ATM Program in C# Console

Introduction

Welcome to this C# console program that simulates an ATM (Automated Teller Machine). In this program, we will use if else if statements to implement the different functionalities of an ATM such as withdrawing money, depositing money, and checking account balance.

The program starts by displaying a menu that allows the user to select the desired transaction. The user can enter 'w' or 'W' to withdraw money, 'd' or 'D' to deposit money, or 'i' or 'I' to check their account balance.

Based on the user's input, the program will execute the appropriate if else if statement to perform the selected transaction. If the user selects to withdraw or deposit money, they will be prompted to enter the amount to withdraw or deposit. The program will then calculate the new account balance and display it to the user.

If the user selects to check their account balance, the program will simply display the current balance. If the user enters an invalid input, an error message will be displayed.

Overall, this program provides a simple simulation of an ATM using if else if statements to perform different transactions.

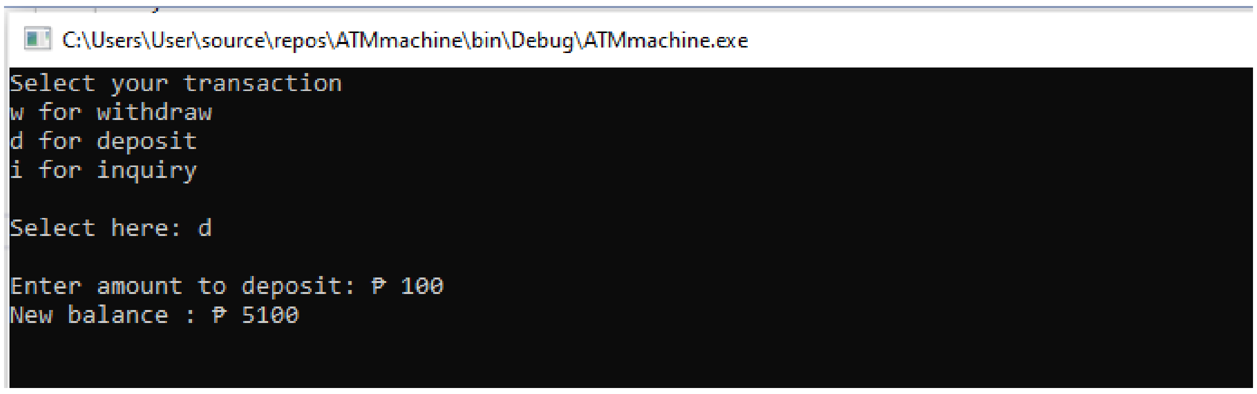
Objectives

1. To provide a basic understanding of how if else if statements can be used to simulate the functionality of an ATM. By implementing different if else if statements for different transactions such as withdrawing money, depositing money, and checking account balance, the program demonstrates how conditional statements can be used to control the flow of the program.
2. To provide a practical example for beginners to learn how to work with console input and output in C#. The program prompts the user to input values for different transactions and displays the results back to the console. This provides an opportunity for beginners to learn how to work with console input and output in C#.

Code Example

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ATMmachine
{
    class Program
    {
        static void Main(string[] args)
        {
            double balance = 5000;
            double newbalance;
            double withdraw;
            double deposit;
            string select;

            Console.OutputEncoding = System.Text.Encoding.Unicode;

            Console.WriteLine("Select your transaction");
            Console.WriteLine("w for withdraw");
            Console.WriteLine("d for deposit");
            Console.WriteLine("i for inquiry");
            Console.WriteLine();
            Console.Write("Select here: ");
            select = Console.ReadLine();
            Console.WriteLine();

            if (select.Equals("w") || select.Equals("W"))
            {
                Console.Write("Enter amount to withdraw : ₱ ");
                withdraw = double.Parse(Console.ReadLine());
                newbalance = balance - withdraw;

                Console.WriteLine("New balance : ₱ " + newbalance);
                Console.ReadKey();
            }

            else if (select.Equals("d") || select.Equals("D"))
            {
                Console.Write("Enter amount to deposit: ₱ ");
                deposit = double.Parse(Console.ReadLine());
                newbalance = balance + deposit;

                Console.WriteLine("New balance : ₱ " + newbalance);
                Console.ReadKey();
            }

            else if (select.Equals("i") || select.Equals("I"))
            {
                Console.WriteLine("balance : \u20B1 " + balance);
                Console.ReadKey();
            }

            else
            {
                Console.WriteLine("Error....!!!!");
            }
        }
    }
}
```

Output:


C:\Users\User\source\repos\ATMmachine\bin\Debug\ATMmachine.exe

```
Select your transaction
w for withdraw
d for deposit
i for inquiry

Select here: d

Enter amount to deposit: ₱ 100
New balance : ₱ 5100
```

Code Explanation

Line 13-17

This part of the program declares and initializes five variables that will be used to store the current balance, new balance, amount to withdraw, amount to deposit, and user's selected transaction.

Line 13

This line declares a double variable named 'balance' and initializes it to 5000. This variable stores the current balance in the user's account.

Line 14

This line declares a double variable named 'newbalance' without initializing it. This variable will be used to store the new balance after a transaction such as a withdrawal or deposit.

Line 15

This line declares a double variable named 'withdraw' without initializing it. This variable will be used to store the amount to withdraw from the user's account.

Line 16

This line declares a double variable named 'deposit' without initializing it. This variable will be used to store the amount to deposit into the user's account.

Line 17

This line declares a string variable named 'select' without initializing it. This variable will be used to store the user's selected transaction ('w' or 'W' for withdrawal, 'd' or 'D' for deposit, 'i' or 'I' for inquiry).

Line 19

The purpose of the code Console.OutputEncoding = System.Text.Encoding.Unicode; is to set the encoding of the console output to Unicode. Unicode is a character encoding standard that represents almost all of the written languages and scripts in the world, which means that it can display a wide range of characters, including non-English characters and special characters, such as the "₱" symbol used in the program.

Setting the console output encoding to Unicode ensures that the program can correctly display the "₱" symbol and other special characters on the console output. If the encoding is not set to Unicode, the console may not be able to display certain characters correctly, which can result in garbled or incorrect output.

By setting the output encoding to Unicode, the program ensures that the console output is correctly encoded and can display a wide range of characters, making it more user-friendly and accessible to users who speak different languages or use non-standard characters.

Line 21-24

This part of the program is responsible for displaying a menu of available transactions to the user and prompting the user to enter their choice of transaction.

Line 26-27

These lines use the Console class to prompt the user to enter their selection from the available options. The message "Select here: " is printed on the console using the Console.Write method, and the user's input is read and stored in the 'select' variable using the Console.ReadLine method.

Line 30-38

This part of the program uses an if statement to check whether the user selected a transaction to withdraw money from their account.

Line 30

This line checks whether the value of the 'select' variable is equal to "w" (lowercase) or "W" (uppercase), which indicates that the user wants to withdraw money from their account. The '||' operator is used to check whether either condition is true, meaning that the user can enter either uppercase or lowercase "w" to select this option.

Line 32-33

These lines use the Console class to prompt the user to enter the amount they want to withdraw from their account. The message "Enter amount to withdraw : ₱ " is printed on the console using the Console.Write method. The user's input is read using the Console.ReadLine method, which returns a string value. The double.Parse method is then used to convert the string input to a double value, which is stored in the 'withdraw' variable.

Line 34

This line calculates the new balance in the user's account by subtracting the value of the 'withdraw' variable from the 'balance' variable, and stores the result in the 'newbalance' variable.

Line 36-37

These lines use the Console class to display the new balance in the user's account, which is stored in the 'newbalance' variable. The Console.WriteLine method is used to print the message "New balance : ₱ " and the value of the 'newbalance' variable concatenated using the '+' operator. Finally, the Console.ReadKey method is used to pause the program and wait for the user to press any key before continuing.

Line 40-48

This part of the program is executed when the user enters "d" or "D" to select the deposit option. It uses the else if statement to check whether the user input is equal to "d" or "D" and then executes the code inside the block if the condition is true.

The first line of code inside the block prompts the user to enter the amount to deposit using Console.Write(), and displays the "₱" sign as part of the prompt. The amount entered by the user is then read using Console.ReadLine(), and is converted to a double using double.Parse() method.

Next, the code adds the deposit amount to the current balance to calculate the new balance, which is stored in the newbalance variable. The new balance is then displayed to the user using Console.WriteLine(), which again displays the "₱" sign before the new balance.

Finally, Console.ReadKey() is used to pause the program until the user presses any key, allowing the user to view the new balance before the program exits.

Line 50-54

This part of the program is executed when the user enters "i" or "I" to select the inquiry option. It uses the else if statement to check whether the user input is equal to "i" or "I" and then executes the code inside the block if the condition is true.

The first line of code inside the block displays the current balance to the user using Console.WriteLine(), which includes the "₱" sign before the balance. The "₱" sign is displayed using the Unicode character escape sequence \u20B1, which represents the "₱" symbol in Unicode.

The current balance is obtained from the balance variable, which is initialized at the beginning of the program with a default value of 5000. The value of the balance is concatenated with the string literal "balance : \u20B1 " using the + operator.

Finally, Console.ReadKey() is used to pause the program until the user presses any key, allowing the user to view the current balance before the program exits.

Line 56-59

This part of the program is executed if the user enters any input other than "w", "W", "d", "D", "i", or "I". It uses the else statement to execute the code inside the block if none of the previous conditions are met.

The code inside the block simply displays an error message to the user using Console.WriteLine(), informing the user that the input is invalid. The message "Error....!!!!" is displayed as a string literal in the console window.

Since there are no further statements after the Console.WriteLine(), the program terminates immediately after displaying the error message.

Summary

In this lesson, we have created a C# console program to simulate an ATM machine. The program uses the if-else statement to provide three options to the user: withdraw, deposit, or inquiry.

The program prompts the user to select one of these options and reads the user's input using the Console.ReadLine() method. If the user selects withdraw or deposit, the program prompts the user to enter the amount to withdraw or deposit and calculates the new balance accordingly. If the user selects inquiry, the program displays the current balance.

The program also uses the Unicode character escape sequence to display the "₱" symbol in the console window and Console.ReadKey() to pause the program until the user presses any key. Finally, the program displays an error message if the user enters an invalid input.

Overall, this program provides a simple example of how to use the if-else statement, console input/output, and string concatenation in C#.