Lab Programming Exercises- Ch09– Using Classes and Objects

Purpose: The purpose of this Lab exercise is to:

- Practice the use of Classes and Objects

References: Read the course's text book chapter 09 on classes and objects, lecture notes, ppts. This material provides the necessary information that you need to complete the exercises.

Instructions: Be sure to read the following general instructions carefully:

These lab exercises should be completed individually by all the students to get better understanding of classes and objects and their use. These will prove helpful in completing your lab assignments and tests. *Your IDE is Visual Studio 2017 community edition or higher version such as Professional or Enterprise.*

Apply the naming conventions for variables, methods, classes, and namespaces:

- variable names start with a lowercase character
- classes start with an uppercase character
- namespaces use only lowercase characters
- methods start with a uppercase character

Exercise #1:

Write a C# application that implements the following class(es) as per business requirements:

Create a **CommissionEmployee** class (CommissionEmployee.cs) that has the following instance variables:

- First name , last name , social security number, gross sales (amount in dollars) and commission rate
- Define properties with validations (e.g. gross sales and commission rate should be positive) for all the above.
- Social Security number should only have getter method.
- Gross sales cannot be negative and commission rate should be between 0.1 and 2%
- Class should have defined two overloaded constructors:
  - One for initializing all the instance data members
  - Second for initializing only first name, last name and social security number.
- Define a public method **double earnings()** which calculates employee's commission (commission rate * gross sales)
- Define a public method – **String toString()** which is used to display the object's data

Create a test class – **CommissionEmployeeTest** (CommissionEmployeeTest.cs) which tests above class by at least creating two objects of the CommissionEmployee class.

Exercise #2:

Write a C# application that implements the following class(es) as per business requirements mentioned below:

Create a **BasePlusCommissionEmployee** class (BasePlusCommissionEmployee.java) that has the following instance variables:

- First name , last name , social security number, base salary, gross sales (amount in dollars) and commission rate
- Define properties with validations for all the above.
- Social Security number should only have getter method.
- Gross sales and base salary cannot be negative and commission rate should be between 0.1 and 1.0%

- Class should have defined two overloaded constructors:
    - One for initializing all the instance data members
    - Second for initializing first name, last name and social security number only.
- Define a public method **double earnings()** which calculates employee's commission (commission rate * gross sales + base salary)
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class – **BasePlusCommissionEmployeeTest** (BasePlusCommissionEmployeeTest.c#) which tests above class by at least creating two objects of the BasePlusCommissionEmployee class.

Exercise #3:

Write a C# application that implements the following class(es) as per business requirements mentioned below:

Create a **CheckingAccount** class (*CheckingAccount.cs*) that has the following instance variables:

- Account number , customer name, account balance
- Define properties with validations for all the above.
- Account number should only have getter method.
- Account balance cannot be negative
- Class should have defined two overloaded constructors:
    - One for initializing all the instance data members
    - Second for initializing only Account number, customer name
- Define one public method - **double withdraw (double amount)** which is used for taking out money
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class – **CheckingAccountTest** (*CheckingAccountTest.cs*) which tests above class by at least creating two objects of the CheckingAccount class.

Exercise #4:

Write a C# application that implements the following class(es) as per business requirements mentioned below:

Create a **SavingsAccount** class (SavingsAccount.cs) that has the following instance variables:

- Account number , customer name, account balance, interest rate
- Define properties with validations for all the above.
- Account number should only have getter method
- Account balance and interest rate cannot be negative and interest rate should be between 0.1 and 2.0%
- Class should have defined two overloaded constructors:
    - One for initializing all the instance data members
    - Second for initializing only account number , customer name.
- Define two public methods **double deposit(double amount)** which is used for depositing money to account balance and another method - **double calculateMonthlyInterest()** to calculate monthly interest on balance.
- Define a public method – **String toString()** which is used to display the object's data

Create a driver class – **SavingAccountTest** (SavingAccountTest.cs) which tests above class by at least creating two objects of the SavingAccount class.