**Assignment 1 – BCS 426 C# Programming**

**Due: 2/5/2020 @ 1:40pm**

You will need to create two Visual Studio solutions for this assignment. When you submit the assignment in the Blackboard dropbox make sure you zip both solutions and submit them.

IMPORTANT – Make sure you ***properly comment*** AND ***properly indent*** your program. The commenting and indenting documents are on Blackboard in the “Handouts” folder. ***If you fail to properly comment or properly indent I will deduct points.***

***VERY IMPORTANT – IF THE PROGRAM DOES NOT COMPILE THERE WILL BE MAJOR POINTS TAKEN OFF.***

***Overview***

Create the first program in the development of the payroll application.

DLL solution. You will need to create a DLL solution which will contain multiple class definitions and testing code. This DLL will contain all the classes we use for our application. We will also add classes to this DLL later in the semester.

Console solution. Create a console application that will import the DLL and use the classes we wrote.

You will also need to use regions in your code.

***Part 0 – Regions***

All code that you write should be inside of C# regions. You should create regions for the following:

* For each class create one region containing all the private member variables
* For each class create one region containing all the properties
* For each class create one region containing all the methods
* Create one region in main for the unit testing calls

***Part 1 – Class Definitions in the DLL***

Create a DLL solution in Visual Studio (C# Class Library project type). There should be two projects within the solution.

* Payroll Solution
  + Payroll project – Contains the classes that we define.
  + Payroll testing project – Contains the testing code.

The DLL solution should contain the definitions of the classes described below.

**Class – Worker**

***Member Variables (all private)***

name (string), id (int), payrate(double)

***Member Method Signatures and Descriptions (all public)***

|  |  |
| --- | --- |
| **Signature** | **Description** |
| Worker() | Default constructor. Sets the values of each member variable to a default value. |
| C# properties | Write properties for each member variable. Do NOT write traditional get/set methods. Use normal C# Properties (not auto-implemented properties). |
| String ToString() | This method should return a string that contains descriptive text and data for all member variables. |

**Class – Shift**

***Member Variables (all private)***

workerId (string), hoursWorked (double), date(DateTime)

***Member Method Signatures and Descriptions (all public)***

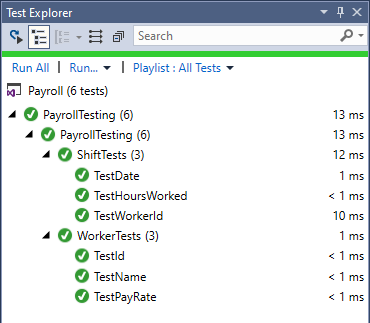
|  |  |
| --- | --- |
| **Signature** | **Description** |
| Shift() | Default constructor. Sets the values of each member variable to a default value. |
| C# Properties | Write properties for each member variable. Do NOT write traditional get/set methods. Use normal C# Properties (not auto-implemented properties). |
| String ToString() | This method should return a string that contains descriptive text and data for all member variables. |

***Part 2 – Testing Project***

Create a new testing project in the DLL solution. The type of the project should be “Unit Test Project (.NET Framework)”. Add the following classes and methods.

* Class - WorkerTests
  + TestName
  + TestId
  + TestPayRate
* Class – ShiftTests
  + TestWorkerId
  + TestHoursWorked
  + TestDate

You should use Assert methods within your testing methods to make sure the results are correct. Check at the end of the unit testing slides for an example.

Here is a screenshot after running unit tests:  


***Part 3 – Main Project***

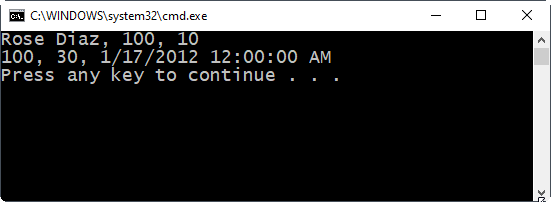
Create a new solution for a console application project in Visual Studio. It should import the payroll DLL solution. DO NOT COPY THE DEFINITION OF ANY OF THE CLASSES INTO THE CONSOLE APPLICATION, IMPORT THE DLL SOLUTION!!!

**Main**

Create one instance of each class. In main do the following:

* Read the data for each instance from input files (one input file for each class). Check below for sample input files.
* Call ToString on each instance and display whatever ToString returns on the screen.
* Write data for both instances to separate output files. Data should be written out in the order listed in the file formats detailed below.

Here is a screenshot of the program running:



***Worker Input File Format***

Name

Id

PayRate

***Worker Sample Input File***

Rose Diaz

100

10.00

***Shift Input File Format***

WorkId

HoursWork

Year

Month

Day

***Shift Sample Input File***

100

30

2020

1

17