20	Yes
6	Oratile Tshabalala	2010-05-17	No
7	Ben Motshwari	2009-02-12	Yes
8	Innocent Maela	2007-09-16	No
9	Thapelo Erasmus	2008-09-22	Yes
10	Gabriela Salgado	2005-02-11	No

(5)

1.4 Display the model and speed of all spaceships that have an above average speed. Display the list in descending order of speed.

The correct output is shown below:

Model	Speed
S336	5.58
T941	5.49
M313	5.34
S390	5.23
S897	5.17
T419	4.6
M845	4.29
S766	4.29
S174	4.21
S178	4.16
T666	3.97
S379	3.96
M641	3.74
M327	3.3

(5)

1.5 Display a list showing the number of missions that have been scheduled in the first 6 months of each year. The list should show the year of the mission called '**YearOfMission**' and a count called '**NumberOfMissions**'.

The correct output is shown below:

YearOfMission	NumberOfMissions
2021	2
2022	3
2023	2
2024	3
2025	1

(7)

1.6 Display a list showing the **MissionID**, the full name of the captain, the **Model** of the spaceship, the speed of the spaceship, the destination solar system, the distance and a calculated field that will determine the time in years it would take to reach the destination. Round this number to two decimal places and name the field **TimeInYears**. The time can be calculated as follows:

Time in years = Distance to solar system/speed of the spaceship

The first five records are shown below:

MissionID	FullName	Model	Speed	SolarSystem	Distance	TimeInYears
1	Innocent Maela	S897	5.17	X-645G	6	1.16
2	Sipho Hlanti	S766	4.29	F-331B	3	0.7
3	Nomvula Moodaly	S174	4.21	U-184A	4	0.95
4	Andile Hlatshwayo	T494	2.87	F-306E	9	3.14
5	Ronwen Williams	M256	2.87	F-479C	7	2.44

(9)

1.7 The company would like to change some of the main objectives of some of the missions. Identify all the missions that have not yet departed, are Military Exercises and whose solar system codename does not end with an A, B or C. Change the main objective of these missions to 'Scouting'.

Below are the new details of the records that were changed:

MissionID	CaptainID	SpaceShipID	MainObjective	DepartureDate	Distance	SolarSystem
1	8	28	Scouting	2024-04-19	6	X-645G
4	16	8	Scouting	2024-04-20	9	F-306E
9	33	23	Scouting	2026-10-01	1	I-179G
17	9	1	Scouting	2025-10-24	3	V-594E
24	42	15	Scouting	2023-11-17	14	T-978G

(7)

1.8 Some of the captains have no assigned missions. Write a query that will add missions for these captains. Assign them a random ship number (1–30 inclusive), a main objective of 'Exploration', a departure date for 01/01/2030, the solar system 'Alpha-Centauri' with a distance of 5 light years.

Below are the details of the newly added records. Note that your ship number will be different depending on the random number generated.

MissionID	CaptainID	SpaceShipID	MainObjective	DepartureDate	Distance	SolarSystem
31	1	22	Exploration	2030-01-01	5	Alpha-Centauri
32	3	17	Exploration	2030-01-01	5	Alpha-Centauri
33	6	18	Exploration	2030-01-01	5	Alpha-Centauri
34	10	9	Exploration	2030-01-01	5	Alpha-Centauri
35	11	10	Exploration	2030-01-01	5	Alpha-Centauri
36	12	25	Exploration	2030-01-01	5	Alpha-Centauri
37	17	1	Exploration	2030-01-01	5	Alpha-Centauri
38	21	24	Exploration	2030-01-01	5	Alpha-Centauri
39	22	26	Exploration	2030-01-01	5	Alpha-Centauri
40	24	22	Exploration	2030-01-01	5	Alpha-Centauri
41	25	2	Exploration	2030-01-01	5	Alpha-Centauri
42	27	13	Exploration	2030-01-01	5	Alpha-Centauri
43	28	27	Exploration	2030-01-01	5	Alpha-Centauri
44	31	25	Exploration	2030-01-01	5	Alpha-Centauri
45	36	12	Exploration	2030-01-01	5	Alpha-Centauri
46	38	30	Exploration	2030-01-01	5	Alpha-Centauri

(9)

50 marks

SECTION B OBJECT-ORIENTATED PROGRAMMING

XSpaceSystems has stored the details of all the crew and officers in a text file called **crewmembers.txt**. Recently, a training course was held for some of the crew and officers who wish to be promoted. The course was followed by a test and the results of the test have been captured and stored in a text file called **testResults.txt**. Based on their test results, the following criteria apply for promotion.

- A crew member must achieve at least 75% to be promoted to an officer with the rank of 1 and title of 'Ensign'.

- Officers must achieve a minimum score according to their current rank and title. If the officer achieves the minimum score, their rank will increase by one. A captain (rank 5) cannot increase their rank even if they have passed the test.

Below is a table showing the minimum scores an officer would need to be promoted to their next rank:

Score (inclusive)	Title	Rank
75–79	Ensign	1
80–84	Lieutenant	2
85–89	*Lt Commander	3
90–100	Commander	4
	Captain	5

** Lt is the abbreviation for Lieutenant*

If the officer has the rank of 5 and is the captain, they can write the test but cannot be further promoted regardless of their test scores.

The details of the crew members and officers have been stored in a text file called **crewmembers.txt**.

Below are the first 10 lines of the text file:

```
Tayla Wentzel#892#Engineering#2#04/07/2020
Kwagga Kolisi#721#Medical
Nomathamsanqa Geldenhuys#371#Security
Malcolm Malherbe#279#Security
Sibusiso Louw#277#Medical#1#17/07/2020
Francois de Jager#317#Medical
Bongi le Roux#461#Security#3#29/12/2020
Jesse Nyakane#397#Engineering#3#22/09/2019
Lamla Weston#270#Flight
Siviwe Tywaleni#131#Medical#5#20/10/2018
```

Each line in the text file represents either a crew member or an officer.

CrewMember
The details for a crewmember are stored in the text file as follows:

< fullname >#<crewID>#<department>

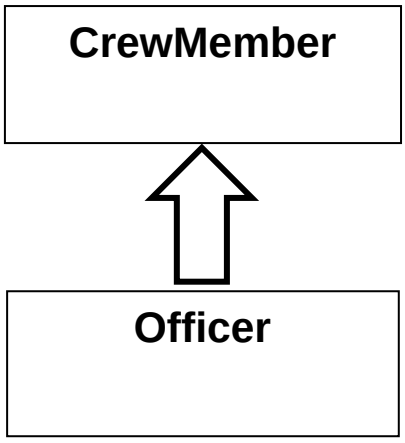
- **Fullname**: a string representing the name and the surname of the crew member.
- **Crew ID**: a unique integer value that identifies each crew member.
- **Department**: a string representing the crew member's department.

Officer
The details for an officer are stored in the text file as follows:

<fullname>#<crewID>#<department>#<rank>#<date promoted>

- **Fullname**: a string representing the name and the surname of the crew member/officer.
- **Crew ID**: a unique integer that identifies each crew member/officer.
- **Department**: a string representing the crew member's department.
- **Rank**: an integer representing the rank of the officer.
- **Date promoted**: a date value in the format dd/MM/yyyy representing the date that the officer was promoted to their current title and rank.

You will need to create two classes with the parent **CrewMember** class and the child **Officer** class as shown in the diagram below:



QUESTION 2

Use the class diagram below to create a class called **CrewMember**. This class will be used to create objects to store the details of a crew member. The diagram indicates the required fields and methods.

CrewMember
Fields: – fullname : string – crewID : integer – department : string + numPromotedCrew : integer = 0
Methods: + Constructor(inFN : string, inCID : integer, inDT : string) + getFullName() : string + getCrewID() : integer + getDepartment() : string + toString() : string

2.1 Create a new class called **CrewMember** with **fullname**, **crewID** and **department** fields as indicated above. (4)

2.2 Add the static/class field named **numPromotedCrew** to the class as shown in the class diagram. (2)

2.3 Code a **constructor** method for the class that accepts a string **inFN** representing the **fullname** field, an integer **inCID** as a parameter representing the **crewID** field and a string **inDT** that will represent the **department** field. Use these parameters to assign the values for the fields of the class. (3)

2.4 Create **assessor/getter** methods for the **fullname**, **crewID** and **department** fields of the class. (3)

2.5 Add a **toString** method to the class that will return a string containing all the information of the class in the following format:

fullname<tab>'Crew ID:'<space>crewID<tab>'['department']'
For example,
Lamla Weston Crew ID: 270 [Flight] (5)

*Note. The **numPromotedCrew** field will be updated in Question 6.*

[17]

QUESTION 3

Use the class diagram below to create a class called **Officer**. This class will inherit from the **CrewMember** class and will be used to create objects that will store the details of an officer. The diagram below indicates required fields and methods.

Officer
Fields: – rank : integer – datePromoted : Date + numPromotedOfficers: integer = 0
Methods: + Constructor(inFN : string, inCID : integer, inDT : string, inRK : integer, inDP : Date) + getRank () : integer + getTitle() : string + toString() : string + promote ()

3.1 Write code to create a new class called **Officer** that extends the **CrewMember** class. (2)

3.2 Add the **rank** field and **datePromoted** (in the format dd/MM/yyyy) field as indicated in the class diagram. Note that the **datePromoted** field must make use of an appropriate object type to store date information. (2)

3.3 Add a static/class field called **numPromotedOfficers** as shown in the class diagram. (2)

3.4 Code a **constructor** method that will initialise the fields **fullname**, **crewID** and **department** of the **CrewMember** parent class and the fields **rank** and **datePromoted** of the **Officer** child class. (4)

3.5 Code an **accessor**/**getter** method for the **rank** property. (1)

3.6 Code a method named **getTitle** that will return a string containing the title of the officer based on their rank using the table below:

Title	**Rank**
Ensign	1
Lieutenant	2
Lt Commander	3
Commander	4
Captain	5

(5)

3.7 Code a **toString** method that will override the parent class's **toString** method and combine the title (using the **getTitle** method in Question 3.6) to the string returned by the **toString** method of the **CrewMember** class in the following format:

fullname<tab>'Crew ID:'<space>crewID<tab>'['department']'<space>title
For example,
Tayla Wentzel CrewID: 892 [Engineering] Lieutenant (4)

3.8 Code a method named **promote** that will increase the rank of the officer by one if they have a rank of four or below. The method should update their date promoted to the current date and increase the number of promoted officers by one.

(5)
[25]

QUESTION 4

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

4.2 Add two instance variables to this class:
- An array called **cArr** capable of storing 80 **CrewMember** or **Officer** objects. Name this array **cArr**.
- An integer called **size** to keep track of the number of objects added to the array called **cArr**. These two variables must not be accessible outside the class. (4)

4.3 Create a **constructor** for the class that will read the contents of the text file **crewmembers**.**txt**. Each line of the file contains information for a single **CrewMember** or an **Officer** object. Your method should do the following:
- Read each line from the file and extract the data.
- Instantiate the appropriate type of object (**CrewMember** or an **Officer**) using the data extracted from the line.
- Add the **CrewMember** or **Officer** object to the next available position in the array.
- Increase the **size** appropriately. (11)

4.4 Code a **toString** method to combine the values of each object in the array **cArr** into a string. Each crew member's or officer's information should appear on a separate line. Use the object's **toString** methods that you created in Questions 2.5 and 3.7. Sample output is shown in Question 5.3. (5)
[21]

QUESTION 5

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

5.2 Create a **CrewMemberManager** object in an appropriate place in the class using an appropriate method. (2)

5.3 Display a list of all the information about the crew members or officers using the appropriate method in the **CrewMemberManager** class.

The first six and last six crew members in the order they appear in the text file.

```
Tayla Wentzel  Crew ID: 892   [Engineering] Lieutenant
Kwagga Kolisi  Crew ID: 721   [Medical]
Nomathamsanqa Geldenhuys Crew ID: 371   [Security]
Malcolm Malherbe    Crew ID: 279   [Security]
Sibusiso Louw  Crew ID: 277   [Medical] Ensign
Francois de Jager   Crew ID: 317   [Medical]
     …

Denita Macingwana   Crew ID: 593   [Flight] Lt Commander
Nolusindiso Cant    Crew ID: 482   [Science] Ensign
Siya Brits     Crew ID: 758   [Security]
Zenay Ngxatu   Crew ID: 926   [Command]
Vuyolwethu Adonis   Crew ID: 361   [Science] Captain
Duane Gelant    Crew ID: 970    [Command]
```

(2)
[5]

QUESTION 6

The results of the tests are stored in the text file named **testResults.txt**. The file contains the crew IDs and test scores of the crew members and officers stored on separate lines. The test centre can only assess 10 crew members and/or officers producing a text file with exactly 20 lines. Each test result is stored on two separate lines: the first for the crew ID and the second for the test score out of 100.

Below are the first 6 lines from the text file:

```
892
90
461
95
990
89
```

Add the following two methods to the **CrewMemberManager** class:

6.1 Code a method called **findCrewMember** to search for a crew member or officer object in the array called **cArr** who has the matching **crewID**. The method must accept an integer parameter for the **crewID** and return the integer position of the object in the array. (4)

6.2 Code a method named **processTestResults** to read all the lines from the text file called **testResults.txt**. Using an appropriate loop read the **crewID** and test score from the text file. (5)

In this method add code to:

6.2.1 Determine the position of the crew member or officer in the array called **cArr** using the method you coded in Question 6.1. (1)

6.2.2 Using the above position, check if the related person is:

(a) an officer, and has met their test requirements (indicated in the column **Score**) according to their rank, then their rank value is increased by 1.

Score (inclusive)	Title	Rank
75–79	Ensign	1
80–84	Lieutenant	2
85–89	*Lt Commander	3
90–100	Commander	4
	Captain	5

Use the method called **promote** that you coded in Question 3.8 to update the office object by increasing the rank of the officer, incrementing the **numPromotedOfficers** field and setting the date **datePromoted** field to the current date. (7)

(b) a crew member, and has met the test requirements. The crew member must be promoted to an officer with a rank of 1 and today's date. The **CrewMember** object in the array must be changed to an **Officer** object with crew member's details, a rank of 1 and today's date. Increment the static/class **numPromotedCrew** field you created in Question 2.2. (6)

6.2.3 The method should return a string containing a list of the names of all the crew members or officers who took the test.

- Add the label '**Crewmember**' or '**Officer**' depending on if the person is a crewmember or an officer.
- If the person passed, include the word '**has passed and is promoted to**' after their name, their new title and rank.
- If the person failed, include the message '**has failed**' after their name.
- Lastly, if the person is a captain, display their name and the message '**participated in the test**'.

The correct output is shown in Question 6.3. (6)

6.3 In the **CrewMemberUI** class call the **processTestResults** method and display the list showing the names of all the crew members or officers who took the test and whether they passed or failed.

```
Officer Tayla Wentzel has passed and is promoted to Lt
Commander Rank 3
Officer Bongi le Roux has passed and is promoted to
Commander Rank 4
Crew Member Cindy Ntoyanto has passed and is promoted to
Ensign Rank 1
Officer Cobus Koch has failed
Crew Member Eben Mapimpi has failed
Crew Member Pieter-Steph Pollard has failed
Crew Member Vincent Reinach has passed and is promoted to
Ensign Rank 1
Officer Celeste Faleni participated in the test
Officer Steven Jantjies has passed and is promoted to
Captain Rank 5
Officer Thantaswa Nobele has passed and is promoted to
Captain Rank 5
```
(1)

6.4 In the **CrewMemberUI** class display the number of crew members who were promoted and the number of officers who were promoted. Use the static/class fields you created in Questions 2.2 and 3.3.
Your output should appear as:

```
Number of promoted crew members: 2
Number of promoted officers: 4
```
(2)

[32]

100 marks

Total: 150 marks