Lab Assignment 3 (Fall 2020)

To do this lab, you will need to use **C#** in **Visual Studio Professional 2019**. You can access this program in **Mohawk Apps**, while either on campus or at home. Alternatively, while on campus a local version can be accessed from the **Start Menu**, or, you can download and install it as described by the instructions in the **Student Resources** sub-section located in the **Modules** section of the course page.

To Be Graded – General Details:

- This program will be marked for 6% of your final grade
- Please examine the **Marking Scheme (https://mycanvas.mohawkcollege.ca/courses/45074/pages/lab-assignment-3-fall-2020#jump)** to see the marks breakdown
- This program needs to have appropriate internal comments, as well as **XML comments** for *every class* and *every method*
- This program also needs to have an appropriate comment block at the top of all code files that contains:
  - Your name and student number
  - The file date
  - The program's purpose
  - Your **Statement of Authorship (https://mycanvas.mohawkcollege.ca/courses/45074/pages/statement-of-authorship)**
- Bundle your project into one Zip file, and upload it to the appropriate **Lab Assignment (https://mycanvas.mohawkcollege.ca/courses/45074/assignments/366641)** on MyCanvas
- Please read about **documentation (https://mycanvas.mohawkcollege.ca/courses/45074/pages/program-documentation)** style
- Programs that are late will be penalized 10% per day (includes each day of a weekend)
- Programs that do not compile or do not include a **Statement of Authorship (https://mycanvas.mohawkcollege.ca/courses/45074/pages/statement-of-authorship)** will be penalized 10% for each

Part A: Media is the Message

Project Name: Lab3A Create Class: *Various (one file for each class)*

Write a Console App (.NET Framework) that:

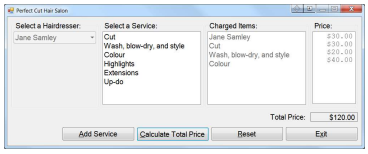
- Makes use of an interface called **IEncryptable (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314861/download)**  **(https://mycanvas.mohawkcollege.ca/courses/45074/files/6314861/download)** that contains method signatures for **Encrypt( )** and **Decrypt( )** (right-click and save as *IEncryptable.cs*)
- Makes use of an interface called **ISearchable (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314859/download)** 

(https://mycanvas.mohawkcollege.ca/courses/45074/files/6314859/download) that contains a method signature for **Search( )** (right-click and save as *ISearchable.cs*)

- Makes use of an abstract class called **Media** (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314863/download) (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314863/download) which represents one single media object (right-click and save as *Media.cs*)
- Creates additional classes derived from *Media*:
  - **Book** (represents one book and has two string properties, *Author* and *Summary*)
  - **Movie** (represents one movie and has two string properties, *Director* and *Summary*)
  - **Song** (represents one song and has two string properties, *Album* and *Artist*)
- The main class (**Lab3A**) should have the following features:
  - A method called **ReadData( )** that will read the **Data.txt** (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314865/download) (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314865/download) file (right-click and save as Data.txt) and store up to 100 searchable media objects into an array
    - Examine the data file structure to see how the different media information has been formatted and stored
    - The data file will have the *Summary* information for both *Books* and *Movies* encrypted using a simple Rot13 algorithm (see Wikipedia)
    - Include exception handling for the file I/O
  - Prompts the user via a menu to display your media objects in a variety of ways:
    1. **List All Books** - a neat list of all Book objects (no *Summary* displayed)
    2. **List All Movies** - a neat list of all Movie objects (no *Summary* displayed)
    3. **List All Songs** - a neat list of all Song objects
    4. **List All Media** - a neat list of all derived Media objects (no *Summary* displayed)
    5. **Search All Media by Title** - a neat list of all objects with the search key anywhere in the *Title* (display decrypted *Summary* where available)
    6. **Exit Program**
  - Continues to prompt until the user selects the exit option
  - Error checking for user input
  - The **Main( )** method should be highly modularized
- You may download this **sample program** (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314977/download) for a demonstration of program behaviour

Part B: Hair of the Dog

Project Name: Lab3B



(right-click to view)

Write a Windows Form App (.NET Framework) that:

- Makes use of an alternate GUI interface (shown to the right) that determines pricing for a hair salon
- The user must:

1. Select one **Hairdresser** from a ComboBox (DropDownList style), each of which has a different base rate:
   - Jane - $30
   - Pat - $45
   - Ron - $40
   - Sue - $50
   - Laurie - $55
2. Select one or more **Services** from a ListBox, each of which has a different rate:
   - Cut - $30
   - Wash, blow-dry, and style - $20
   - Colour - $40
   - Highlights - $50
   - Extension - $200
   - Up-do - $60
3. The **Add Service** Button will:
   - Display the selected **Hairdresser** (one) in the **Charged Items** ListBox
   - Display the selected **Services** (one or more) in the **Charged Items** ListBox
   - Display the corresponding price of the **Charged Item** in the **Price** ListBox
4. The **Calculate Total Price** Button will display the total cost of all items in the **Price** ListBox in currency format
5. The **Reset** Button will select the first entry in the **Hairdresser** ComboBox, clear the **Charged Items** and **Prices** ListBoxes, clear the **Total Price** Label, disable the **Add Service** and **Calculate Total Price** Buttons, and set focus to the **Hairdresser** ComboBox

- Specification Notes:
  - The first time the **Add Service** Button is used, both the hairdresser selected and the first service selected will be added to the **Charged Items** ListBox. Every time after that, the **Add Service** Button will only add the selected service
  - Using the Enabled property of controls, disable/enable appropriately to prevent erroneous selections from being made and control the order in which selections are made. For example:
    a. Initially the **Add Service** and **Calculate Total Price** Buttons should be disabled
    b. The **Add Service** Button is enabled once a selection is made from **Service** ListBox
    c. The **Calculate Total Price** Button is enabled when the **Add Service** Button is used for the first time

d. The **Hairdresser** ComboBox is disabled when the **Add Service** Button is used for the first time

- You may download this **sample program (https://mycanvas.mohawkcollege.ca/courses/45074/files/6314999/download)** for a demonstration of program behaviour

Marking Scheme

Part A: Media is the Message	
Documentation: Comments, Naming Conventions	/ 5
Interfaces: IEncryptable, ISearchable	/ 3
Classes: Book, Movie, Song	/ 3
Methods: ReadData	/ 3
Lists: Books, Movies, Songs, All, Search	/ 4
Menu: All Options, Re-Prompts	/ 1
Output: Neat, Complete	/ 1
Part B: Hair of the Dog	
Documentation: Comments, Naming Conventions	/ 5
GUI Appearance: Correctly Implemented	/ 4
GUI Behaviour: Correct Enables/Disables	/ 4
Services Button: Correctly Adds to Both ListBoxes	/ 3
Calculate Button: Correct and Properly Formatted Output	/ 2
Reset Button: Resets Controls	/ 2
Total:	**/ 40**