CIS 605 – Fall 2024
Assignment Set 3
Due: Sunday, September 29 @ 11:59 PM

Develop the projects described below using good visual design and program coding practices that includes

- Professional Appearance (Layout, placement, spelling, formatting)
- Meaningful title on title bar of form(s)
- Identifiers (names) for objects, variables, and constants are meaningful and follow a consistent naming convention
- General remarks at the start of every class in your program including Class Name, Class Description, Developer Name, Date Created, Date Last Modified
- Descriptive remarks for every method
- Proper indentation & blank line after each full comment line
- All variables and constants are local whenever possible (scope)
- Modular programming – i.e., breaking down a "large" programming task into multiple, independent modules, with each module performing one part of the required functionality.

For all programs

- Use NumericUpDown controls for numeric input
- Declare and use constants when appropriate
- Format output data

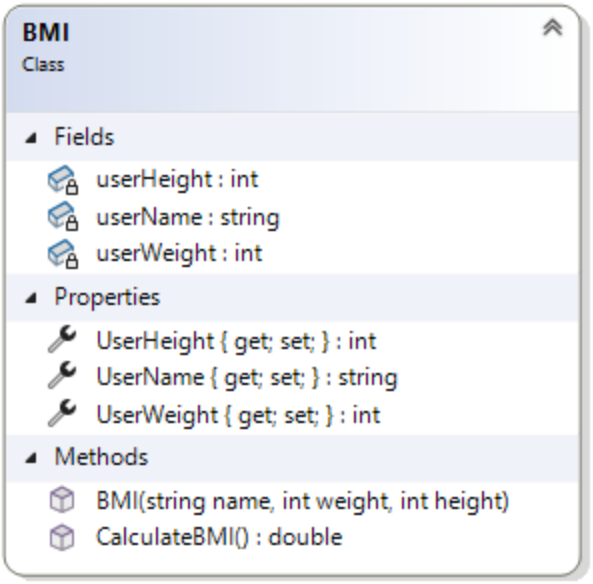
Program 6

Create a class (BMI) that has

- 3 Instance Properties (with a private field and public get and set accessors for each)
  - User name
  - User weight
  - User height

- 1 Constructor
  - To instantiate object and set three properties – user name, user weight and user height - using parameters)

- 1 Public Instance Method
  - To calculate and return the user's BMI
    - User BMI = user weight * 703 / height^2



Create a Form class that has:

- appropriate controls to input the following:
    - User's name
    - User's weight (in pounds)
    - User's height (in inches)
- a Create BMI button to instantiate a BMI object, call the instance method, and display the user's BMI in a label
- a Clear/Reset button to clear or reset the values displayed on the form
- an Exit button to exit the application

Program 7

Create a class (TruckRental) that has

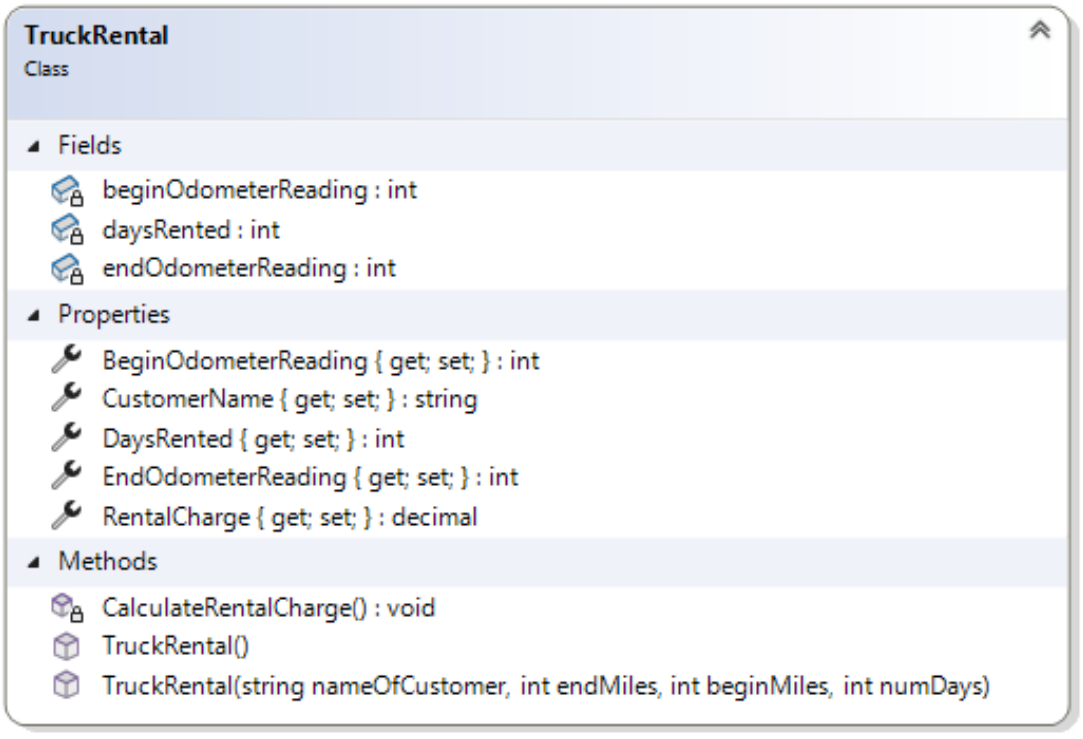
- 5 Instance Properties
  - Customer name (auto-implemented – public get and set)
  - Rental charge (auto-implemented – public get and private set)
  - Ending odometer reading (with a private field and public get and set accessors)
  - Beginning odometer reading (with a private field and public get and set accessors)
  - Number of days rented (with a private field and public get and set accessors)

  After setting their respective values, the set accessors for ending odometer reading, beginning odometer reading, and number of days rented should call the instance method that calculates the rental charge.

- 2 Constructors
  - A default constructor (with no parameters) to instantiate object
  - An overloaded constructor to instantiate object and set four properties – customer name, ending odometer reading, beginning odometer reading, and number of days rented - using parameters

- 1 Private Instance Method

  - To calculate rental charge and set the rental charge property
    - Rental charge = $.63.90 per day + $0.81 per mile



Create a Form that has:

- appropriate controls to input the following:
    - Customer name
    - Beginning odometer reading
    - Ending odometer reading
    - Number of days rented
- a Create Truck Rental button to instantiate a Truck Rental object and display the rental charge in a label
- a Clear/Reset button to clear or reset the values displayed on the form
- an Exit button to exit the application

Program 8

Create a class (PaySlip) that has

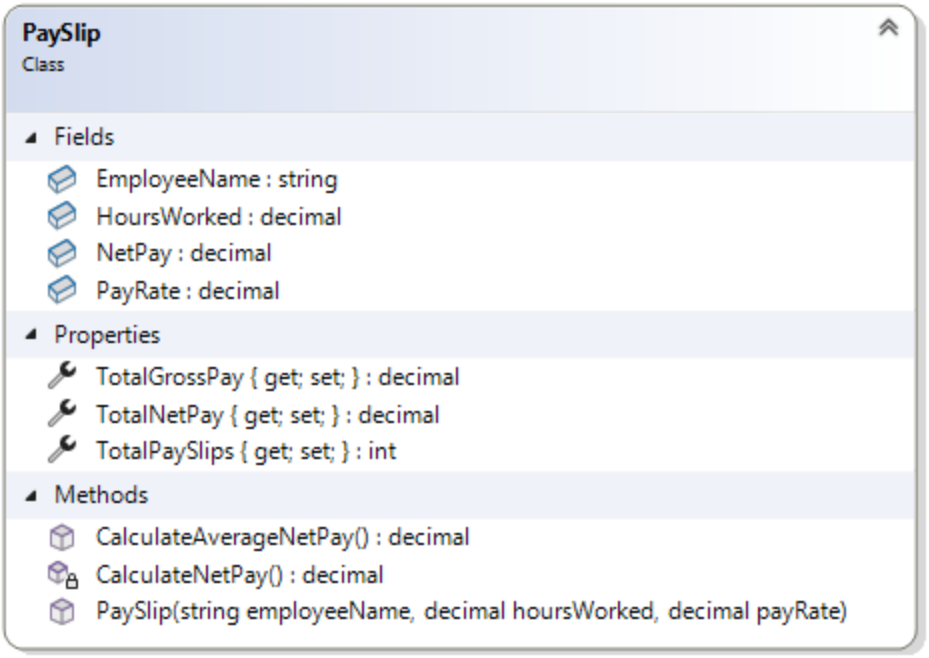
- 4 Instance Properties (read only fields)
  - Employee name
  - Hours worked
  - Pay rate
  - Net pay

- 3 Static Properties (auto-implemented – public get and private set)
  - Total number of pay slips
  - Total gross pay
  - Total net pay

- 1 Constructor
  - A constructor that a) instantiates object b) sets three properties – employee name, hours worked, and pay rate - using parameters, and c) sets the net pay property by calling the instance method that calculates net pay (see below)

- 1 Private Instance Method that
  - Calculates net pay
    - Net pay = Gross Pay – Federal Income Tax – State Income Tax - Social Security Tax – Medicare Tax
      - Gross Pay = hours worked * pay rate
      - Federal Income Tax = gross pay * 13.29%
      - State Income Tax = gross pay * 3.55%
      - Social Security Tax = gross pay * 7.17%
      - Medicare Tax = gross pay * 1.68%
  - Increments the three static properties
  - Returns the net pay

- 1 Public Static Method
  - To calculate and return the average net pay



Create a Form that has:

- appropriate controls to input the following:
    - Employee name
    - Hours worked
    - Pay rate
- a Create Pay Slip button to instantiate a Pay Slip object and display the net pay in a label
- a Display Summary button to display the total number of pay slips, total gross pay, total net pay, and the average net pay
- a Clear/Reset button to clear or reset the values displayed on the form
- an Exit button to exit the application