Southern New Hampshire University

IT 511 Stepping Stone Labs Two Through Six Rubric

Overview: Beginning with Stepping Stone Two in Module Three, you will complete a series of related stepping stone labs that will help you build object-oriented programming skills that relate to your final project. Be sure to incorporate feedback from your instructor as you develop your work through the stepping stone labs and develop your final project.

Note that Stepping Stone Lab One: Pseudocode is graded with a separate rubric.

Stepping Stone Labs Two through Six specifics:

Stepping Stone	**Module**	**Title/Topic**	**Task Specifics**
Two	Three	Data Types	Produce basic data types for a recipe manager program.
Three	Four	Validating Input With Branches	Develop a branched structure for a recipe manager program.
Four	Five	Entering Ingredients With Loops	Develop iterative loops for a recipe manager program.
Five	Six	Collection and Item Class With Accessors and Mutators	Develop the first version of the Recipe class and create a test class.
Six	Eight	The RecipeBox Driver Application	Produce a driver application for a recipe manager program.

Each of your programming assignments includes two parts:

- Code file(s)
- Written reflection as text file or Word document

The assignment parts are submitted together with the written submission and with the code as a file attachment.

Critical Elements

The following **critical elements** should be addressed in your project submission:

I. **Code Reflection**
A brief explanation of the code, its purpose, and a brief discussion of your experience in developing it, including any issues that you encountered while completing the stepping stone and what approaches you took to solve them

II. **Specifications**
Source code must meet its specifications as defined by the data and problem. However, this may require multiple attempts or iterations. You will be given credit for code that is well on its way to meeting specifications or solving the problem.

III. **Correctness**
Source code must behave as desired. While correct code produces the correct output as defined by the data and problem, for the stepping stones you will receive credit for producing fully functioning code (producing no errors) that aligns with as many of the specifications as possible. Note: You should write your code in such a way that the submitted files execute, even if they do not produce the correct output.

IV. **Readability**
Code needs to be readable to a knowledgeable programmer. In this course, readable code requires the following:

- Consistent, appropriate white space (blank lines, spaces) and indentation to separate, distinct parts of the code and operations
- Explicit, consistent variable names that clearly indicate the data they hold and are formatted consistently
- Organized structure—clear design separating components with different responsibilities

V. **Annotation**
All code should also be well-commented. This is a practiced “art” that requires striking a balance between commenting everything, which adds a great deal of unneeded noise to the code, and commenting nothing. Well-annotated code requires you to do the following:

- Explain the purpose of lines or sections of your code, detailing the approach and method you took to achieve a specific task in the code.
- Document any section of code that is producing errors or incorrect results.

Rubric

Your assignment should be submitted as a zip file of the exported code and reflection text, as required.

Note that, although the stepping stone labs are graded, their main purpose is to provide useful practice and feedback that you can incorporate as you build the knowledge and skills you need to succeed in the final project.

Critical Element	Proficient (100%)	Needs Improvement (70%)	Not Evident (0%)	Value
Code Reflection	Describes purpose of the code, techniques implemented to solve the problem, challenges encountered, and the approaches to overcome the challenges	Lacks details of code purpose, techniques implemented, or challenges encountered	Does not describe purpose of code, techniques used, or challenges encountered	20
Code Requirements	All or most algorithm specifications are fully met	Details of the specifications are not met in significant instances	The program does not meet the specifications	20
Code Correctness	The program functions as designed in most cases	The program functions as designed in limited cases	The program does not function as designed	20
Code Readability	Code follows proper syntax and demonstrates deliberate attention to spacing, white space, and variable naming	Code contains variations from established syntax and conventions	Code exhibits consistent and significant variations from established syntax and conventions	20
Annotation	Code annotations explain and facilitate navigation of the code	Code annotations are incomplete or provide insufficient assistance with understanding the code	Code annotations do not explain the code, do not facilitate navigation of the code, or are not present	20
			Total	**100%**