**Module Three Assignment**

Placing all of your code into the Main() method is not recommended, nor is it good programming practice.  Prior to object-oriented programming, developers used functions, which are synonymous with methods, to break up the functionality of the application into smaller chunks.  Each method should have a single task to perform.

In later modules, you will begin to create class files to represent the items we have been using so far.  You will create class files for Students, Teachers, Courses, etc.  The class files will consist of attributes and methods.  To prepare for this, the assignment for this module will ask you to think about some basic methods that you can implement in your code.

For example, class files make use of a special method known as a constructor.  The constructor is used to initialize some values for an object when it is created.  This assignment requires you to create some methods for getting data for your variables and sending that data to the console window.

In the assignment, you are to practice getting values from a user and assigning the to local variables.  As a result, move the variables into the appropriate Get methods.

Then, from within the Get methods, assign the input values to the local variables.

Then, call an output method, passing in the variables, and use an appropriate message to print the content to the console window.

The first example is a guide for you, the rest you will need to create on your own.

Create a method that prompts a user of your console application to input the information for a student:

static void GetStudentInfo()  
{  
      Console.WriteLine("Enter the student's first name: ");  
      string firstName = Console.ReadLine();  
      Console.WriteLine("Enter the student's last name");  
      string lastName = Console.ReadLine();  
       // Code to finish getting the rest of the student data  
      .....  
}

static void PrintStudentDetails(string first, string last, string birthday)  
{  
    Console.WriteLine("{0} {1} was born on: {2}", first, last, birthday);  
}

1. Using the above partial code sample, complete the method for getting student data.
2. Create a method to get information for a teacher, a course, and program, and a degree using a similar method as above
3. Create methods to print the information to the screen for each object such as static voidPrintStudentDetails(...)
4. Just enter enough information to show you understand how to use methods.  (At least three attributes each).
5. Call the Get methods from the Main method in the application
6. Call the Print methods from within each Get method

**Exceptions**

1. At times, developers create method signatures early on in the development process but leave the implementation until later.  This can lead to methods that are not complete if a developer forgets about these empty methods.  One way to help overcome the issue of not remembering to complete a method is to throw an exception in that method if no implementation details are present.
2. For this task, use MSDN to research the NotImplementedException exception.
3. Create a new method for validating a student's birthday.  You won't write any validation code in this method, but you will throw the NotImplementedException in this method
4. Call the method from Main() to verify your exception is thrown

**Challenge**(This challenge is for your own study and does not need to be submitted for peer review)

Using MSDN, research the System.DateTime type.  Using the information you learn, modify your birth date field for the student and/or teacher to ensure it used a DateTime type if you did not already include that in your data for these objects.

* Remove your NotImplementedException statement in the validate method you created above.
* Create a try/catch block to catch invalid date entries and display a message to the user if this occurs.  (Console output)
* Assume that your student must be at least 18 years of age to enroll at a university.
* Write code that validates the student is at least 18 years old.  You can use birth year and math or you can calcuate from today's date and work back.
* Output an error message to the console window if the student's age is less than 18

*NOTE: Typically you would write validation code using regular expressions or some other method to verify the date entered but this is a way to practice exception handling but we are not teaching regular expressions in this course.*