# IT 230 Coding Activity Submission Template

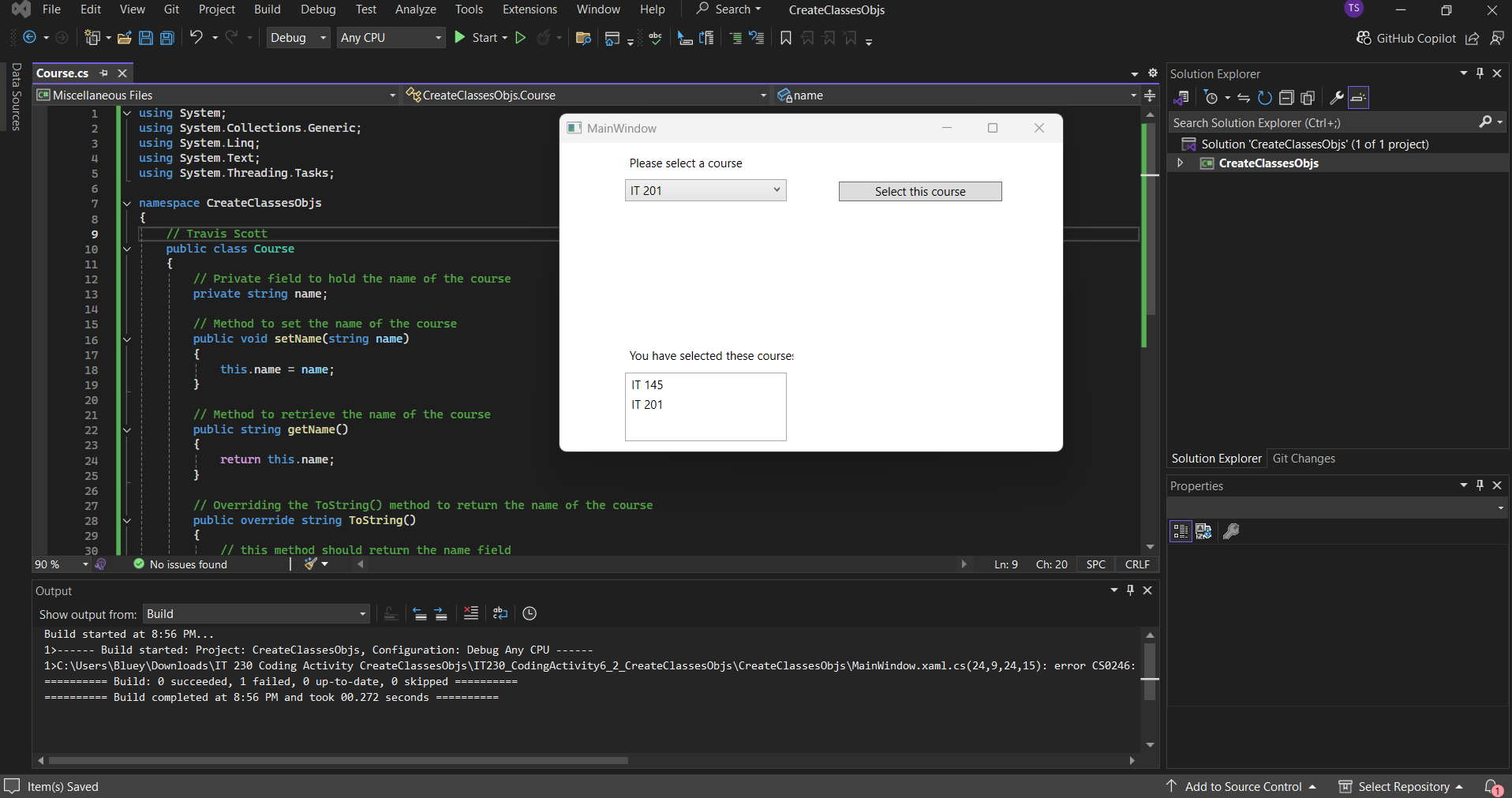
Submit your work on the coding activities for Modules One, Two, Three, Four, and Six in this document. In addition to this document, you should submit a ZIP file containing all your Visual Studio project files and source code that can be run in Visual Studio on a different computer.

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;

namespace CreateClassesObjs

{

// Travis Scott

public class Course

{

// Private field to hold the name of the course

private string name;

// Method to set the name of the course

public void setName(string name)

{

this.name = name;

}

// Method to retrieve the name of the course

public string getName()

{

return this.name;

}

// Overriding the ToString() method to return the name of the course

public override string ToString()

{

// this method should return the name field

return this.name;

}

}

}

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 required the creation of a Course class in C# to be integrated into an existing application. The primary goal was to allow the program to manage and display course names through instances of this class. The process began with identifying the necessary components for the Course class based on the requirements provided. I designed the class with a private string field to hold the course name, ensuring encapsulation. Next, I implemented a method to set the course name (setName) and another to retrieve it (getName). These methods allow the rest of the application to interact with the Course object. To ensure that the course name would be displayed correctly in the UI elements like ComboBox and ListBox, I overrode the ToString() method to return the course name. This design ensures that when a Course object is selected or added to the UI, the correct course name is shown, meeting the application's functional requirements. This approach ensures a clear and logical flow from object creation to display within the application.

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

Through this activity, I gained valuable experience in designing and implementing a class in C#, reinforcing my understanding of object-oriented programming principles such as encapsulation and method overriding. I learned how to effectively integrate a custom class within a WPF application, allowing the class to interact seamlessly with UI elements. This task highlighted the importance of designing methods that not only manage internal data but also present that data appropriately when interfacing with other components of an application. Overall, this experience deepened my understanding of how to structure code to create flexible, maintainable, and user-friendly software.