PROG2121 – WINDOWS PROGRAMMING

A01 – C# AND OBJECT-ORIENTED PROGRAMMING

OVERVIEW

The concepts of Object-Oriented Programming are generally universal. However, the implementation of certain aspects may vary from language to language.

This assignment is intended to give you a chance to review some OOP basics. This is done with C# and .NET. Constructors, inheritance, overloaded methods, overridden methods and properties are implemented.

OBJECTIVES

- Create Windows Applications using the .NET Framework

ACADEMIC INTEGRITY AND LATE PENALTIES

- Please refer to the SET Policies document regarding Academic Integrity Information
- Please refer to the SET Policies document regarding Late Policy

EVALUATION

- Assignment is worth 10% of the final mark
- You must use the .NET Framework (not .NET / .NET Core)
- For details on evaluating the assignment, please see the rubric in eConestoga

PREPARATION

- Review Object-Oriented concepts from the Object-Oriented Programming course:
  - Constructors
  - Inheritance
  - Access modifiers
  - Overloading methods
  - Overriding methods

REQUIREMENTS

You must use the .NET Framework (not .NET / .NET Core).

Good Object-Oriented Principles must be applied!

You must follow SET coding standards.

You may add components if you think it is necessary.

You must use GitHub with the link provided in class/lab. As you develop your solution, please commit regularly to the repository.

You will implement a class hierarchy representing bank accounts. The following describe the attributes of the three existing bank accounts:

Savings Account

- Account number
- Current balance
- Interest rate
- Deposit transaction
- Withdraw transaction
- Apply interest
- Get Savings account information

Chequing Account

- Account number
- Current balance
- Annual fee
- Deposit transaction
- Withdraw transaction
- Apply annual fee
- Get Chequing account information

Loan Account

- Account number
- Current balance (a negative balance can be used here)
- Loan interest rate (per annum)
- Deposit transaction (this is a loan payment, where some goes to interest and the rest goes towards the balance owing)
- Withdraw transaction (this is usually only done once when the loan is given out)
- Interest calculation
- Get Loan account information

For the scenario above, please create the following:

Class diagram:

- Use a simple UML class diagram to show the four classes, their relationship and the methods and properties in each
- It is highly recommended you do this first – before any coding

Classes (in C#)

- Base class called Account that each of the account types will inherit from
- For each of the three accounts:
    - Inherit Account
    - Create a default constructor (it may be empty)
    - Create at least one overloaded constructor to allow initialization of specific properties
    - Create any new properties that will be needed
    - If necessary, overload or override any inherited methods
    - If necessary, create any new methods

Test Program:

- Instantiate each of the three subclasses
- Demonstrate accessing the inherited properties (both write and read)
- Demonstrate accessing the unique property(s) of the subclasses
- Demonstrate the method(s) inherited from the parent
- Demonstrate the overridden methods
- Demonstrate the overloaded methods
- Demonstrate the unique methods

FILE NAMING REQUIREMENTS

- There are no specific file naming requirements. Please keep the name short, but meaningful.
- The submitted file must be a .zip file (or fully compatible)

SUBMISSION REQUIREMENTS

- The full C# solution and class diagram should be submitted in *one* zip folder. Only *.zip* (or fully compatible) format is acceptable.
- Several GitHub commits to show development progress