**Final C# Project:**

**ATM/Banking Application**

ISDS 309 S03

Spring 2024

Professor Hoda Diba

**Prepared By**: ISDS Group #5

Moises Gonzalez

Blake Demarest

Kylie Ebrahimi

Jesus Gutierrez

Sean Smith

Project Summary

The Simple ATM Interface project aims to create a user-friendly interface for conducting basic ATM transactions. It will include a login screen for user authentication and a transaction screen for performing common ATM operations. Our project focuses on a few fundamental components, including user authentication and transaction execution. The login screen serves as the gateway to the banking system, ensuring that only authorized users can access their accounts. Once authenticated, users are granted access to a transaction screen. From checking their account balance to depositing funds, withdrawing, and transferring money, our interface offers services tailored to meet the needs of our banking customers.

Project Detail Page

The Simple ATM Interface program offers users a seamless and intuitive banking experience through an interactive Windows Forms interface. With a focus on user convenience and security, each customer is assigned a unique username and password, securely stored in a designated file. Upon successful authentication, users gain access to a comprehensive range of banking functionalities, including checking their account balance, depositing funds, withdrawing cash, and transferring funds between their accounts.

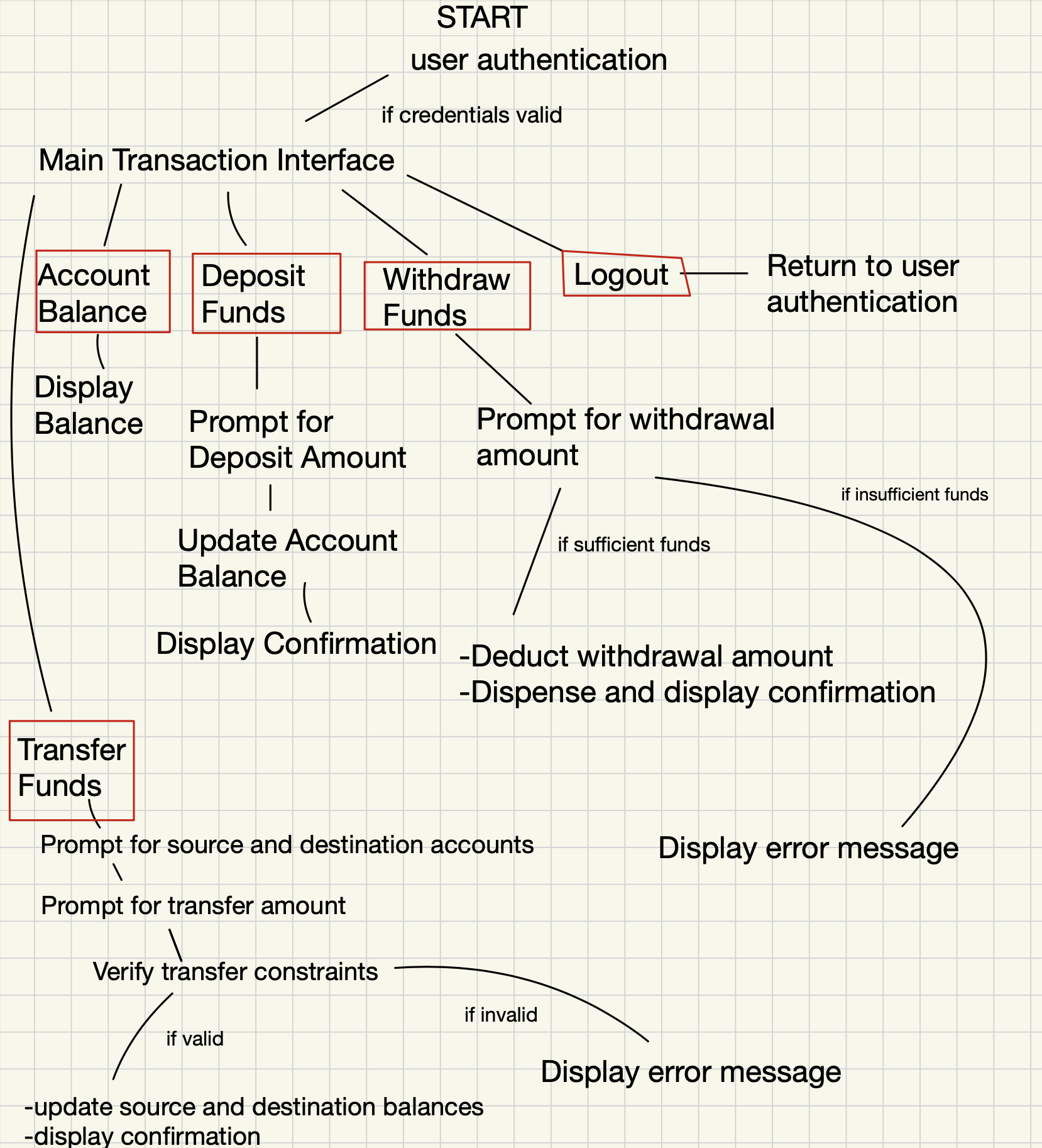
Behind the scenes, the program utilizes advanced data management techniques to ensure efficient and secure access to user information. By establishing a stream to the specified file, the program effectively organizes user data into arrays, enabling swift retrieval and manipulation of relevant information tied to each individual's username. This streamlined approach not only enhances user experience but also reinforces data privacy and security measures.

The program's transactional capabilities are straightforward and user-friendly. Deposits increase the user's available balance by adding the deposited amount to the relevant array and updating the file accordingly. Conversely, withdrawals deduct funds from the user's balance, ensuring accurate and real-time account management. Transfers between accounts are seamlessly executed, with the program manipulating both source and destination arrays to reflect the transaction accurately in the file.

Users receive a comprehensive banking receipt generated in a log file format after completing their transactions. This receipt includes essential details such as the user's username, the timestamp of their platform access, and a record of the tasks they completed during their session. By logging this information and generating receipts, the program enhances transparency and accountability, providing users with a tangible record of their banking activities.

In summary, the Simple ATM Interface program represents a modern approach to banking, combining user-centric design principles with data management techniques. By prioritizing simplicity, security, and accountability, the program empowers users to manage their finances confidently and conveniently, ushering in a new era of digital banking excellence.

Logic Flow Chart



User Guide Pages

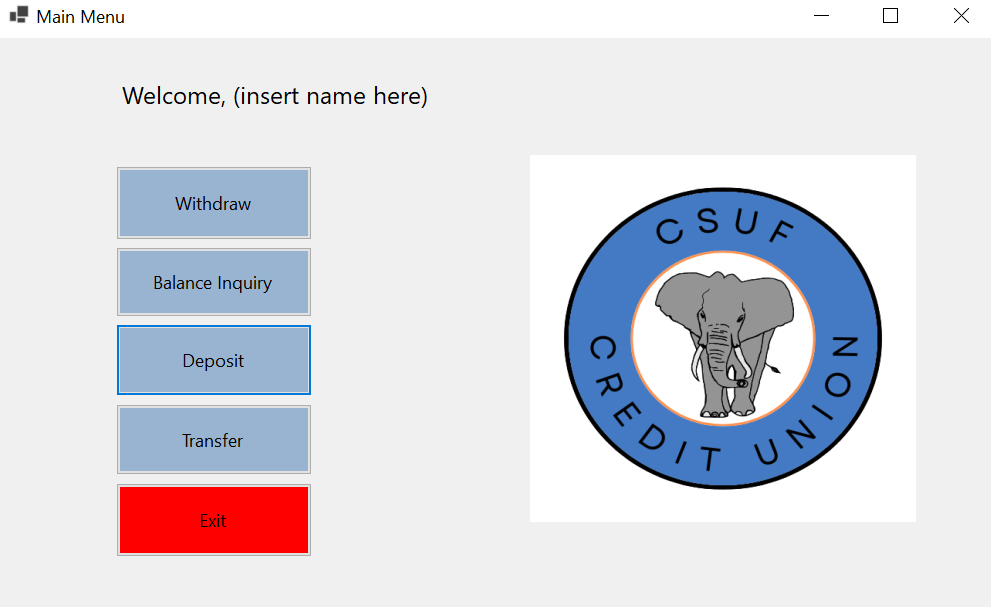
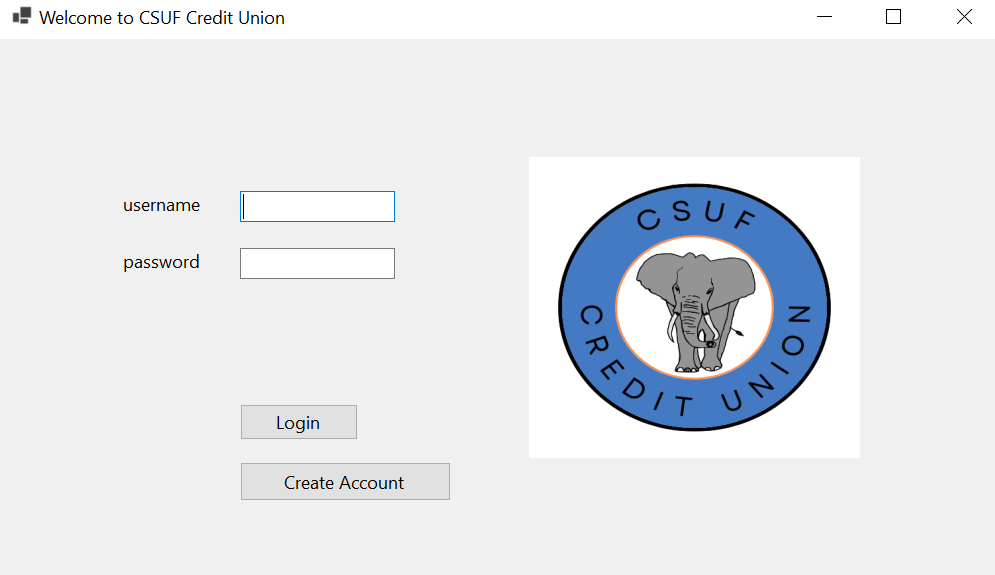
1. Logging In

-The program prompts users to enter both their username and password on the login screen.

-User input is validated against the stored usernames and passwords in a designated text file.

-If the provided credentials match any existing records, the program retrieves the user's information using the GetUserInformation command.

-The program then proceeds to utilize only the attributes associated with the authenticated user throughout the session, ensuring privacy and security.



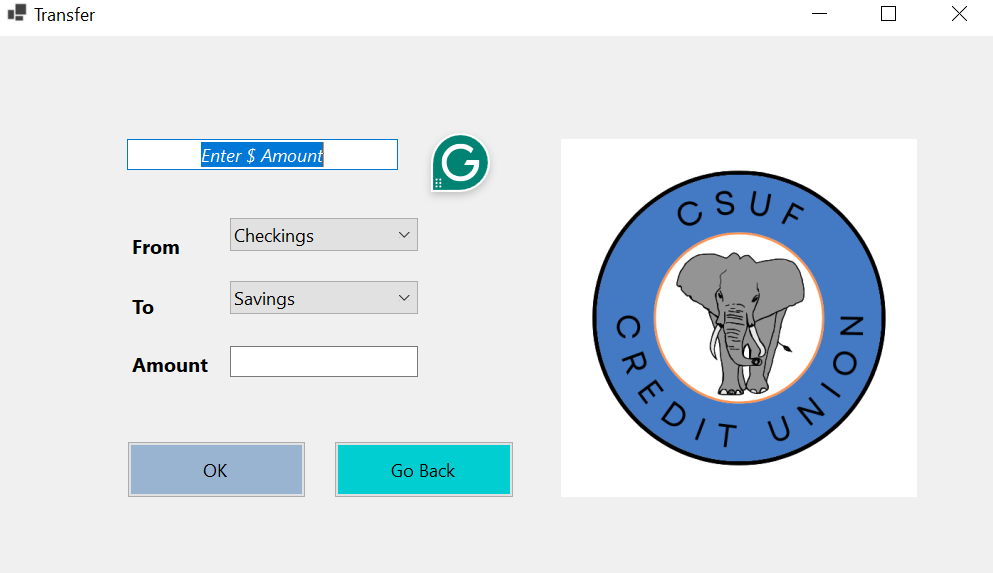
1. Transfers

-Users initiate transfers by selecting the appropriate option from the transaction menu.

-The transfer function employs nested if-else statements to validate user inputs and manage error control.

-Parameters such as transferring to the same account or exceeding available funds are checked.

-If the transfer meets the correct parameters, the program updates the available balances in both accounts and displays a success message along with the new remaining balance.



1. Deposits

-Users initiate deposits by selecting the deposit option from the transaction menu.

-The deposit function writes to the master account file through a stream, updating the specific user's account balance.

-Users can enter any deposit amount, and upon completion, the program alerts the user of the successful transaction.

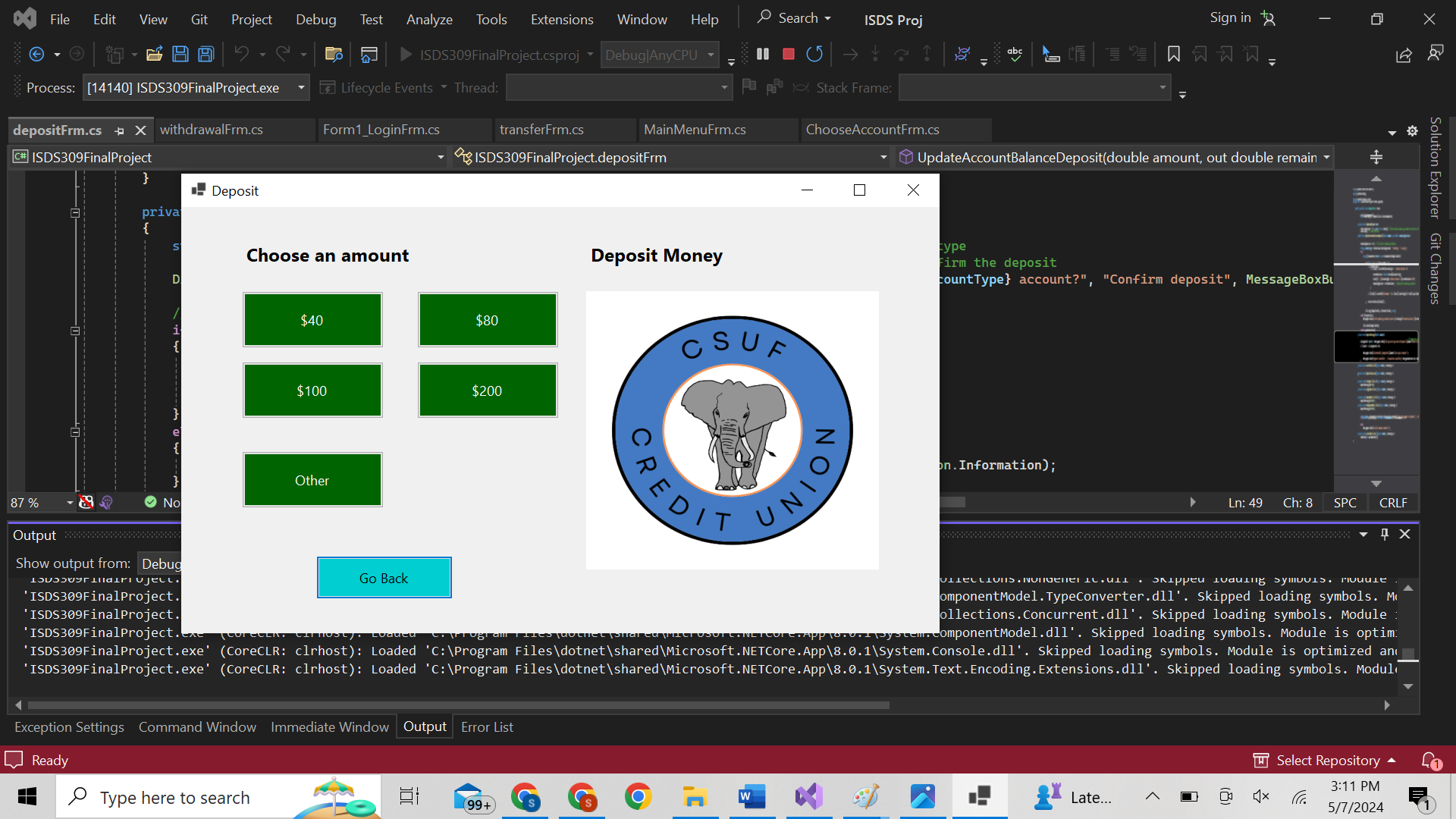
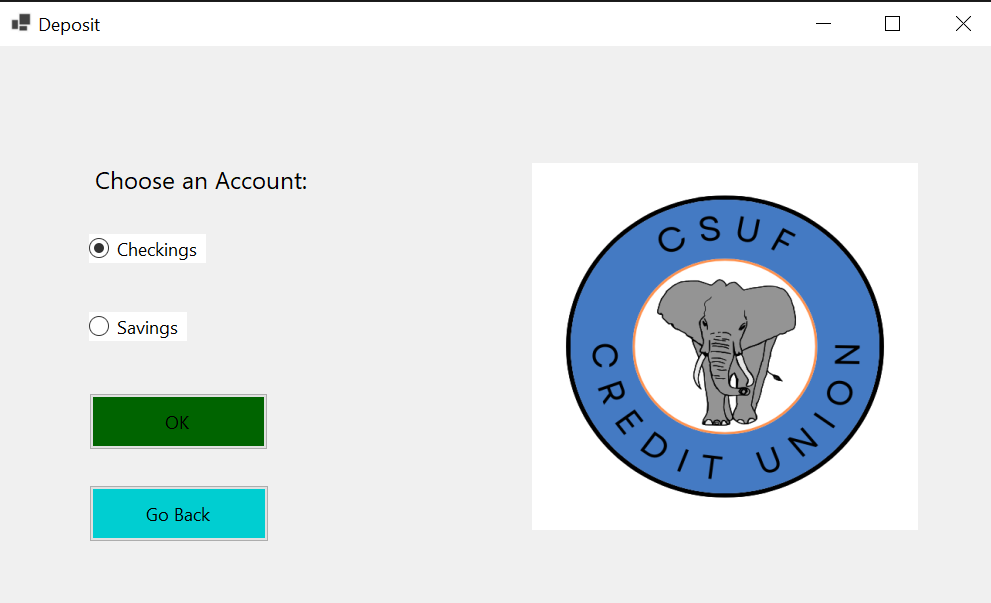
1. Withdrawals

-Similar to deposits, users initiate withdrawals by selecting the withdrawal option from the transaction menu.

-The withdrawal function reverses the process of deposits, updating the user's account balance accordingly.

-Error control mechanisms ensure that users can only withdraw funds available in their account.

-If users attempt invalid actions, such as withdrawing more than their available balance, an error message prompts them to enter a valid value.

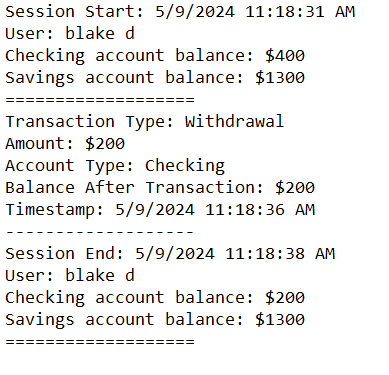


1. Logout/Receipts

-The program updates the log file with the transaction details when users log out of the ATM.

-A personalized receipt is generated using stream writer commands, providing users with a record of their account changes.

-The receipt includes a before and after section, displaying the changes in account balances after each transaction.

Sample receipt after withdrawing $200 From checkings 

Features

**Login Screen**: Users input their credentials (username and password) to access ATM functionalities.

**Transaction Screen**: After login, users are presented with a menu of available transactions, including withdrawal, deposit, and balance inquiry.

**Receipt Output**: A receipt is printed at the end of each transaction, providing detailed information such as date, time, and transaction details.

**Transaction Storage**: Transactions are stored until the end of the day, ensuring accurate record-keeping.

**Efficient Data Management**: Emphasis is placed on efficient storage and retrieval of information to optimize program performance.

**Error Handling**: Comprehensive error handling is implemented to manage unexpected scenarios, providing informative messages to guide users.

**Windows Forms (WinForms)**: The user interface is developed using WinForms, offering a familiar and intuitive environment for users.

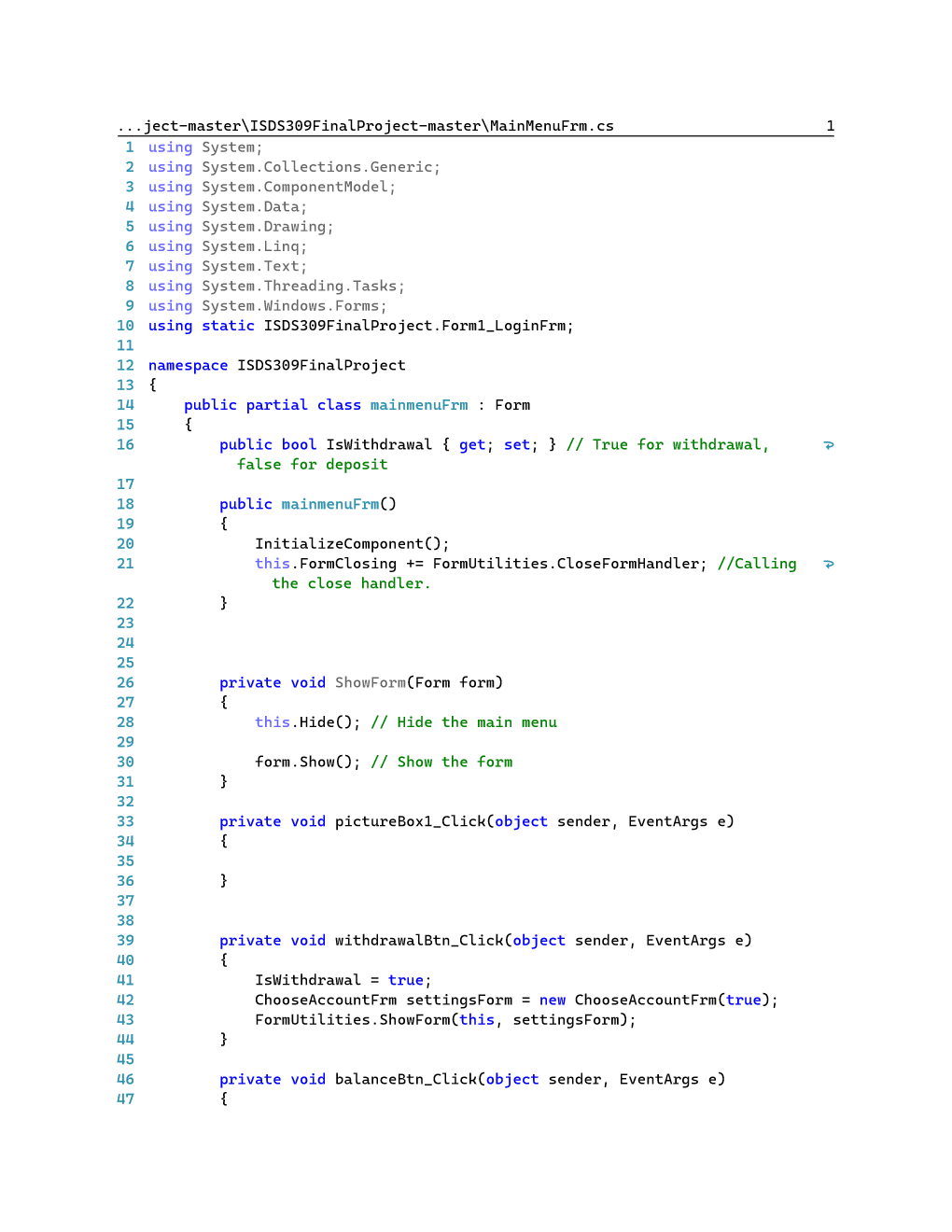
Code Print-Out Pages

-Main Form (Form1)

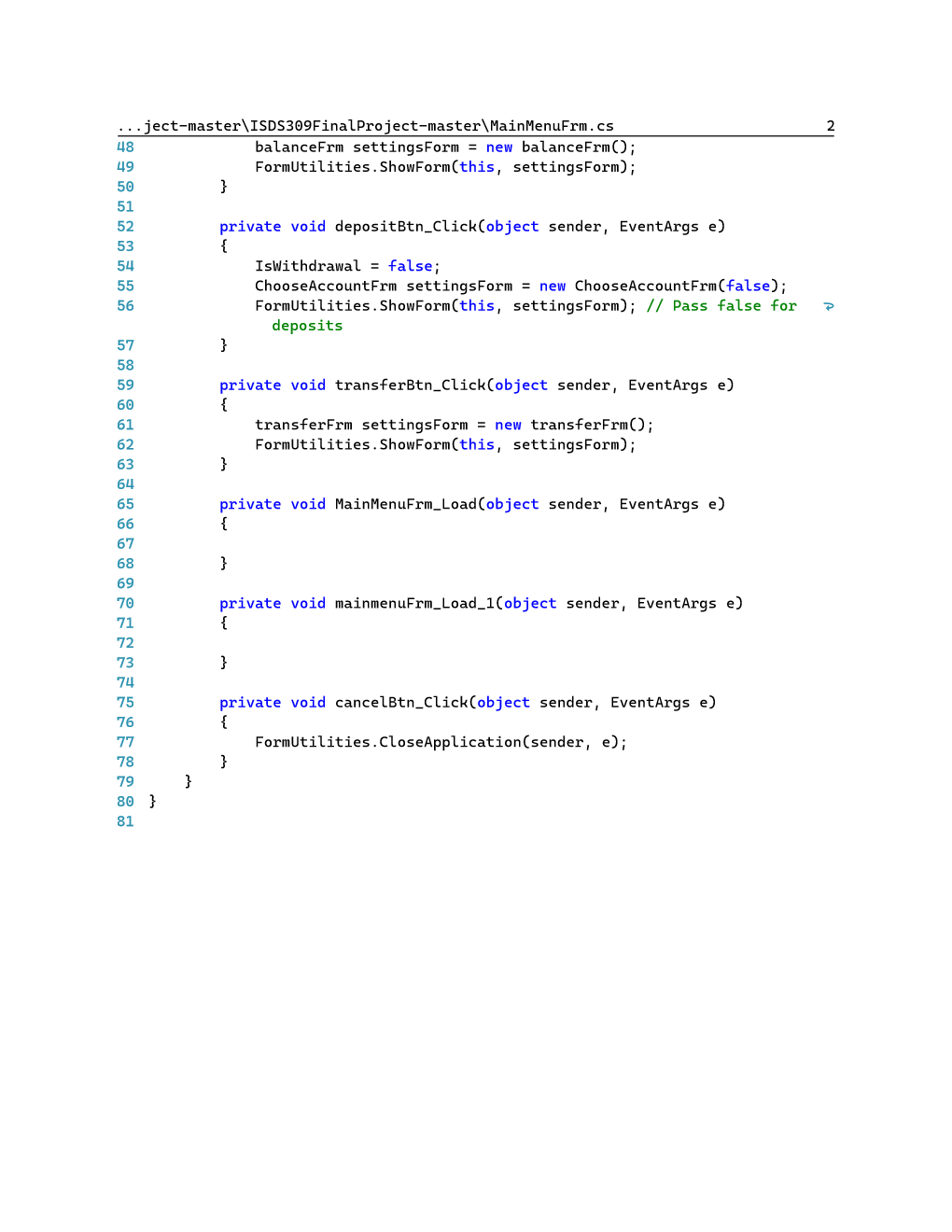
-Form screenshots

-Content of Form1.cs (and the ".cs" of any other forms you may have, if you have multiple forms)

-Same for any additional Form you may use

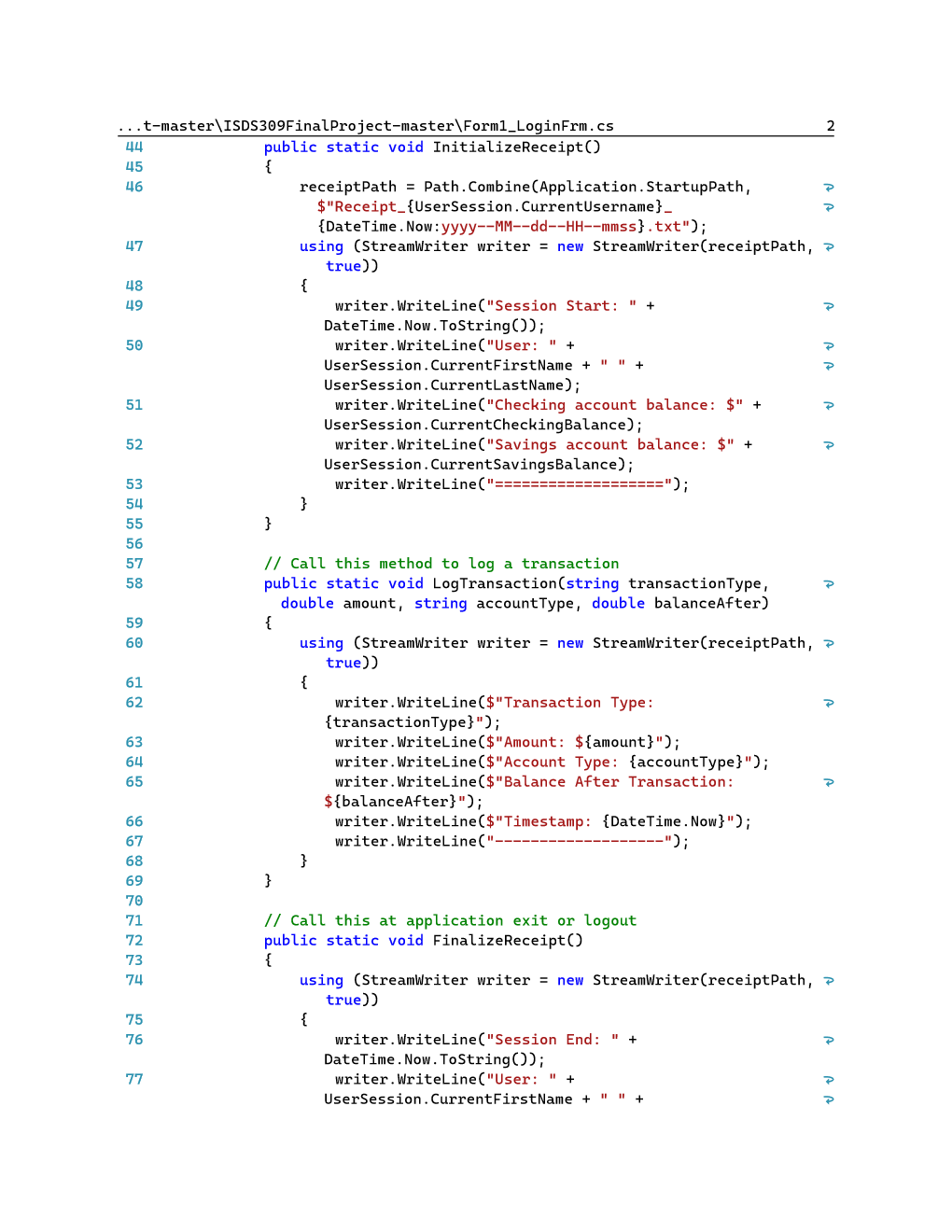


Main Menu

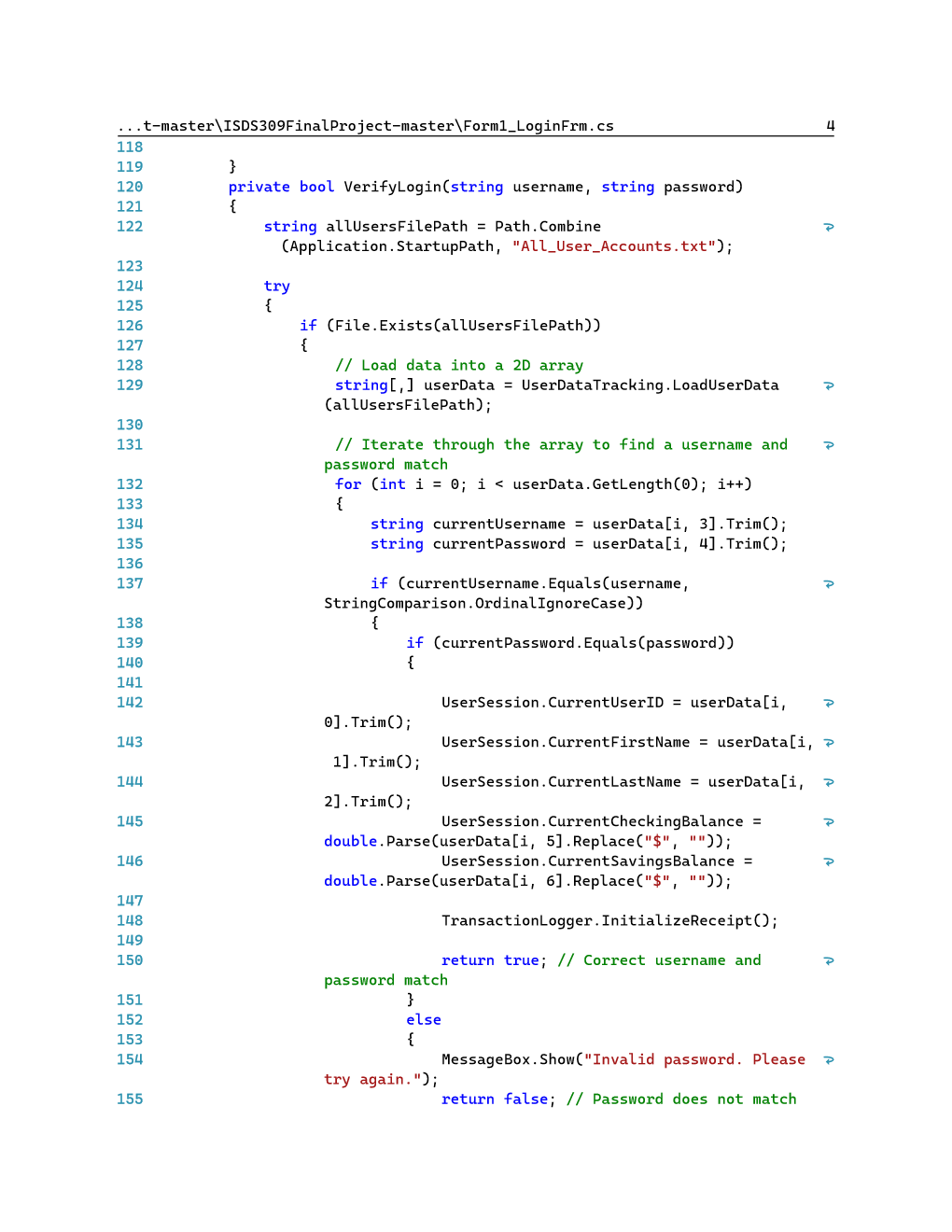


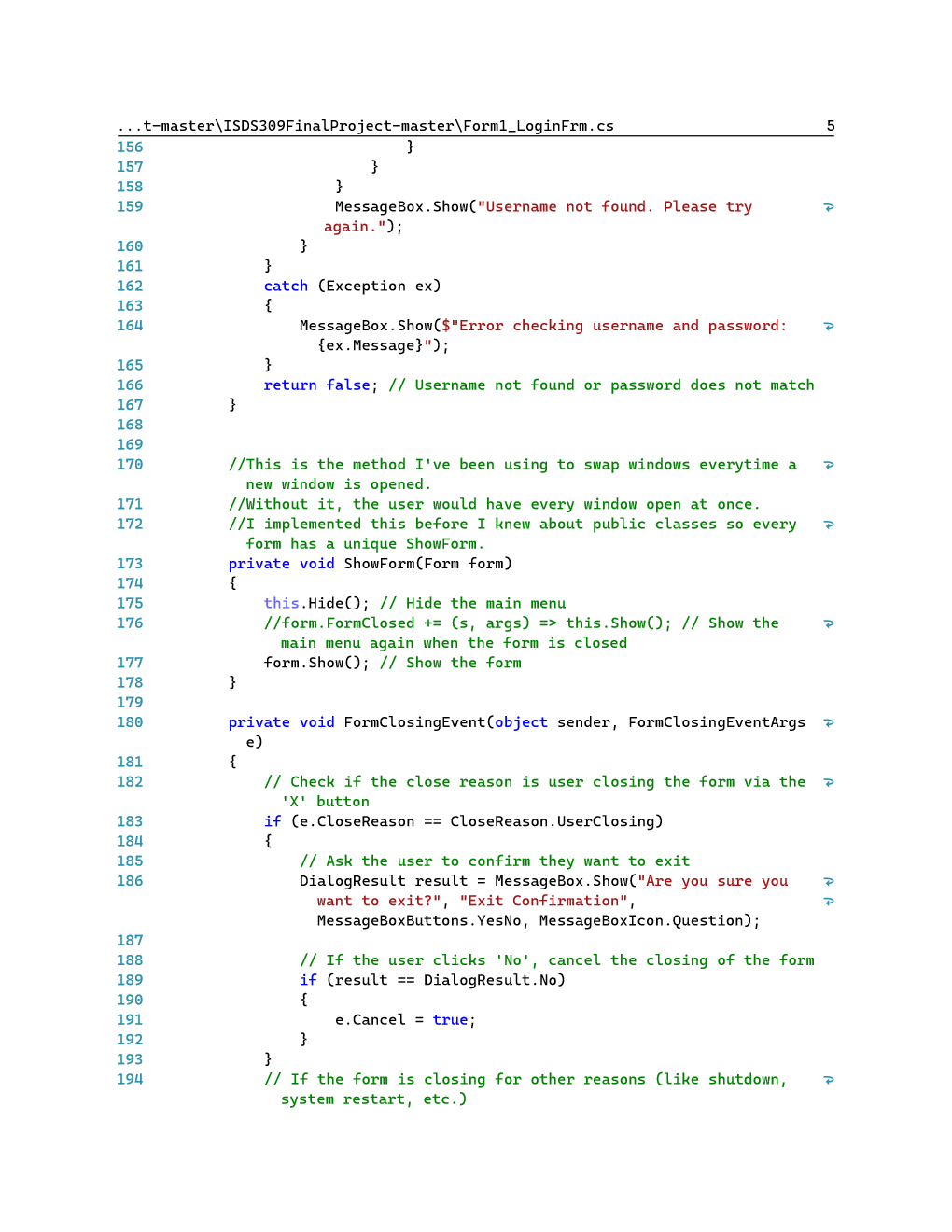
Form1\_LoginFrm

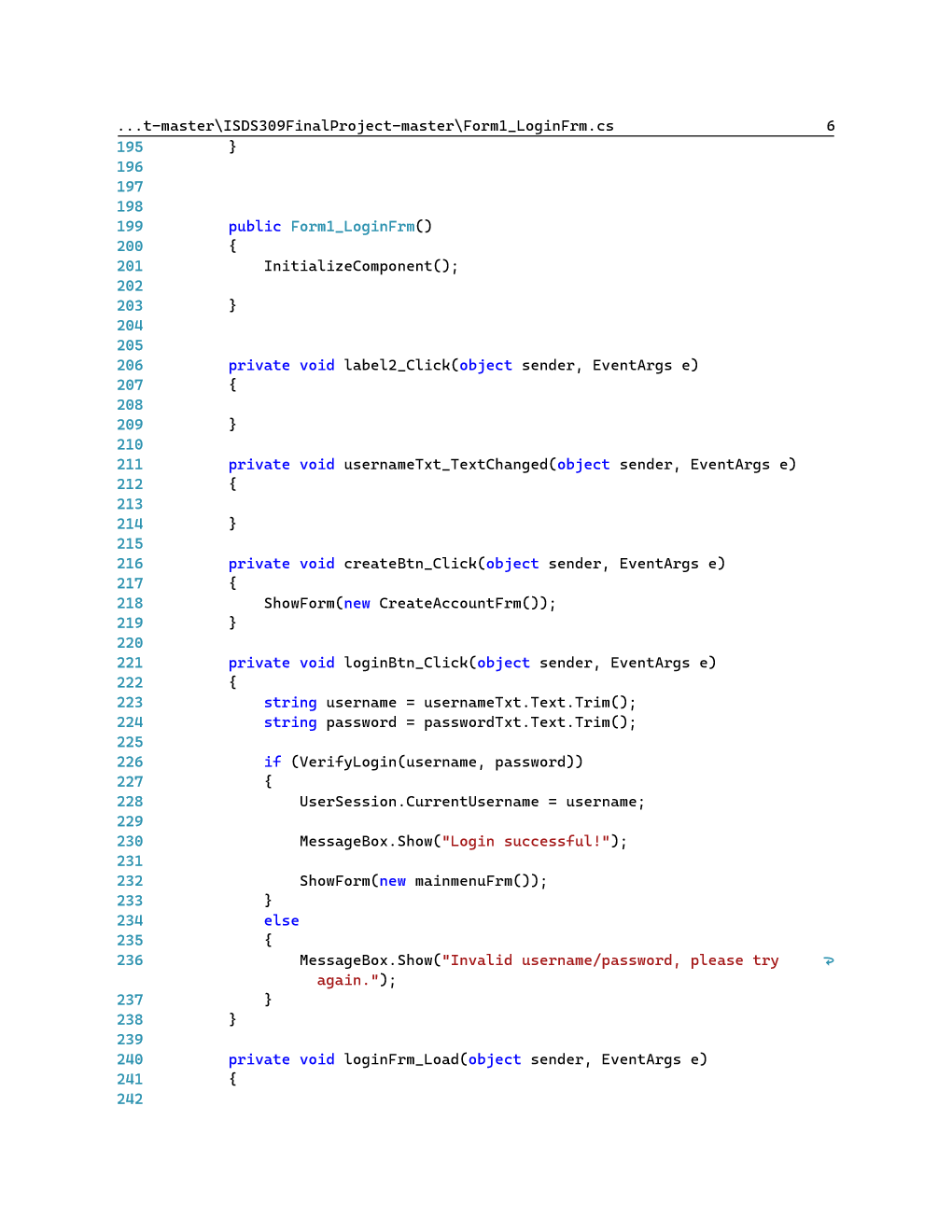


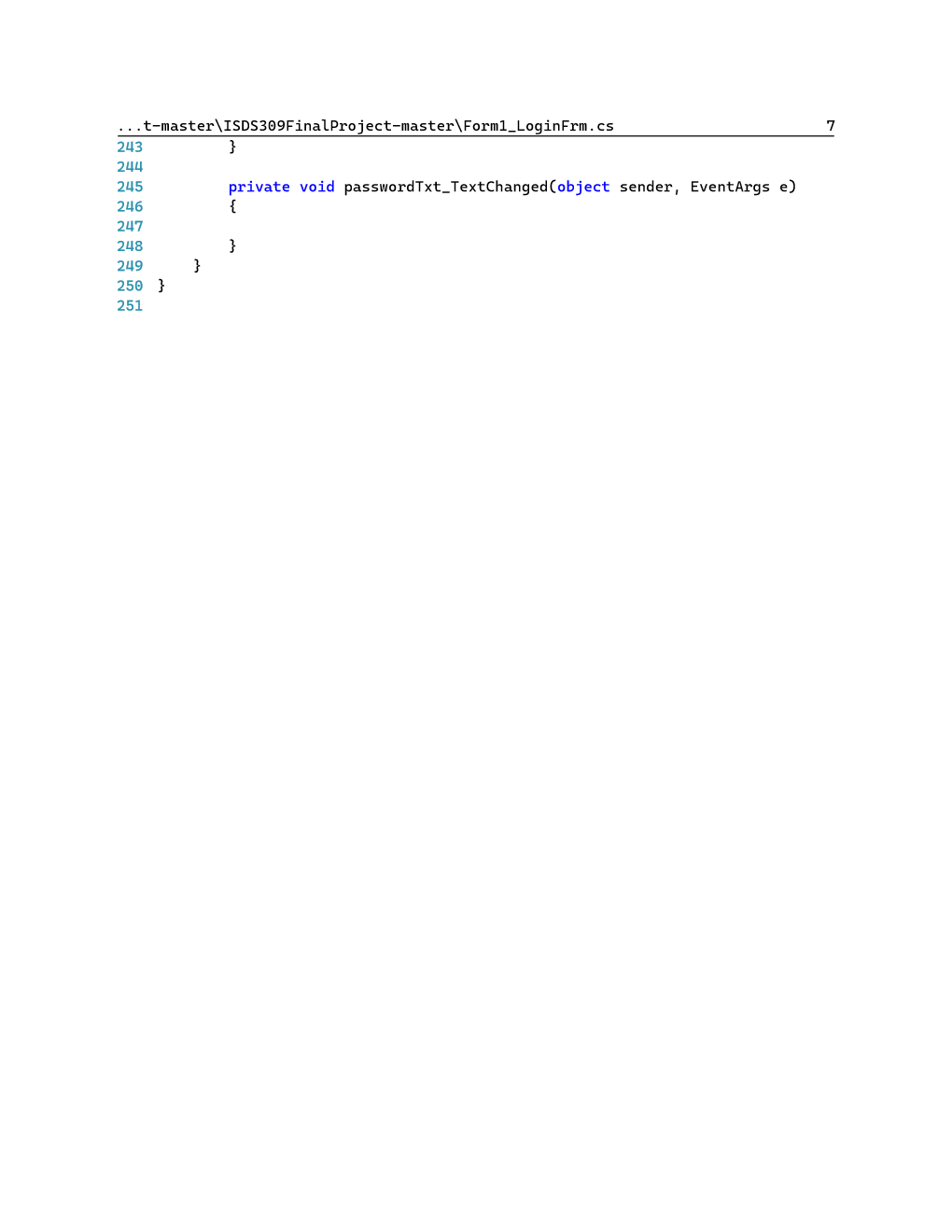




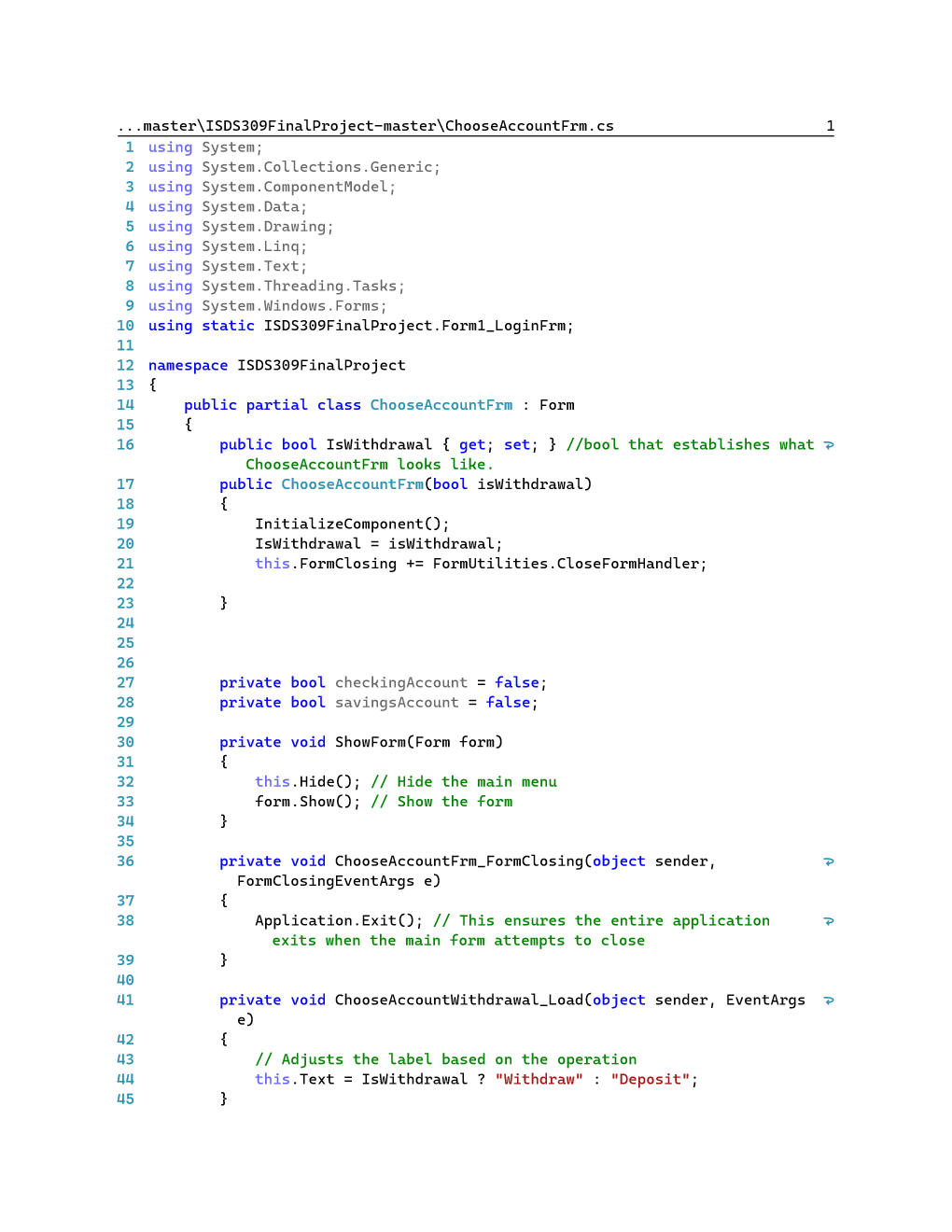


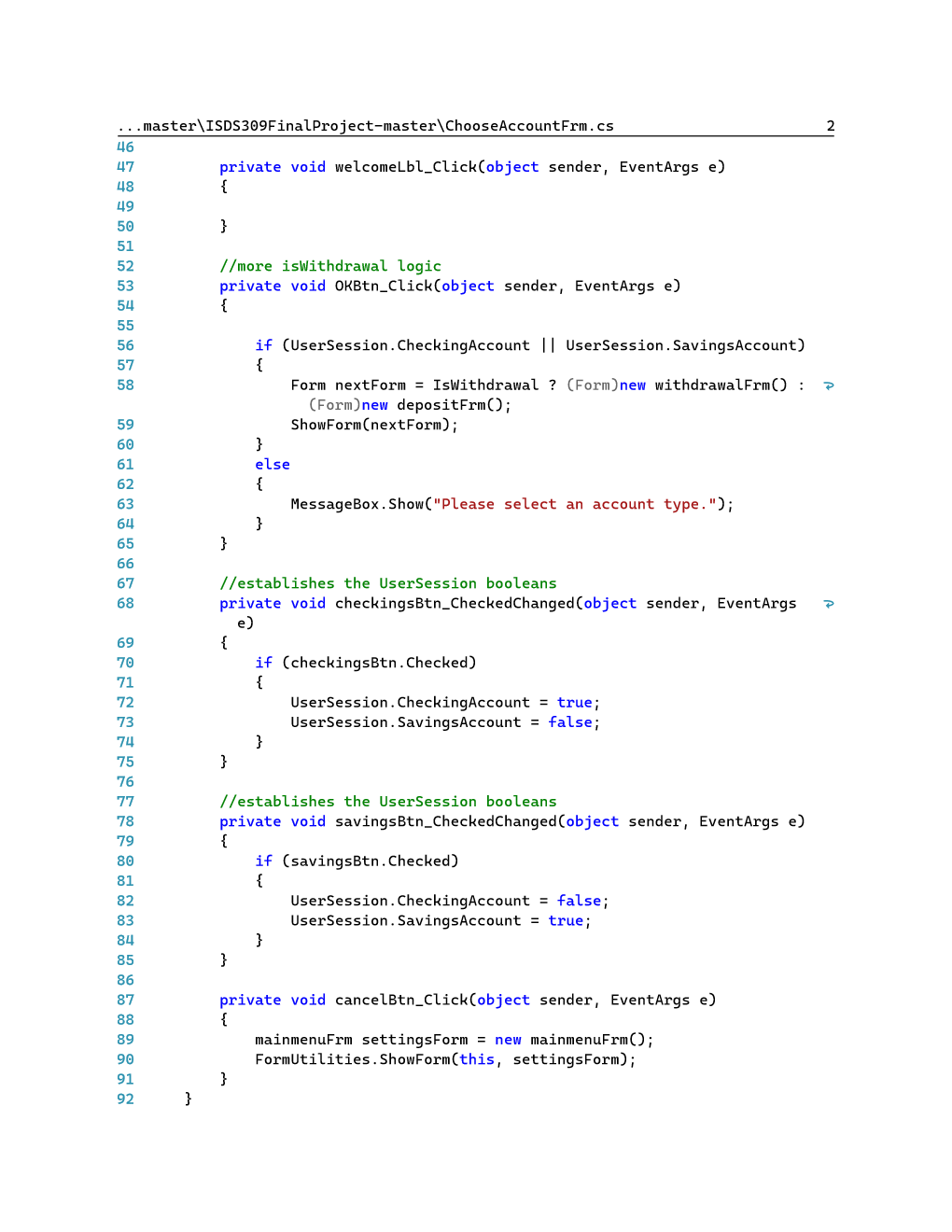


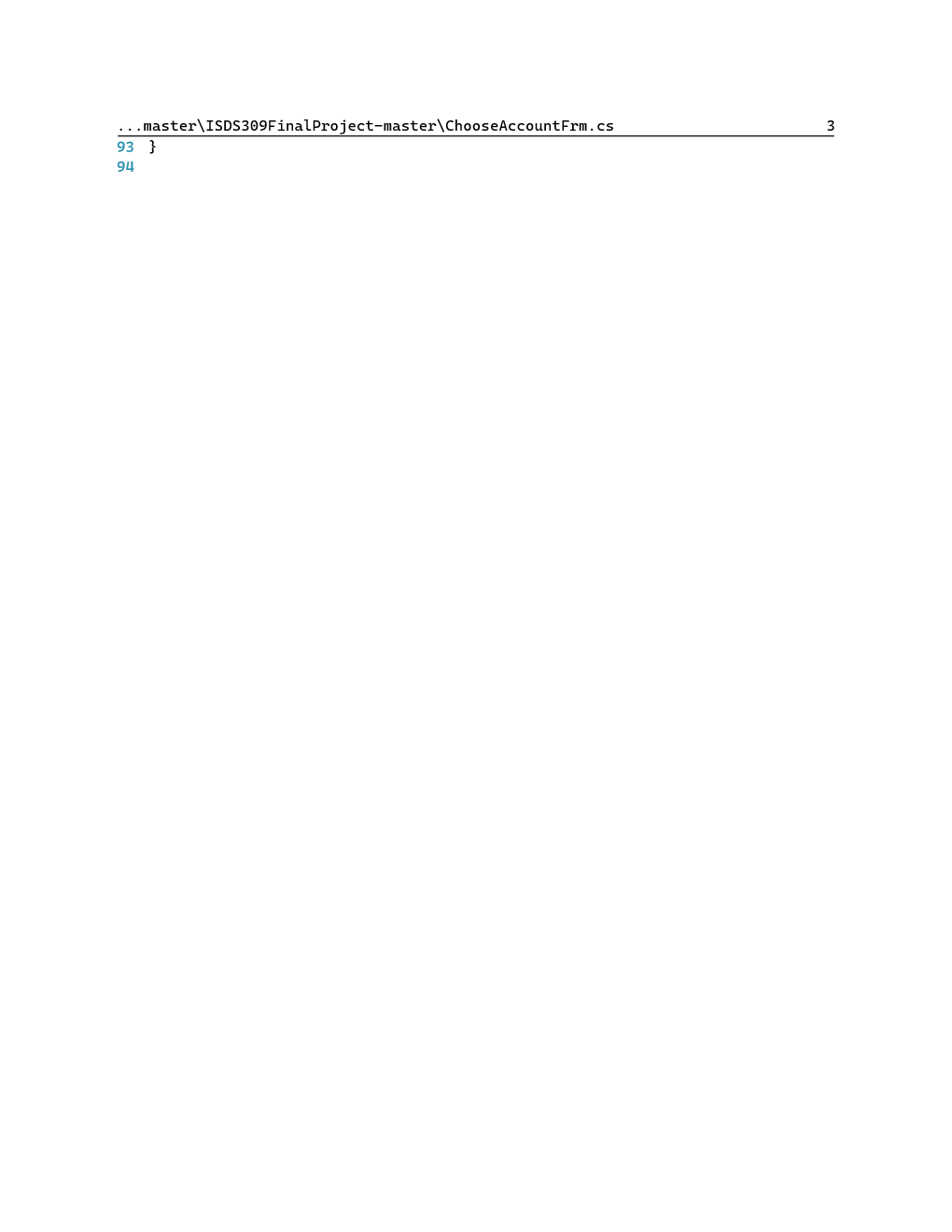




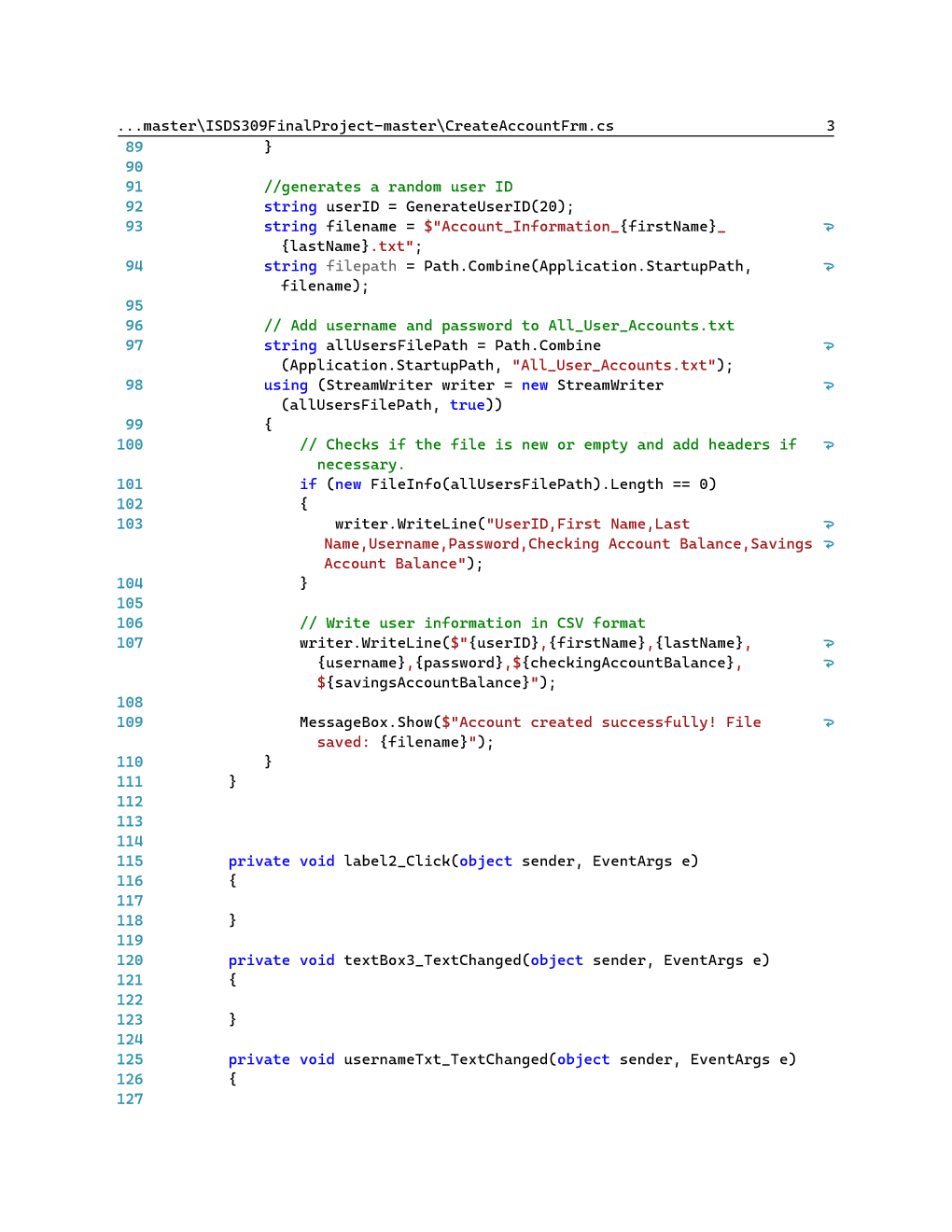
ChooseAccountFrm



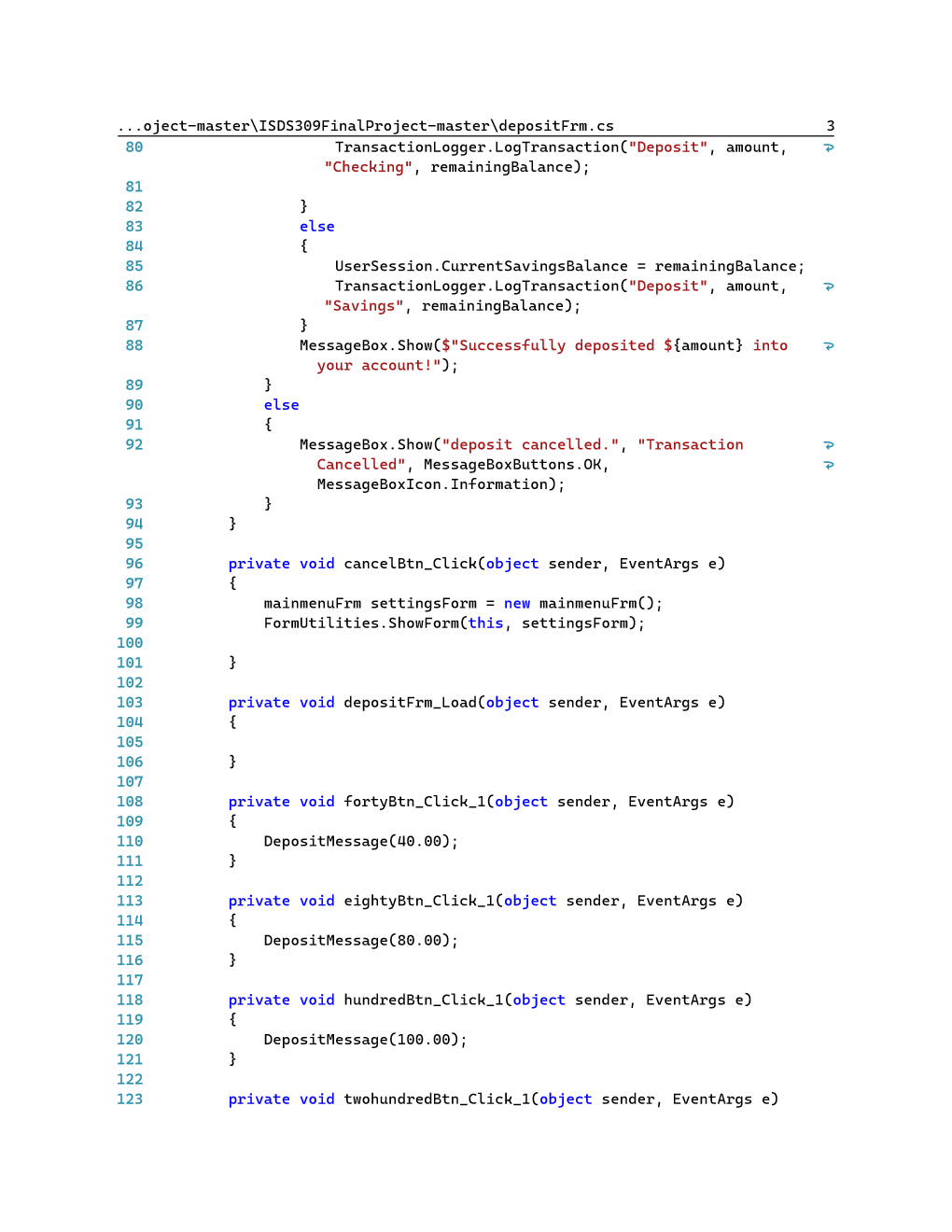
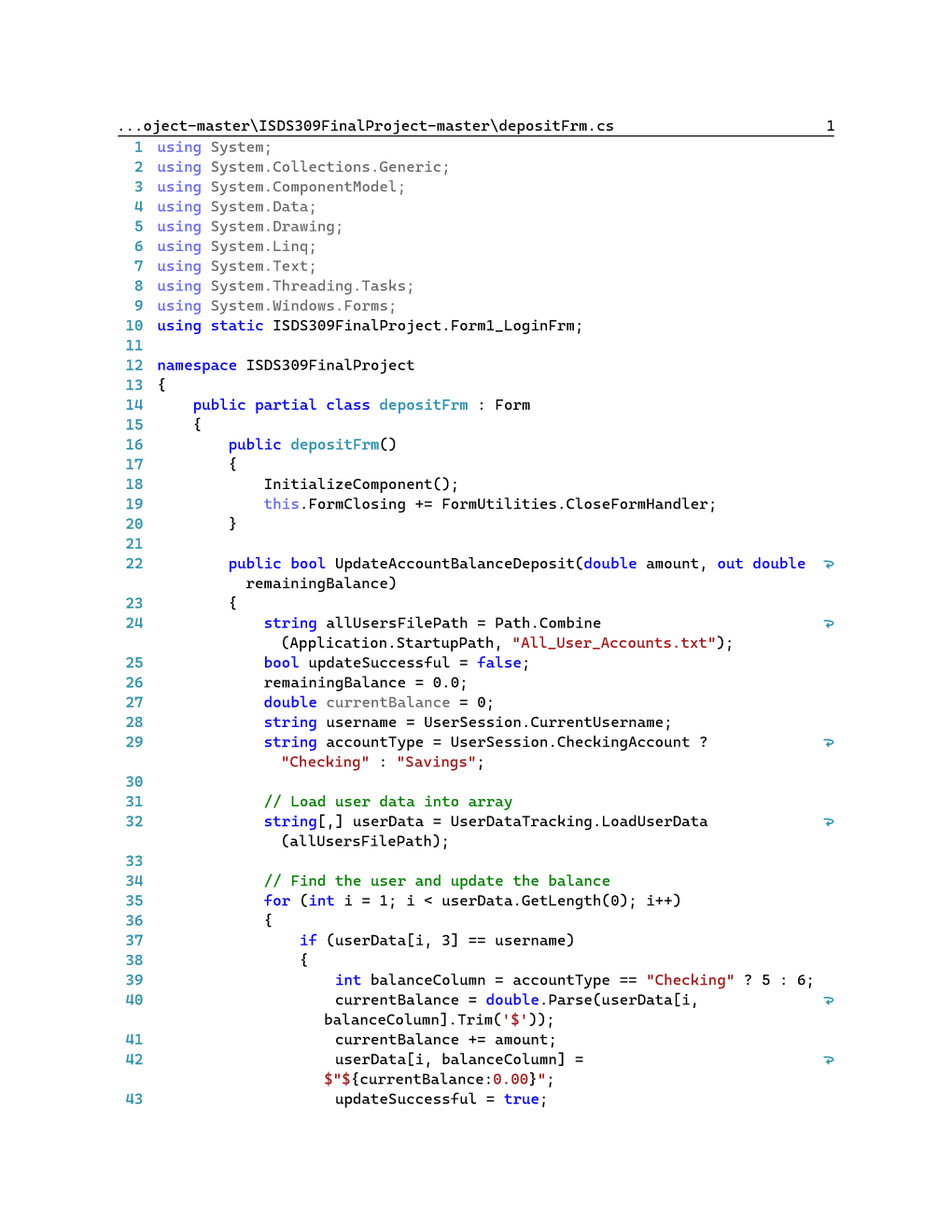




CreateAccountFrm

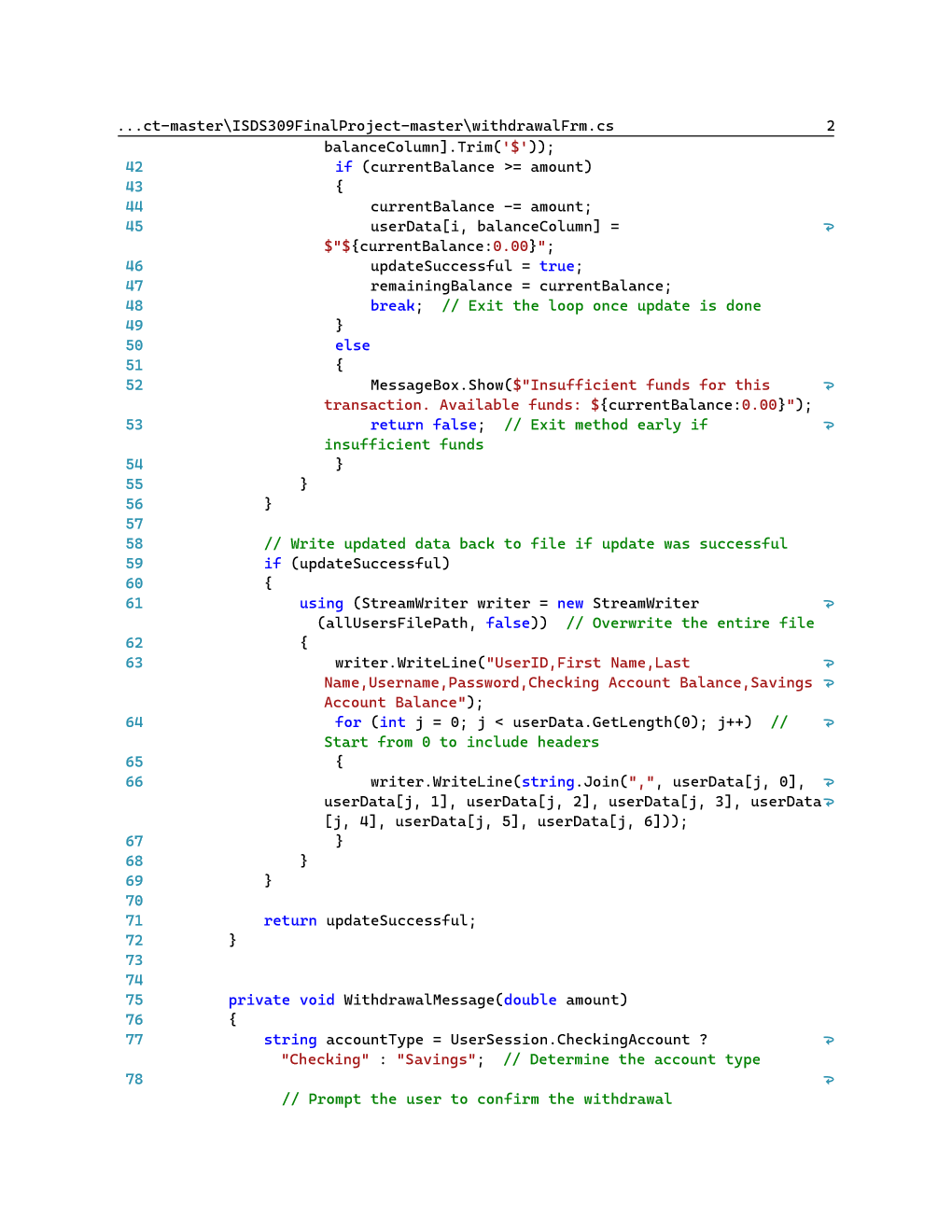


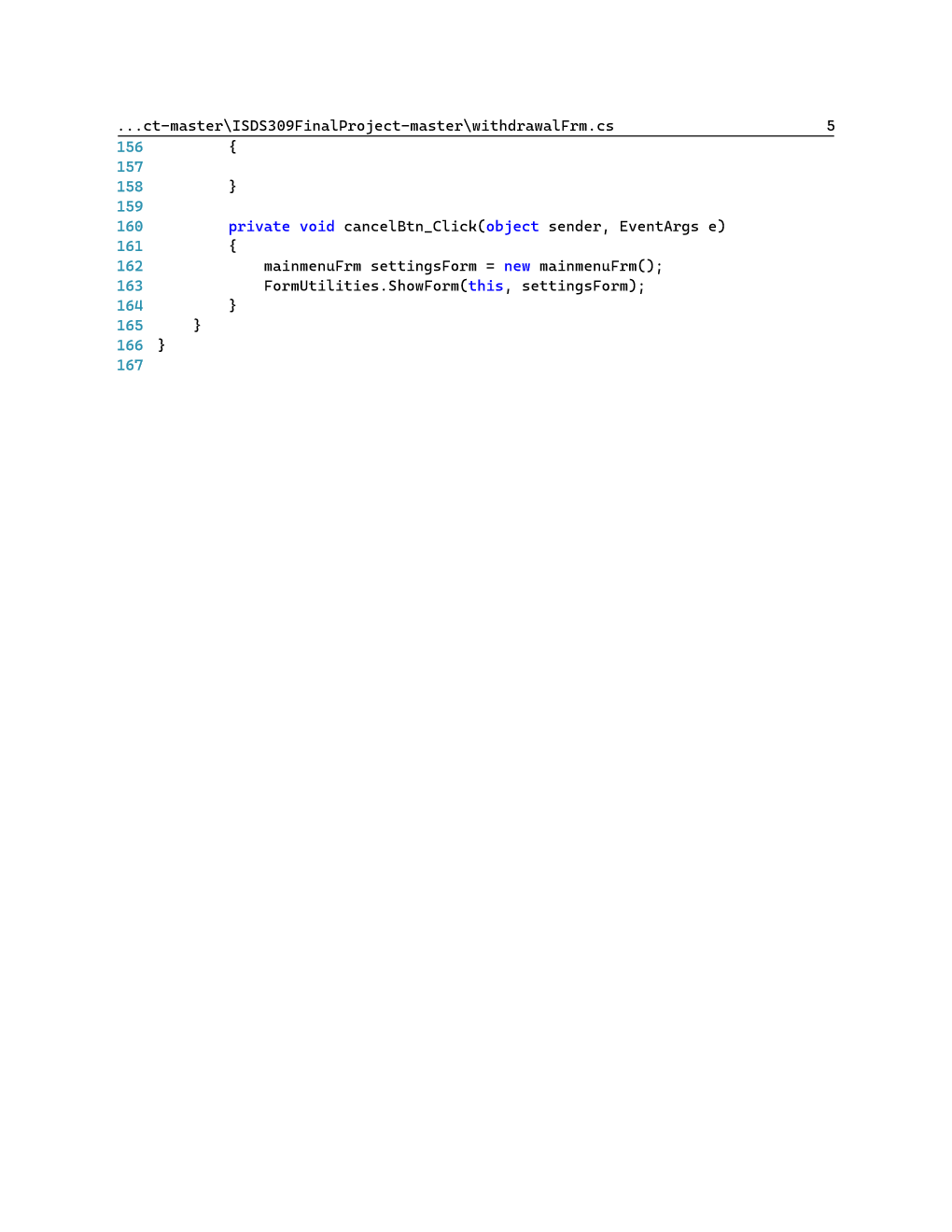
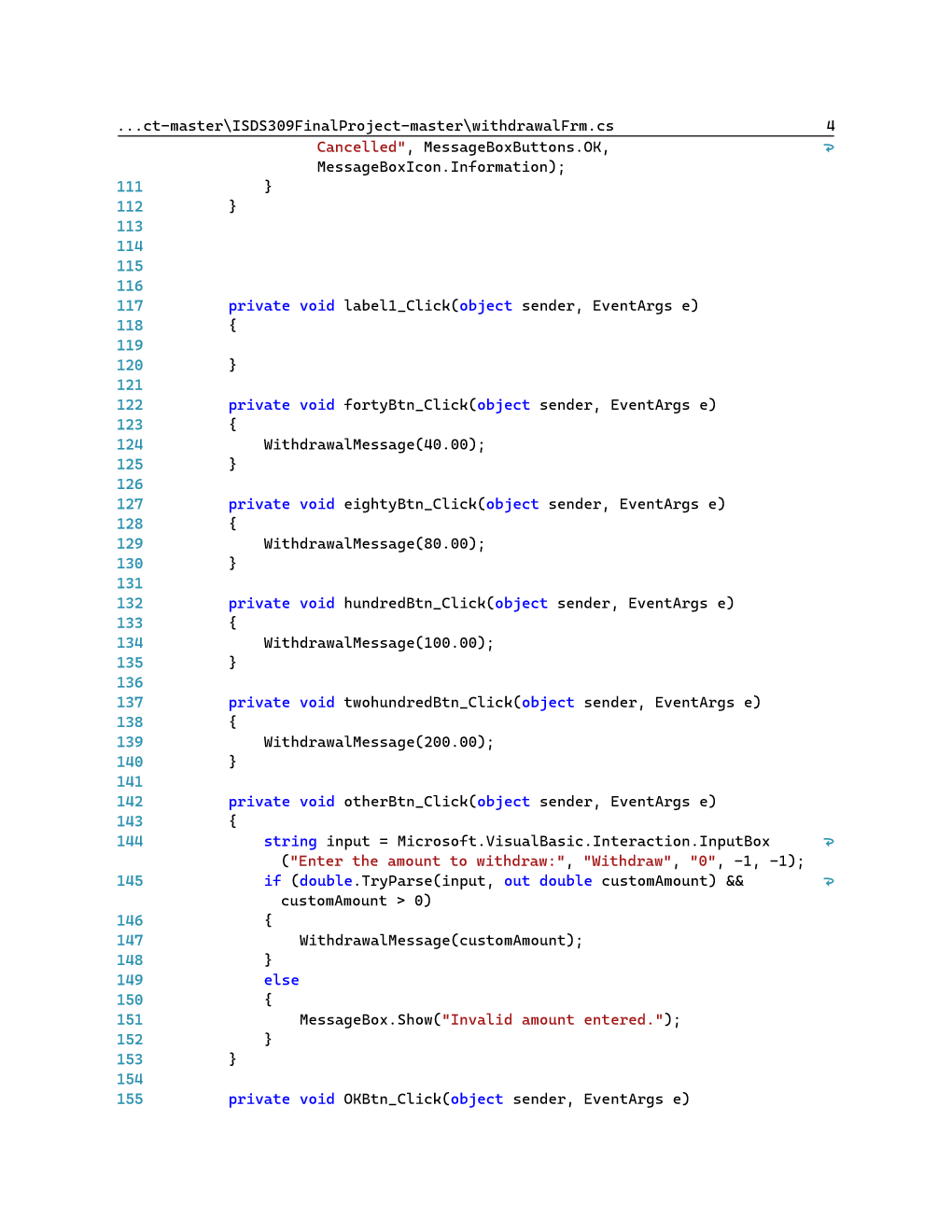
depositFrm



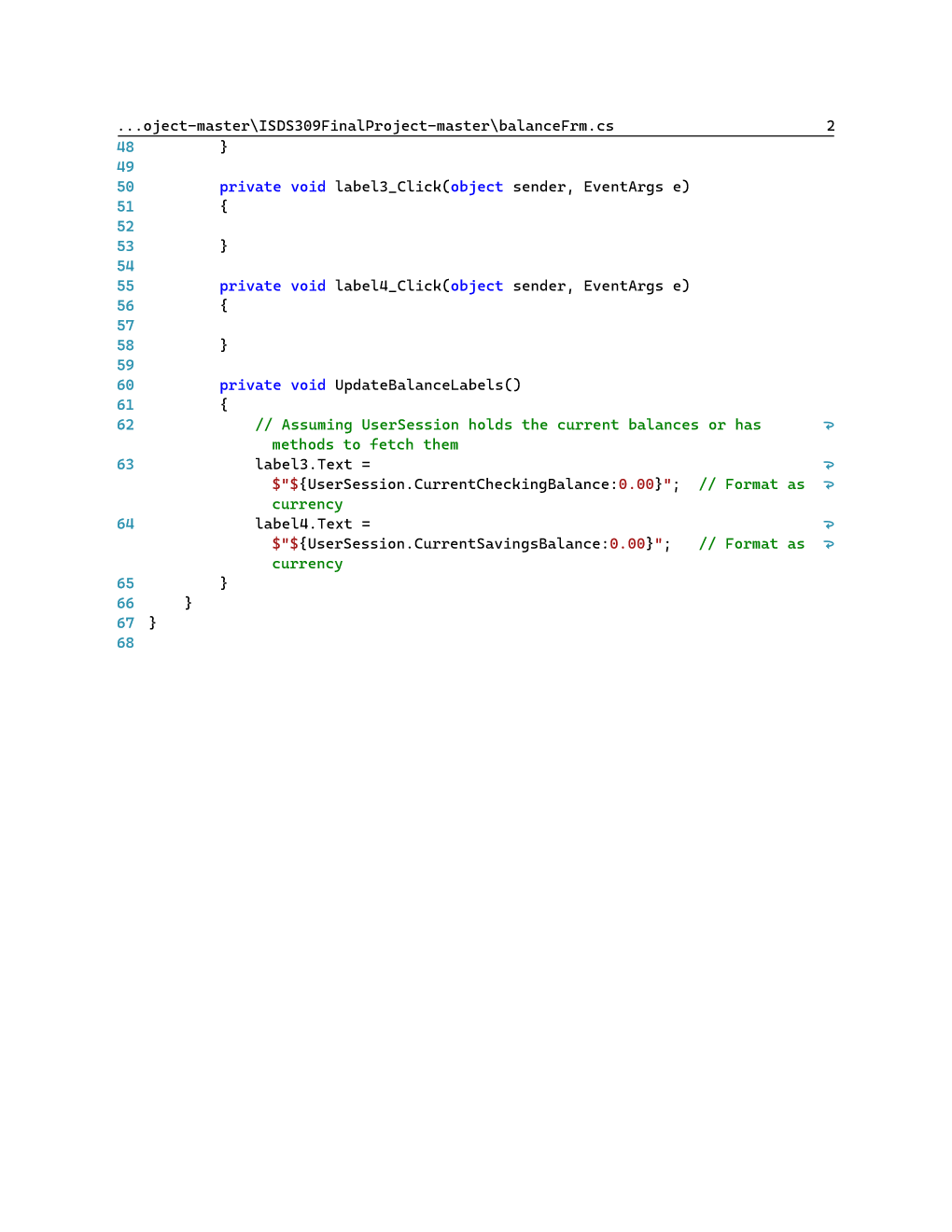
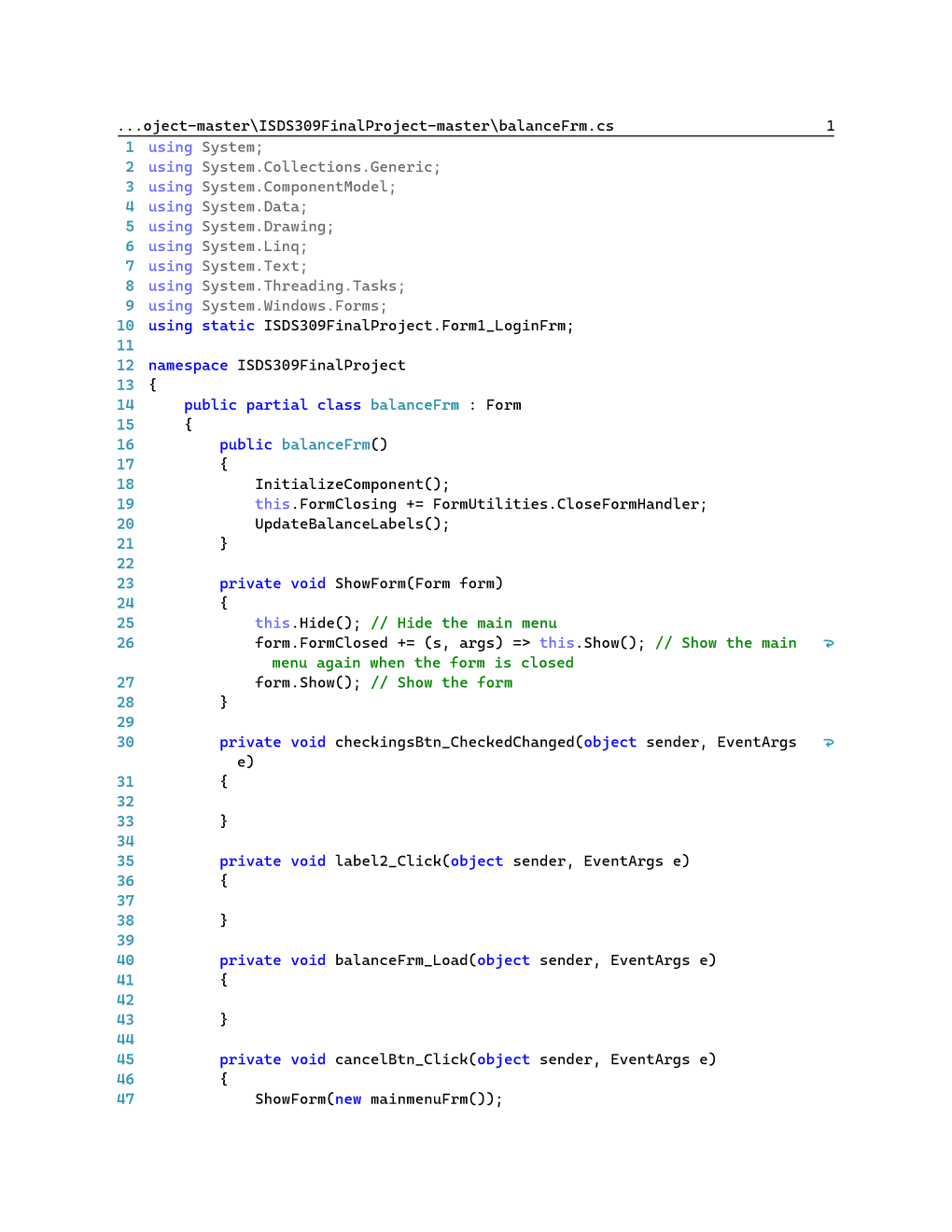
withdrawalFrm



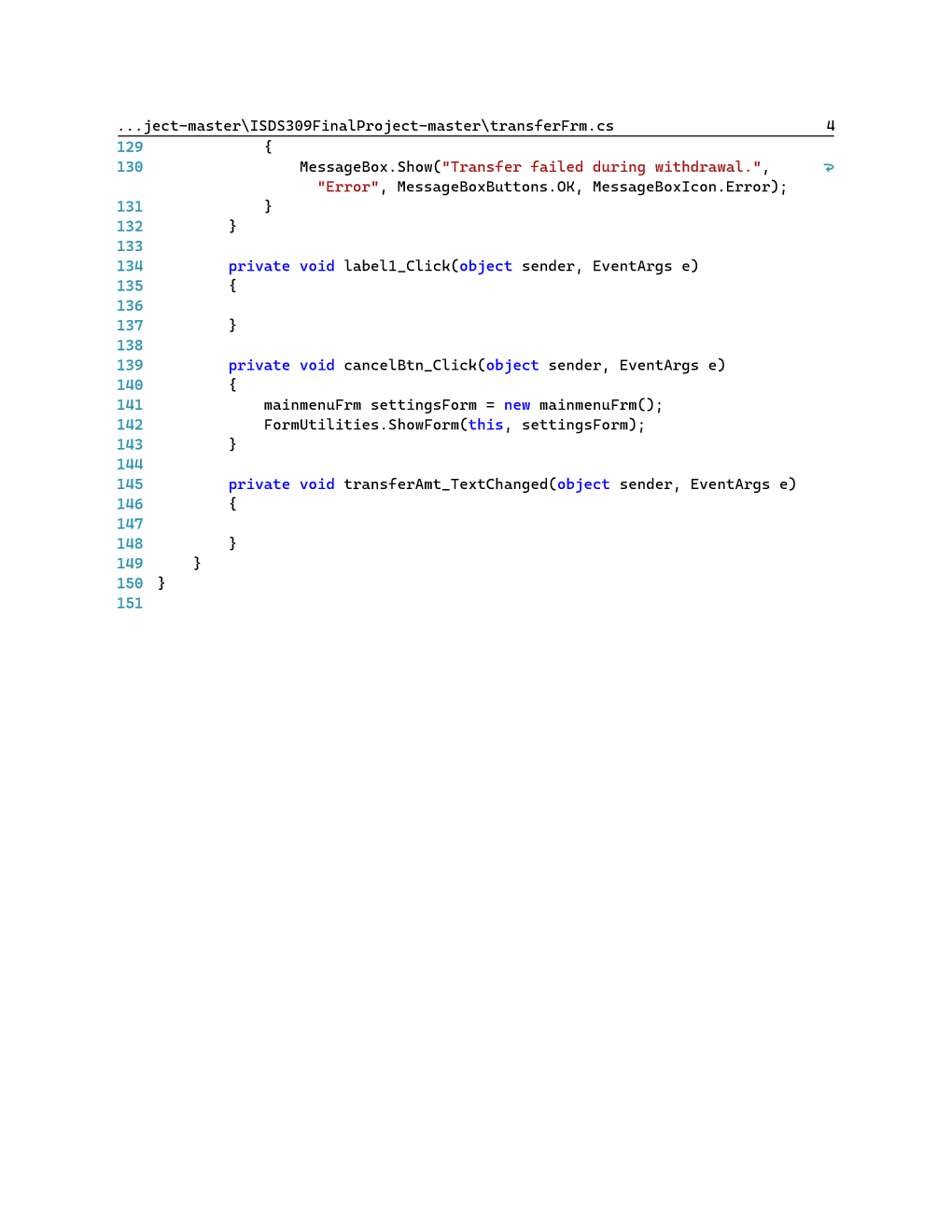




balanceFrm



transferFrm



Program.cs File

