



PECANWOOD
COLLEGE

Prepared for Life

INFORMATION TECHNOLOGY NOVEMBER PRACTICAL EXAMINATION
GRADE 11

NAME: _____

GRADE: _____

118 3 bonus marks section 1

DATE: 8 NOVEMBER 2022

MODERATOR: MR C SEEWALD

EXAMINER: MR SC EILERTSEN

MARKS: 115

TIME: 180 MINUTES

INSTRUCTIONS:

1. This examination paper is made up of 9 pages which includes a blank page at the back for notes you may want to make during the examination.
2. Note the mark allocation for each question.
3. The screen shots and output examples in each question are part of the question and should be followed.
4. You will be provided with three CSV files for question two from which you will build a database. The files are tblBird, tblFood and tblBirdFood.
5. You will be provided with two text files "thefile.txt" and "newFile2.txt" for questions three and five.
6. You will be provided with a single text file "nationality.txt" for question 4.
7. Marks are not awarded for code that has been recognised earlier in the marking memorandum.
8. Marks are awarded for indentation, variable names, class name, good use of white space, comments, and good readable layout.
9. Save your work on a regular basis.
10. Question one. This GUI JFrameForm assignment has already been handed in.
11. For question three you may use the two pages of notes that you have prepared from Learning Unit 6 in your textbook (pages 98 to 107 only). Only notes pertaining to the chapter "Date and Time" may be written on these notes. Your notes must be handed in for moderation the day before the exam.
Only notes signed, and with the school stamp are approved for use with this examination.
12. You may use the help files and language prompts that are part of the IDE you are using (NetBeans for example) but you may not use any other help (you are not allowed to access the internet)

Section One

GUI using JFrameForm

Create a game of chance much like a One-Armed Bandit.

There must be a GUI welcome screen

There must be the game interface screen with labels, fields and buttons

The player must hit the play button and win or loose.

Your program must work the following way

1. Your welcome screen will have the main method
2. All other methods must be in a separate method class
3. When the player hits the play button, the button must call the methods it needs
4. Your program must read in data from a text file into an array
5. Your program must generate random numbers to determine the winner
6. There must be different levels of rewards for winners
7. Your program must have at least three different GUI interfaces
8. At least one of your GUIs must use an image (or images) as part of the creative design

Here is the rubric

- The overall fun aspect of playing the game - the level of creativity, originality etc (3)
- The creative design - background images, fonts, colours, layout etc (3)
- The flow from one screen to the next (3)
- The code in the UI class (3)
- The code in the manager class - comments, layout, variable names, whitespace etc (5)
- The level of sophistication found in the code, with regard to the game itself, random number selection etc (3)
- The code that reads in the text file (3)
- Bonus marks will be awarded if the program writes the winners details to a new text file (3)

Total: 23 marks (plus a possible 3 bonus marks)

26

C8

Section Two

Databases Design and SQL

Using SQLite studio to build a SQLite database with three separate tables.







The tables are as follows

1. tblBirds – for this table import the CSV file “tblBird”
2. tblFood – for this table import the CSV file “tblFood”
3. tblBirdFood – for this table import the CSV “fileBirdFood”

The bird table has a lot of data about the various bird. Each bird has a **primary key**.

The food table identifies 25 different types of foods that birds eat. Each food has a **primary key**

The food bird table is the clever part. It does not have its own primary key but instead has **two foreign keys** – the combination of these two foreign keys makes what is called a “**composite primary key**”. This enables one bird type to eat more than one food type. This is called a **many-to-many relationship**. See below.

What the table looks like in data sheet view	Explanation of the composite primary key																																										
<div><div>StructureDataCons</div><div>Grid viewForm view</div><div></div></div> <table><thead><tr><th></th><th>birdID</th><th>foodID</th></tr></thead><tbody><tr><td>1</td><td>1</td><td>5</td></tr><tr><td>2</td><td>1</td><td>4</td></tr><tr><td>3</td><td>1</td><td>19</td></tr><tr><td>4</td><td>1</td><td>12</td></tr><tr><td>5</td><td>2</td><td>1</td></tr><tr><td>6</td><td>2</td><td>20</td></tr><tr><td>7</td><td>2</td><td>8</td></tr><tr><td>8</td><td>2</td><td>21</td></tr><tr><td>9</td><td>3</td><td>5</td></tr><tr><td>10</td><td>3</td><td>8</td></tr><tr><td>11</td><td>4</td><td>5</td></tr><tr><td>12</td><td>4</td><td>16</td></tr><tr><td>13</td><td>4</td><td>18</td></tr></tbody></table>		birdID	foodID	1	1	5	2	1	4	3	1	19	4	1	12	5	2	1	6	2	20	7	2	8	8	2	21	9	3	5	10	3	8	11	4	5	12	4	16	13	4	18	<p>Bird ID one eats four different food types i.e. 5,4,19, and 12</p> <p>Bird ID three eats only two different food types i.e. 5, and 8</p> <p>FoodID number 5 is eaten by three different types of birds i.e. bird 1, 3 and 4</p> <p>FoodID number 8 is eaten by two different types of birds ie bird 2 and 3</p> <p>Therefore,</p> <ul style="list-style-type: none">• each bird can eat many different types of food,• each food can be eaten by many different types of birds <p>(Hence a many-to-many relationship)</p> <p>This “joining” table’s primary key is the combination of two different foreign keys – each complete line (record) is unique.</p> <p>A similar design can now allow one learner to play many sports. Equally one sport can be play be many learners.</p>
	birdID	foodID																																									
1	1	5																																									
2	1	4																																									
3	1	19																																									
4	1	12																																									
5	2	1																																									
6	2	20																																									
7	2	8																																									
8	2	21																																									
9	3	5																																									
10	3	8																																									
11	4	5																																									
12	4	16																																									
13	4	18																																									

2.1) Create a SQLite database called “Birds”

2.2) Import and create the bird table

2.3) Import and create the food table

2.4) Import and create the bird food table

2.5) Set the datatypes for the bird table (including the primary key)

2.6) Set the datatypes for the food table (including the primary key)

2.7) Set the datatypes for the bird food table (including the composite primary key)

(1)

(1)

(1)

(11)

(2)

(4)

[20]

dfu

How to submit your work for marking.

Select the database, right click and export the database. Export (select) everything, the tables and the data
Set the export format type as **SQL** and save it onto your desktop.

At the end of the exam, you can print your database.

Set the paper orientation to **landscape** (not portrait) and set the paper size to **A4**.

Your database export will look like this – this is just a tiny section of the file . . .

```
-- File generated with SQLiteStudio v3.3.3 on Thu Oct 27 17:08:35 2022
--
-- Text encoding used: Shift_JIS
--
PRAGMA foreign_keys = off;
BEGIN TRANSACTION;

-- Table: tblBirds
CREATE TABLE tblBirds (birdID INTEGER PRIMARY KEY AUTOINCREMENT, birdname
INSERT INTO tblBirds (birdID, birdname, height, wingspan, eggs, broods, in
INSERT INTO tblBirds (birdID, birdname, height, wingspan, eggs, broods, in
INSERT INTO tblBirds (birdID, birdname, height, wingspan, eggs, broods, in
INSERT INTO tblBirds (birdID, birdname, height, wingspan, eggs, broods, in
INSERT INTO tblBirds (birdID, birdname, height, wingspan, eggs, broods, in
```

Section Three

Date and time - Reading from a text file – no delimiters using the Scanner class

For this question you may use the two pages of notes that you have prepared from Learning Unit 6 in your textbook (pages 98 to 107 only). Only notes pertained to the chapter “Date and Time” may be written on these notes. Your notes must be handed in for moderation the day before the exam. **Only notes signed, and with the school stamp, are approved for use with this examination.**

Write a **one class program** with static variables and static methods that reads dates in from a text file. This is what the text file looks like.

2019-08-20
2020-10-17
2019-02-12
2021-11-05
2020-01-23
2019-10-07
2019-07-15
2021-12-24
2019-04-28

3.1) Using a static method, read in the dates from a text file into one of two arrays. Note that the two arrays have essentially the same data, except that the first stores the date as a String, while the second stores the date as a LocalDate object

- The first array must be an array of String
- The second array must be an array of LocalDate.

(17)

3.2) Using a static method print out the String array.

(3)

3.3) Using a static method print out the array of LocalDate.

(1)

3.4) The name of the text file must now be captured from the keyboard – do this in the main method.

(2)

3.5) Using a static method determine and print to screen the oldest (earliest) date in the date array.

(5)

The output should look like this . . .

String array 2019-08-20
String array 2020-10-17
String array 2019-02-12
String array 2021-11-05
String array 2020-01-23
String array 2019-10-07
String array 2019-07-15
String array 2021-12-24
String array 2019-04-28

Date array 2019-08-20
Date array 2020-10-17
Date array 2019-02-12
Date array 2021-11-05
Date array 2020-01-23
Date array 2019-10-07
Date array 2019-07-15
Date array 2021-12-24
Date array 2019-04-28

The oldest date is 2019-02-12

[28]

Section Four

Two class program. Reading from a text file – with delimiters.

Here is a text file of South African citizens and Botswana citizens that you have seen before. Now however you are going to use OOP principals and create a two-class program with a UI class and a manager class.

```
FGD-ZA-7674884A#andrew alpha#T#76567.76
6672889-BO#erric echo#654567.71
876553-BO#cindy charlie#465.67
GFD-ZA-9565776C#betty bravo#F#75769.37
3456278-BO#foxy foxtrot#6465476.87
6672889-BO#david dandy#6750.65
9876567-BO#gary golf#65576.67
NBV-ZA-9763537B#inky india#T#7869.76
DDR-ZA-4256297A#jacky jackson#F#86876.01
NBV-ZA-6697899A#henry hotel#T#87879.78
```

The South Africans have a ZA in their passport numbers while those from Botswana have a BO.

Create a two-class program

1. **NationalityUI** - has the main method that has the variables and calls the methods in the manager object in the correct order. (3) ✓
2. **NationalityManager** - has the variables and methods needed. Your class may or may not have a constructor method. (5) ✓

Your manager class will have variables and methods to read in the text file and output the result to the monitor. By using the String method `indexOf("-")` and `substring`, you will be able to copy all the South Africans into a South African array and all the Botswana citizens into a Botswana array.

The methods in the manager class should be as follows . . .

- One method to read in the text file and create one array for South Africans and one for Botswana citizens. Note: You may use the constructor method to do this. (14) ✓
- One method to display the South African citizens. (3) ✓
- One method to display the Botswana citizens. (2) ✓
- One method to return total amount of currency from both South Africans and Botswana citizens together back to the UI class. The **UI class** must then print the total amount to the monitor. (6) ✓
 - Suggestion: You can determine the total amount in the while loop as it reads in the text file line by line.

Oban [33]

Here is the expected output – South African passport holders followed by Botswana passport holders.
Note the capital letters. Your columns do not have to be perfect like those shown below.

```
=====
Country ZA
=====
FGD-ZA-7674884A      ANDREW ALPHA      true      76567.76
GFD-ZA-9565776C      BETTY BRAVO       false     75769.37
NBV-ZA-9763537B      INKY INDIA        true      7869.76
DDR-ZA-4256297A      JACKY JACKSON     false     86876.01
NBV-ZA-6697899A      HENRY HOTEL       true      87879.78
```

```
=====
Country BO
=====
6672889-BO ERRIC ECHO      654567.71
876553-BO CINDY CHARLIE    465.67
3456278-BO FOXY FOXTROT    6465476.87
6672889-BO DAVID DANDY     6750.65
9876567-BO GARY GOLF       65576.67
```

```
=====
Total Amount - ZA and BO - R7527800.25
```

Section Five

Extension of question 3

Return to your coded solution for question 3 and add a **new method** at the bottom of the class.


4.1) This new method will read in a new text file called “newFile2.txt” shown below.

```
2019-8-20
2020-10-17
2019-2-12
2021-11-5
2020-1-23
2019-10-7
2019-7-15
2021-12-24
2019-4-28
```

Although similar to “theFile.txt”, this file stores the dates without leading zeros (eg 2019-8-20 instead of 2019-08-20). Therefore, the length of the dates changes from line to line. Your new method must read in the dates in the same way as question 3.1. – these dates from this new text file must be read into a **new LocalDate** array. (9)

4.2) Add one more method to print out this new date array (2)

Study the final output below for more detail . . .

<pre>String array 2019-08-20 String array 2020-10-17 String array 2019-02-12 String array 2021-11-05 String array 2020-01-23 String array 2019-10-07 String array 2019-07-15 String array 2021-12-24 String array 2019-04-28 Date array 1: 2019-08-20 Date array 1: 2020-10-17 Date array 1: 2019-02-12 Date array 1: 2021-11-05 Date array 1: 2020-01-23 Date array 1: 2019-10-07 Date array 1: 2019-07-15 Date array 1: 2021-12-24 Date array 1: 2019-04-28 The oldest date is 2019-02-12 Date array 2: 2019-08-20 Date array 2: 2020-10-17 Date array 2: 2019-02-12 Date array 2: 2021-11-05 Date array 2: 2020-01-23 Date array 2: 2019-10-07 Date array 2: 2019-07-15 Date array 2: 2021-12-24 Date array 2: 2019-04-28</pre>	<p>NOTE: The second date array has had the leading zeros re-inserted by the date class. This is correct – the LocalDate class will do that.</p> <p>[11]</p> 
---	--

TOTAL: 115 marks

Additional Paper for notes you may want to make as you code.



PECANWOOD
COLLEGE
Prepared for Life

Information Technology: Moderation Sheet

Question Paper

Name of exam: IT Practice Grade: 11 Date: 8 Nov 2022

Branding and layout	Satisfactory	Not satisfactory
Question numbering	Satisfactory	Not satisfactory
Page numbering	Satisfactory	Not satisfactory
Mark allocation	Satisfactory	Not satisfactory
Variety of question styles	Satisfactory	Not satisfactory
Enrichment source material	Satisfactory	Not satisfactory
Analysis grid	Satisfactory	Not satisfactory
Detailed memo with mark allocation	Satisfactory	Not satisfactory

Comments: _____

C. Sewald Handels 31/10

Name of moderator:

Signature:

Date:

