Discount Calculator in C#

Introduction

Welcome to the C# programming lesson on creating a Discount Calculator. In this tutorial, we'll explore the fundamental concepts of user input handling, conditional statements, and basic arithmetic operations to design a practical application. The Discount Calculator prompts users to input the original price of an item and the discount percentage they wish to apply. Using if-else statements, the program then calculates the final price after the discount and displays both the original and discounted prices. This lesson serves as a hands-on opportunity to sharpen your skills in C# and gain practical experience in building a useful and interactive program.

This journey isn't just about calculating discounts; it's about:

- Building your C# skills: Solidify your understanding of user input, conditional statements, and mathematical operations within the C# language.
- Boosting your problem-solving abilities: Break down the discount calculation into steps, design your program's logic, and overcome any coding challenges you encounter.
- Opening doors to new possibilities: Imagine applying your newfound skills to calculate sales tax, price markups, or even complex financial formulas!

Objectives

The objectives of this Discount Calculator lesson are to:

1. Understand:

- Grasp the core concept of discounts: Decipher the relationship between original price, discount percentage, and final price, internalizing the mathematical formula involved.
- Navigate C# functionalities: Understand how to gather user input (price and discount) and utilize "if-else" statements to handle discounted and full-price scenarios within your C# program.
- Visualize the program flow: Break down the calculation process into manageable steps, identifying how each element (input, calculation, output) fits into the bigger picture.

2. Learn:

- Harness the power of "if-else" statements: Master the application of these conditional statements to determine if a discount applies, ensuring accurate calculations based on user input.

- Implement mathematical operations: Translate the discount formula into C# code, utilizing built-in operators like multiplication and division to perform the necessary calculations.

- Craft user-friendly interactions: Design your program to ask users for information and display results in a clear and engaging way, making the experience intuitive and interactive.

3. Practice:

- Roll up your sleeves and code!: Translate your understanding and knowledge into action by actively writing C# code to build your discount calculator step by step.

- Experiment and refine: Test your program with different input values, identify and fix errors, and iterate on your design to enhance its functionality and clarity.

- Challenge yourself: Explore variations of the calculator, like handling multiple discounts, incorporating tax calculations, or even building a tip calculator using similar principles.

4. Apply:

- Showcase your creation: Share your completed discount calculator with classmates, friends, or family, demonstrating your newly acquired C# skills and understanding of discount calculations.

- Extend the concept: Utilize the core principles learned in this lesson to tackle other real-world calculations that involve percentages, user input, and conditional statements.

- Become a C# coding master!: Remember, this is just the beginning of your coding journey. Keep practicing, exploring, and applying your skills to new projects and challenges, expanding your coding mastery step by step.

By focusing on these objectives, we will gain a solid understanding of discounts, practice essential programming concepts in C#, and apply our knowledge to create a functional and user-friendly Discount Calculator.

Source code example

```
using System;

namespace DiscountCalculatorApp
{
    class Program
    {
        static void Main()
        {
            Console.WriteLine("iNetTutor.com");
            Console.WriteLine("Welcome to the Discount Calculator!");

            // Prompt user for the original price
            Console.Write("Enter the original price: ");
            if (double.TryParse(Console.ReadLine(), out double originalPrice))
            {
                // Prompt user for the discount percentage
                Console.Write("Enter the discount percentage: ");
                if (double.TryParse(Console.ReadLine(), out double discountPercentage))
                {
                    // Calculate discounted price
                    double discountAmount = (discountPercentage / 100) * originalPrice;
                    double discountedPrice = originalPrice - discountAmount;

                    // Display results
                    Console.WriteLine($"Original Price: ${originalPrice}");
                    Console.WriteLine($"Discount Percentage: {discountPercentage}%");
                    Console.WriteLine($"Discounted Price: ${discountedPrice}");

                }
                else
                {
                    Console.WriteLine("Invalid discount percentage. Please enter a valid number.");
                }
            }
            else
            {
                Console.WriteLine("Invalid original price. Please enter a valid number.");
            }

            Console.WriteLine("Press any key to close the application.");
            Console.ReadKey();
        }
    }
}
```

Explanation

The provided source code is a C# console application that calculates the discounted price of a product based on the original price and the discount percentage entered by the user. It prompts the user to enter the original price and the discount percentage, performs the necessary calculations, and displays the original price, discount percentage, and discounted price.

The source code consists of a single class named Program within the DiscountCalculatorApp namespace. Here's a breakdown of the code:

1. Welcoming the User:

- Console.WriteLine("iNetTutor.com"); and Console.WriteLine("Welcome to the Discount Calculator!"); greet the user with a warm welcome message.

2. Gathering User Input:

- Console.Write("Enter the original price: "); and Console.Write("Enter the discount percentage: "); prompt the user to enter the original price and discount percentage.
- double.TryParse() carefully checks if the entered values are valid numbers before proceeding, ensuring a robust and error-proof experience.

3. Calculating the Discount:

- double discountAmount = (discountPercentage / 100) * originalPrice; performs the core calculation, determining the amount of money saved based on the discount percentage.
- double discountedPrice = originalPrice - discountAmount; subtracts the discount from the original price to reveal the final, discounted price.

4. Presenting the Results:

- Console.WriteLine($"Original Price: ${originalPrice}");, Console.WriteLine($"Discount Percentage: {discountPercentage}%");, and Console.WriteLine($"Discounted Price: ${discountedPrice}"); display the calculated values in a clear and informative manner, empowering the user with financial clarity.

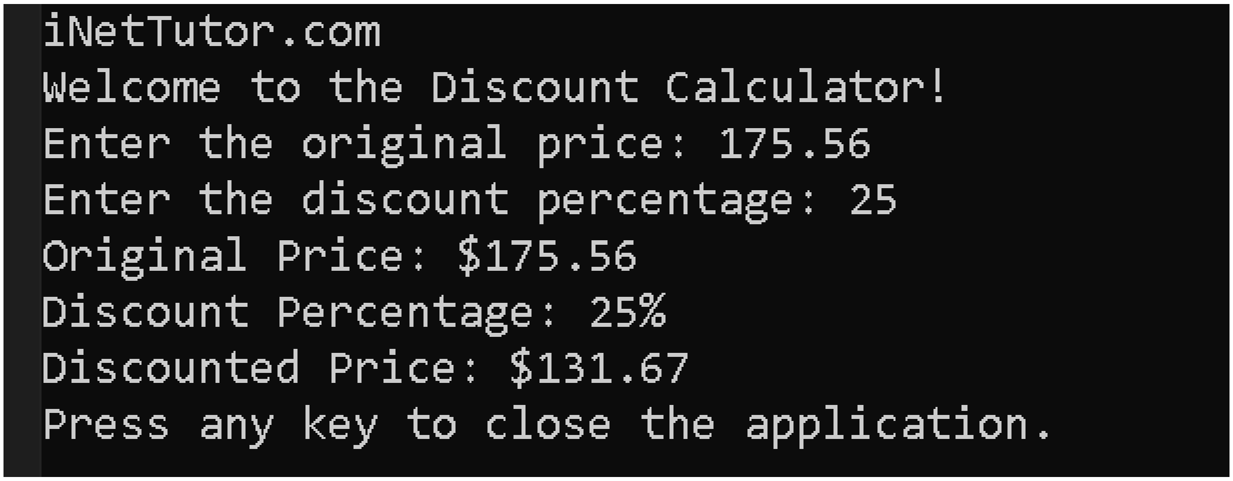
5. Handling Errors Gracefully:

- If the user enters invalid input (like letters instead of numbers), the code provides helpful error messages to guide them towards correct usage, ensuring a smooth and user-friendly experience.

6. Closing with a Pause:

- Console.WriteLine("Press any key to close the application."); and Console.ReadKey(); create a pause, allowing the user to review the results before ending the program, providing a sense of closure and control.

Output



```
iNetTutor.com
Welcome to the Discount Calculator!
Enter the original price: 175.56
Enter the discount percentage: 25
Original Price: $175.56
Discount Percentage: 25%
Discounted Price: $131.67
Press any key to close the application.
```

Summary

The provided source code showcases a C# console application that calculates the discounted price of a product based on the original price and the discount percentage entered by the user. The code prompts the user to input the original price and discount percentage, validates the input, performs the necessary calculations, and displays the original price, discount percentage, and discounted price. It demonstrates key programming concepts such as user input handling, data validation, mathematical calculations, and output formatting.

To improve the source code, one can engage in exercises and a lab exam. Exercises may include performing a code review to identify potential issues, solving algorithmic problems to enhance problem-solving skills, and practicing coding exercises to strengthen understanding of programming concepts. A lab exam can be designed to test students' understanding of the code and their ability to identify and fix potential issues, covering areas such as input validation, error handling, code optimization, and best practices. These activities aim to enhance coding skills, improve code quality assessment, and prepare individuals for real-world programming challenges.

Exercises and Assessment

By engaging in these exercises and lab exams, you will gain practical experience in code review, algorithmic problem-solving, and applying best practices. These activities will help you improve your coding skills, enhance your understanding of code quality, and prepare you for real-world programming challenges.

Exercises:

1. Enhance User Experience:

   - Modify the program to accept input for the item name along with the original price and discount percentage. Display the item name in the output for a more personalized experience.

2. Add Multiple Discounts:

   - Extend the program to allow users to apply multiple discounts successively. Prompt users for additional discount percentages until they choose to stop. Calculate and display the final price after all discounts.

3. Create a Discount Class:

   - Refactor the code by creating a Discount class. Encapsulate the discount-related logic within this class, making the main program more modular and organized.

Lab Exam:

Scenario: Department Store Discounts

You are tasked with building an advanced version of the Discount Calculator for a department store. The store offers different types of discounts based on membership levels: Bronze, Silver, and Gold. Each membership level provides a specific percentage discount on the original price.

Requirements:

1. Implement a membership system where users can choose their membership level (Bronze, Silver, Gold) and input the original price.

2. Based on the chosen membership level, apply the corresponding discount percentage.

3. Display the original price, membership level, discount percentage, and the final discounted price.

Grading Criteria:

- Correct implementation of the membership system.

- Accurate application of the discount based on the chosen membership level.

- Clear and concise user prompts and output.

- Effective error handling for invalid inputs.

- Code organization and readability.

Meta Description

Learn C# console application development with this code example. Calculate discounted prices based on user input. Improve your coding skills now!