UNIVERSITY OF JOHANNESBURG

IFM01B1 | IFM1B10 INFORMATICS 1B

Practical Assignment P2022-09
Due: 30 Sep 2022 09:00 SAST

Competencies Checklist (Your Practical Should Demonstrate the Following):

□ Downcasting □ Sequential Files

INSTRUCTIONS

- **ALL** classes must begin with the following comment (completed with relevant information)

```
‘ ****************************************************************************
‘  Surname, Initials:
‘  Student Number:
‘  Practical: P2017-09
‘  Class/Interface name: (this is the name of the class/interface you are working on)
‘ ****************************************************************************
```

- All practical submissions MUST be accompanied by a **handwritten design** that must be reviewed by your assistant at the end of the session today (29 September 2017)

THE PROBLEM

It’s almost holiday time and instead of knocking back drinks on a beach, you have decided to go do some volunteer work at the various national parks in South Africa. With your amazing IT skills, you have been tasked with designing and developing a Visual Basic application that will help National Parks Management with documenting the various wildlife they have on their parks.

OBJECT ORIENTED PROGRAMMING – best programming practices must be implemented

a) Create a class called Wildlife with the following attributes:

Name	Type	Description	Example
_TrackerID	String	Tracker code assigned to wildlife	“LION-0”
_RarityLevel	Integer	Rarity level of wildlife (1-5, 1 for common, 5 for rare)	3
_Sightings	Integer()	Times wildlife spotted each month	5, 2, 9, …, 25
_Months	Integer	Number of months that the wildlife is monitored for	6
_LeastMonth	Integer	Month where wildlife was least spotted	2 *(calculated)*

Only _Sightings has a property method.

In addition, Wildlife also has the following methods:

- A constructor that accepts the parameters _TrackerID, _RarityLevel, and _Months.
- CalcAve – calculates and returns the average of the _Sightings array.
- LowestSightings - a method that calculates the month in which the wildlife was least spotted. This value is stored in _LeastMonth and then returns the actual number of Sight-

ings (not the month number, but the Sightings in that month).

- `CalculateRarityRating` - a method that returns a Rarity Rating (A, B, C, or D) based on the following table:

Rarity Level	Average Sightings <= 15	Average Sightings > 15
1 or 2	"C"	"D"
3	"B"	"C"
4 or 5	"A"	"B"

  The above must be implemented using a `Select Case` and three `If` Statements. `CalculateRarityRating` may differ in derived classes.
- `Display` - provides the Wildlife's Tracker ID, Rarity Level, Average Sightings (display up to 2 decimal places using the `Format` command), the Least Sightings, and the Rarity Rating (data must be presented in this exact order) in a single line as follows:
  `"Tracker ID: Rarity Level / Average Sightings / Least Sightings / Rarity Rating"`

b) In addition, create the following classes which inherit from Wildlife. The following table contains the additional properties and methods for the two derived classes:

- **Lion**

Property	Type	Description	Example
`Pride`	`Integer`	Number of lions that belong to this lion's pride	4
`IsAlpha`	`Boolean`	Indicates whether lion is the alpha male	`True`

  ◊ A constructor accepting parameters `TrackerID`, `RarityLevel`, and `NumMonths` in order to do any necessary initialisations.

  ◊ `CalculateRarityRating` - returns "C"

  ◊ `Display` - provides the Wildlife's Tracker ID, Rarity Level, Average Sightings, the Least Sightings, the Rarity Rating, the Number of Lions in the Lion's Pride, and whether the Lion is the Alpha Male (data must be presented in this exact order) in a single line (see `Wildlife` class for a similar example).

- **Rhino**

Attribute	Type	Description	Example
`_Crash`	`Integer`	Number of rhinos in this rhino's "family"	3

  ◊ A constructor accepting parameters `TrackerID`, `RarityLevel`, and `NumberInCrash` in order to do any necessary initialisations. Rhinos are always monitored for 2 months.

  ◊ `CalculateRarityRating` - returns "A"

  ◊ `Display` - provides the Wildlife's Tracker ID, Rarity Level, Average Sightings, the Least Sightings, the Rarity Rating, as well as the Number of Rhinos in the Rhino's Crash (data must be presented in this exact order) in a single line (see `Wildlife` class for a similar example).

Questions c, d, f, and g must be answered using separate methods (i.e. each of the questions must be answered within its own function, subroutine or event such as button click).

c) Using a single array, instantiate objects to store information of the different Lions and rhinos. The number of Wildlife, type of Wildlife (Lion or Rhino), and values for each Wildlife will be input by the user of the program. You may assume that all Lions are monitored for the same period of time.

d) Using the single array created in Question c) above, display the details of all the Wildlife in a multi-line textbox.

e) Without removing existing code, the Director General of National Parks Management would like your program to calculate the Rarity Rating of all Lions and Rhinos by following the table provided in the `Wildlife` class.

FILES – the following questions (f and g) MUST be completed working with files. Failure to do so will result in the student being awarded 0 for the section below. Any code pertaining to the incorrect file type will also be given 0. It is advised that file-related code is developed in your Form class. Exception handling is not required.

f) Using the single array of `Wildlife` objects you created in Question c), save all the `Rhino` objects to a sequential file. You may assume that the file is saved in the local folder of the Visual Basic application. You may also create the file and save all the `Rhino` objects without having to create, close, open, write, and close again.

g) The Director General of National Parks Management would like to ensure that all the information on their Rhinos have been captured correctly. In a separate multi-line textbox, display the details of all the Rhino objects that have been saved to file. Your code may not rely on any variables that have been previously used to determine how many records there are in the file.

MARKSHEET

0.1	Design (Input, Output, Events, Actions, UML, Variables, Interfaces, Algorithms)	5
	Wildlife Class	
A.1	Attributes (2), Property Method (3)	5
A.2	Utility method (for validation)	2
A.3	Constructor	4
A.4	`CalcAve`	4
A.5	`LowestSightings`	6
A.6	`CalculateRarityRating`	4
A.7	`Display`	4
	Lion Class	
B.1	Attributes (1), Property Methods (2)	4
B.2	Constructor	3
B.3	Other Methods	3
	Rhino Class	
B.4	Constructor	3
B.5	Other Methods (other than Property Method)	2
	Form Class	
C.1	Input of Wildlife information in single dynamic array – Code	9
C.2	Input of Wildlife information in single dynamic array – Correct	9
D.1	Calculation & display of Wildlife information – Code	2
E.1	Code change	1
E.2	Calculation & display of Wildlife information with code change - Correct	3
	Files	
F.1	Setup of serialisation	5
F.2	Saving of `Rhino` objects to sequential file – Code	6
F.3	Saving of `Rhino` objects to sequential file – Correct	6
G.1	Display of `Rhino` objects from sequential file – Code	5
G.2	Display of `Rhino` objects from sequential file – Correct	5
	Total	**100**