C# classes.

1. Rename the current class „Programm“ to „TestingClass“
2. **Class declaration and structure**: https://www.tutorialspoint.com/csharp/csharp_classes.htm examine the „Box“ class example, view the example here http://csharp.net-tutorials.com/classes/introduction/
    a. Write a new class called „HelloWorldPrinter“
    b. Print „Hello world“ in method „PrintText()“ (create the method)
    c. Create a new instance of the „HelloWorld“ class in „TestingClass“ main method and call out the method PrintText()
3. **Class fields:** read http://www.completecsharptutorial.com/basic/staticmethod-variables/, https://syntaxdb.com/ref/csharp/class-var
    a. Create a new class called Person
    b. Add field Name, age
    c. Create a new instance of the class and try to set different
4. **Class methods**
    a. Add a method to class Person called „SetName(string name)“
    b. Add a method called GetName() what prints out the name
    c. Add a method SetAge(int age) which takes age as integer value
    d. Add another method called SetAge(DateTime birthYear) which takes birthyear as parameter and calculates the age (age = DateTime.Now – birthYear)
    e. Add third method called SetAge(string value) which has 3 behaviours:
        - If the length of the string is 1-3 then uses the string as age
        - If the length of the string is 4 then uses the string as birthyear (create new DateTIme())
        - If the length of the string is less than 1 or bigger than 4, then write „Invalid value for setting the year“.
5. **Class constructor**. Read: https://www.dotnetperls.com/constructor
    a. Create the default constructor which sets the name as „Jhon Smith“
    b. Create constructor accepting string calling out method SetName() to set the name
    c. Create a constuctor accepting string and age and setting them

Exercise 2:

Add a class to the application called “VendingMachine“.This class will be composed of one field (i.e., instance variable or attribute) and several methods as follows:

- A field of type int called “DepositedAmount“.
- A constructor that takes no parameters that simply initializes the machine’s DepositedAmount to 0.
- A method (no return value) called “DepositCoin” with a single int parameter called “coinAmount”.
  This method adds the value passed in the coinAmount parameter to the machine’s DepositedAmount field.
- A method (no return value) called “GetDrink” with no parameters
  This method does one of two things. If the machine’s DepositedAmount is is sufficient to cover the cost of a drink (75 cents or more), then the method calculates the change (i.e.,DepositedAmount less 75 cents), writes “Your change is {change amount} cents” to the Console, and resets the DepositedAmount back to 0. If the machine’s DepositedAmount is not sufficient (less than 75 cents), it writes “Insert more coins” to the Console.
- A method (no return value) called “GetRefund” with no parameters
  This method simply displays “You were refunded {refund amount}” (refund amount should be the whole DepositedAmount) to the console and resets the machine’s DepositedAmountback to 0.
- Your Main method should create a single instance of your VendingMachine class. You should then call methods on this object to fully test that the machine is functioning properly. No need to get input from the user. Just “hard code” instructions in your Main method.