BANK MANAGEMENT SYSTEM

SHAPING THE FUTURE OF FINANCE

CSIT 25, FIRST YEAR.
COURSE CODE: CT-175
COURSE NAME: PROGRAMMING FUNDAMENTALS.

Presented by

AATIQA BATOOL CT-25056.
SAITA AHUJA CT-25065.
ARBISH TEHSEEN CT-25053

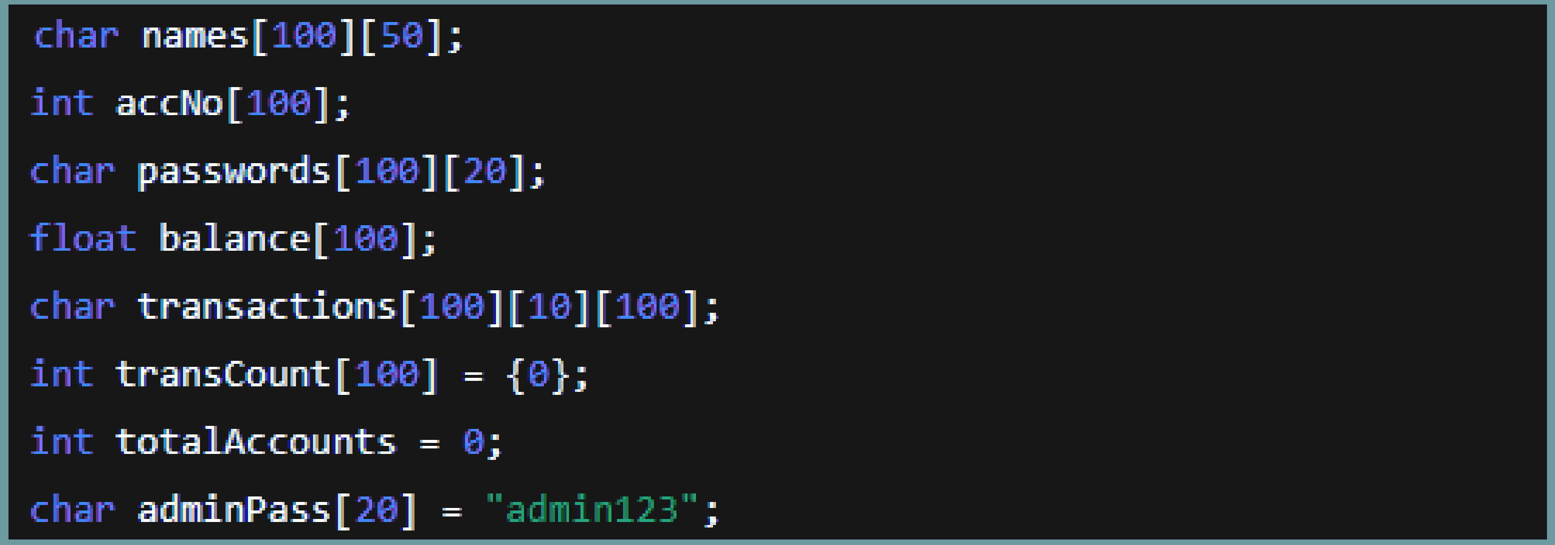
INDEX

- Introduction
- Data Storage
- Main Menu
- Admin Login
- ATM Services
- Help Desk
- Additional Features
- Problem we faced and how we overcame them.
- Code Run.
- Architectural Diagram
- Conclusion

INTRODUCTION

- The Bank Management System is a console-based application developed in the C programming language.
- It is designed to perform basic banking operations such as:
- Account creation and deletion
- Deposits and withdrawals
- Balance inquiry
- Fund transfers
- Viewing account summaries
- The system allows management of multiple customer accounts efficiently.
- It includes password protection to ensure data security and prevent unauthorized access.
- The program prevents invalid transactions, such as negative balance withdrawals.
- It features transaction tracking for maintaining records of user activities.
- The project provides a realistic simulation of banking operations within a simple interface.
- It is user-friendly, easy to navigate, and suitable for beginners.
- Demonstrates key C programming concepts like:
- Arrays for data storage
- Loops for repeated operations
- Conditionals for decision-making
- String handling for names and passwords
- The project helps learners understand how data is managed, secured, and processed in financial systems.

DATA STORAGE

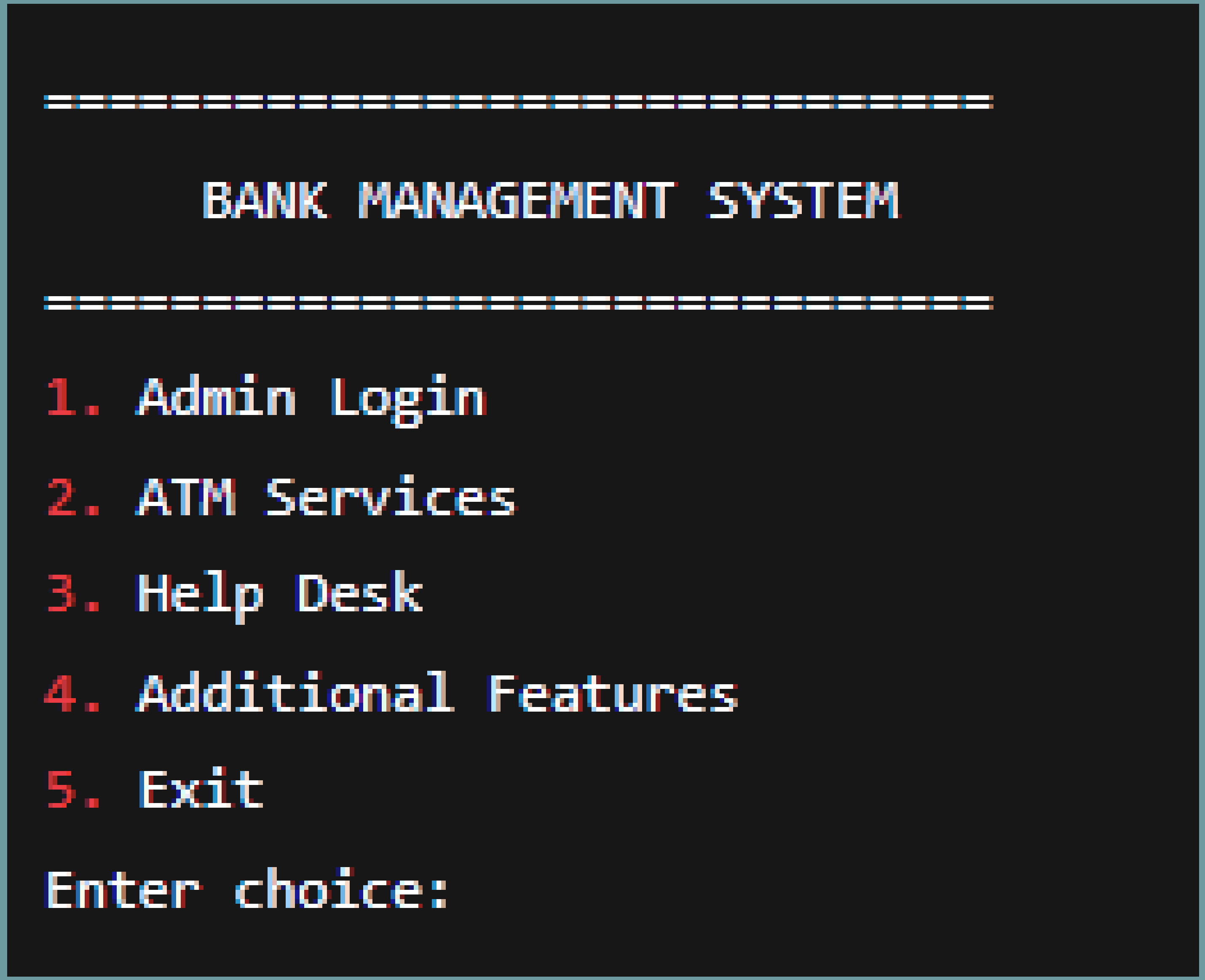


```c
char names[100][50];
int accNo[100];
char passwords[100][20];
float balance[100];
char transactions[100][10][100];
int transCount[100] = {0};
int totalAccounts = 0;
char adminPass[20] = "admin123";
```

- names[100][50] – Stores names of up to 100 account holders (each name up to 50 characters).
- accNo[100] – Holds unique account numbers for identifying each account.
- passwords[100][20] – Stores passwords (up to 20 characters) for securing each account.
- balance[100] – Keeps track of each account’s monetary balance.
- transactions[100][10][100] – Stores transaction history (up to 10 records per account, each up to 100 characters).
- transCount[100] – Tracks the number of transactions made by each account.
- totalAccounts – Maintains the total number of accounts created.
- adminPass[20] – Stores the administrator password for authentication.

MAIN LOOPS AND USER INTEFERENCE

The program runs inside an infinite loop using `while(1)`, repeatedly displaying the main menu:
1. Admin Login
2. ATM Services
3. Help Desk
44. Additional Features
5. Exit
Users can select options multiple times without restarting the program. The `switch(choice)` structure executes the code corresponding to the user's choice, ensuring continuous operation for a realistic banking simulation.



```
========================================

      BANK MANAGEMENT SYSTEM

========================================
1. Admin Login
2. ATM Services
3. Help Desk
4. Additional Features
5. Exit
Enter choice:
```

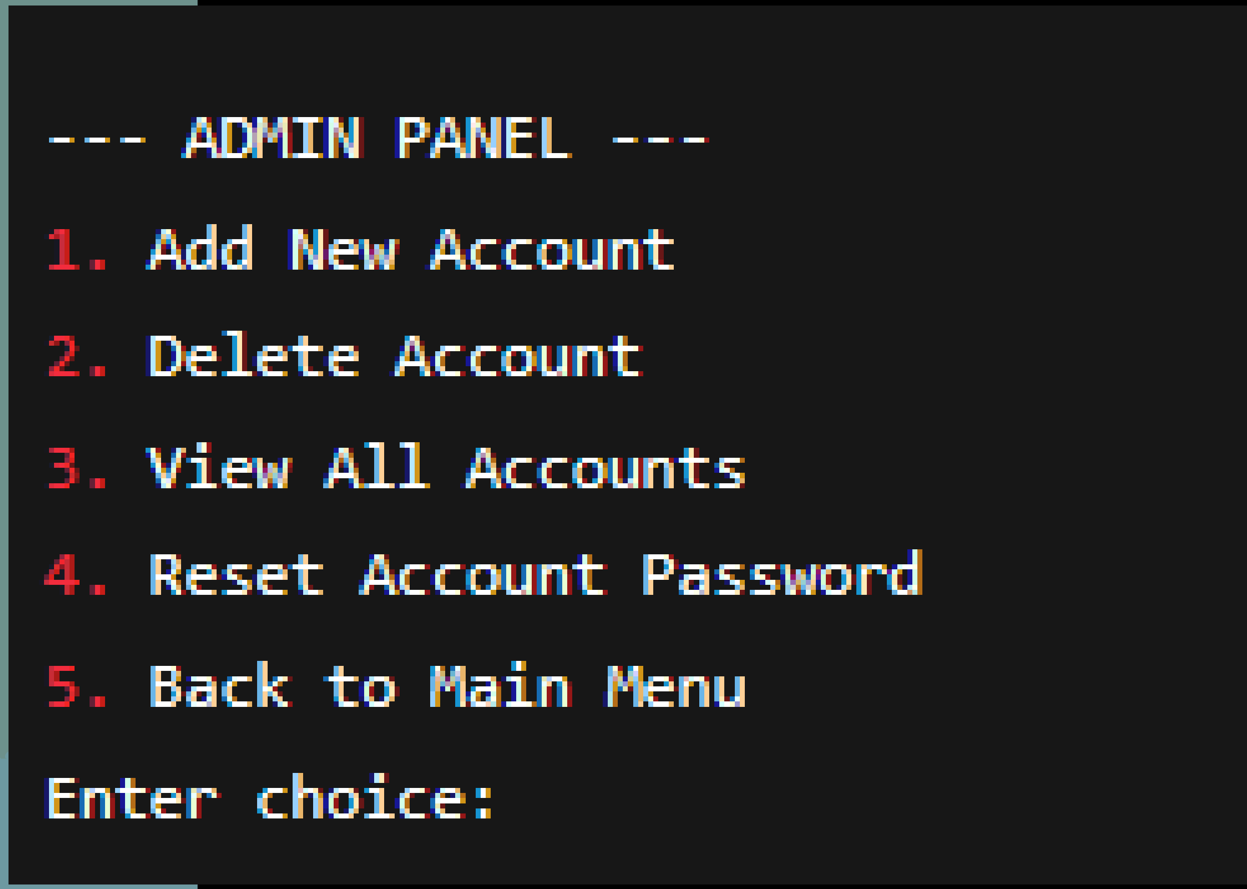
ADMIN PANEL

The Admin Login option prompts the user to enter the admin password, verified using `strcmp()`. Authenticated admins can:

- Add New Account: Store account details in arrays and increment `totalAccounts`.
- Delete Account: Remove an account by shifting array elements.
- View All Accounts: Display name, account number, and balance.
- Reset Password: Reset account password to default.

The admin panel allows complete account management with safe handling of data using loops and arrays

Once you enter your choice and put in correct password then this screen will pop up..

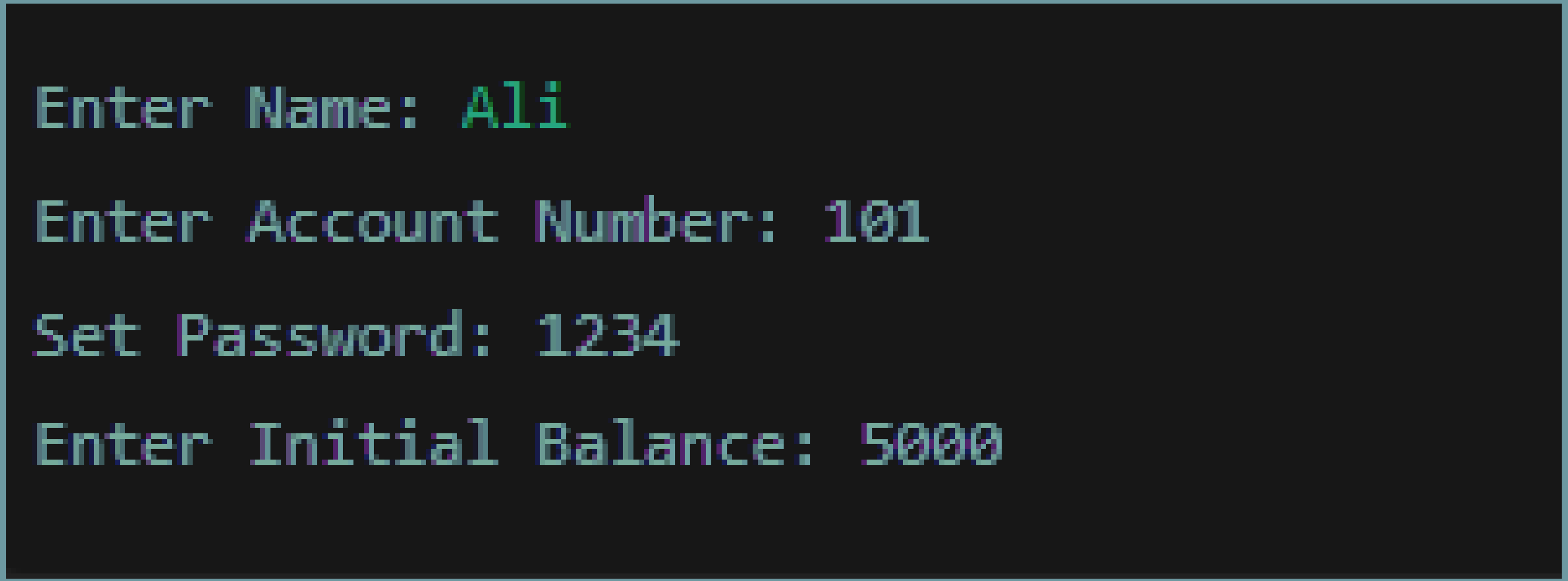


```
--- ADMIN PANEL ---
1. Add New Account
2. Delete Account
3. View All Accounts
4. Reset Account Password
5. Back to Main Menu
Enter choice:
```

1.*ADD NEW ACCOUNT*

A new record is saved in arrays:

- names[0] = "Ali"
- accNo[0] = 101
- passwords[0] = "1234"
- balance[0] = 5000
- totalAccounts = 1



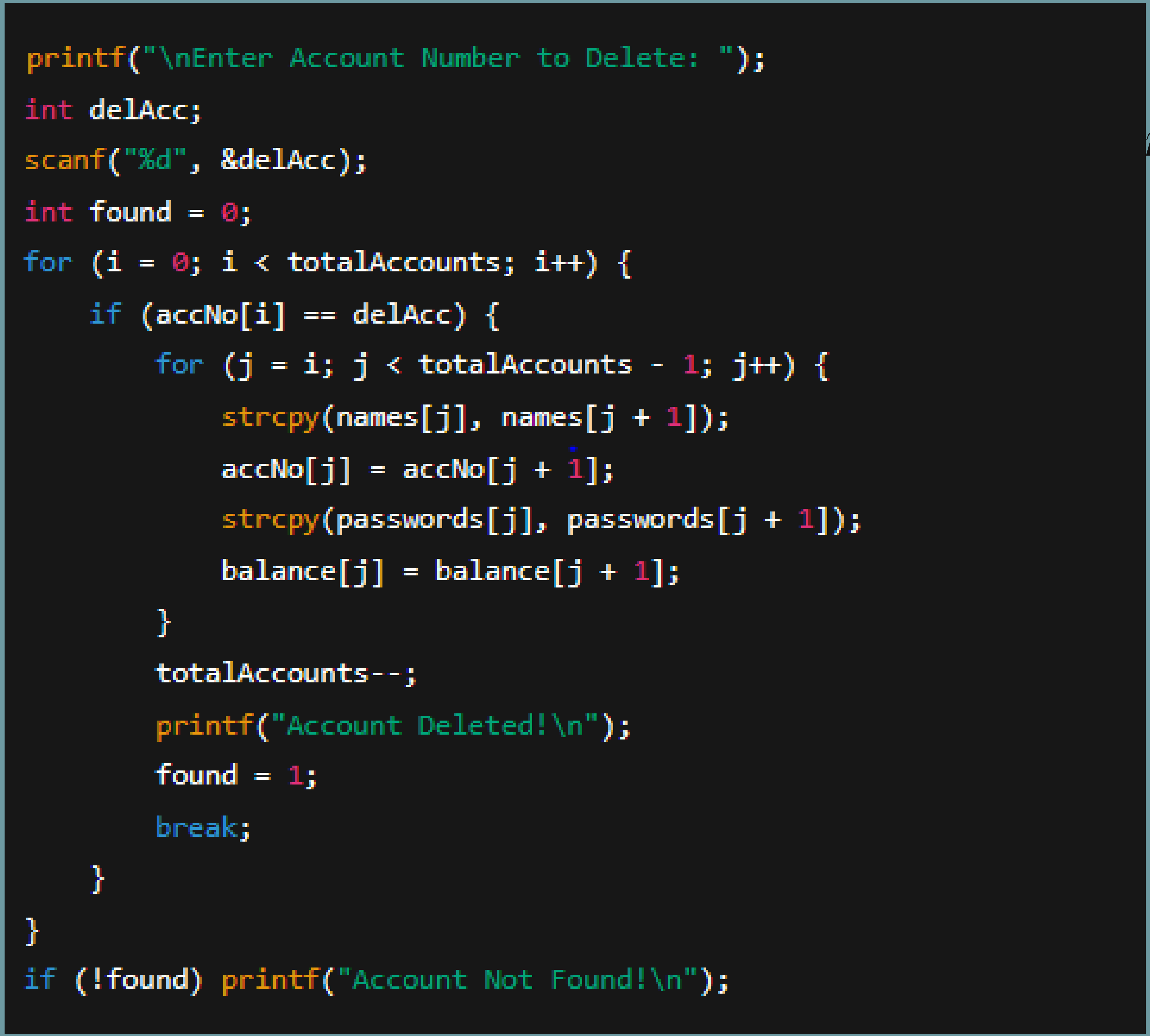
```
Enter Name: Ali

Enter Account Number: 101

Set Password: 1234

Enter Initial Balance: 5000
```

2. DELETE ACCOUNT



```
printf("\nEnter Account Number to Delete: ");
int delAcc;
scanf("%d", &delAcc);
int found = 0;
for (i = 0; i < totalAccounts; i++) {
    if (accNo[i] == delAcc) {
        for (j = i; j < totalAccounts - 1; j++) {
            strcpy(names[j], names[j + 1]);
            accNo[j] = accNo[j + 1];
            strcpy(passwords[j], passwords[j + 1]);
            balance[j] = balance[j + 1];
        }
        totalAccounts--;
        printf("Account Deleted!\n");
        found = 1;
        break;
    }
}
if (!found) printf("Account Not Found!\n");
```

- *. Input:*

The program asks the admin to enter the account number to delete.

- *Search Loop:*

It loops through all stored accounts using

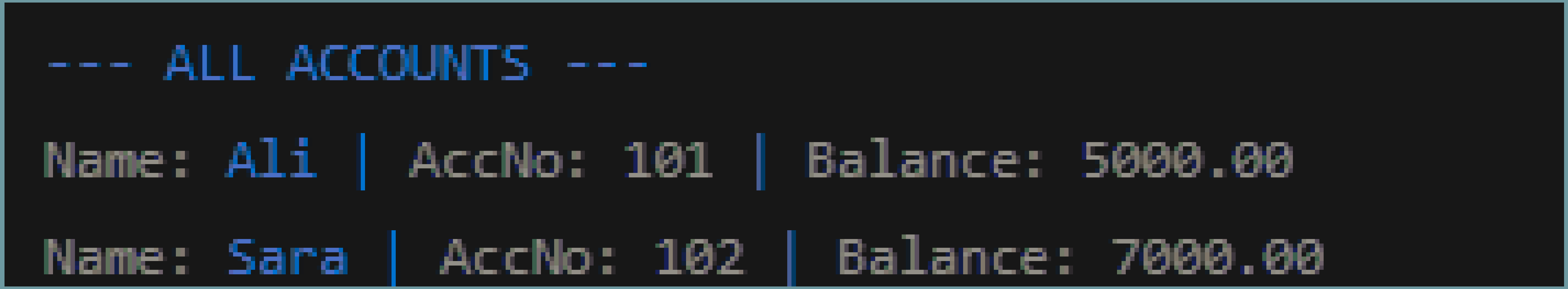
- *To find the matching account number (accNo[i] == delAcc),If Match Found:*

The system removes the account by shifting all data (name, password, balance) of next accounts one position left.

- *This ensures that no empty space remains in between array elements.*
- *Update Total Accounts.*
- *After deletion,*

3. VIEW ALL ACCOUNTS

- If there are multiple accounts (say Ali and Sara), you'll see:

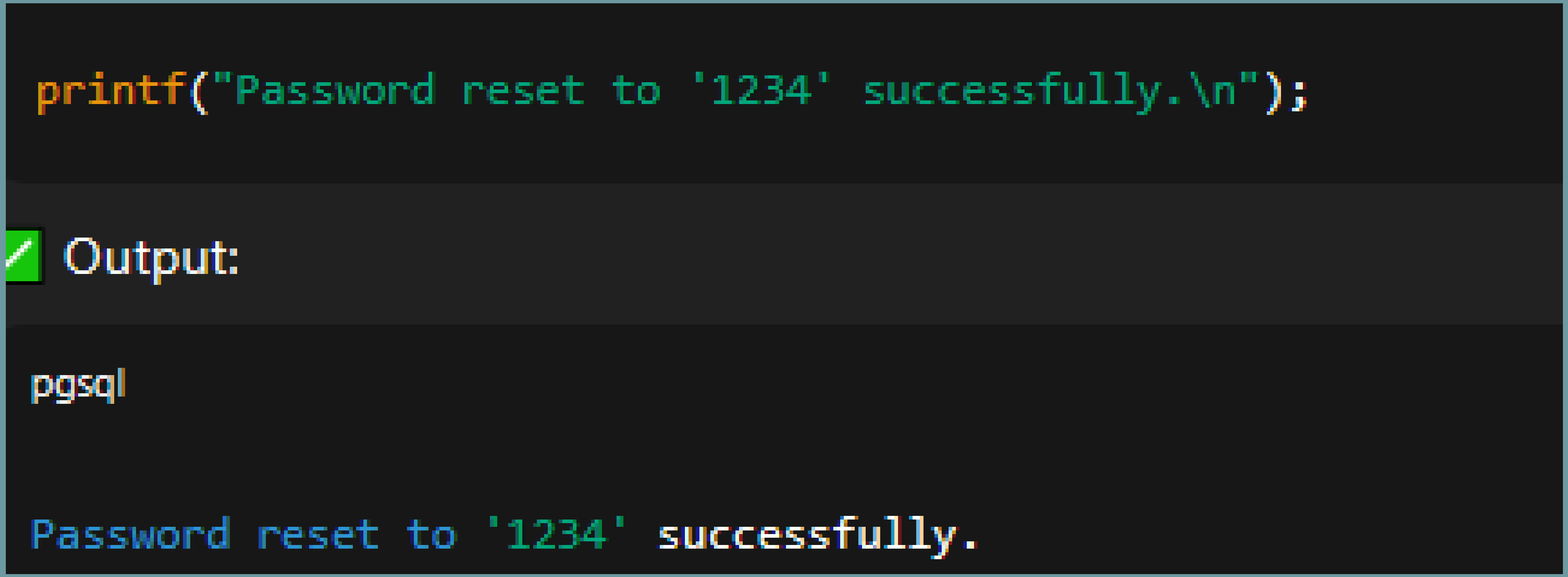


```
--- ALL ACCOUNTS ---
Name: Ali | AccNo: 101 | Balance: 5000.00
Name: Sara | AccNo: 102 | Balance: 7000.00
```

4. RESET PASSWORD

When user enters 4. If the entered account number exists in the system, the condition becomes true.

- i loops through all created accounts.
- accNo[i] is compared with the entered number (resAcc)



```c
printf("Password reset to '1234' successfully.\n");
```

✅ Output:

```
pgsql

Password reset to '1234' successfully.
```

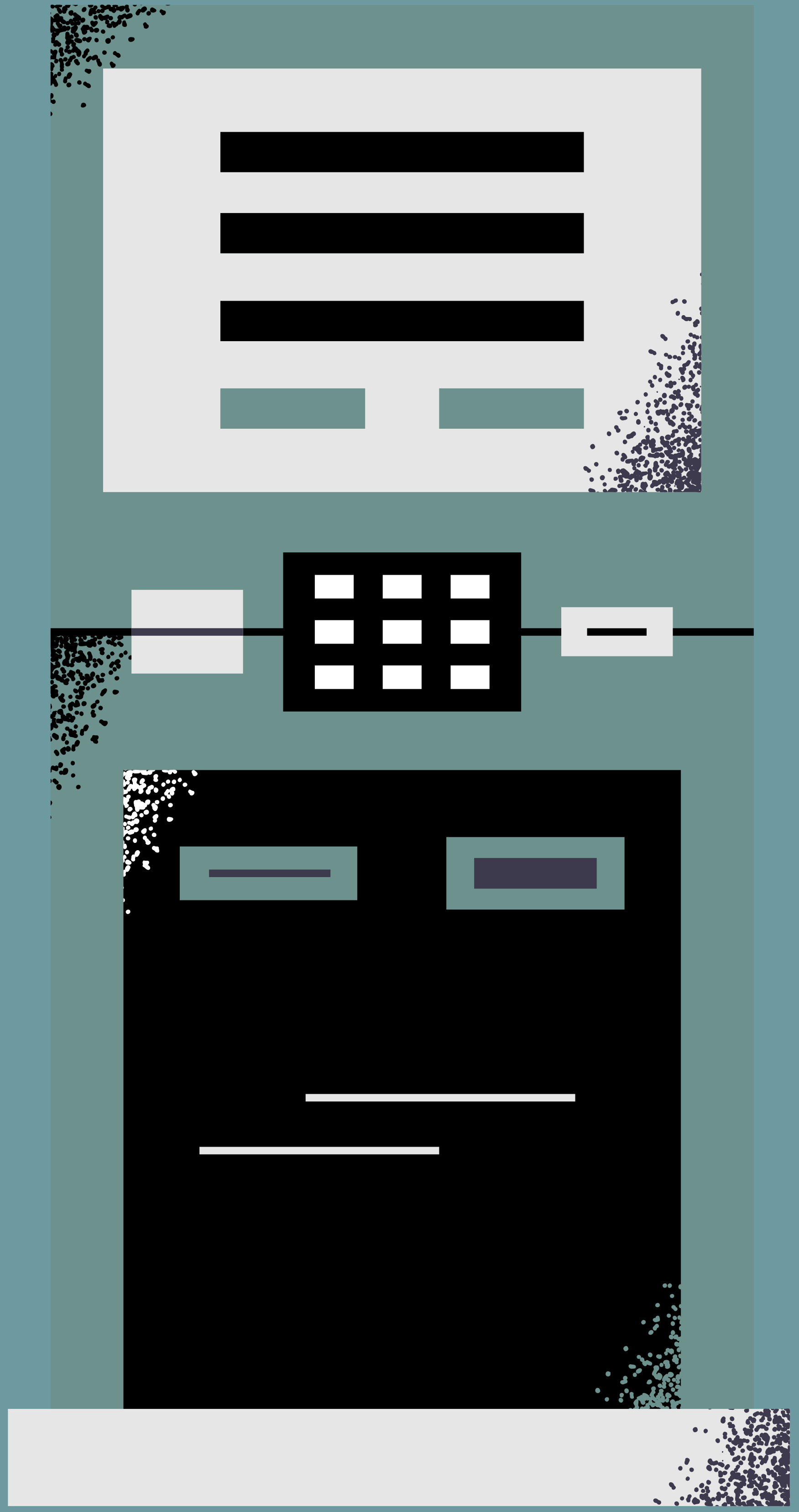
5.BACK TO MAIN MENU

Return to first screen.

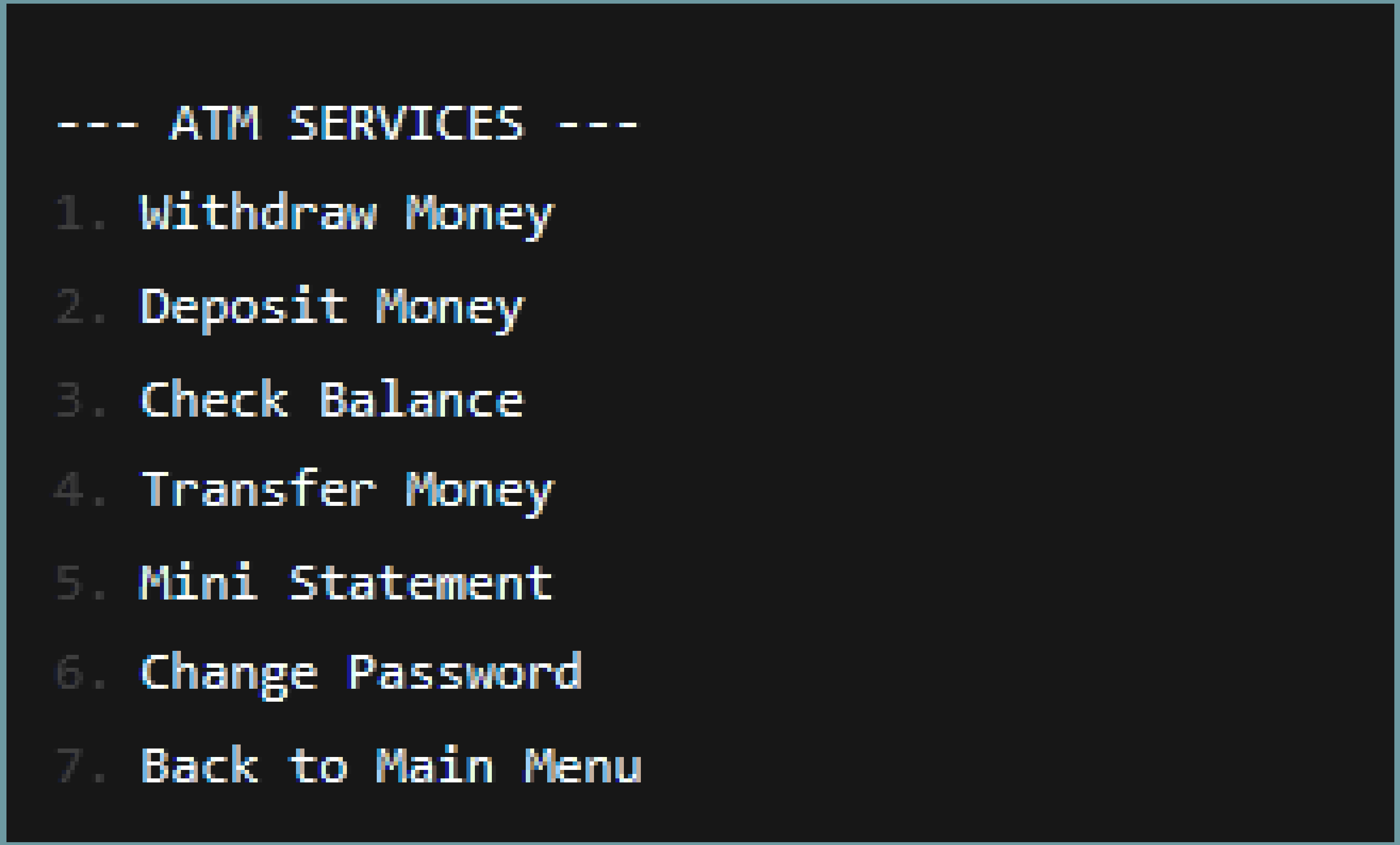
ATM SERVICES

Users log in with account number and password.
Options include:
- Withdraw Money
- Deposit Money
- Check Balance
- Transfer Money
- Mini Statement
- Change Password
- Back to Main Menu

Transactions are stored in the `transactions` array with counts in `transCount` to ensure accurate tracking and prevent overdrawing

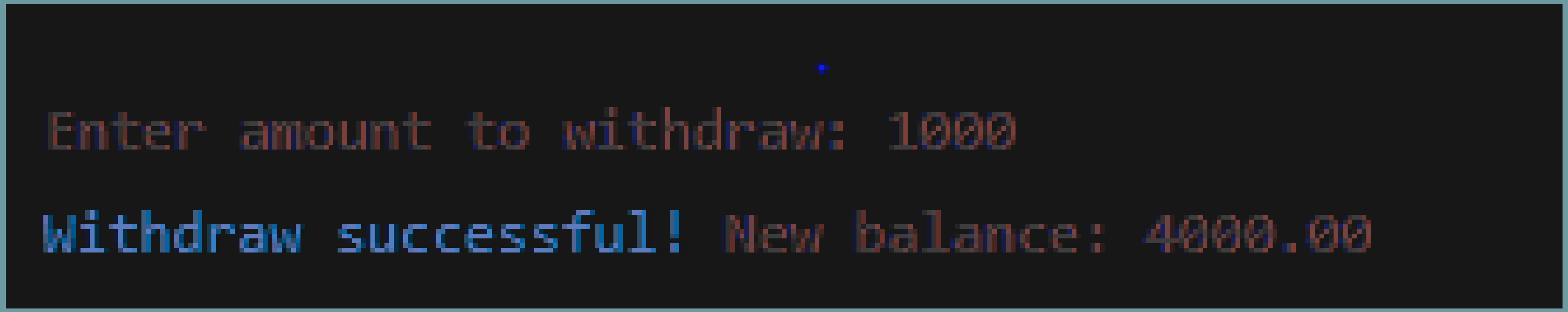


ATM MENU APPEARS



```
--- ATM SERVICES ---
1. Withdraw Money
2. Deposit Money
3. Check Balance
4. Transfer Money
5. Mini Statement
6. Change Password
7. Back to Main Menu
```

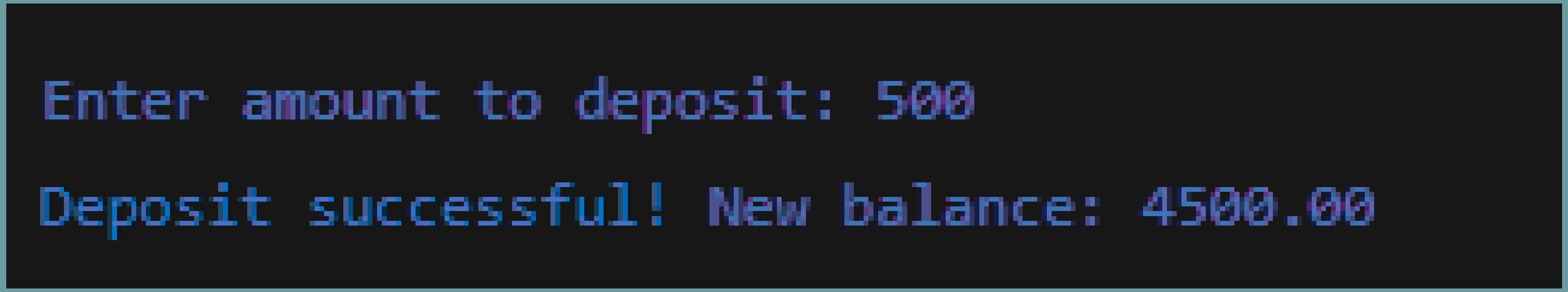
1.WITHDRAW MONEY



```
Enter amount to withdraw: 1000
Withdraw successful! New balance: 4000.00
```

As you withdraw your money, new balance will appear. Transaction is saved as:transactions[accIndex][0] = "Withdraw: -1000.00"

2.Deposit Money



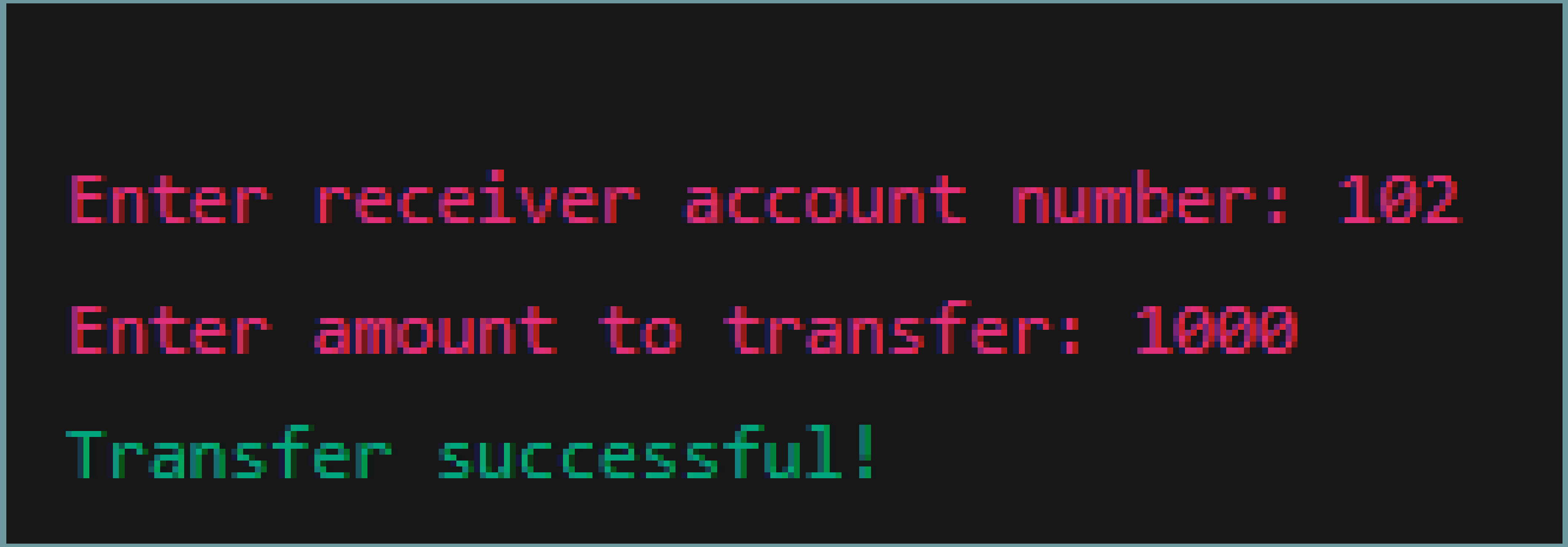
```
Enter amount to deposit: 500
Deposit successful! New balance: 4500.00
```

- Transaction:"Deposit: +500.00"

3. Check Balance

- By simply clicking the option you may check the balance.

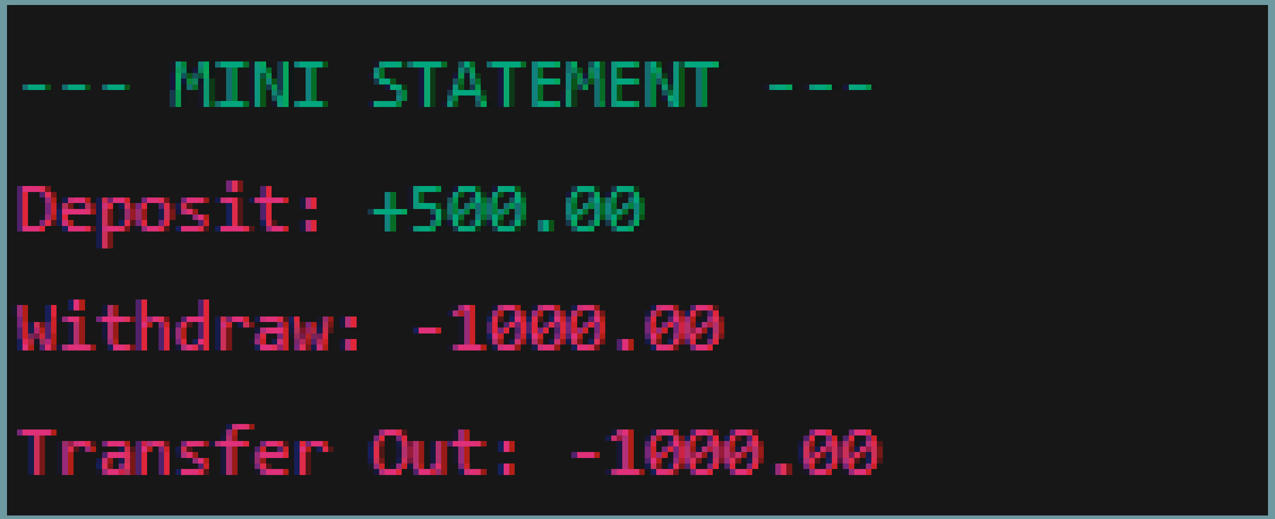
4.Transfer Money



```
Enter receiver account number: 102
Enter amount to transfer: 1000
Transfer successful!
```

Deducts 1000 from sender (101)
Adds 1000 to receiver (102)
Saves both transactions:
"Transfer Out: -1000.00"
"Transfer In: +1000.00"

5.Mini Statement



```
--- MINI STATEMENT ---
Deposit: +500.00
Withdraw: -1000.00
Transfer Out: -1000.00
```

- Shows last few transition for the account.

HELP DESK.

Provides instructions for:

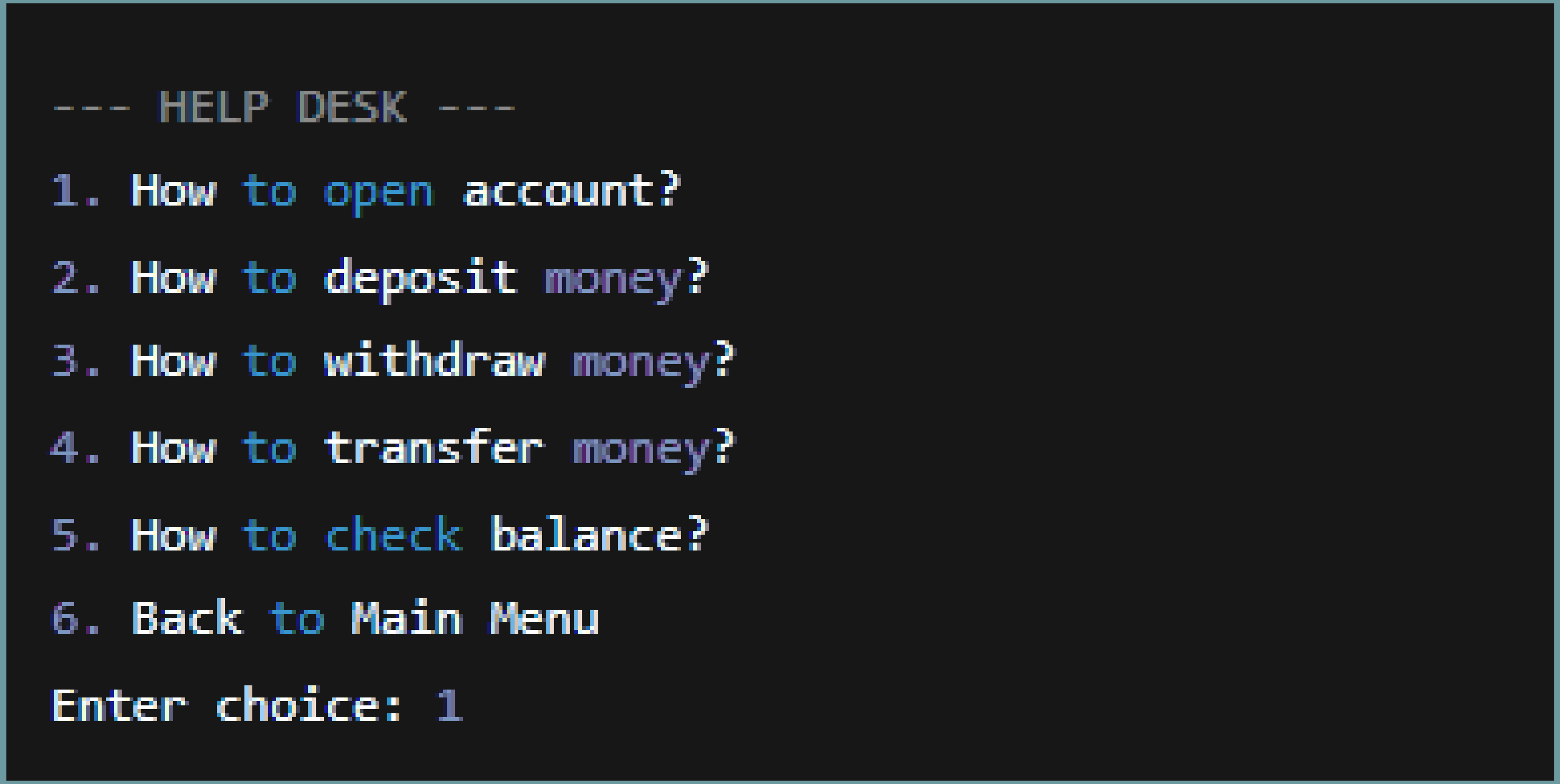
- Opening accounts
- Depositing
- Withdrawing
- Transferring money
- Checking balances

A loop allows multiple views before returning to the main menu.

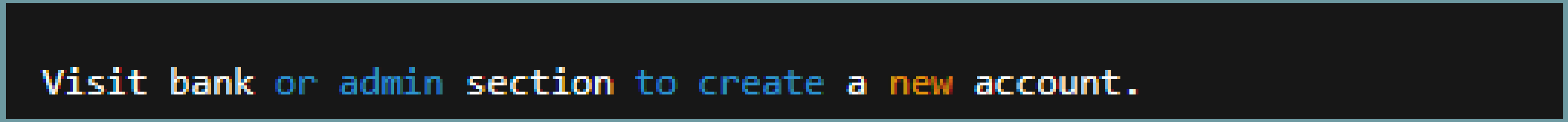
HELP DESK

- This menu runs in a loop (so you can ask multiple questions).
- When you choose option 6, it breaks and goes back to the main menu.

t

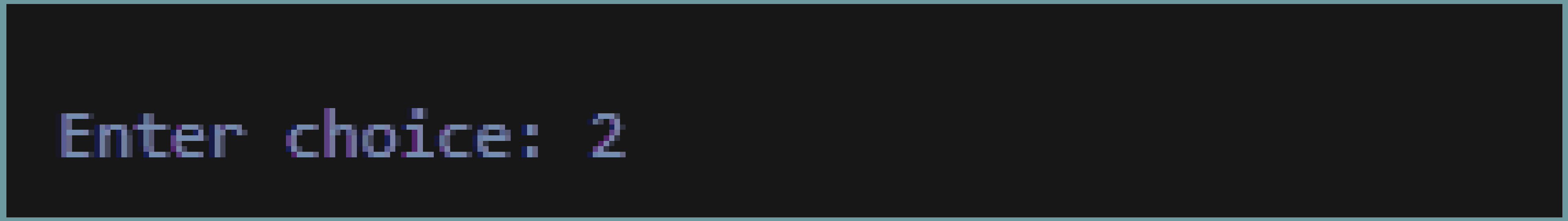


```
--- HELP DESK ---
1. How to open account?
2. How to deposit money?
3. How to withdraw money?
4. How to transfer money?
5. How to check balance?
6. Back to Main Menu
Enter choice: 1
```

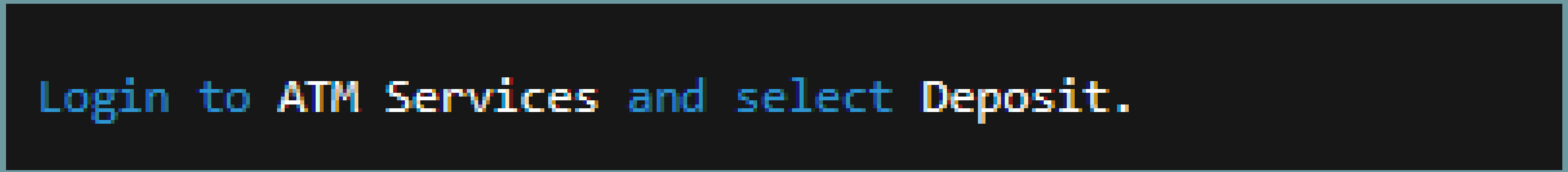


```
Visit bank or admin section to create a new account.
```

- The program tells you that account creation is only possible by the Admin (through option 1 in the main menu).
- Normal users cannot create accounts themselves

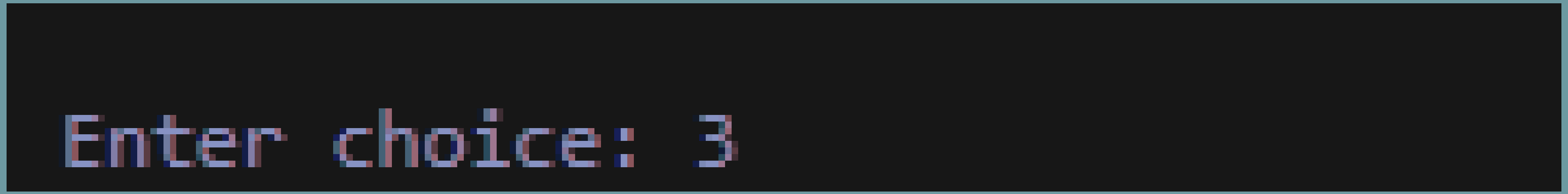


```
Enter choice: 2
```

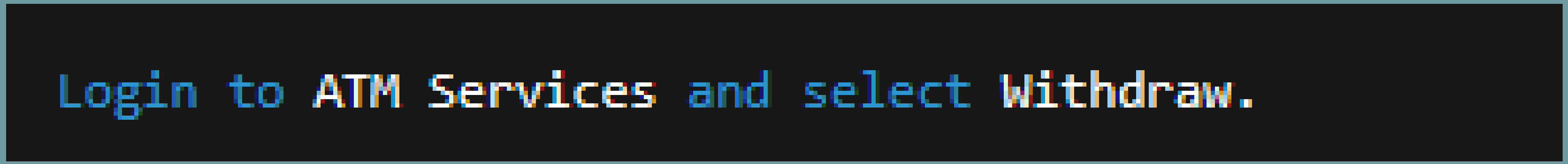


```
Login to ATM Services and select Deposit.
```

- It reminds users that to deposit money, they must log in through ATM Services (option 2) in the main menu.
- Once logged in, they should select the Deposit option (ATM menu option 2).

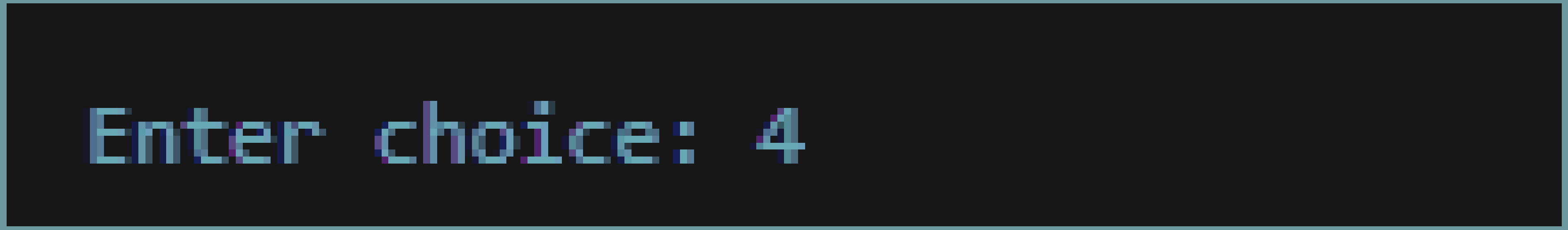


```
Enter choice: 3
```

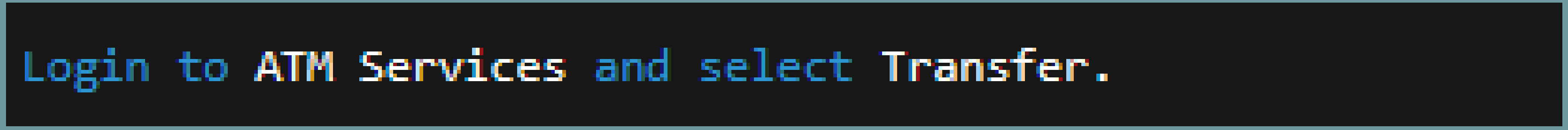


```
Login to ATM Services and select Withdraw.
```

- Users can withdraw money only from the ATM Services menu after logging in.
- They should select option 1 (“Withdraw Money”) once inside.



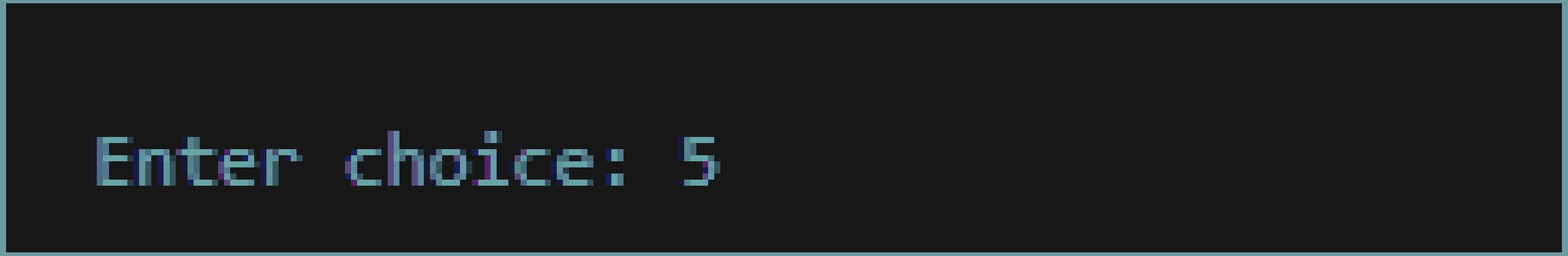
```
Enter choice: 4
```



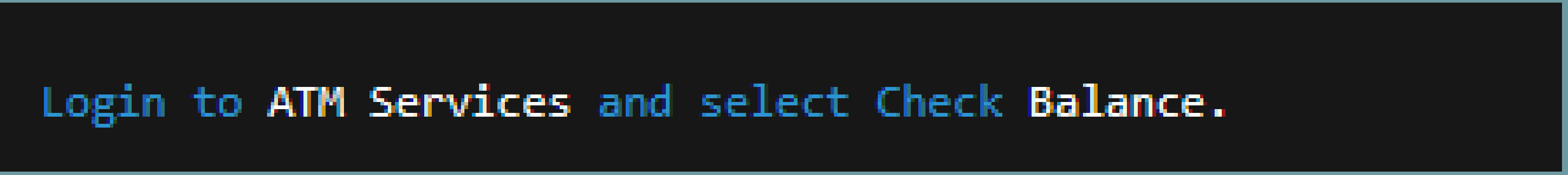
```
Login to ATM Services and select Transfer.
```

t

- Explains that money transfer between accounts is done inside ATM Services.
- Once logged in, select option 4 (“Transfer Money”).



```
Enter choice: 5
```

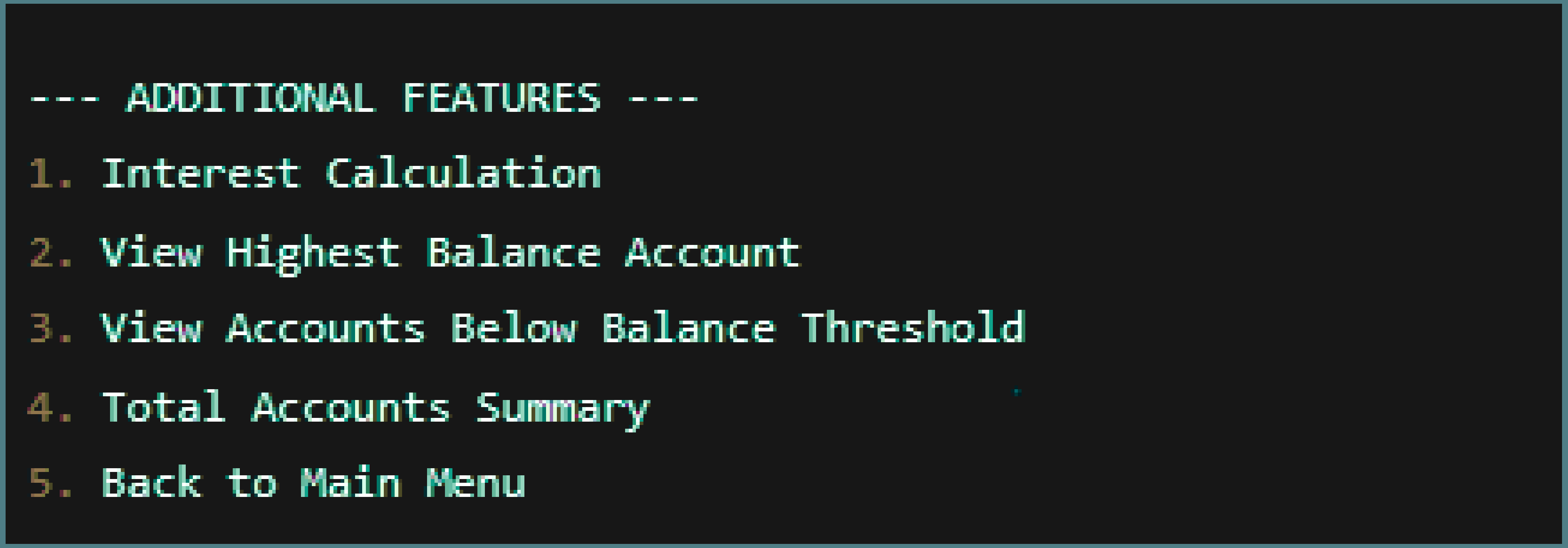


```
Login to ATM Services and select Check Balance.
```

- Informs the user that to check their account balance, they need to log in to ATM Services.
- Then select option 3 (“Check Balance”).

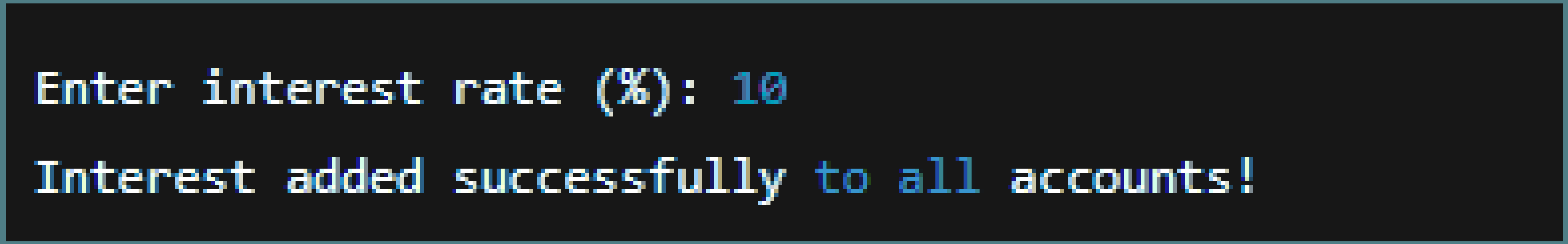
ADDITIONAL FEATURES

- Interest Calculation: Apply interest to all accounts.
- - View Highest Balance Account: Find the account with maximum
- balance.
- - Accounts Below Balance Threshold: List accounts under a
- specified balance.
- - Total Accounts Summary: Display total accounts.



```
--- ADDITIONAL FEATURES ---
1. Interest Calculation
2. View Highest Balance Account
3. View Accounts Below Balance Threshold
4. Total Accounts Summary
5. Back to Main Menu
```

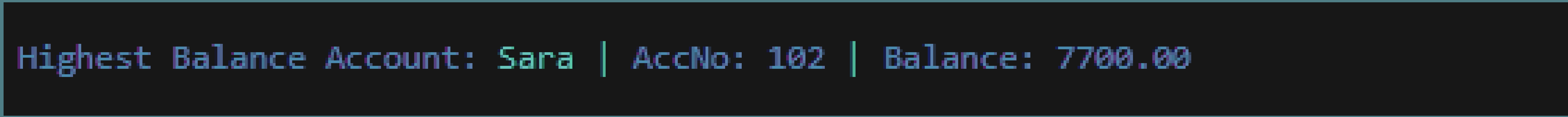
1. INTEREST CALCULATION!



```
Enter interest rate (%): 10
Interest added successfully to all accounts!
```

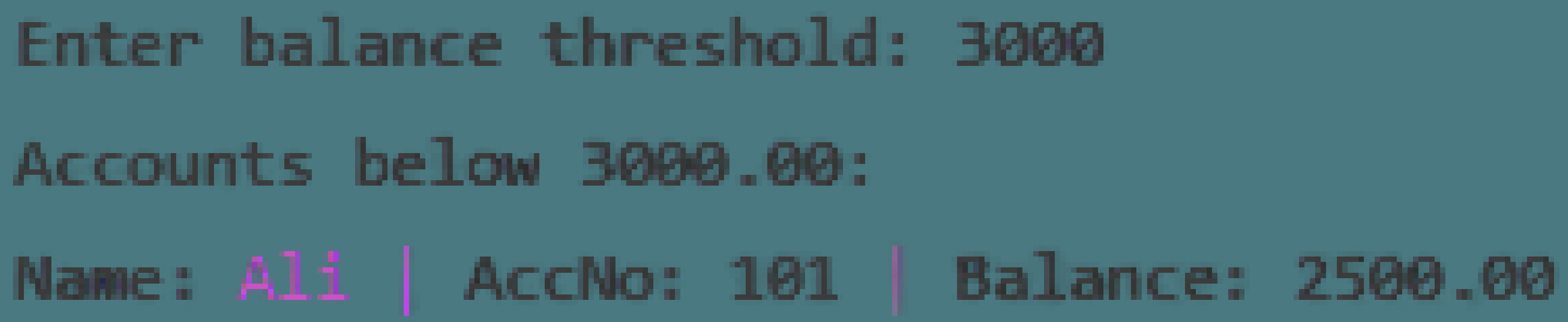
- Every balance increases by 10%

2. HIGHEST BALANCE ACCOUNT



```
Highest Balance Account: Sara | AccNo: 102 | Balance: 7700.00
```

3. ACCOUNTS BELOW BALANCE THRESHOLD



```
Enter balance threshold: 3000
Accounts below 3000.00:
Name: Ali | AccNo: 101 | Balance: 2500.00
```

4. Total Accounts Summary

5. EXIT

PROBLEMS WE FACED AND HOW WE OVERCOME THEM?

1. Managing Multiple Accounts

Solution: Arrays (names[], accNo[], passwords[], balance[]) store multiple accounts efficiently.

2. Tracking Transactions

Solution: 3D array transactions[100][10][100] and transCount[100] track each account's transactions.

3. String Comparison for Passwords

Solution: strcmp() used instead of ==.

4. Deleting Accounts Without Gaps

Solution: Shift subsequent array elements after deletion.

5. Avoiding Negative Balances

Solution: Conditional check if(balance[accIndex] >= amt)

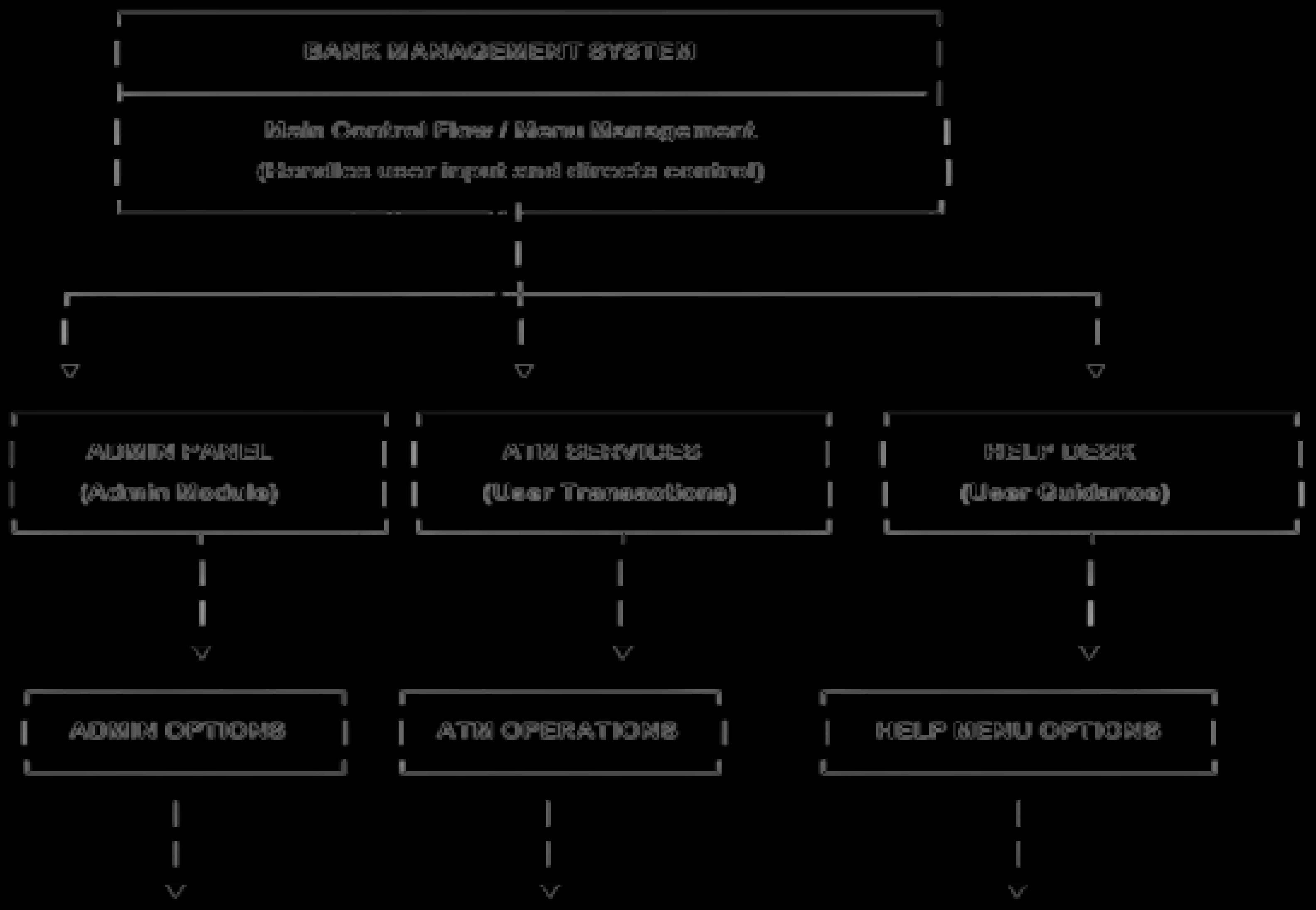
6. Continuous Operation

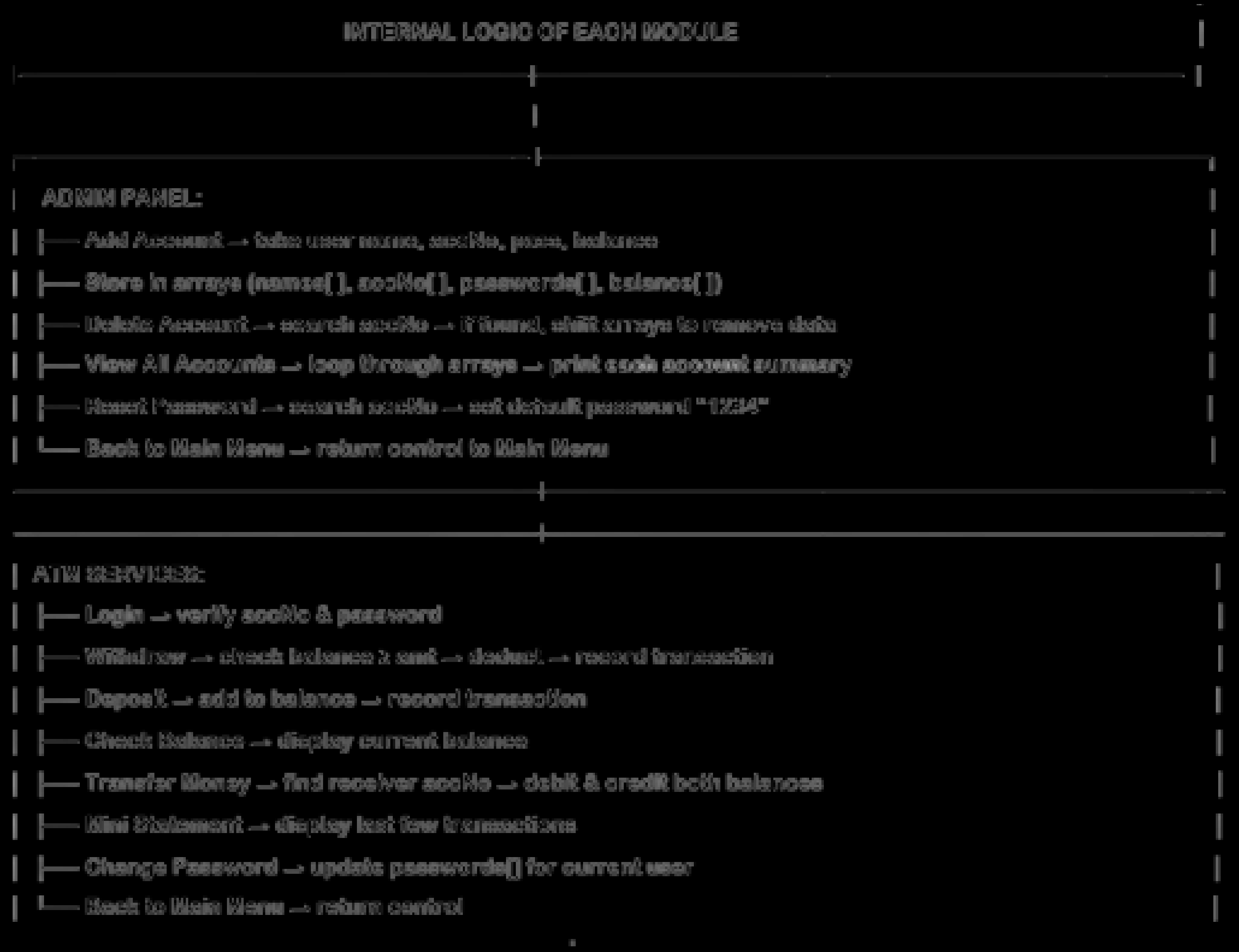
Solution: Infinite loop while(1) allows multiple operations.

7. Admin Authentication Security

Solution: Separate admin variable required for admin access.

ARCHITECTURAL DIAGRAM

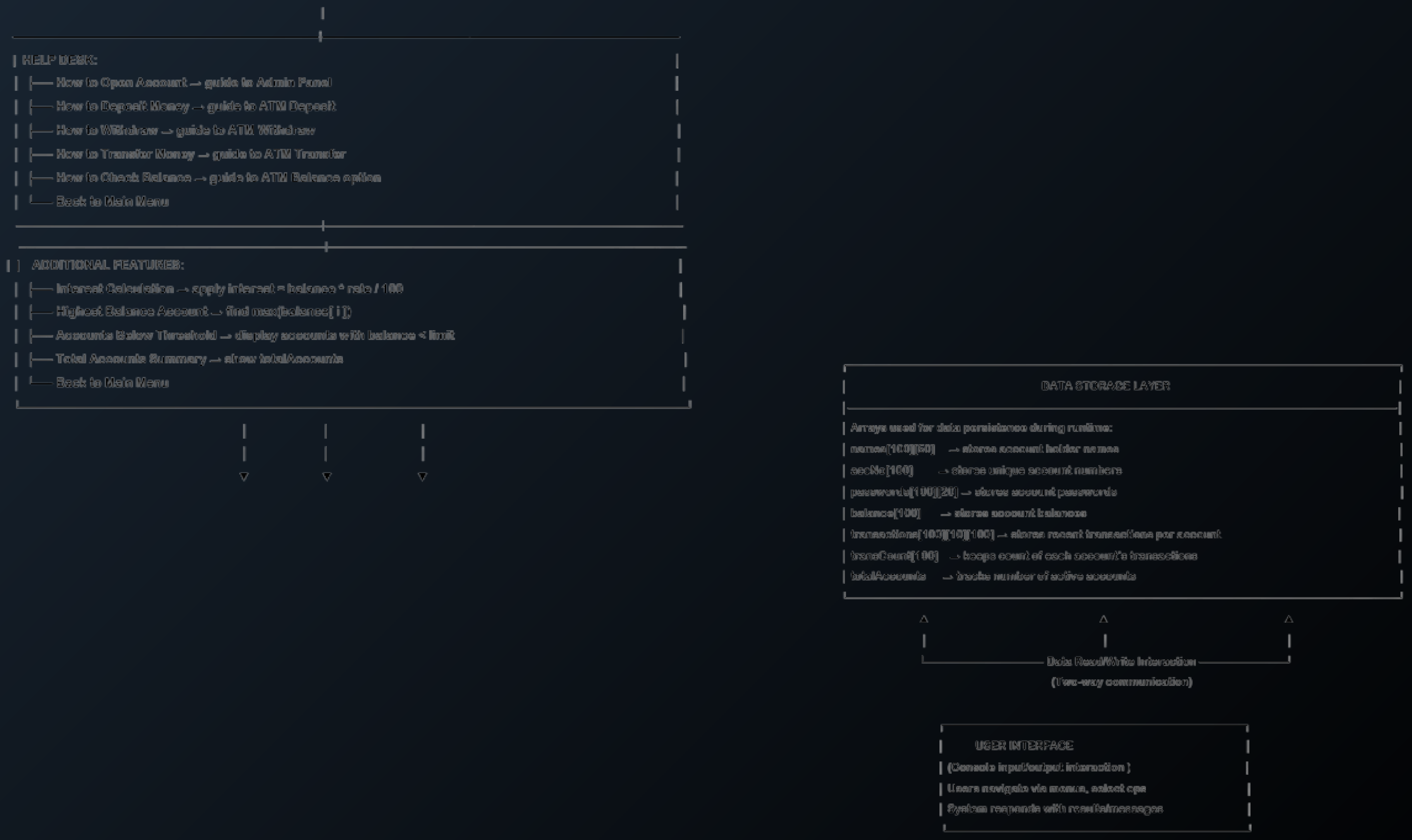




INTERNAL LOGIC OF EACH MODULE

```
ADMIN PANEL:
├── Add Account → take user name, accNo, pass, balance
├── Store in arrays (names[ ], accNo[ ], passwords[ ], balance[ ])
├── Delete Account → search accNo → if found, shift arrays to remove data
├── View All Accounts → loop through arrays → print each account summary
├── Reset Password → search accNo → set default password "1234"
└── Back to Main Menu → return control to Main Menu
```

```
ATM SERVICES:
├── Login → verify accNo & password
├── Withdraw → check balance ≥ amt → deduct → record transaction
├── Deposit → add to balance → record transaction
├── Check Balance → display current balance
├── Transfer Money → find receiver accNo → debit & credit both balances
├── Mini Statement → display last few transactions
├── Change Password → update passwords[] for current user
└── Back to Main Menu → return control
```



```
HELP DESK:
├── How to Open Account → guide to Admin Panel
├── How to Deposit Money → guide to ATM Deposit
├── How to Withdraw → guide to ATM Withdraw
├── How to Transfer Money → guide to ATM Transfer
├── How to Check Balance → guide to ATM Balance option
└── Back to Main Menu

ADDITIONAL FEATURES:
├── Interest Calculation → apply interest = balance * rate / 100
├── Highest Balance Account → find max(balance[ i ])
├── Accounts Below Threshold → display accounts with balance < limit
├── Total Accounts Summary → show totalAccounts
└── Back to Main Menu
```

```
DATA STORAGE LAYER

Arrays used for data persistence during runtime:
names[100][50]     → stores account holder names
accNo[100]         → stores unique account numbers
passwords[100][20] → stores account passwords
balance[100]       → stores account balances
transactions[100][10][100] → stores recent transactions per account
transCount[100]    → keeps count of each account's transactions
totalAccounts      → tracks number of active accounts
```

Data Read/Write Interaction
(Two-way communication)

```
USER INTERFACE
(Console input/output interaction )
Users navigate via menus, select ops
System responds with results/messages
```

CONCLUSION

- Built using C programming to simulate real banking operations.
- Uses arrays to manage multiple accounts and transactions securely.
- Key functions: create/delete accounts, deposit, withdraw, transfer, mini-statement.
- Handles passwords, balance checks, and ensures smooth operation.
- Includes admin controls, interest calculation, and account summaries.
- Strengthens problem-solving, data handling, and user interaction skills

THANK YOU!