University of Johannesburg

Academy of Computer Science & Software Engineering

IFM01B1: Introduction to Data Structures (VB)

Practical Assignment 07 (Due: 23 September 2022 @ 09h00)

Programming Instructions

Copy and paste the following text at the beginning of each class you create and fill in the relevant details:

```
'*******************************************************************************************
' Surname, Initials:
' Student Number:
' Practical: P07
'*******************************************************************************************
```

Question

Semester Test Question

Highly concerned with the poor administration of bursary funds, the Utopian Social Grants Department has approached you to develop a Visual Basic application that will assist them with managing and allocating the funds better for High School and University students.

OBJECT ORIENTED PROGRAMMING – ensure that you implement best programming practices

a) Begin by creating a class called Student with the following attributes:

- _FullName – Full name of the student applying for the bursary
- _NumMarks – an integer value representing the number of Marks that the student has
- _Marks – an Integer array representing the student's marks as percentages (i.e. values 0 to 100 – any invalid entries should be defaulted to 0)
- _Average – a Double value representing the student's average mark
- _Qualifies – a Boolean value indicating whether the student qualifies for bursary or not

Only _Marks (*can read and modify*), _Average, and _Qualifies (*both cannot be modified*) have Property methods.

In addition, cStudent also has the following methods:

- A constructor that accepts the parameters _FullName, and _NumMarks in order to do any necessary initialisations (such as setting size of _Marks) and settings during instantiation. You may assume that _NumMarks will always be correctly provided.
- CalculateAverage – calculates and stores the average of the _Marks array in the variable _Average. Once the average is calculated, it invokes the EvaluateStudent method
- EvaluateStudent – a method that depends on rules that differ depending on whether the student is a High School or University student
- Display – returns a String with a description of the student's Full Name and Average in brackets

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

Class	Properties	Methods

HighSchool	• ExtraMural
	• Prefect
	Both Boolean
		• EvaluateStudent – sets _Qualifies to False by default and True if:
		  ○ Average is >= 80; or
		  ○ Average is >= 70 and
		  ▪ Either ExtraMural or Prefect is True
		• Constructor – accepts the parameter Fullname. All High School students have will three (3) marks.

University	No additional properties	• EvaluateStudent
		  ○ Average is >= 65; _Qualifies set to True
		  ○ Average < 65; _Qualifies set to False
		• Constructor – accepts the parameter Fullname. All University students have will two (2) marks.

The following questions (Questions c to f) 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 High School and University students who apply for bursaries. The number of students, type of student (High School or University) and values for each student will be input by the **user** of the program (*do not make use of random data generation techniques)*.
Hint: Make use of the Marks property method to enter the marks of the student AFTER the object has been instantiated

d) Calculate the averages for each student and display the details of all the students in a multi-line textbox. **Using a comment, clearly indicate the single line of code where polymorphism takes place.** ***You may need to revisit other classes to answer this question.***

FILES – the following questions (e and f) 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.***

e) **Using the single dynamic array of student objects**, save all University objects to a sequential file with the same name as your project name with the file extension ipb. 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 University objects without having to create, close and open again.

The following question requires that you reopen the sequential file saved in Question e) to read the University objects. **Your code must NOT be dependent on knowing how many objects exist in the sequential file.** Your code must also cater for the possibility that no University students were saved to the file *(i.e. there are no records in the sequential file)*.

f) Every year, the Barrier Foundation puts aside R175 000 to be split equally amongst the university students qualifying for a bursary (i.e. if 5 university students qualify for a bursary, each will get R35 000).

Furthermore, Utopian Social Grants Department awards the top University student with an additional R10 000.00 in bursary money. This is the student with the highest average. *You may assume that only one University Student will have the highest average.*

In a single-line textbox, display the details of this student, along with their total bursary income.

0.1	Design (Input, Output, Events, Actions, UML, Variables, Interfaces, Algorithms)	5
Student Class		
A.1	Definition	1
A.2	Attributes	4
A.3	Property & Utility Methods	5
A.4	Constructor	3
A.4	Other Methods	6
HighSchool Class		
B.1	Definition	1
B.2	Instance Variables & Property Methods	2
B.3	Method	2
B.4	Constructor	1
University Class		
B.5	Definition	1
B.6	Method	2
B.7	Constructor	1
Form Class		
C.1	Input of student information in single dynamic array – Code	12
C.2	Input of student information in single dynamic array - Code	12
D.1	Calculation & display of student information – Code	2
D.2	Calculation & display of student information – Correct	2
D.3	Comment indicating where polymorphism takes place	3
Files		
E.1	Setup of serialisation	5
E.2	Saving of University objects to sequential file – Code	6
E.3	Saving of University objects to sequential file – Correct	6
F.1	Calculation & display of highest average student – Code	9
F.2	Calculation & display of highest average student – Correct	9
	TOTAL	100