Final Project Rubric

Scalable Data Infrastructures: MDV2330

Bare Minimum Requirements

These requirements must be satisfied before any points are awarded. Failing to meet these requirements will result in a zero (0) grade.

1. Working C# file with no major syntax errors and no runtime errors.
2. You must submit the whole project folder and not just the .cs file.

Topic	%	Excellent (100%)	Good (75%)	Fair (30%)	Poor (0%)
Technical					
Naming	5%	The submitted files follow the correct naming convention of Lastname_Firstname_FinalProject			Files are not named properly.
Programming Fundamentals					
User Item List	10%	User item list is prompted and validated correctly in the Main Method.	User item list is prompted and validated in the Main Method, but there are minor errors.	There are major errors in prompting the user for item list.	User is not prompted and item list is hard-coded.
TextToArray Function	15%	TextToArray function is setup correctly, accepts a text string as a parameter and returns a string array to the Main.	TextToArray function is setup correctly, accepts a text string as a parameter and returns a string array to the Main, but there are minor errors.	There are major errors in the TextToArray function.	TextToArray function does not function or exist.
PromptForCosts Function	15%	PromptForCosts function is setup correctly, accepts a string array as a parameter and returns a decimal array to the Main.	PromptForCosts function is setup correctly, accepts a string array as a parameter and returns a decimal array to the Main, but there are minor errors.	There are major errors in the PromptForCosts function.	PromptForCosts function does not function or exist.
User Prompted Costs	15%	User is prompted in the PromptForCosts Function for the cost of each event. Cost is validated, converted to the correct data type and stored in an array.	User is prompted in the PromptForCosts Function for the cost of each event. Cost is validated, converted to the correct data type and stored in an array. However, there are minor errors.	There are major errors in prompting the user for costs.	User is not prompted for costs of the items or they are hard-coded.
SumOfCosts Function	15%	SumOfCosts function is setup correctly, accepts a decimal array as a parameter and returns a decimal variable.	SumOfCosts function is setup correctly, accepts a decimal array as a parameter and returns a decimal variable. However there are minor errors.	There are major errors in the SumOfCosts function.	SumOfCosts function does not function or exist.
Final Result	20%	Final Total is correct and formatted correctly.	Final Total is correct but is not formatted correctly.	Final result is missing or is not correct.	
Test Values	5%	Test values are present in a multi-lined comment at the end of each section, contain the required values to test and are all correct.			No test values are present

Activity: Final Project

OVERVIEW:

For this last assignment we will be combining everything that you have learned this month into one final project that will test all of your skills.

LEVEL OF EFFORT:

This activity should take approximately 240 to complete. It will require:

- 0m Research
- 15m Prep & Delivery
- 225m Work

If you find that this activity takes you significantly less or more time than this estimate, please contact me for guidance.

READING & RESOURCES:

Final Project - Rubric (necessary)
The rubric on the first page of this document outlines the points for the assignment.
Make sure you check off each one as done before submitting your assignment!

INSTRUCTIONS:

1. Before you begin, you should read the rubric on page 1. This is extremely important, as it will tell you exactly how this assignment will be graded.

2. Create a project called **Lastname_Firstname_FinalProject**.

3. In this assignment you will have the following objectives:
    a. Add comments that include your name, term, assignment name and class.
    b. Create a ReadLine in the Main method to accept input from the user.
    c. Create a multiple functions that will solve the problem. All functionality to solve the problem should be contained in these methods as described below.
    d. Final output must be in the Main method.

4. How you solve the problem is not as important as finding a way to do it, so do not feel limited in what you're allowed to do.

5. The only limitation is that you cannot go outside the scope of this course; that is, you **cannot** use anything outside the scope of this class to solve the problem.
    a. If do use anything outside of this class as part of your solution, you will receive a 0.

6. Place your name, date, and assignment at the top of your code in a multi-lined comment.

7. Make sure to comment every important line of code so that you are explaining exactly what you are trying to do.

8. Your code should give the user meaningful output. So, after your calculations are complete, your code should report back to the user the final values with a Console.WriteLine().
    a. This should contain the variables that you calculated and a concatenation text string that describes the value.
    b. e.g. Console.WriteLine ("The area of the rectangle is " +calcArea+ "!");

9. Zip your whole project folder and upload this file to FSO.

Item List Check Out

INSTRUCTIONS:

1. You will be asking the user for a comma-separated list of events that they wish to buy tickets for.
    a. Validate this to make sure it is not left blank.
    b. Send this text string into the custom function **TextToArray.**

2. Create a custom function named **TextToArray** that will split apart the user's list text string into an array of individual events.
    a. Make sure to remove any spaces before or after the event itself.
    b. Return this array of events to be purchased to the Main method.

3. Once you have the string array of events back in your Main Method create a 2nd custom function called **PromptForCosts**.
    a. This function should accept the string array of events.
    b. Inside of this function create an array to hold the costs of these events.
        i. Note the length of this Array or ArrayList must be depended on how many events the user typed in. AKA do NOT hard-code this.
    c. Loop through the events array and **prompt the user** for the cost of each event in the array.
        i. Validate that the user is typing in a valid response.
        ii. Once it is validated, store this cost inside of the cost array.
    d. After you get each cost, Return this array to the Main.

4. After catching the returned cost Array, create one last custom function called **SumOfCosts**.
    a. This array should accept the cost array as a parameter.
    b. Inside of this function create a variable to hold the total sum of the events costs.
    c. Loop through each item in the cost Array and add their total to the sum.
    d. Return this sum to the Main method.

5. Use the total sum variable that is return in a final output to the user that is in this format.
    a. **"The total cost of all of your events will be $X."**
        i. Where **X** is a formatted text string that contains the cost rounded to 2 digits.

6. **Test, Test, Test**
    a. Test your code using the data below and at the very least 2 more times and put your Test Results in a multi-lined comment at the end of your Main Method.

7. When you are finished zip your whole folder and submit it on FSO.

Test Values:

- Event List – “Fiddler On The Roof, Guardians Of The Galaxy 2, Wonder Woman”
  - Fiddler On The Roof Cost – 50.00
  - Guardians Of The Galaxy 2 Cost– Twelve Dollars
    - Re-prompted New Cost- 12.00
  - Wonder Woman Cost – 15.00
- Final Results - **“The total cost of all of your events will be $77.00.”**

TURNING IT IN:

- Double-check that you’ve commented your code (You can’t comment too much).
- Compress your **Lastname_Firstname_FinalProject** folder into one zipped file. It should be named **Lastname_Firstname_FinalProject.zip**
- Upload this zipped file to FSO. This is the file I will unzip and run to verify it works and review your code.
- You must zip the whole folder and not just the one individual C# file. If you only submit a .cs file you will get a **zero** for the whole project.

Don’t Forget:

Make sure your project follows this list of criteria:

- The result should appear in the console and include an explanation of the result.
  - **Good example of console print out:** The volume of the sphere is 26 feet cubed.
  - **Bad example of console print out:** 26
- Final output should use string concatenation.
- Comment every line of code (describe what each line is doing in English). Do NOT just label sections of your code.