IT 230 Final Project Part II Guidelines and Rubric

Overview

Real-world programmers are required to edit existing code, work from software requirement documentation, and code from scratch. This final assessment is designed to incorporate all three of those functions.

For your final assessment, you will be given a software requirements list for a final program. You will also be given partially completed source code. You will review the provided source code and fix any errors, and then complete the program based on the software requirements list.

This assessment addresses the following course outcomes:

- Implement variables in C# that are accurate according to the differences in data types
- Create C# code that accurately performs basic math operations
- Incorporate logical operators and Boolean logic in C# to ensure proper support of code components
- Implement functional control structures using C# to provide flow control in software projects
- Implement appropriate classes and methods in C# for creating functionality in software projects

This project is divided into two parts and has three milestone assignments to scaffold learning and ensure quality final submissions. **Part II** has **one milestone,** which will be submitted in **Module Six, with the final submission in Module Seven.**

The milestone assignments are not submitted as part of the final project in their entirety, but the coding techniques, code snippets, and C# classes that you build in these milestones are used in the final project. The milestone assignments build code base that is used to complete the final project. You will not be able to complete the final project unless you complete these formative milestones and receive feedback from the instructor.

Prompt

Overview: The attached zip file (called WPFRegisterStudent.zip) contains the entire Visual Studio project for a WPF program that does compile and does run, but does not produce the required results.

This WPF program is the graphical user interface (GUI) counterpart to the ConsoleRegisterStudent program you completed in Final Project: Part I.

The program enables students to register for courses in a term of study. Students select from a menu of courses for which they wish to register.

The assumptions used by the program are:

- Each course carries three credit hours.
- The program terminates only when the student requires it.

The program must follow these registration business rules:

- No registration more than once for the same course.
- No registration for more than nine credit hours (e.g., no more than three courses).

The program validates the user menu selection, and if valid, registers the student for the selected course. Otherwise, the program outputs an error message. The program then outputs the current list of registered classes. Additionally, the program should output the cumulative total credit hours the student has registered for thus far.

You have been hired to complete the source code of this program by creating additional C# code in the button_Click() event hander according to these requirements:

- Validate the user selection against the above business rules.
- Output an error or a registration confirmation message based on your validation of user selection.
- Update the total credit hours textbox if a registration is confirmed for a selected course.

Once completed, the program output should look like this video, which provides an example of the outcome for a properly functioning program. (Rather than viewing a still image, the video demonstrates the interaction between the application and the user.)

For this assignment, you will submit one deliverable:

I. Project Files: a ZIP file containing:
   a. All of your Visual Studio project files that it can be loaded and run on a separate machine
   b. A copy of your source code
   c. A screenshot of your output

Use the following naming convention for your Word doc:

Lastname_firstname_helloyourname.doc

Given the provided source code and software requirements list, accomplish the following:

Part II:

II. Create additional (source) code.

   A. Add required user **interface** components.

      1. **Code Variables:** Code the required user interface (UI) variables ensuring that variables are defined with proper conventions.
      2. **Boolean Logic:** Implement Boolean statements using proper Boolean logic.
      3. **Map User Input:** Collect appropriate user input and map it properly to branching operations.
      4. **Create UI Code:** Create user interface (UI) code as required.

   B. Implement the required **mathematical** component.

      1. **Program Math Variables:** Code the required math variables ensuring that variables are defined with proper conventions.
      2. **Program Math Operations:** Program the required math operations with no syntax or computation errors.
      3. **Program Math Methods:** Create the method(s) required to support the mathematical component.

   C. Implement the **library** component as described in the software requirements.

      1. **Create Variables:** Create additional variables and associated C# code to support the software requirements, ensuring the variables are defined with proper conventions.
      2. **Add Branching:** Add additional branching code to support flow control.
      3. **Program Library Component:** Program the required math operations related to the library component.
      4. **Incorporate Logic Operators:** Incorporate the required logic operators to support the library component.

Milestones

Final Project Part II, Milestone One Coding Activity: *Create Classes for Final Project*
In **Module Six**, you will be provided with functioning, but incomplete, code and specific guidelines for creating a class to develop the intended functionality of the code. See the Module Six Coding Activity CreateClasses Instructions document for more information. **This milestone is graded with the Coding Activity Rubric.**

Final Project Part II Submission: *WPFRegisterStudent*
In **Module Seven**, you will submit the second part of your final project. For this assignment, you will write new code and complete Section II of the critical elements above. **This submission is graded with the Final Project Part II Rubric (below).**

Southern New Hampshire University

Final Project Part II Rubric

Guidelines for Submission: Submit your completed C# program with modified code for your instructor's evaluation.

Critical Elements	Exemplary	Proficient (100%)	Needs Improvement (55%)	Not Evident (0%)	Value
Interface: Code Variables [IT-230-01]		Codes all required UI variables with proper conventions and no generated errors	Codes some, but not all, required UI variables, does not use proper conventions, or segment(s) of code generate errors	Does not code required UI variables	6.66
Interface: Boolean Logic [IT-230-03]		Implements Boolean statements using proper Boolean logic that have no syntax errors	Implements Boolean statements but with improper Boolean logic or syntax error(s)	Does not implement Boolean statements	10
Interface: Map User Input [IT-230-04]		Properly collects user input and maps it to branching operations	Collects and maps user input to branching operations, but does not properly collect user input or contains gaps in proper mapping to branching operations	Does not collect user input and map it to branching operations	10
Interface: Create UI Code [IT-230-05]		Comprehensively creates UI code without error	Creates UI code but with error(s) or missing component(s)	Does not create UI code	10
Mathematical: Program Math Variables [IT-230-01]		Codes required math variables with proper conventions and no generated errors	Codes required math variables, but does not use proper conventions or segment(s) of code generated errors	Does not code required math variables	6.67
Mathematical: Program Math Operations [IT-230-02]		Programs required math operations with no syntax or computation errors	Programs required math operation code but is missing required math operation(s) or math operation(s) produce incorrect results	Does not program required math operation code	10
Mathematical: Program Math Methods [IT-230-05]		Creates math methods required to support mathematical component that are properly programmed and produce correct results	Creates math methods required to support mathematical component but with gaps in proper coding or mathematically accurate results	Does not create math methods required to support mathematical component	10

Library: Create Variables [IT-230-01]		Code all additional required variables and associated C# code to support the software requirements with proper conventions and no generated errors	Codes additional required variables and associated C# code to support the software requirements, but does not use proper conventions or segment(s) of code generate errors	Does not code additional required variables and associated C# code to support the software requirements	6.67
Library: Add Branching [IT-230-04]		Adds additional branching that supports flow control without generating errors	Adds additional branching code but is missing branching code(s) to properly support flow control or code generates error(s)	Does not add additional branching code	10
Library: Program Library Component [IT-230-02]		Programs all math operation code in the library component so that there are no syntax or computation errors and math operations work properly	Programs math operations in the library component, but math operation(s) in the library component are missing or math operation(s) produce incorrect results	Does not program math operations in the library component	10
Library: Incorporate Logic Operators [IT-230-03]		Incorporates required logic operators to support the library component without generating errors	Incorporates logic operators to support the library component but is missing logic operation(s) or logic operator(s) result in error or incorrect results	Does not incorporate logic operators to support the library component	10
				Total	**100%**