# IT 230 Coding Activity Submission Template

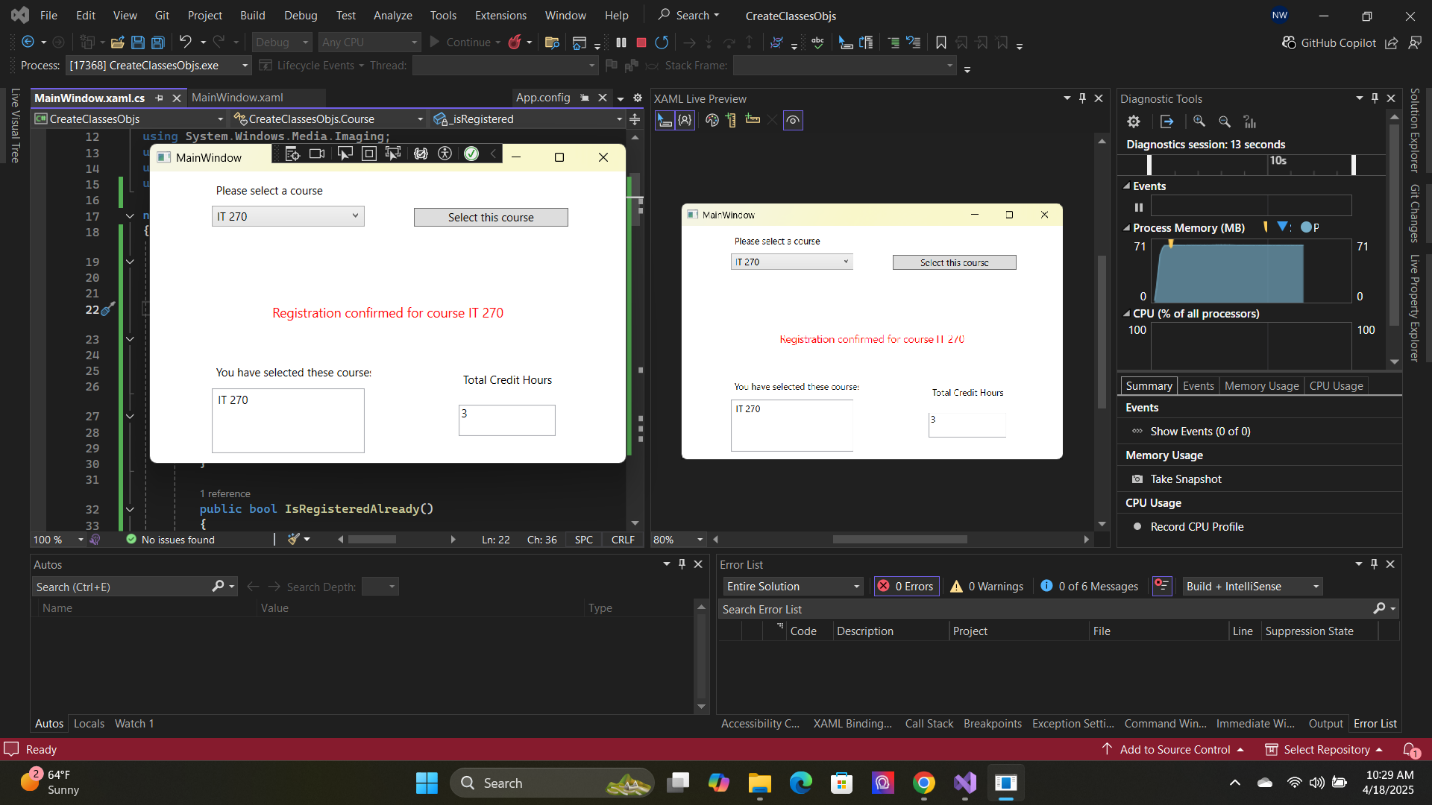
Submit your work on the coding activities for Modules One, Two, Three, Four, and Six in this document. In addition to this document, you should submit a ZIP file containing all your Visual Studio project files and source code that can be run in Visual Studio on a different computer.

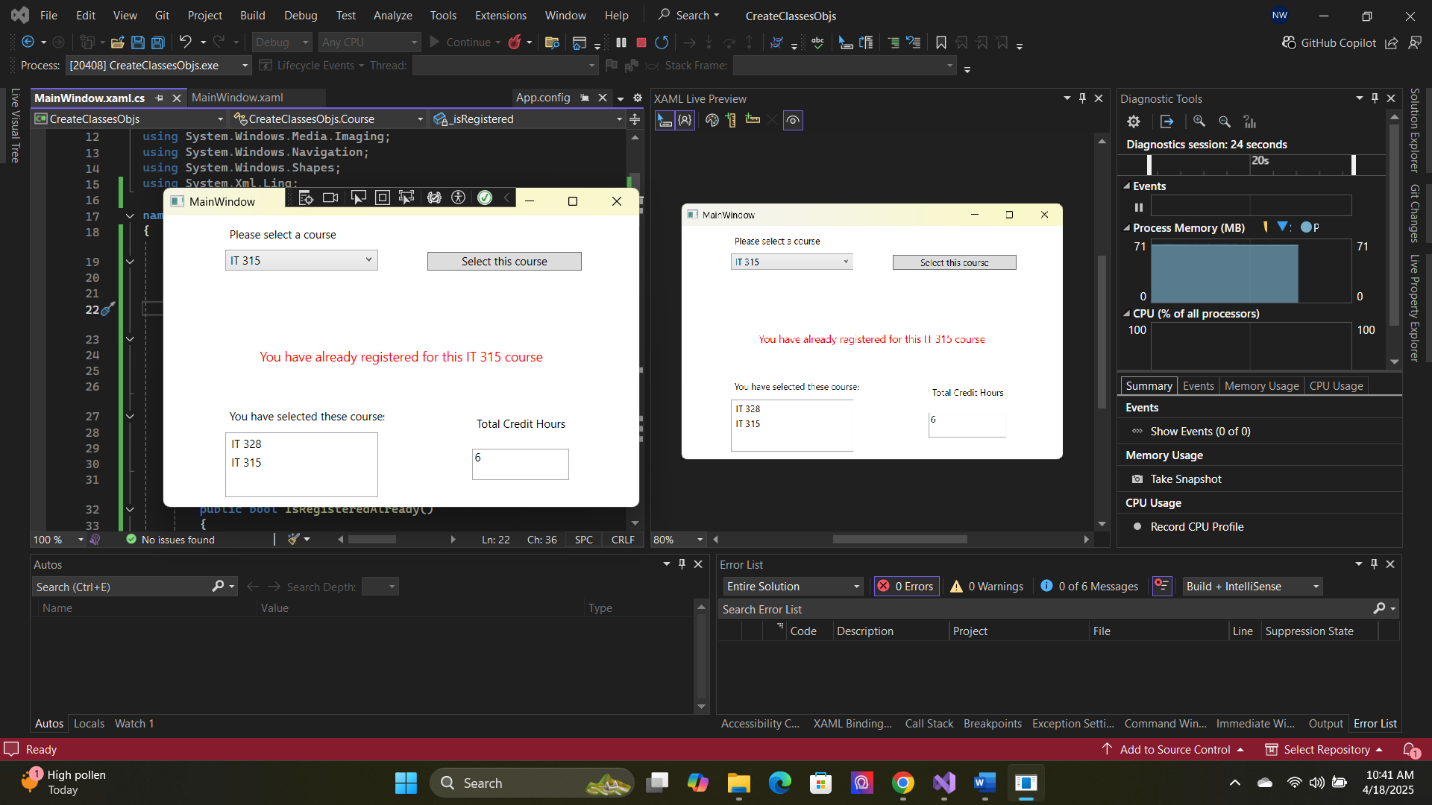
For each coding activity, complete the following steps:

* 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



A screenshot of a computer

AI-generated content may be incorrect.

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;

using System.Xml.Linq;

namespace CreateClassesObjs

{

class Course

{

public string name;

private bool \_isRegistered;

public void setName(string userChoice)

{

name = userChoice;

}

public string getName()

{

return name;

}

public bool IsRegisteredAlready()

{

return \_isRegistered;

}

public void SetToRegistered()

{

\_isRegistered = true;

}

public override string ToString()

{

return getName();

}

}

/// <summary>

/// Interaction logic for MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

Course choice;

private int creditHours = 0;

public MainWindow()

{

InitializeComponent();

}

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

{

Course course1 = new Course();

Course course2 = new Course();

Course course3 = new Course();

Course course4 = new Course();

Course course5 = new Course();

Course course6 = new Course();

Course course7 = new Course();

course1.setName("IT 145");

course2.setName("IT 200");

course3.setName("IT 201");

course4.setName("IT 270");

course5.setName("IT 315");

course6.setName("IT 328");

course7.setName("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);

}

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

{

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

if (this.listBox.Items.Contains(choice) && choice.IsRegisteredAlready())

{

this.textBlock1.Text = "You have already registered for this " + choice.ToString() + " course";

}

else if (creditHours < 9)

{

this.listBox.Items.Add(choice);

choice.SetToRegistered();

creditHours += 3;

this.textBox.Text = Convert.ToString(creditHours);

this.textBlock1.Text = "Registration confirmed for course " + choice.ToString();

}

else

this.textBlock1.Text = "You cannot register for more than 9 credit hours.";

}

}

}

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.

If else statements were added to the code to make sure the conditions for the registration were correct. I also had to add several classes to the WPF XAML window to get the code to run and register with the actions made. A textbox updates with the amount of creditHours per selected course. Another textbox was added to display the messages for the user which include both errors and confirmation messages. I also adjusted the previous lessons code to adjust to what was added.

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

It was a fun lesson to work on. It was quite tricky but having already worked on the code beforehand it became simpler. This challenge required problem solving and not just writing code but adding elements to the output window.