# IT 230 Coding Activity Submission Template

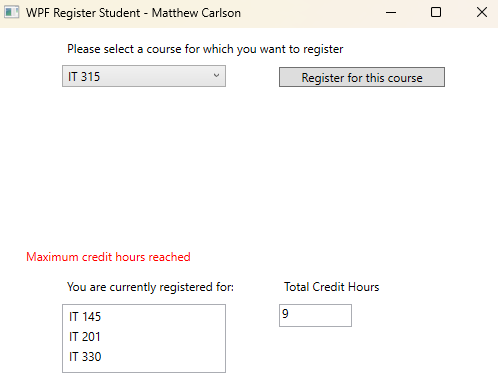
Submit your work on the coding activities for Modules One, Two, Three, Four, and Six in this document. In addition to this document, you should submit a ZIP file containing all your Visual Studio project files and source code that can be run in Visual Studio on a different computer.

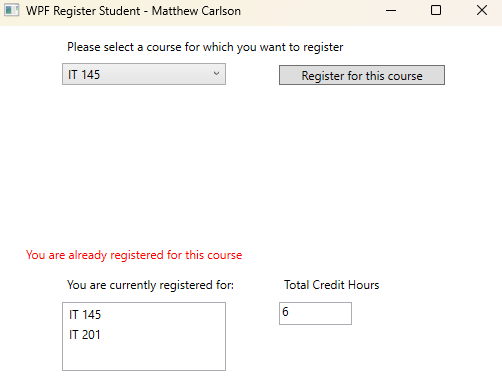
For each coding activity, complete the following steps:

* Download and rename this document to meet the file naming conventions requested in the assignment instructions.
* Fill in the required information below by replacing the bracketed text with the relevant information.
* Submit this document and your ZIP file for grading and feedback. Your ZIP file should follow the same naming conventions.

Document your work in the coding activity by completing each of the following items:

1. Provide a screenshot of the output that resulted from running your program successfully in Visual Studio. See the coding assignment instructions for an example of what should be included in the screenshot. Your screenshot must include the following elements:
   1. Your last name as the first printed text on the screen
   2. Verification that the program is fully functioning and data results are accurate for the given problem





1. Copy and paste the source code text you wrote for this assignment from the \*.cs file into the space below. Only providing the \*.cs files or a screenshot does not meet the requirements for this part of the assignment. Code should be logically organized. It should also follow proper syntax and conventions noted in the Coding Activity Guidelines and Rubric.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace WPFRegisterStudent

{

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

Course choice;

int creditHours = 0; // Integer variable for credit hours

public MainWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

Course course1 = new Course("IT 145");

Course course2 = new Course("IT 200");

Course course3 = new Course("IT 201");

Course course4 = new Course("IT 270");

Course course5 = new Course("IT 315");

Course course6 = new Course("IT 328");

Course course7 = new Course("IT 330");

this.comboBox.Items.Add(course1);

this.comboBox.Items.Add(course2);

this.comboBox.Items.Add(course3);

this.comboBox.Items.Add(course4);

this.comboBox.Items.Add(course5);

this.comboBox.Items.Add(course6);

this.comboBox.Items.Add(course7);

this.textBoxCred.Text = "";

this.textError.Text = ""; // Added Text block for error

}

private void button\_Click(object sender, RoutedEventArgs e)

{

choice = (Course)(this.comboBox.SelectedItem);

if (creditHours >= 9) // For max credit hours

{

this.textError.Text = "Maximum credit hours reached";

}

else if (choice.IsRegisteredAlready()) // If already registered for course

{

this.textError.Text = "You are already registered for this course";

}

else if (choice.IsRegisteredAlready() == false) // If not registered for course & not at max credit hours

{

choice.SetToRegistered();

this.listBox.Items.Add(choice);

creditHours += 3;

}

this.textBoxCred.Text = Convert.ToString(creditHours); // Put creditHours to screen

}

}

}

XAML:

<Window x:Class="WPFRegisterStudent.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:WPFRegisterStudent"

mc:Ignorable="d"

Title="WPF Register Student - Matthew Carlson" Height="395.902" Width="525" Loaded="Window\_Loaded">

<Grid>

<ComboBox x:Name="comboBox" HorizontalAlignment="Left" Margin="66,37,0,0" VerticalAlignment="Top" Width="164" IsDropDownOpen="True"/>

<Button x:Name="button" Content="Register for this course" HorizontalAlignment="Left" Margin="283,39,0,0" VerticalAlignment="Top" Width="166" Click="button\_Click"/>

<ListBox x:Name="listBox" HorizontalAlignment="Left" Height="69" Margin="66,276,0,0" VerticalAlignment="Top" Width="164"/>

<TextBox x:Name="textBoxCred" HorizontalAlignment="Left" Height="23" Margin="283,276,0,0" TextWrapping="Wrap" Text="0" VerticalAlignment="Top" Width="73" IsReadOnly="True"/>

<TextBlock x:Name="textError" HorizontalAlignment="Left" Margin="30,220,0,0" Grid.Row="2" TextWrapping="Wrap" VerticalAlignment="Top" Height="30" Width="463" FontSize="12" Foreground="Red" TextAlignment="Left"/>

<Label x:Name="label" Content="Please select a course for which you want to register" HorizontalAlignment="Left" Margin="66,7,0,0" VerticalAlignment="Top" Width="383"/>

<Label x:Name="label1" Content="You are currently registered for:" HorizontalAlignment="Left" Margin="66,245,0,0" VerticalAlignment="Top" Width="176"/>

<Label x:Name="label2" Content="Total Credit Hours" HorizontalAlignment="Left" Margin="283,245,0,0" VerticalAlignment="Top" Width="106"/>

<Label x:Name="label3" Content="" HorizontalAlignment="Left" Margin="66,213,0,0" VerticalAlignment="Top" Width="383" Foreground="#FFEE0E0E"/>

</Grid>

</Window>

1. Show that you understand the task by explaining the design of your program in the space below. Include the process and steps you took to write your code. Explain how you arrived at the solution to the problem and completed the activity.

The task was to, on button press, add the selected class to the box of classes and then also count total credit hours. I achieved this by utilizing code that adds the selection to the combo box from the previous project where we created the class object. Then, it was a simple task of declaring the creditHours variable, and then moving everything into if statements. I created three statements to determine everything. The first being a check on creditHours---if the student is already registered for the maximum number of classes, do not allow for another and print an error message. The second statement checks if the selected course has the boolean for registered already set to true. If not, move on. The third statement, a simple else, applies the registered boolean to the unique course and adds the course to the combo box.

1. Reflect on your learning experience and what you learned from completing the activity.

This project was genuinely really fun. I enjoyed learning about the XAML stuff, and I greatly enjoyed learning about the button function and realizing that the code executed every time that button was pressed. I have a few little side projects I may utilize this information for, so this was really cool and really fun to do.