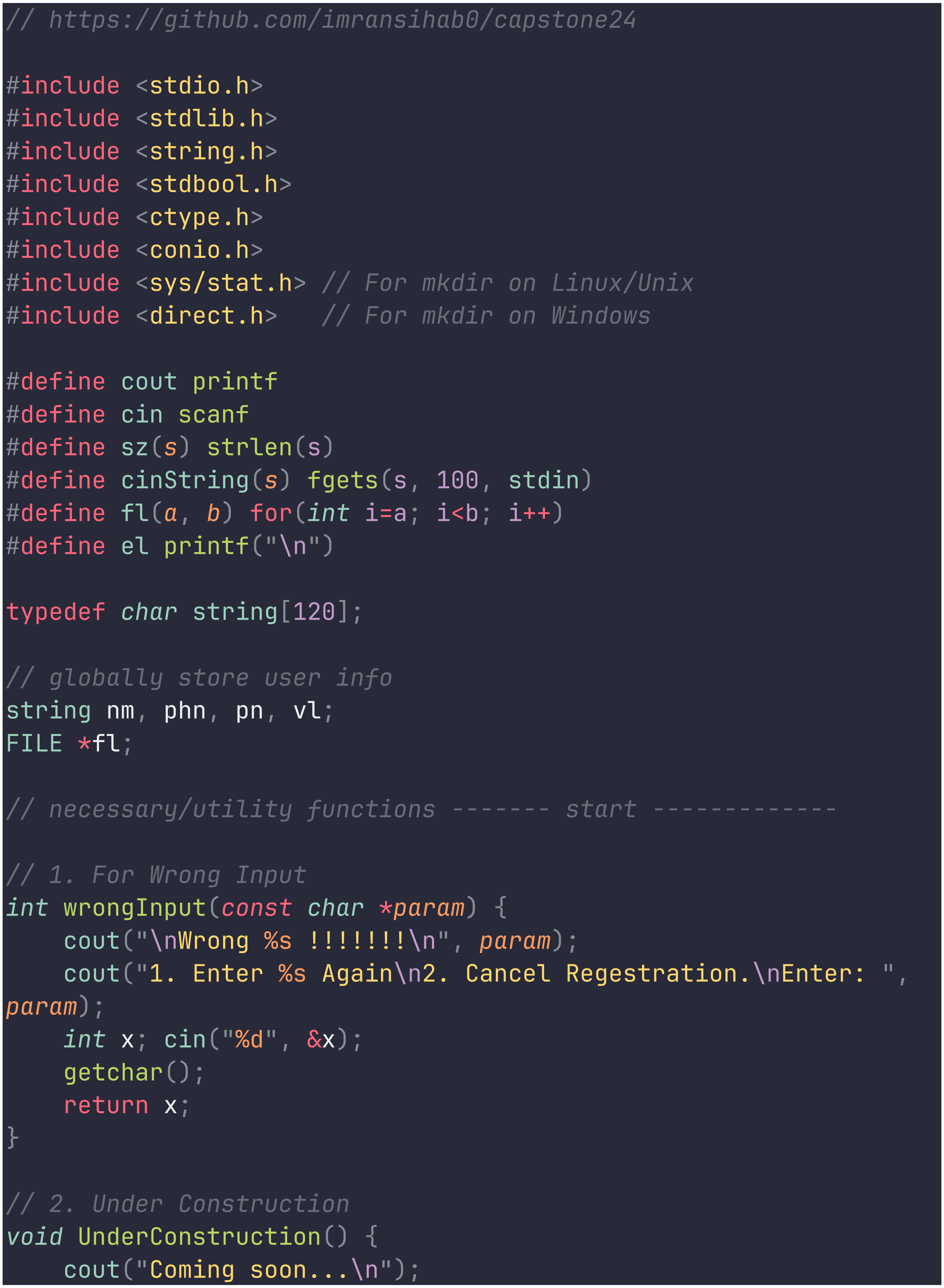
Bkash (Online Banking System)

Project By ->

Imranul Islam Shihab (232-35-733)

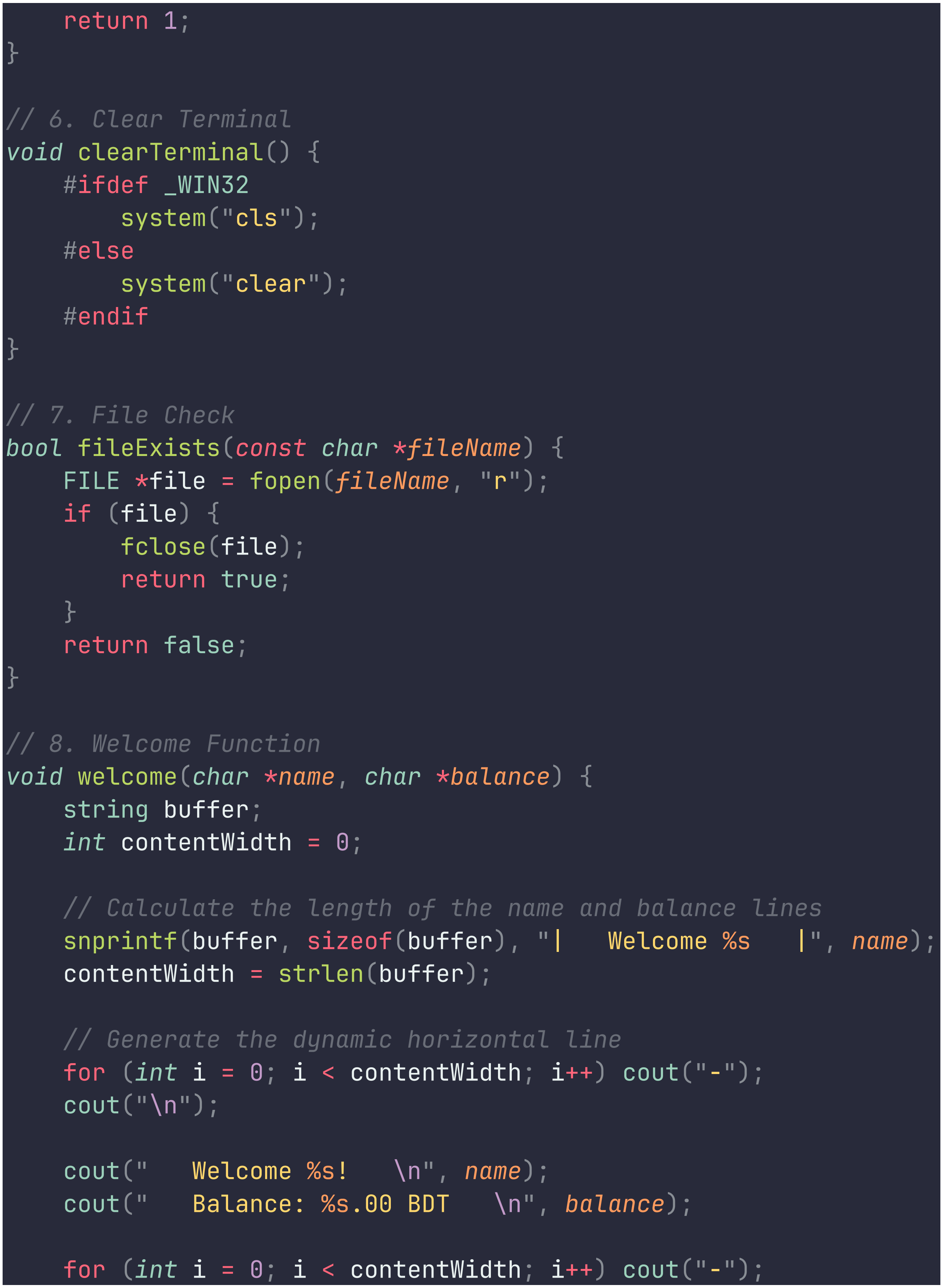
Meherab Hasan Fahim (232-35-670)

Liakat Hossain (232-35-642)

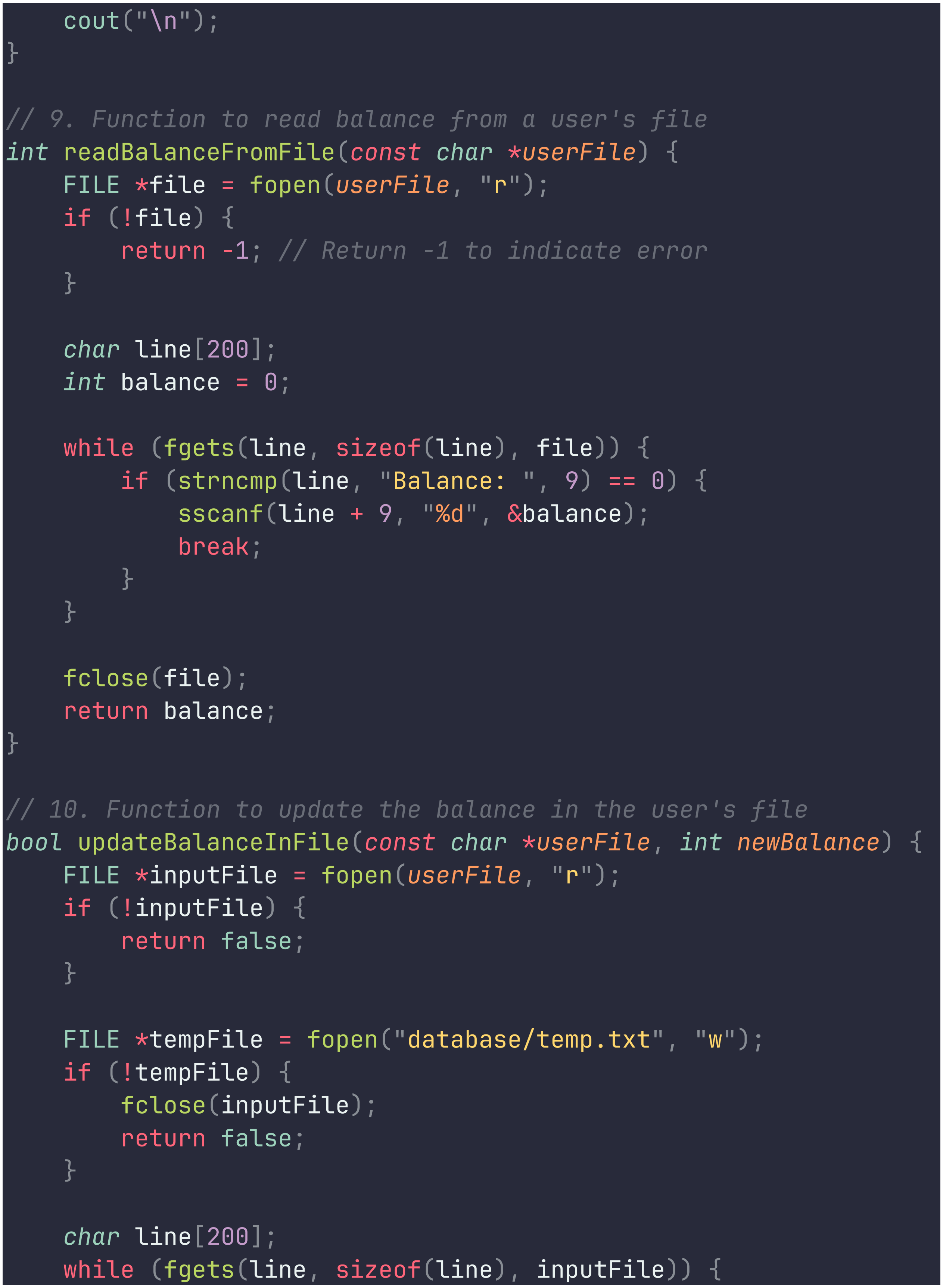


```c
// https://github.com/imransihab0/capstone24

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <ctype.h>
#include <conio.h>
#include <sys/stat.h> // For mkdir on Linux/Unix
#include <direct.h>   // For mkdir on Windows

#define cout printf
#define cin scanf
#define sz(s) strlen(s)
#define cinString(s) fgets(s, 100, stdin)
#define fl(a, b) for(int i=a; i<b; i++)
#define el printf("\n")

typedef char string[120];

// globally store user info
string nm, phn, pn, vl;
FILE *fl;

// necessary/utility functions ------- start -------------

// 1. For Wrong Input
int wrongInput(const char *param) {
    cout("\nWrong %s !!!!!!!\n", param);
    cout("1. Enter %s Again\n2. Cancel Regestration.\nEnter: ",
param);
    int x; cin("%d", &x);
    getchar();
    return x;
}

// 2. Under Construction
void UnderConstruction() {
    cout("Coming soon...\n");
```

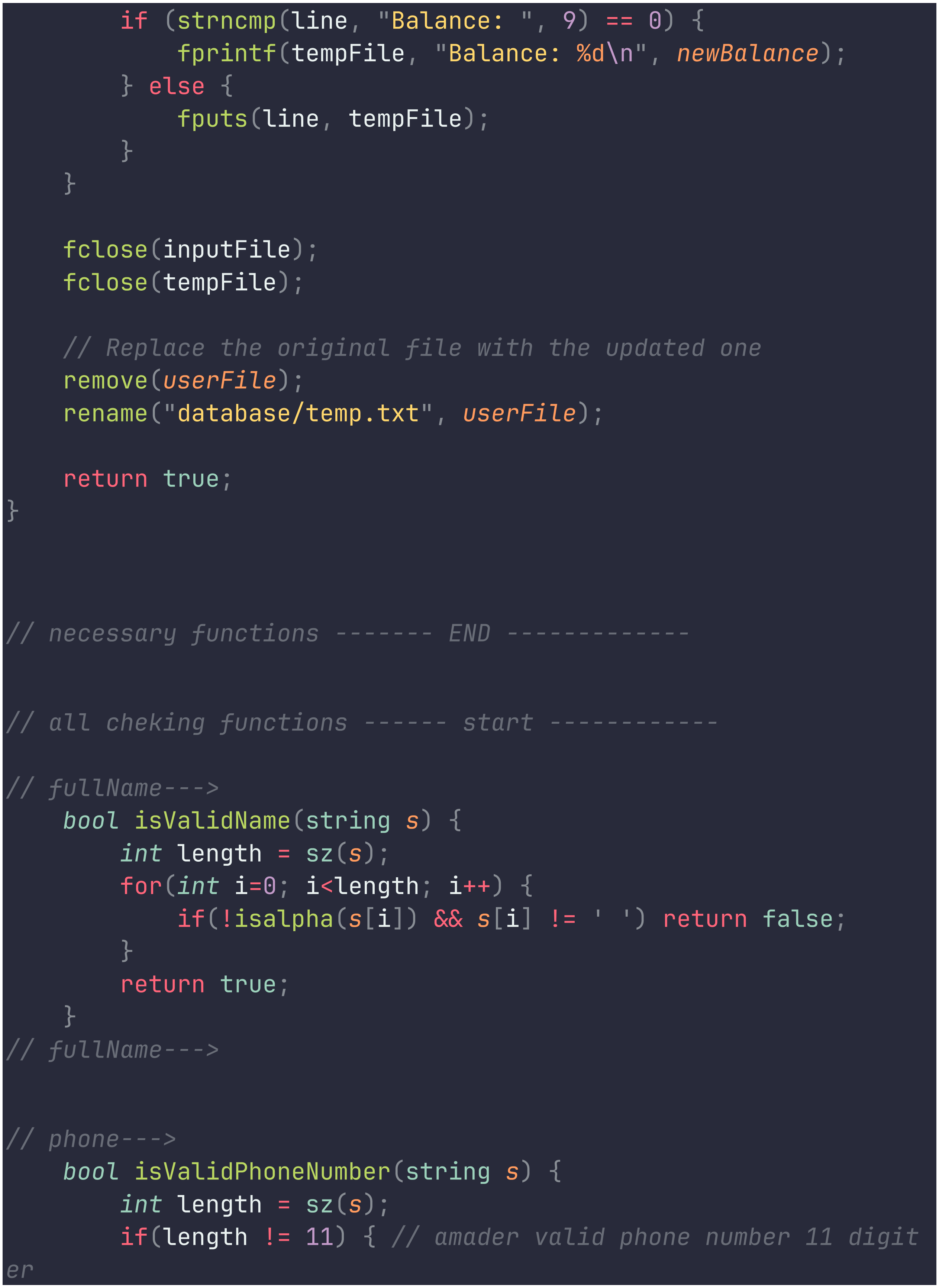
```
}

// 3. Congratulations
void conGo() {
    el;
    cout("|-------------------------------------------------- |\n");
    cout("|   Congratulations. Account Create Successfully!  |\n");
    cout("|-------------------------------------------------- |\n");
    el;
}

// 4. "*****" input function
void maskInput(char *input, int maxLen) {
    int i = 0;
    char ch;

    // _getch() function can take input char without printing in console

    while (i < maxLen - 1 && (ch = _getch()) != '\r') { // '\r' is Enter key in Windows
        if (ch == '\b' || ch == 127) {  // Handle backspace and ch127 is del key
            if (i > 0) {
                printf("\b \b");
                // Erase the last * by printing a backspace (\b), a space,
                // and another backspace to move the cursor back.
                i--;
            }
        } else {
            input[i++] = ch;
            printf("*");
        }
    }
    input[i] = '\0';  // Null-terminate string
    printf("\n");
}

// 5. Store data in database(text file named with the phone
number)
int storeDataToFile(string phoneNumber, string name, string pin)
{
    FILE *file;
    // Create a file with the phone number as the file name
(phoneNumber.txt)
    char fileName[120];

    // Create a "database" folder if it doesn't exist
    #ifdef _WIN32
        _mkdir("database");  // Windows
    #else
        mkdir("database", 0777);  // Linux/Unix
    #endif

    snprintf(fileName, sizeof(fileName), "database/%s.txt",
phoneNumber);

    if (fopen(fileName, "r") != NULL) {
        // File exists, user already registered
        el;
        cout("!!! A user with this phone number already
registered !!!\n");
        return 0;
    }

    file = fopen(fileName, "w");  // Open the file in write mode

    // Write the registration data to the file
    fprintf(file, "Name: %s\n", name);
    fprintf(file, "Number: %s\n", phoneNumber);
    fprintf(file, "PIN: %s\n", pin);
    fprintf(file, "Balance: 100\n");

    fclose(file);  // Close the file
```



```
    return 1;
}

// 6. Clear Terminal
void clearTerminal() {
    #ifdef _WIN32
        system("cls");
    #else
        system("clear");
    #endif
}

// 7. File Check
bool fileExists(const char *fileName) {
    FILE *file = fopen(fileName, "r");
    if (file) {
        fclose(file);
        return true;
    }
    return false;
}

// 8. Welcome Function
void welcome(char *name, char *balance) {
    string buffer;
    int contentWidth = 0;

    // Calculate the length of the name and balance lines
    snprintf(buffer, sizeof(buffer), "|   Welcome %s   |", name);
    contentWidth = strlen(buffer);

    // Generate the dynamic horizontal line
    for (int i = 0; i < contentWidth; i++) cout("-");
    cout("\n");

    cout("   Welcome %s!   \n", name);
    cout("   Balance: %s.00 BDT   \n", balance);

    for (int i = 0; i < contentWidth; i++) cout("-");
```

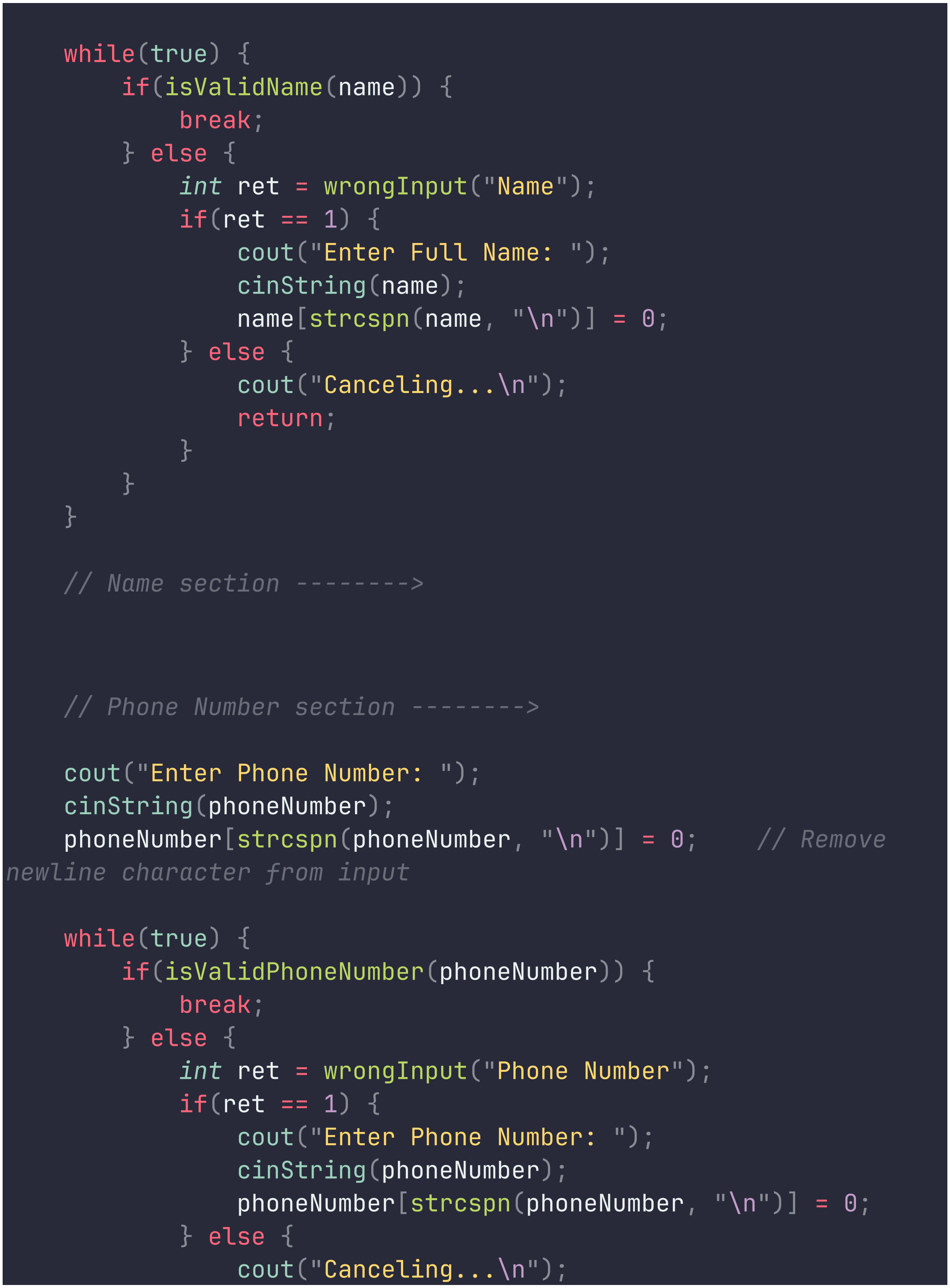


```
    cout("\n");
}

// 9. Function to read balance from a user's file
int readBalanceFromFile(const char *userFile) {
    FILE *file = fopen(userFile, "r");
    if (!file) {
        return -1; // Return -1 to indicate error
    }

    char line[200];
    int balance = 0;

    while (fgets(line, sizeof(line), file)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &balance);
            break;
        }
    }

    fclose(file);
    return balance;
}

// 10. Function to update the balance in the user's file
bool updateBalanceInFile(const char *userFile, int newBalance) {
    FILE *inputFile = fopen(userFile, "r");
    if (!inputFile) {
        return false;
    }

    FILE *tempFile = fopen("database/temp.txt", "w");
    if (!tempFile) {
        fclose(inputFile);
        return false;
    }

    char line[200];
    while (fgets(line, sizeof(line), inputFile)) {
```

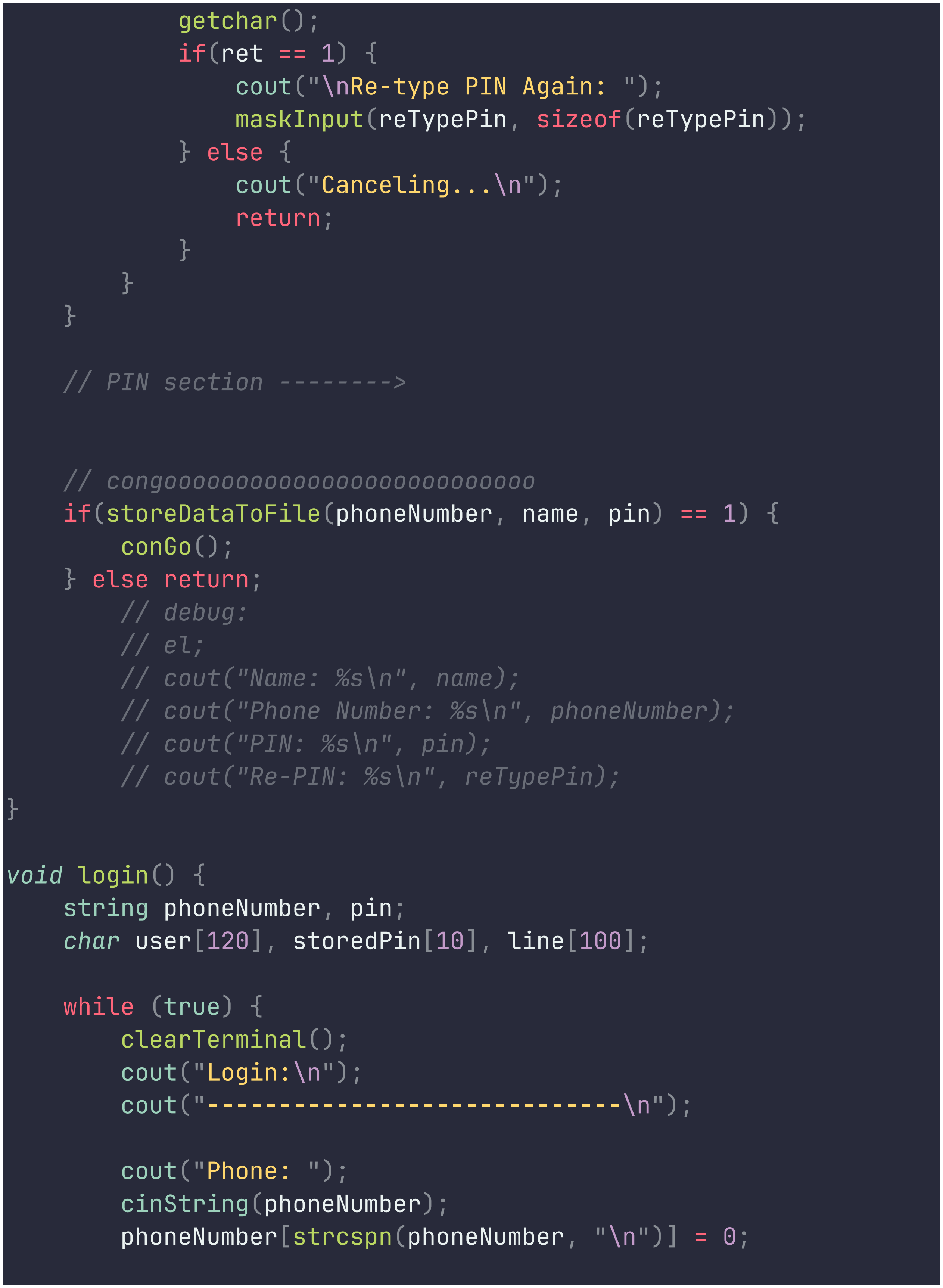


```
        if (strncmp(line, "Balance: ", 9) == 0) {
            fprintf(tempFile, "Balance: %d\n", newBalance);
        } else {
            fputs(line, tempFile);
        }
    }

    fclose(inputFile);
    fclose(tempFile);

    // Replace the original file with the updated one
    remove(userFile);
    rename("database/temp.txt", userFile);

    return true;
}



// necessary functions ------- END -------------


// all cheking functions ------ start ------------

// fullName--->
    bool isValidName(string s) {
        int length = sz(s);
        for(int i=0; i<length; i++) {
            if(!isalpha(s[i]) && s[i] != ' ') return false;
        }
        return true;
    }
// fullName--->


// phone--->
    bool isValidPhoneNumber(string s) {
        int length = sz(s);
        if(length != 11) { // amader valid phone number 11 digit
er
```

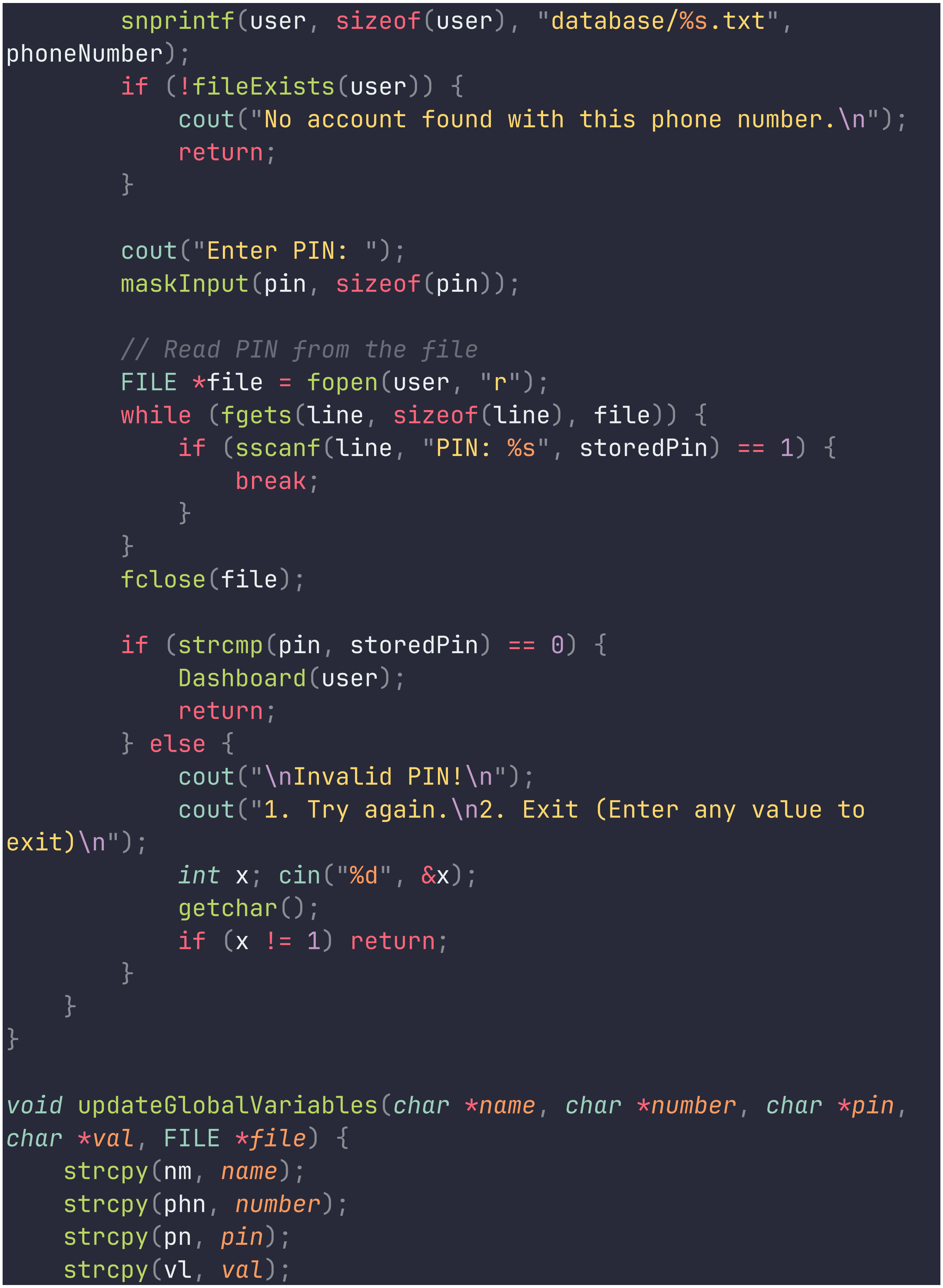
```
            return false;
        }

        for(int i=0; i<length; i++) {
            if(!isdigit(s[i])) return false;
        }
        return true;
    }
// phone--->


// pin---> <security is important>
    bool isValidPin(string s) {
        return sz(s) == 6;
    }
// pin--->



// all cheking functions ------ END ------------



void reg() {
    clearTerminal();

    string name;
    string phoneNumber;
    string pin;
    string reTypePin;

    cout("Regestration:\n");
    cout("-----------------------------\n");

    // Name section -------->

    cout("Enter Full Name: ");
    cinString(name);
    name[strcspn(name, "\n")] = 0;    // Remove newline character from input
```



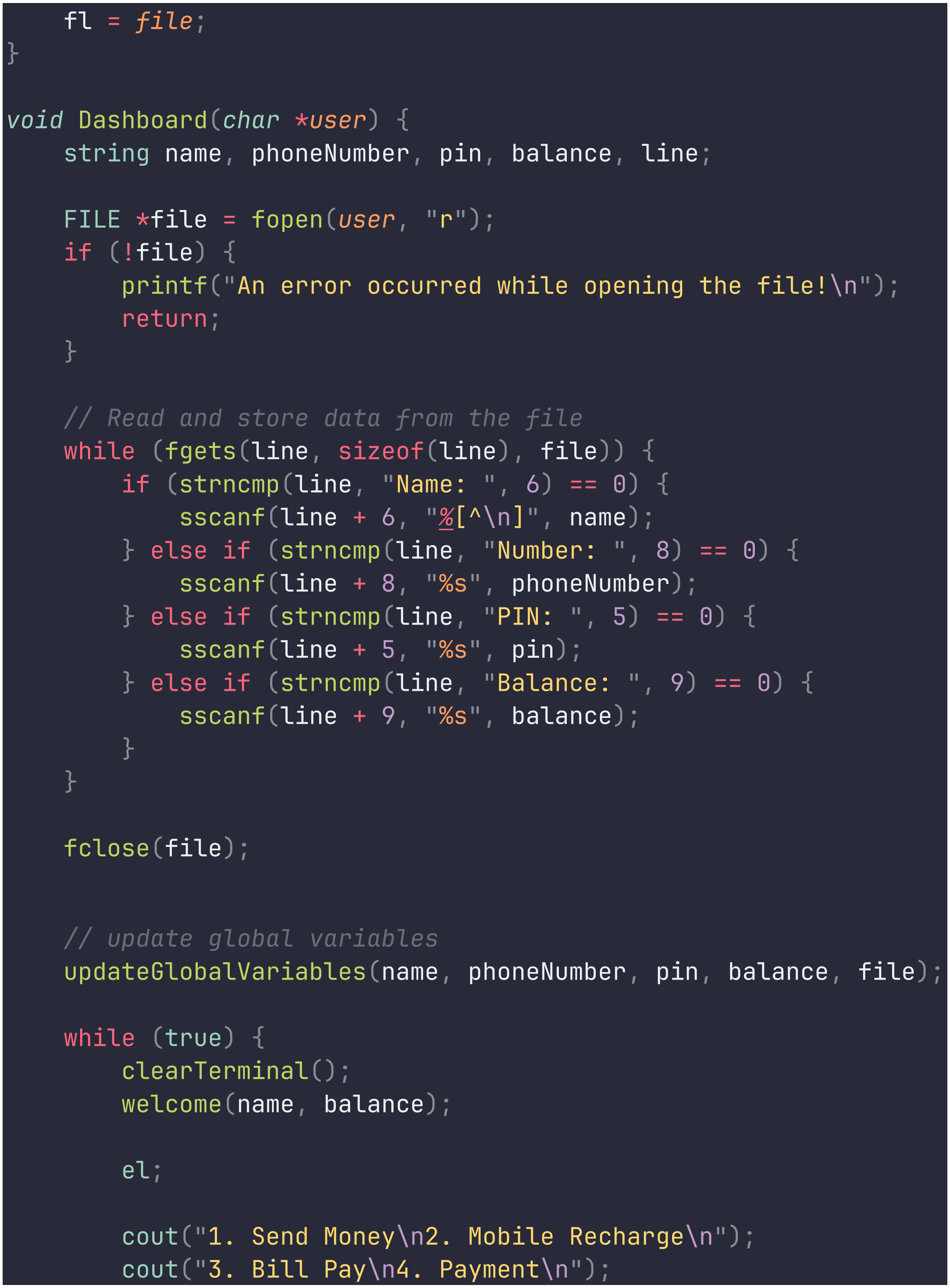
```
    while(true) {
        if(isValidName(name)) {
            break;
        } else {
            int ret = wrongInput("Name");
            if(ret == 1) {
                cout("Enter Full Name: ");
                cinString(name);
                name[strcspn(name, "\n")] = 0;
            } else {
                cout("Canceling...\n");
                return;
            }
        }
    }

    // Name section -------->



    // Phone Number section -------->

    cout("Enter Phone Number: ");
    cinString(phoneNumber);
    phoneNumber[strcspn(phoneNumber, "\n")] = 0;    // Remove
newline character from input

    while(true) {
        if(isValidPhoneNumber(phoneNumber)) {
            break;
        } else {
            int ret = wrongInput("Phone Number");
            if(ret == 1) {
                cout("Enter Phone Number: ");
                cinString(phoneNumber);
                phoneNumber[strcspn(phoneNumber, "\n")] = 0;
            } else {
                cout("Canceling...\n");
                return;
            }
        }
    }

    // Phone Number section -------->


    // PIN section -------->

    cout("Enter PIN (6 digits): ");
    maskInput(pin, sizeof(pin));

    while(true) {
        if(isValidPin(pin)) {
            break;
        } else {
            int ret = wrongInput("PIN");
            if(ret == 1) {
                cout("Enter PIN Again (6 digits): ");
                maskInput(pin, sizeof(pin));
            } else {
                cout("Canceling...\n");
                return;
            }
        }
    }

    // reType sectio-->
    cout("Re-type PIN: ");
    maskInput(reTypePin, sizeof(reTypePin));

    while(true) {
        if(strcmp(pin, reTypePin) == 0) {
            break;
        } else {
            cout("\nPIN did not matched\n1. Re-enter.\n2. Cancel.\nEnter: ");
            int ret; cin("%d", &ret);
```



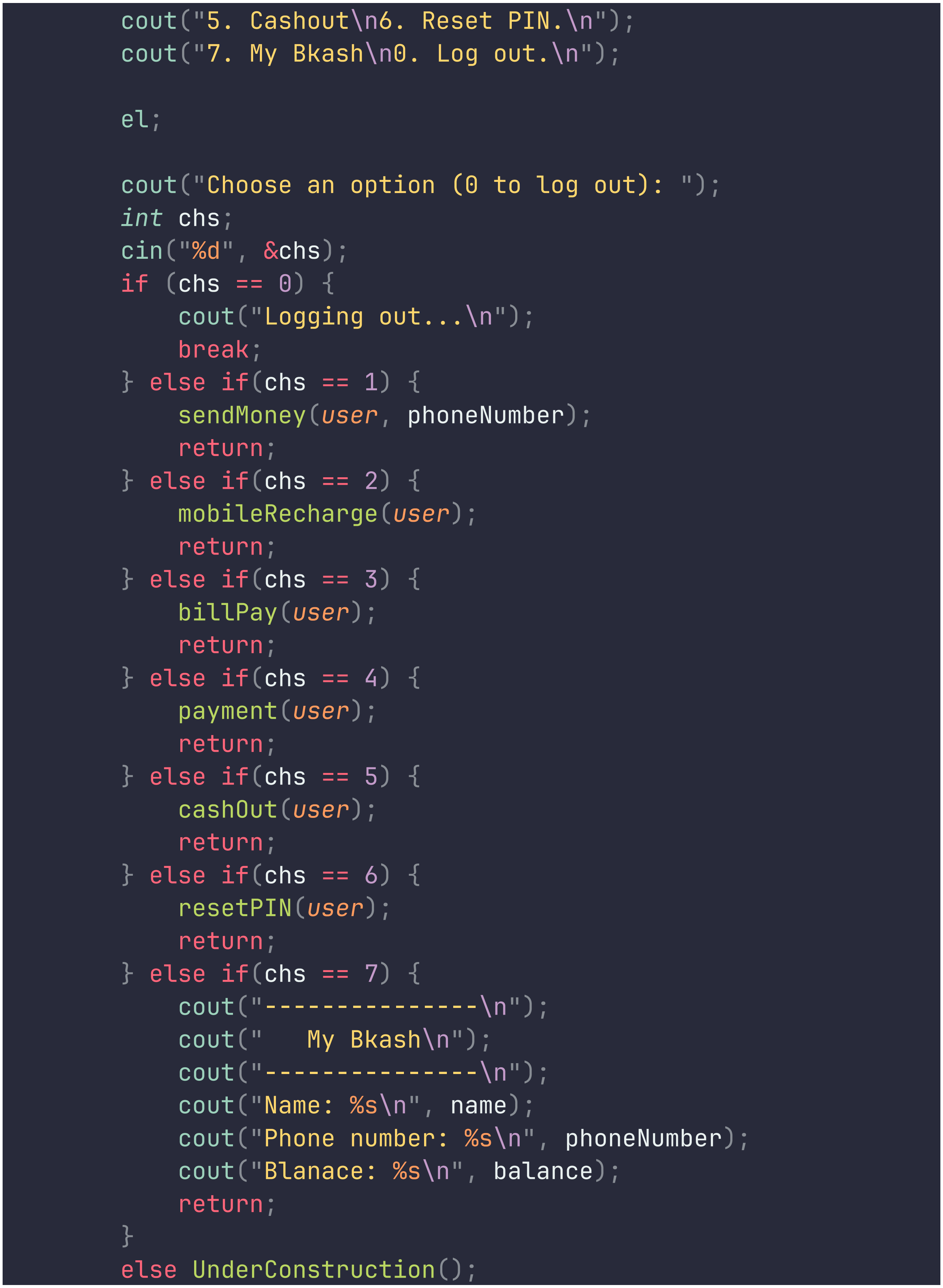
```
            getchar();
            if(ret == 1) {
                cout("\nRe-type PIN Again: ");
                maskInput(reTypePin, sizeof(reTypePin));
            } else {
                cout("Canceling...\n");
                return;
            }
        }
    }

    // PIN section -------->


    // congooooooooooooooooooooooooo
    if(storeDataToFile(phoneNumber, name, pin) == 1) {
        conGo();
    } else return;
        // debug:
        // el;
        // cout("Name: %s\n", name);
        // cout("Phone Number: %s\n", phoneNumber);
        // cout("PIN: %s\n", pin);
        // cout("Re-PIN: %s\n", reTypePin);
}

void login() {
    string phoneNumber, pin;
    char user[120], storedPin[10], line[100];

    while (true) {
        clearTerminal();
        cout("Login:\n");
        cout("-----------------------------\n");

        cout("Phone: ");
        cinString(phoneNumber);
        phoneNumber[strcspn(phoneNumber, "\n")] = 0;
```



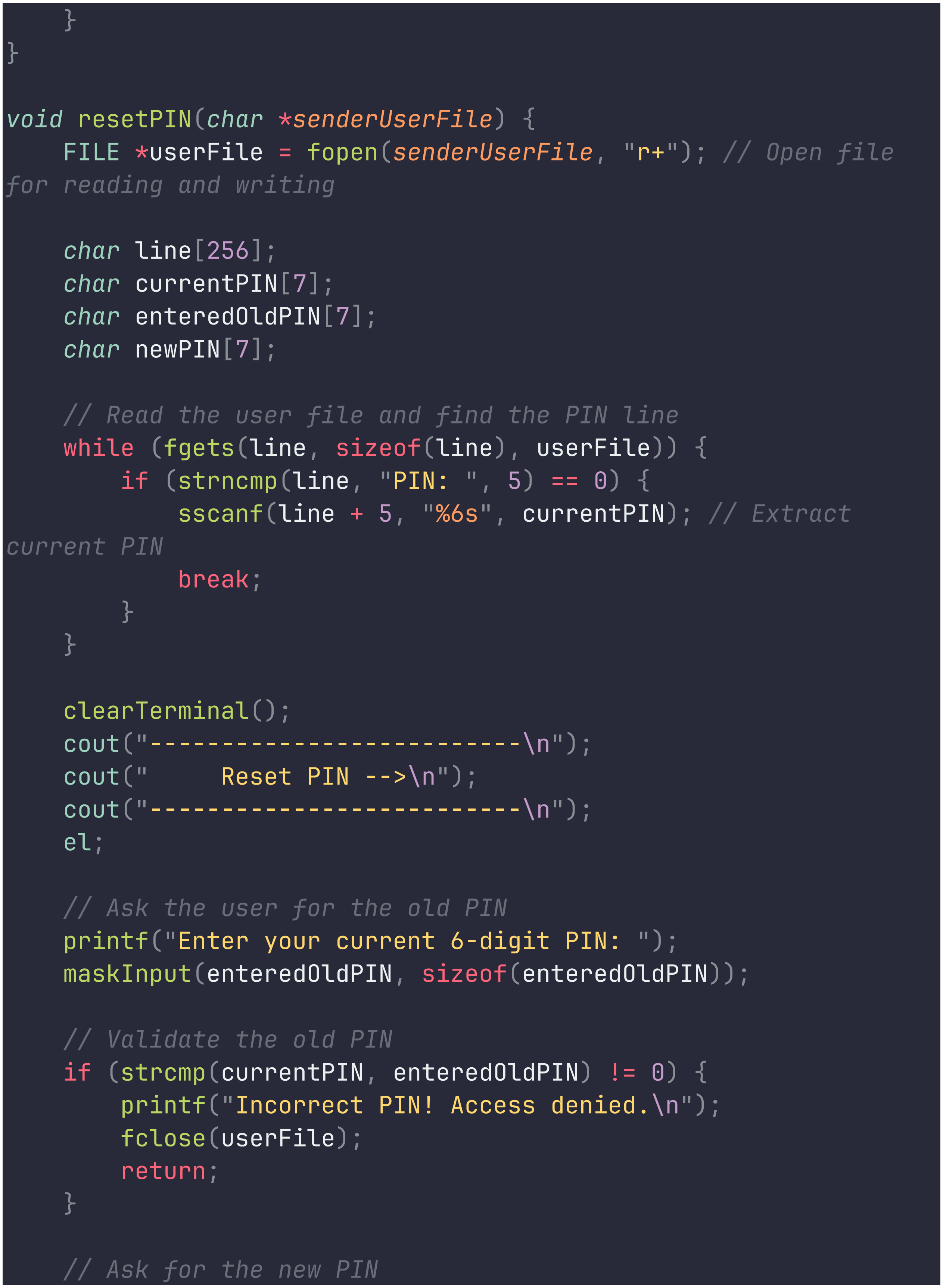
```
        snprintf(user, sizeof(user), "database/%s.txt",
phoneNumber);
        if (!fileExists(user)) {
            cout("No account found with this phone number.\n");
            return;
        }

        cout("Enter PIN: ");
        maskInput(pin, sizeof(pin));

        // Read PIN from the file
        FILE *file = fopen(user, "r");
        while (fgets(line, sizeof(line), file)) {
            if (sscanf(line, "PIN: %s", storedPin) == 1) {
                break;
            }
        }
        fclose(file);

        if (strcmp(pin, storedPin) == 0) {
            Dashboard(user);
            return;
        } else {
            cout("\nInvalid PIN!\n");
            cout("1. Try again.\n2. Exit (Enter any value to
exit)\n");
            int x; cin("%d", &x);
            getchar();
            if (x != 1) return;
        }
    }
}

void updateGlobalVariables(char *name, char *number, char *pin,
char *val, FILE *file) {
    strcpy(nm, name);
    strcpy(phn, number);
    strcpy(pn, pin);
    strcpy(vl, val);
```



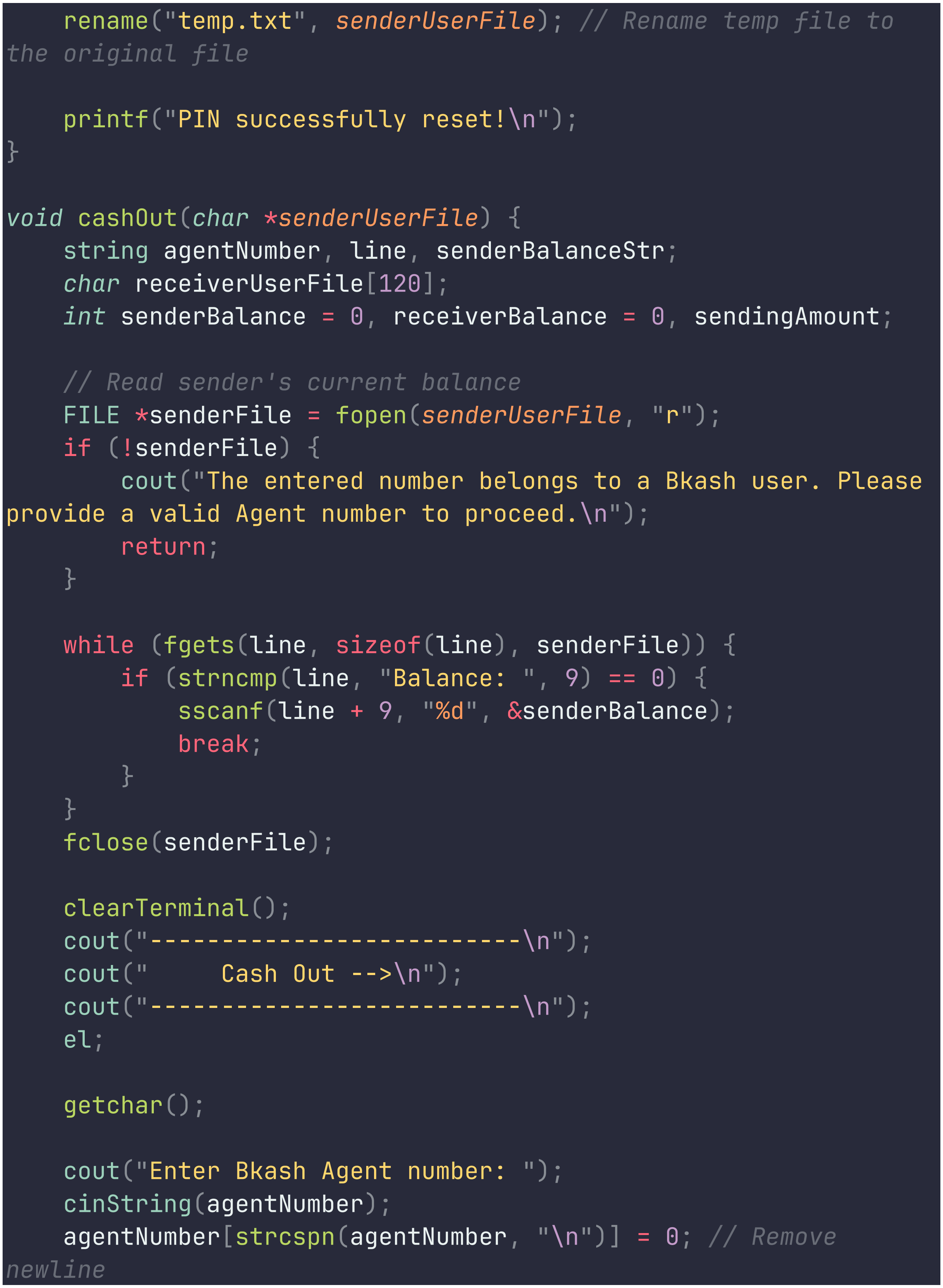
```
    fl = file;
}

void Dashboard(char *user) {
    string name, phoneNumber, pin, balance, line;

    FILE *file = fopen(user, "r");
    if (!file) {
        printf("An error occurred while opening the file!\n");
        return;
    }

    // Read and store data from the file
    while (fgets(line, sizeof(line), file)) {
        if (strncmp(line, "Name: ", 6) == 0) {
            sscanf(line + 6, "%[^\n]", name);
        } else if (strncmp(line, "Number: ", 8) == 0) {
            sscanf(line + 8, "%s", phoneNumber);
        } else if (strncmp(line, "PIN: ", 5) == 0) {
            sscanf(line + 5, "%s", pin);
        } else if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%s", balance);
        }
    }

    fclose(file);


    // update global variables
    updateGlobalVariables(name, phoneNumber, pin, balance, file);

    while (true) {
        clearTerminal();
        welcome(name, balance);

        el;

        cout("1. Send Money\n2. Mobile Recharge\n");
        cout("3. Bill Pay\n4. Payment\n");
```



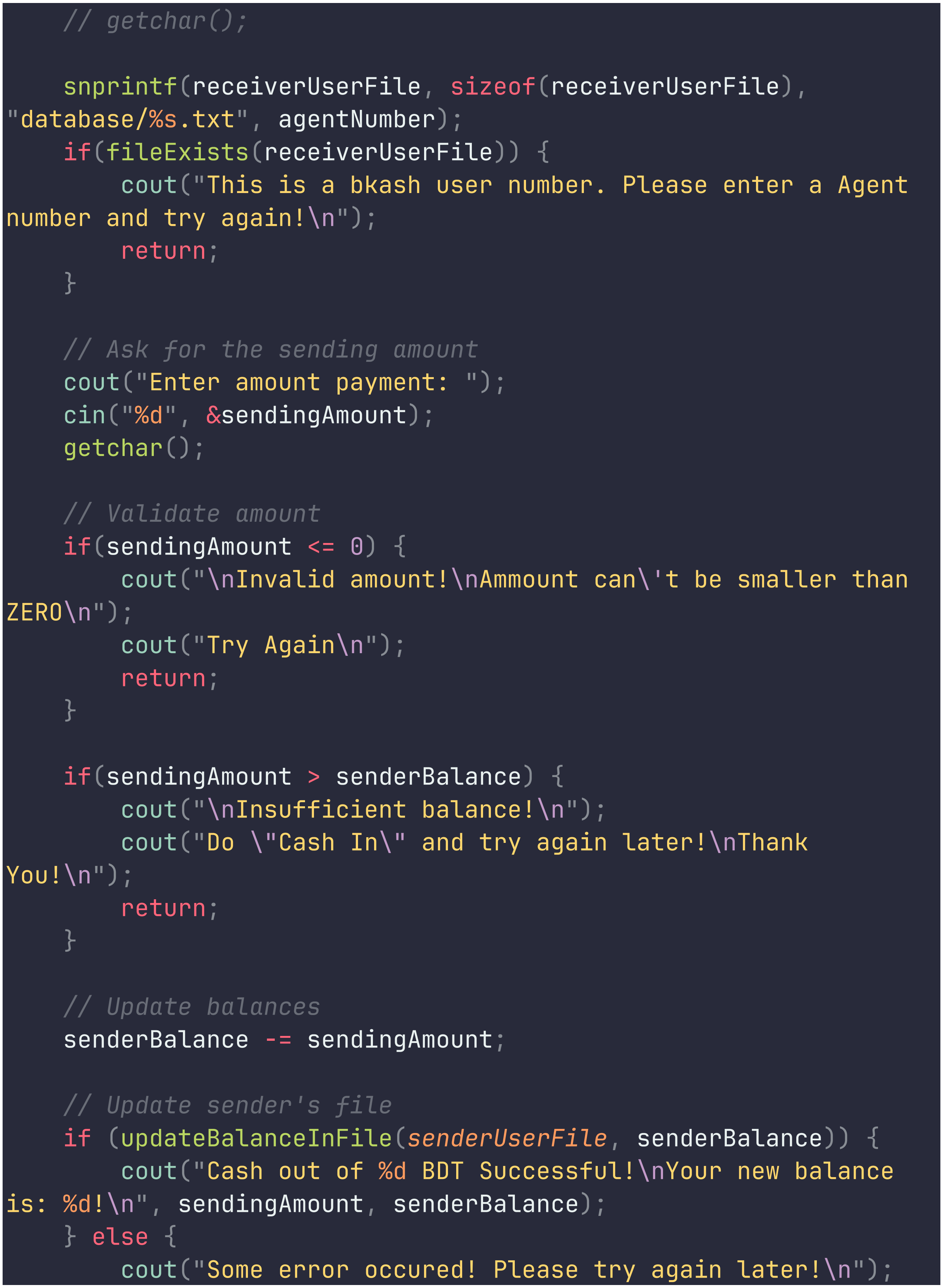
```
        cout("5. Cashout\n6. Reset PIN.\n");
        cout("7. My Bkash\n0. Log out.\n");

        el;

        cout("Choose an option (0 to log out): ");
        int chs;
        cin("%d", &chs);
        if (chs == 0) {
            cout("Logging out...\n");
            break;
        } else if(chs == 1) {
            sendMoney(user, phoneNumber);
            return;
        } else if(chs == 2) {
            mobileRecharge(user);
            return;
        } else if(chs == 3) {
            billPay(user);
            return;
        } else if(chs == 4) {
            payment(user);
            return;
        } else if(chs == 5) {
            cashOut(user);
            return;
        } else if(chs == 6) {
            resetPIN(user);
            return;
        } else if(chs == 7) {
            cout("---------------\n");
            cout("   My Bkash\n");
            cout("---------------\n");
            cout("Name: %s\n", name);
            cout("Phone number: %s\n", phoneNumber);
            cout("Blanace: %s\n", balance);
            return;
        }
        else UnderConstruction();
```



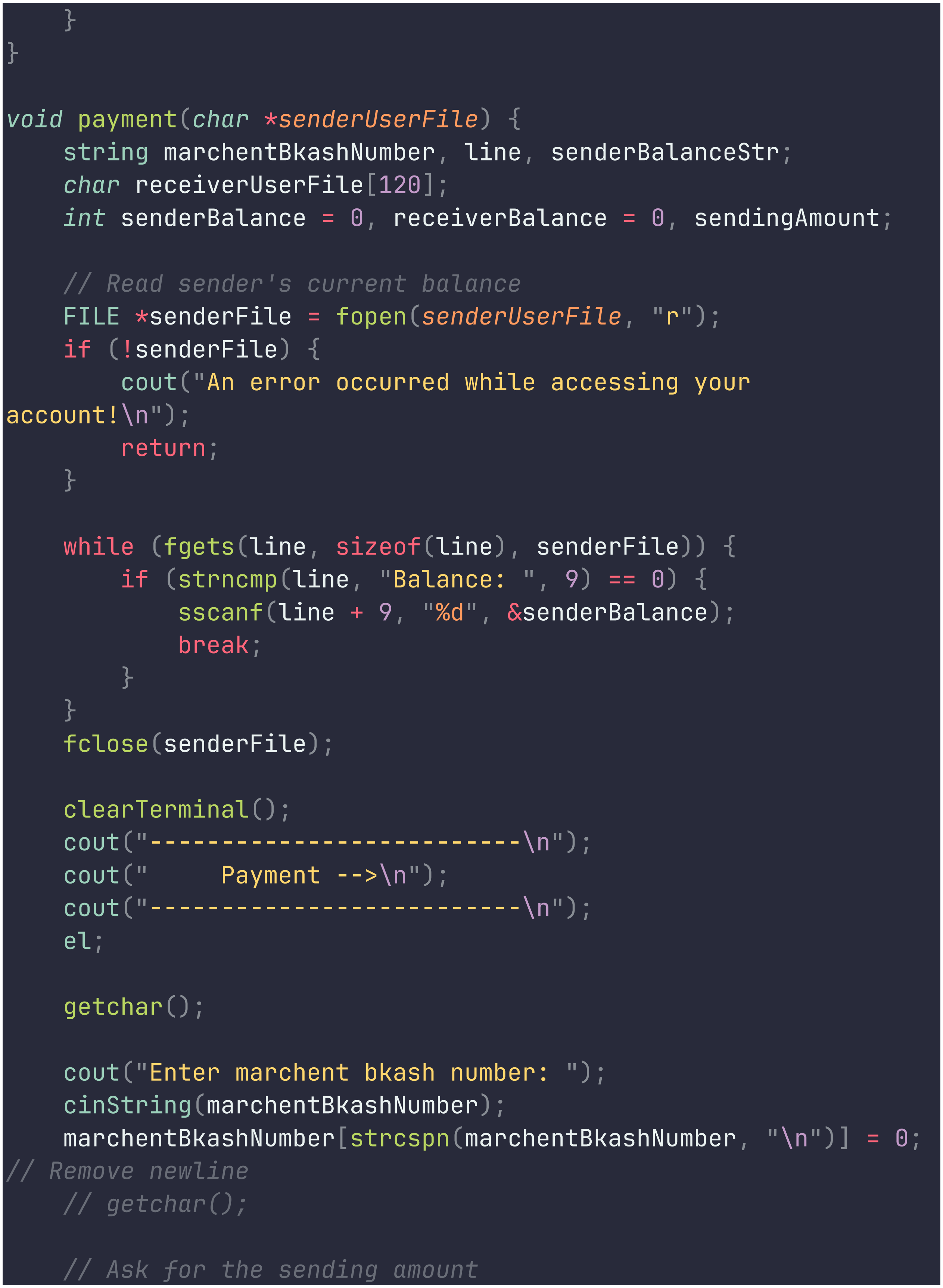
```
    }
}

void resetPIN(char *senderUserFile) {
    FILE *userFile = fopen(senderUserFile, "r+"); // Open file
for reading and writing

    char line[256];
    char currentPIN[7];
    char enteredOldPIN[7];
    char newPIN[7];

    // Read the user file and find the PIN line
    while (fgets(line, sizeof(line), userFile)) {
        if (strncmp(line, "PIN: ", 5) == 0) {
            sscanf(line + 5, "%6s", currentPIN); // Extract
current PIN
            break;
        }
    }

    clearTerminal();
    cout("--------------------------\n");
    cout("     Reset PIN -->\n");
    cout("--------------------------\n");
    el;

    // Ask the user for the old PIN
    printf("Enter your current 6-digit PIN: ");
    maskInput(enteredOldPIN, sizeof(enteredOldPIN));

    // Validate the old PIN
    if (strcmp(currentPIN, enteredOldPIN) != 0) {
        printf("Incorrect PIN! Access denied.\n");
        fclose(userFile);
        return;
    }

    // Ask for the new PIN
    printf("Enter your new 6-digit PIN: ");
    maskInput(newPIN, sizeof(newPIN));

    // Validate the new PIN (must be exactly 6 digits)
    if (!isValidPin(newPIN)) {
        printf("Invalid PIN! PIN must be exactly 6 digits.\n");
        fclose(userFile);
        return;
    }

    // Update the file with the new PIN
    fseek(userFile, 0, SEEK_SET); // Go back to the beginning of the file
    FILE *tempFile = fopen("temp.txt", "w"); // Create a temporary file for writing the updated content
    if (!tempFile) {
        printf("Error opening temporary file!\n");
        fclose(userFile);
        return;
    }

    // Copy content to temporary file while replacing the PIN line
    rewind(userFile);
    while (fgets(line, sizeof(line), userFile)) {
        if (strncmp(line, "PIN: ", 5) == 0) {
            fprintf(tempFile, "PIN: %s\n", newPIN); // Replace old PIN with new PIN
        } else {
            fputs(line, tempFile); // Copy all other lines as is
        }
    }

    fclose(userFile);
    fclose(tempFile);

    // Replace the original file with the temporary file
    remove(senderUserFile);            // Delete the old file
```



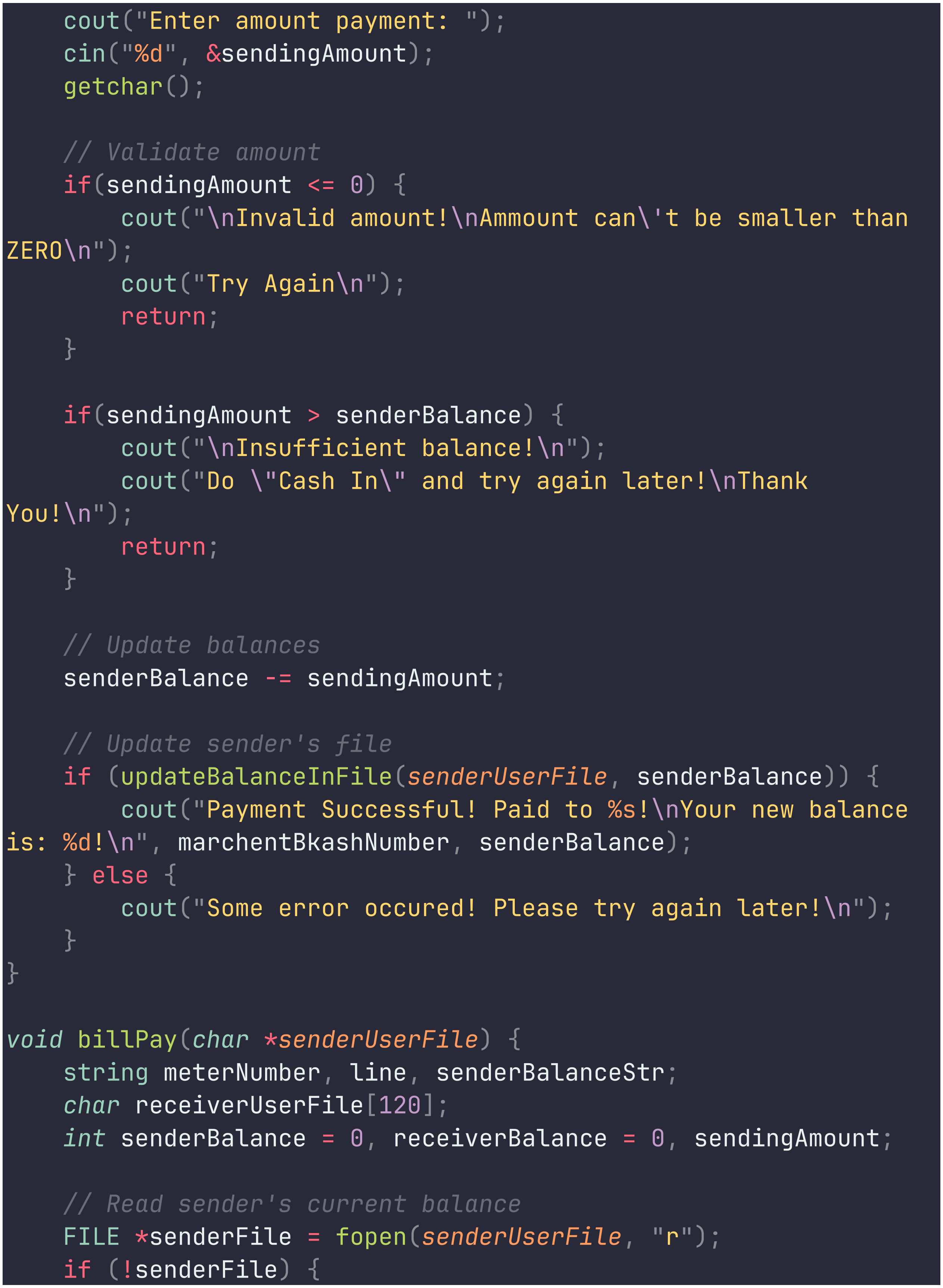
```
    rename("temp.txt", senderUserFile); // Rename temp file to
the original file

    printf("PIN successfully reset!\n");
}

void cashOut(char *senderUserFile) {
    string agentNumber, line, senderBalanceStr;
    char receiverUserFile[120];
    int senderBalance = 0, receiverBalance = 0, sendingAmount;

    // Read sender's current balance
    FILE *senderFile = fopen(senderUserFile, "r");
    if (!senderFile) {
        cout("The entered number belongs to a Bkash user. Please
provide a valid Agent number to proceed.\n");
        return;
    }

    while (fgets(line, sizeof(line), senderFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &senderBalance);
            break;
        }
    }
    fclose(senderFile);

    clearTerminal();
    cout("--------------------------\n");
    cout("     Cash Out -->\n");
    cout("--------------------------\n");
    el;

    getchar();

    cout("Enter Bkash Agent number: ");
    cinString(agentNumber);
    agentNumber[strcspn(agentNumber, "\n")] = 0; // Remove
newline
```



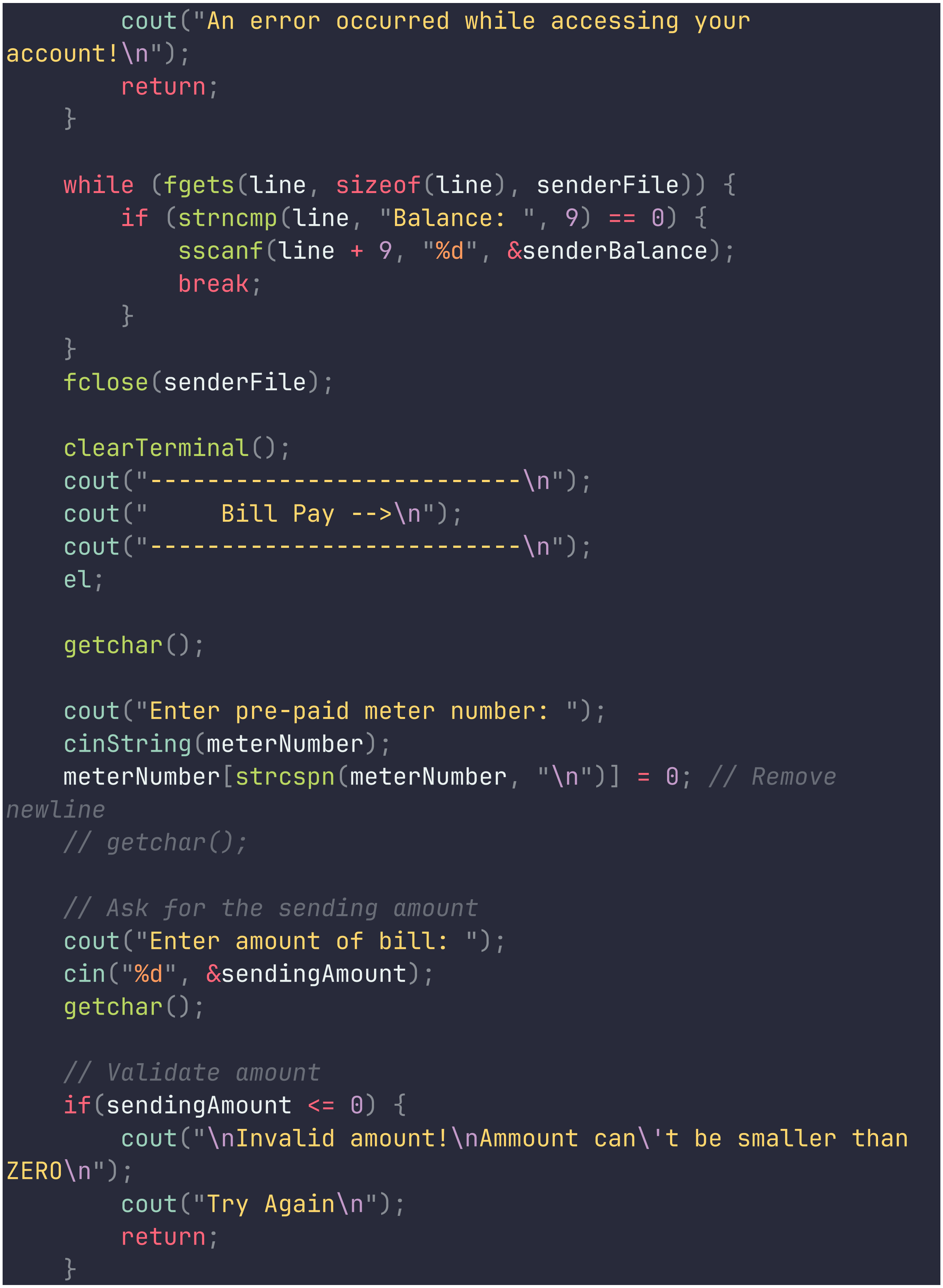
```
    // getchar();

    snprintf(receiverUserFile, sizeof(receiverUserFile),
"database/%s.txt", agentNumber);
    if(fileExists(receiverUserFile)) {
        cout("This is a bkash user number. Please enter a Agent
number and try again!\n");
        return;
    }

    // Ask for the sending amount
    cout("Enter amount payment: ");
    cin("%d", &sendingAmount);
    getchar();

    // Validate amount
    if(sendingAmount <= 0) {
        cout("\nInvalid amount!\nAmmount can\'t be smaller than
ZERO\n");
        cout("Try Again\n");
        return;
    }

    if(sendingAmount > senderBalance) {
        cout("\nInsufficient balance!\n");
        cout("Do \"Cash In\" and try again later!\nThank
You!\n");
        return;
    }

    // Update balances
    senderBalance -= sendingAmount;

    // Update sender's file
    if (updateBalanceInFile(senderUserFile, senderBalance)) {
        cout("Cash out of %d BDT Successful!\nYour new balance
is: %d!\n", sendingAmount, senderBalance);
    } else {
        cout("Some error occured! Please try again later!\n");
```



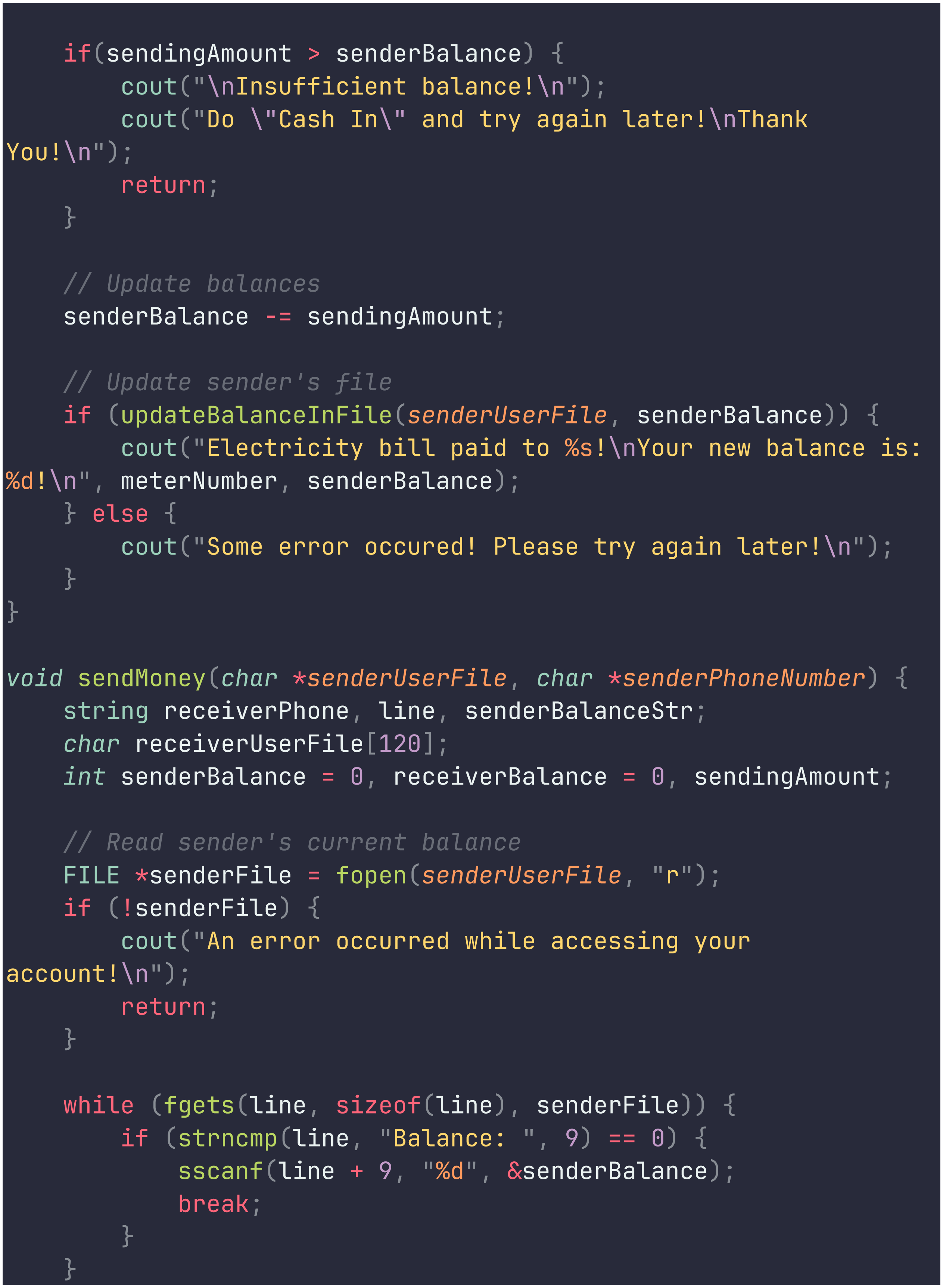
```
    }
}

void payment(char *senderUserFile) {
    string marchentBkashNumber, line, senderBalanceStr;
    char receiverUserFile[120];
    int senderBalance = 0, receiverBalance = 0, sendingAmount;

    // Read sender's current balance
    FILE *senderFile = fopen(senderUserFile, "r");
    if (!senderFile) {
        cout("An error occurred while accessing your
account!\n");
        return;
    }

    while (fgets(line, sizeof(line), senderFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &senderBalance);
            break;
        }
    }
    fclose(senderFile);

    clearTerminal();
    cout("--------------------------\n");
    cout("     Payment -->\n");
    cout("--------------------------\n");
    el;

    getchar();

    cout("Enter marchent bkash number: ");
    cinString(marchentBkashNumber);
    marchentBkashNumber[strcspn(marchentBkashNumber, "\n")] = 0;
// Remove newline
    // getchar();

    // Ask for the sending amount
```



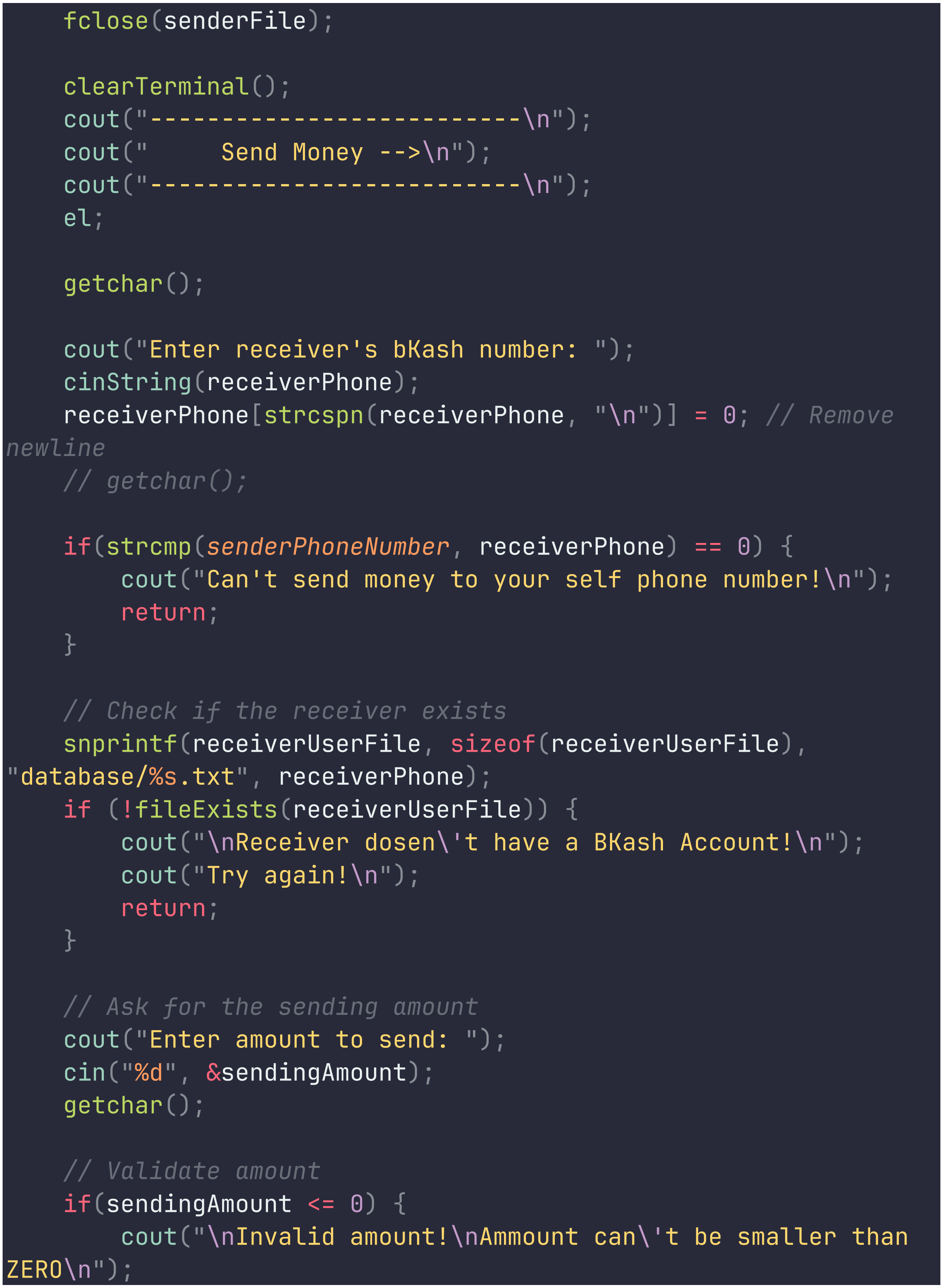
```
    cout("Enter amount payment: ");
    cin("%d", &sendingAmount);
    getchar();

    // Validate amount
    if(sendingAmount <= 0) {
        cout("\nInvalid amount!\nAmmount can\'t be smaller than
ZERO\n");
        cout("Try Again\n");
        return;
    }

    if(sendingAmount > senderBalance) {
        cout("\nInsufficient balance!\n");
        cout("Do \"Cash In\" and try again later!\nThank
You!\n");
        return;
    }

    // Update balances
    senderBalance -= sendingAmount;

    // Update sender's file
    if (updateBalanceInFile(senderUserFile, senderBalance)) {
        cout("Payment Successful! Paid to %s!\nYour new balance
is: %d!\n", marchentBkashNumber, senderBalance);
    } else {
        cout("Some error occured! Please try again later!\n");
    }
}

void billPay(char *senderUserFile) {
    string meterNumber, line, senderBalanceStr;
    char receiverUserFile[120];
    int senderBalance = 0, receiverBalance = 0, sendingAmount;

    // Read sender's current balance
    FILE *senderFile = fopen(senderUserFile, "r");
    if (!senderFile) {
```



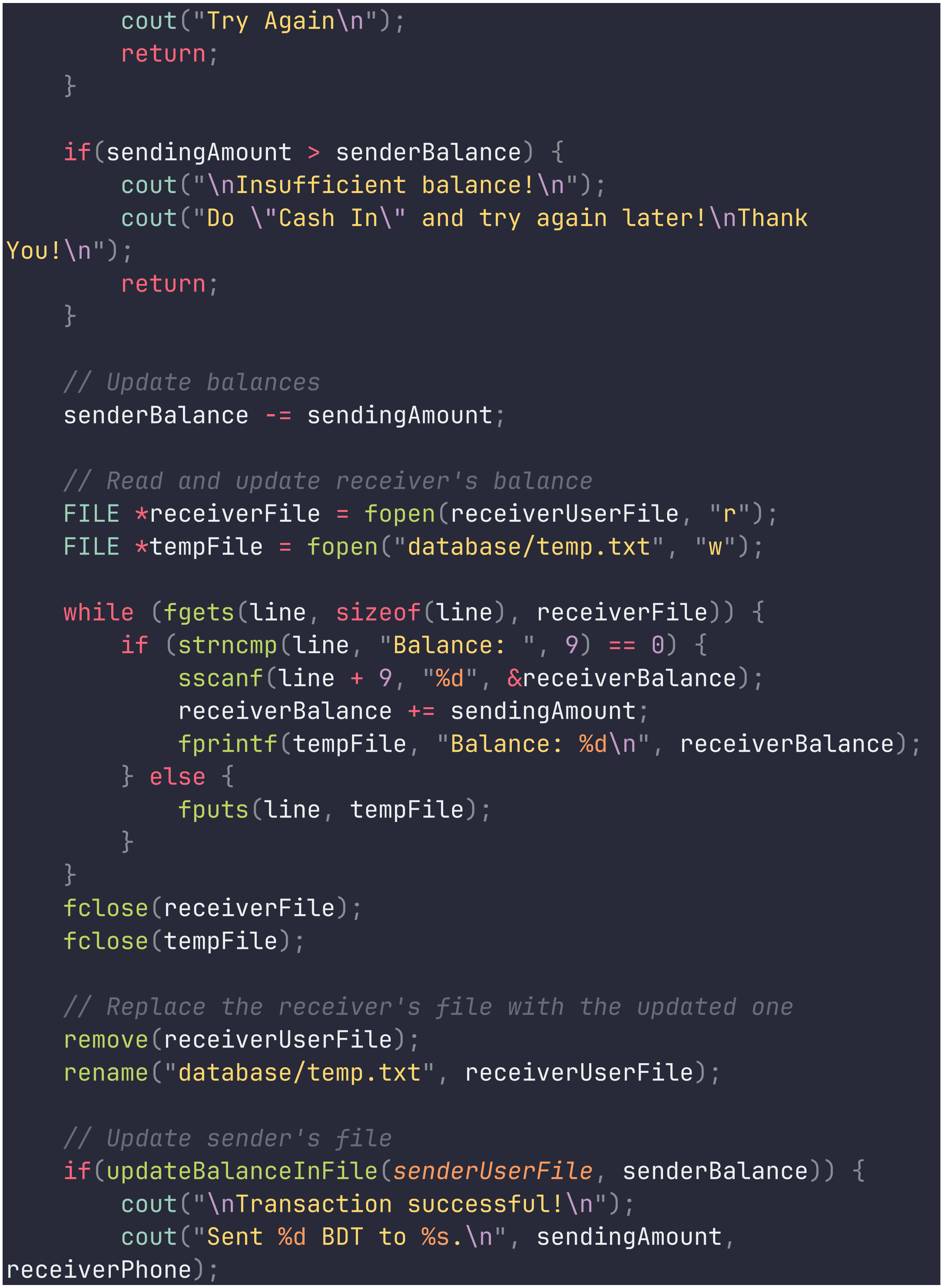
```
        cout("An error occurred while accessing your
account!\n");
        return;
    }

    while (fgets(line, sizeof(line), senderFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &senderBalance);
            break;
        }
    }
    fclose(senderFile);

    clearTerminal();
    cout("--------------------------\n");
    cout("     Bill Pay -->\n");
    cout("--------------------------\n");
    el;

    getchar();

    cout("Enter pre-paid meter number: ");
    cinString(meterNumber);
    meterNumber[strcspn(meterNumber, "\n")] = 0; // Remove
newline
    // getchar();

    // Ask for the sending amount
    cout("Enter amount of bill: ");
    cin("%d", &sendingAmount);
    getchar();

    // Validate amount
    if(sendingAmount <= 0) {
        cout("\nInvalid amount!\nAmmount can\'t be smaller than
ZERO\n");
        cout("Try Again\n");
        return;
    }
```



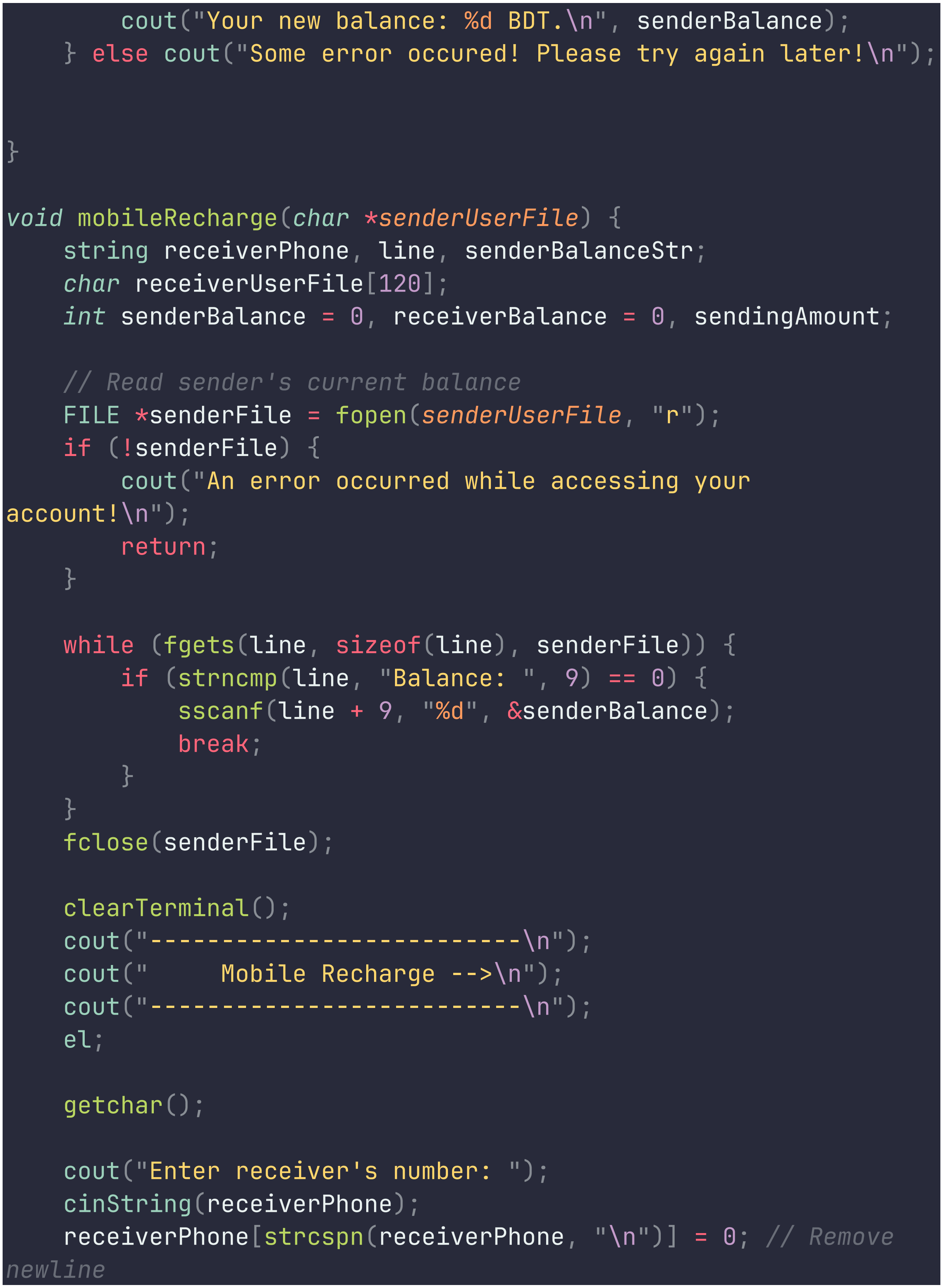
```
    if(sendingAmount > senderBalance) {
        cout("\nInsufficient balance!\n");
        cout("Do \"Cash In\" and try again later!\nThank
You!\n");
        return;
    }

    // Update balances
    senderBalance -= sendingAmount;

    // Update sender's file
    if (updateBalanceInFile(senderUserFile, senderBalance)) {
        cout("Electricity bill paid to %s!\nYour new balance is:
%d!\n", meterNumber, senderBalance);
    } else {
        cout("Some error occured! Please try again later!\n");
    }
}

void sendMoney(char *senderUserFile, char *senderPhoneNumber) {
    string receiverPhone, line, senderBalanceStr;
    char receiverUserFile[120];
    int senderBalance = 0, receiverBalance = 0, sendingAmount;

    // Read sender's current balance
    FILE *senderFile = fopen(senderUserFile, "r");
    if (!senderFile) {
        cout("An error occurred while accessing your
account!\n");
        return;
    }

    while (fgets(line, sizeof(line), senderFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &senderBalance);
            break;
        }
    }
```



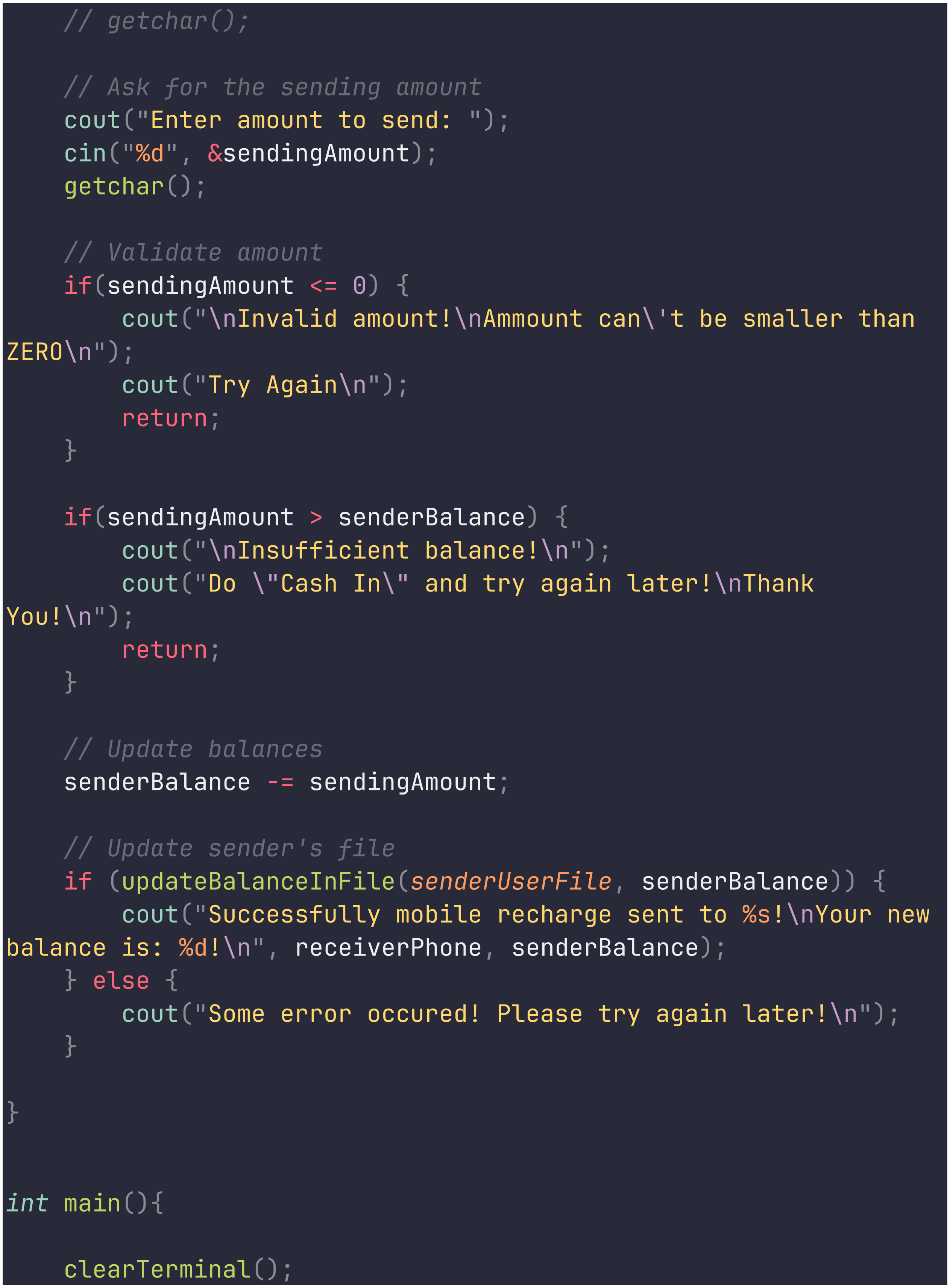
```
    fclose(senderFile);

    clearTerminal();
    cout("--------------------------\n");
    cout("     Send Money -->\n");
    cout("--------------------------\n");
    el;

    getchar();

    cout("Enter receiver's bKash number: ");
    cinString(receiverPhone);
    receiverPhone[strcspn(receiverPhone, "\n")] = 0; // Remove
newline
    // getchar();

    if(strcmp(senderPhoneNumber, receiverPhone) == 0) {
        cout("Can't send money to your self phone number!\n");
        return;
    }

    // Check if the receiver exists
    snprintf(receiverUserFile, sizeof(receiverUserFile),
"database/%s.txt", receiverPhone);
    if (!fileExists(receiverUserFile)) {
        cout("\nReceiver dosen\'t have a BKash Account!\n");
        cout("Try again!\n");
        return;
    }

    // Ask for the sending amount
    cout("Enter amount to send: ");
    cin("%d", &sendingAmount);
    getchar();

    // Validate amount
    if(sendingAmount <= 0) {
        cout("\nInvalid amount!\nAmmount can\'t be smaller than
ZERO\n");
```



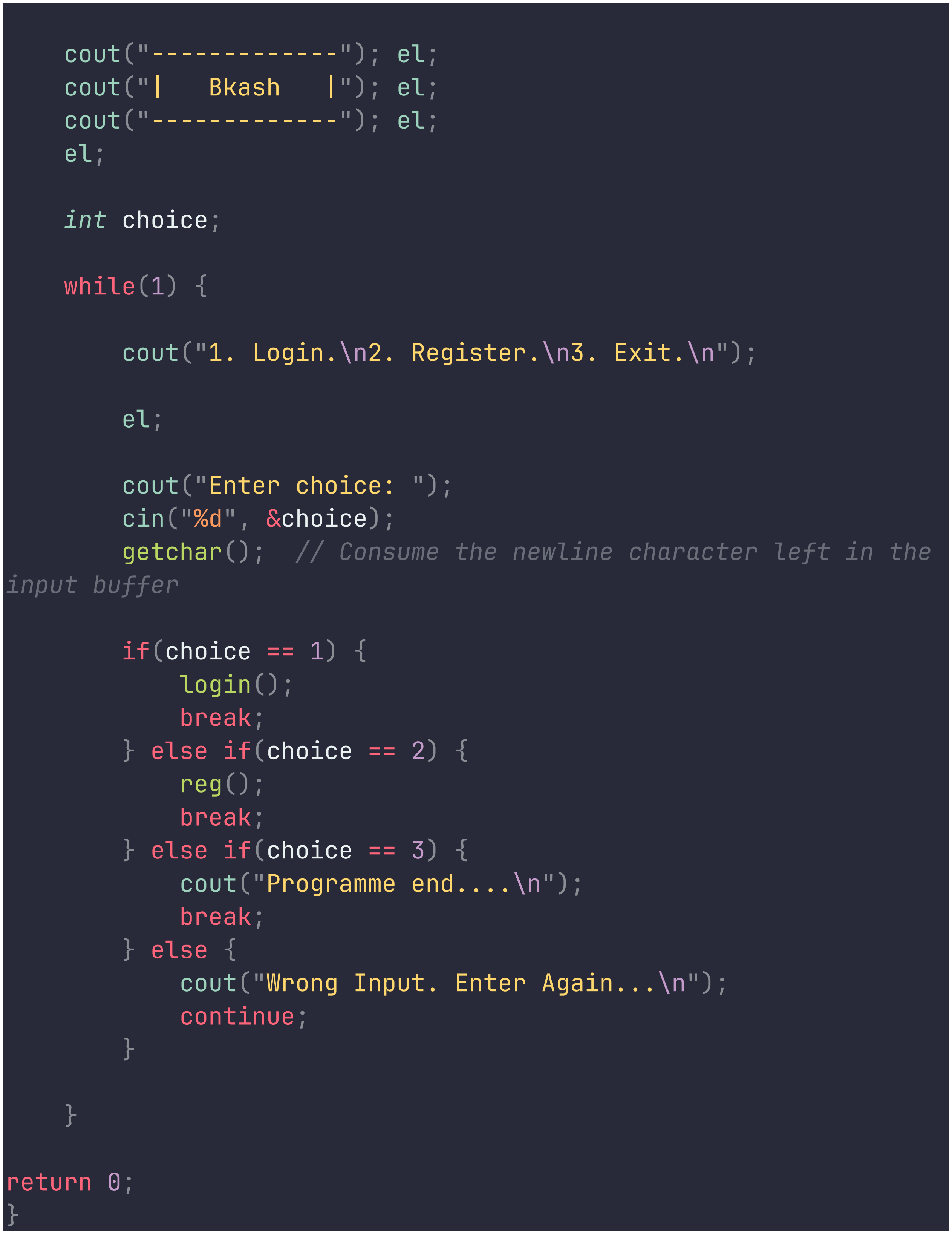
```
        cout("Try Again\n");
        return;
    }

    if(sendingAmount > senderBalance) {
        cout("\nInsufficient balance!\n");
        cout("Do \"Cash In\" and try again later!\nThank
You!\n");
        return;
    }

    // Update balances
    senderBalance -= sendingAmount;

    // Read and update receiver's balance
    FILE *receiverFile = fopen(receiverUserFile, "r");
    FILE *tempFile = fopen("database/temp.txt", "w");

    while (fgets(line, sizeof(line), receiverFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &receiverBalance);
            receiverBalance += sendingAmount;
            fprintf(tempFile, "Balance: %d\n", receiverBalance);
        } else {
            fputs(line, tempFile);
        }
    }
    fclose(receiverFile);
    fclose(tempFile);

    // Replace the receiver's file with the updated one
    remove(receiverUserFile);
    rename("database/temp.txt", receiverUserFile);

    // Update sender's file
    if(updateBalanceInFile(senderUserFile, senderBalance)) {
        cout("\nTransaction successful!\n");
        cout("Sent %d BDT to %s.\n", sendingAmount,
receiverPhone);
```



```
        cout("Your new balance: %d BDT.\n", senderBalance);
    } else cout("Some error occured! Please try again later!\n");


}

void mobileRecharge(char *senderUserFile) {
    string receiverPhone, line, senderBalanceStr;
    char receiverUserFile[120];
    int senderBalance = 0, receiverBalance = 0, sendingAmount;

    // Read sender's current balance
    FILE *senderFile = fopen(senderUserFile, "r");
    if (!senderFile) {
        cout("An error occurred while accessing your
account!\n");
        return;
    }

    while (fgets(line, sizeof(line), senderFile)) {
        if (strncmp(line, "Balance: ", 9) == 0) {
            sscanf(line + 9, "%d", &senderBalance);
            break;
        }
    }
    fclose(senderFile);

    clearTerminal();
    cout("--------------------------\n");
    cout("     Mobile Recharge -->\n");
    cout("--------------------------\n");
    el;

    getchar();

    cout("Enter receiver's number: ");
    cinString(receiverPhone);
    receiverPhone[strcspn(receiverPhone, "\n")] = 0; // Remove
newline
```



```
    // getchar();

    // Ask for the sending amount
    cout("Enter amount to send: ");
    cin("%d", &sendingAmount);
    getchar();

    // Validate amount
    if(sendingAmount <= 0) {
        cout("\nInvalid amount!\nAmmount can\'t be smaller than 
ZERO\n");
        cout("Try Again\n");
        return;
    }

    if(sendingAmount > senderBalance) {
        cout("\nInsufficient balance!\n");
        cout("Do \"Cash In\" and try again later!\nThank 
You!\n");
        return;
    }

    // Update balances
    senderBalance -= sendingAmount;

    // Update sender's file
    if (updateBalanceInFile(senderUserFile, senderBalance)) {
        cout("Successfully mobile recharge sent to %s!\nYour new 
balance is: %d!\n", receiverPhone, senderBalance);
    } else {
        cout("Some error occured! Please try again later!\n");
    }

}


int main(){

    clearTerminal();
```



```
    cout("-------------"); el;
    cout("|   Bkash   |"); el;
    cout("-------------"); el;
    el;

    int choice;

    while(1) {

        cout("1. Login.\n2. Register.\n3. Exit.\n");

        el;

        cout("Enter choice: ");
        cin("%d", &choice);
        getchar();  // Consume the newline character left in the
input buffer

        if(choice == 1) {
            login();
            break;
        } else if(choice == 2) {
            reg();
            break;
        } else if(choice == 3) {
            cout("Programme end....\n");
            break;
        } else {
            cout("Wrong Input. Enter Again...\n");
            continue;
        }

    }

return 0;
}
```