Southern New Hampshire University

IT 145 Module Two Assignment Guidelines and Rubric

Overview: This assignment will allow you to use pseudocode to implement the program to see the value in planning first, coding later. While coding is the glamorous part of the job, software development is a process with many steps. The program itself will focus on operators to complete an equation.

Prompt: Before completing this assignment, be sure to review the Module Two resources. Use the Guide to Pseudocode document to write out the pseudocode for the given problem:

Imagine that you are a merchant and need to keep better tabs on your merchandise to know when to reorder supplies. First, write out pseudocode, and then create a program to help you by accomplishing the following tasks:

- Use command line interface to ask the user to input the following. (You will need to convert this from a string to an integer.)
    - How many apples are on hand
    - How many apples should be in stock
    - How many oranges are on hand
    - How many oranges should be in stock
- Perform an operation to determine how many of each item should be ordered to maintain the stock.
- Use command line interface to output the number of apples and oranges that need to be ordered.

Once you have the program laid out, build it by creating a functional program. Use your pseudocode as a guide. Be sure to remember the following important items:

- Follow the style guidelines found in ZyBooks as you develop.
- Use commenting to describe the code.
- Practice debugging if you encounter errors.
- Ensure your program performs its intended function.

Specifically, the following **critical elements** must be addressed:

I. **Documentation: Pseudocode:** Break down the problem statement into programming terms through creation of pseudocode following the guidelines provided in the course.
II. **Functioning Code:** Produce fully functioning code (a code that produces no errors) that aligns with the accompanying annotations.
III. **Code Results:** Results are properly generated.
    A. Code results generate **accurate output**.
    B. Code results produce results that are streamlined, **efficient**, and error free.

IV. **Comments:** All code should 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.
   A. **Explain the purpose** of lines or sections of your code, detailing the approach and method the programmer took to achieve a specific task in the code.

V. **Style and Structure:** Part of the lesson to be learned in this course is how to write code that is clearly readable and formatted in an organized manner.
   A. Develop **logically organized** code that can be modified and maintained.
   B. Utilize proper **syntax**, style, and language conventions and best practices.

Guidelines for Submission: Submit your pseudocode as a DOC or DOCX file. Submit the program as a JAVA file.

Instructor Feedback: This activity uses an integrated rubric in Blackboard. Students can view instructor feedback in the Grade Center. For more information, review these instructions.

Rubric

Critical Elements	Exemplary (100%)	Proficient (85%)	Needs Improvement (55%)	Not Evident (0%)	Value
Documentation: Pseudocode	Meets "Proficient" criteria and demonstrates thorough understanding of the creation of pseudocode	Breaks down the problem statement into programming terms through the creation of pseudocode	Breaks down the problem statement into programming terms through the creation of pseudocode, but pseudocode contains inaccuracies or explanation is illogical or incomplete	Does not break down the problem statement into programming terms through the creation of pseudocode	10
Functioning Code	Produces fully functioning code (a code that produces no errors) that aligns with the accompanying annotations		Produces partially functioning code (a code that produces almost no errors) that partially aligns with the accompanying annotations	Does not produce functioning code	20
Code Results: Accurate Output	Meets "Proficient" criteria and the code is capable of producing accurate results beyond the specifications of the given problem	Generates code results with accurate output	Generates code with incorrect results for the given problem	Does not produce results for the given problem	15
Code Results: Efficiency	Meets "Proficient" criteria and includes sophisticated techniques such as error handling or reference to user-created functions	Produces code results that are streamlined, efficient, and error free	Produces results that are minimally inefficient (e.g., multiple minor occurrences of convoluted code; alternative code element would achieve results in a simpler manner)	Does not produce code results that are streamlined, efficient, and error free	20

Southern New Hampshire University

Comments: Explanation of Purpose	Meets "Proficient" criteria and clarity of annotations facilitates code navigation for a varied audience; code is written in a concise manner	Explains the purpose of lines or sections of the code, detailing the approach and method the programmer took to achieve a specific task in the code	Explains the purpose of lines or sections of the code, detailing the approach and method the programmer took to achieve a specific task in the code, but explanation has inaccurate and/or missing details	Does not explain the purpose of lines or sections of the code	20
Style and Structure: Logically Organized Code	Meets "Proficient" criteria and demonstrates deliberate attention to spacing, whitespace, and variable naming	Develops logically organized code that can be modified and maintained	Develops logically organized code that can be modified and maintained but with portions that are not logically organized	Does not develop logically organized code that can be modified and maintained	5
Style and Structure: Syntax	Meets "Proficient" criteria and demonstrates an understanding of why certain techniques are considered best practices	Utilizes proper syntax, style, and language conventions and best practices	Utilizes proper syntax, style, and language conventions and best practices, but with some errors	Does not utilize proper syntax, style, and language conventions and best practices	10
				Total	**100%**