Object Oriented Programming
Lab Assignment 11

SUBMITTED BY:

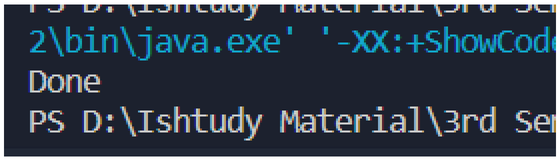
Hasaan Ahmad SP22-BSE-017

SUBMITTED TO: Sir Muzaffar

GLT1:
Code:

```java
package LAB11;

import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.ArrayList;

/*
Create a class Book that has name(String), publisher (String) and an author
(Person). Write five objects
of Book Class in a file named “BookStore”.*/
class Person implements Serializable {
    String Name;
    int age;

    public Person(String name, int age) {
        Name = name;
        this.age = age;
    }

    public String getName() {
        return Name;
    }

    public void setName(String name) {
        Name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

}

class Book implements Serializable {
    String name;
    String publisher;
    Person author;
    public Book(String name, String publisher, Person author) {
        this.name = name;
        this.publisher = publisher;
        this.author = author;
    }

    public String getName() {
        return name;
    }

    public String getPublisher() {
        return publisher;
    }

    public Person getAuthor() {
        return author;
    }
}

public class GLT1 {
    public static void main(String[] args) {
        ArrayList al = new ArrayList();
        al.add(new Book("Book1", "Publisher1", new Person("Author1", 20)));
        al.add(new Book("Book2", "Publisher2", new Person("Author2", 21)));
        al.add(new Book("Book3", "Publisher3", new Person("Author3", 22)));
        al.add(new Book("Book4", "Publisher4", new Person("Author4", 23)));
        al.add(new Book("Book5", "Publisher5", new Person("Author5", 24)));
        try {
            FileOutputStream fos = new FileOutputStream("BookStore.txt");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(al);
            oos.close();
            fos.close();
            System.out.println("Done");
        } catch (Exception e) {
            System.out.println(e);
        }
    }

}
```

Output:



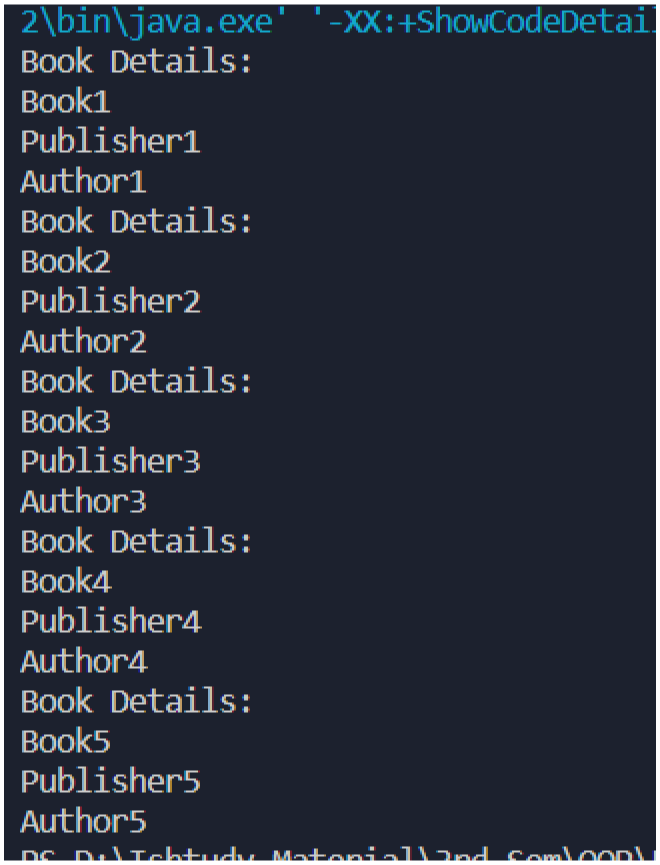
```
2\bin\java.exe' '-XX:+ShowCod
Done
PS D:\Ishtudy Material\3rd Se
```

GLT2:

Code:

```
package LAB11;

import java.io.FileInputStream;
import java.io.ObjectInputStream;
import java.io.Serializable;
import java.util.ArrayList;

public class GLT2 implements Serializable {
    public static void main(String[] args) {
        // Consider the Book class of GLT1 and write a function that displays all
        // Books present in file “BookStore”.
        try {
            FileInputStream fis = new FileInputStream("BookStore.txt");
            ObjectInputStream ois = new ObjectInputStream(fis);
            ArrayList<Book> books = (ArrayList<Book>) ois.readObject();
            for (Book book : books) {
                System.out.println("Book Details: ");
                System.out.println(book.getName());
                System.out.println(book.getPublisher());
                System.out.println(book.getAuthor().getName());
            }
            ois.close();
            fis.close();

        } catch (Exception e) {
            System.out.println(e);
        }

    }

}
```

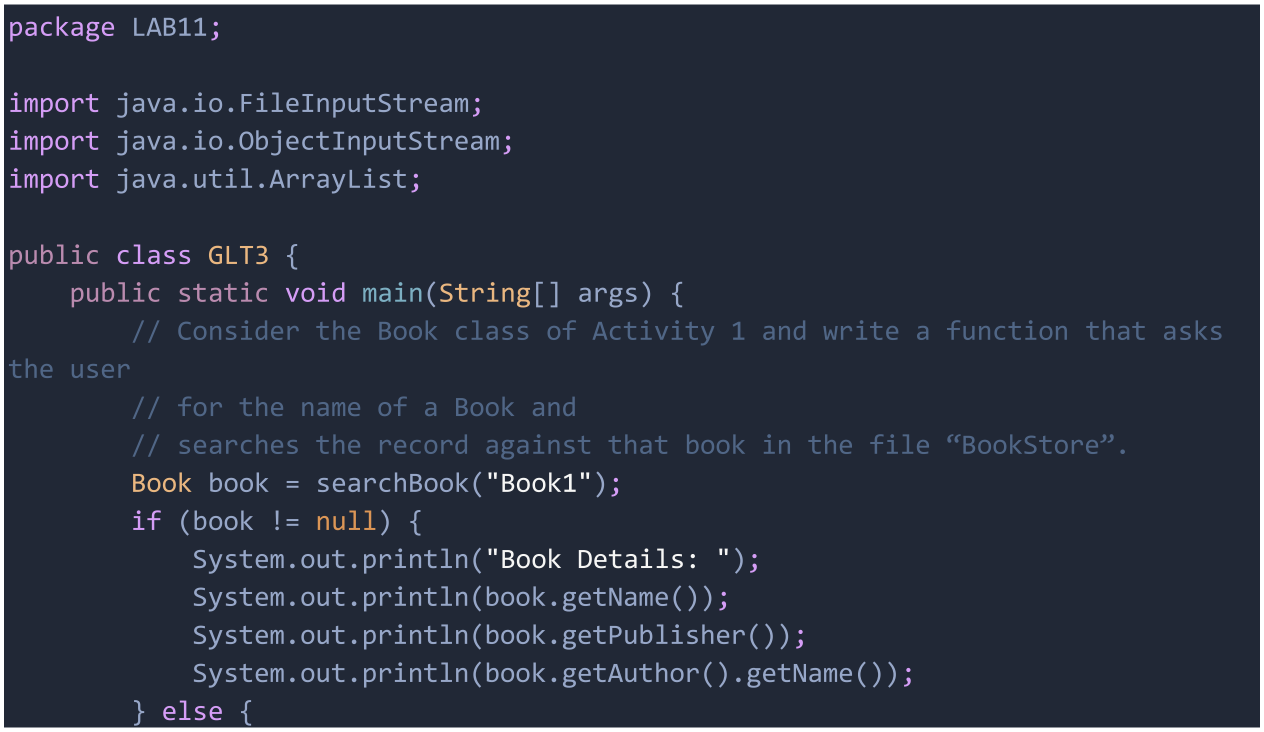
Output:



```
2\bin\java.exe' '-XX:+ShowCodeDetai
Book Details:
Book1
Publisher1
Author1
Book Details:
Book2
Publisher2
Author2
Book Details:
Book3
Publisher3
Author3
Book Details:
Book4
Publisher4
Author4
Book Details:
Book5
Publisher5
Author5
```

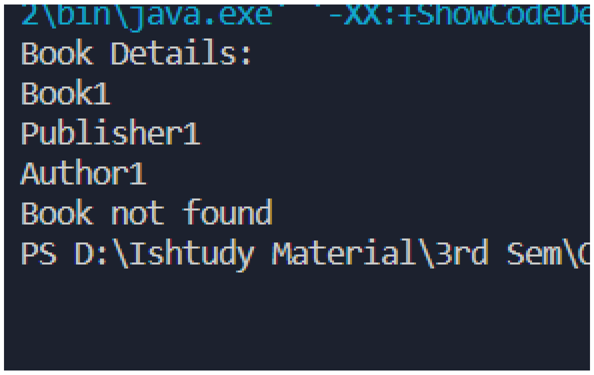
GLT3:

Code:



```
package LAB11;

import java.io.FileInputStream;
import java.io.ObjectInputStream;
import java.util.ArrayList;

public class GLT3 {
    public static void main(String[] args) {
        // Consider the Book class of Activity 1 and write a function that asks
the user
        // for the name of a Book and
        // searches the record against that book in the file “BookStore”.
        Book book = searchBook("Book1");
        if (book != null) {
            System.out.println("Book Details: ");
            System.out.println(book.getName());
            System.out.println(book.getPublisher());
            System.out.println(book.getAuthor().getName());
        } else {
            System.out.println("Book not found");
        }
        // Not found
        book = searchBook("Book 10");
        if (book != null) {
            System.out.println("Book Details: ");
            System.out.println(book.getName());
            System.out.println(book.getPublisher());
            System.out.println(book.getAuthor().getName());
        } else {
            System.out.println("Book not found");
        }

    }

    static Book searchBook(String name) {
        try {
            FileInputStream fis = new FileInputStream("BookStore.txt");
            ObjectInputStream ois = new ObjectInputStream(fis);
            ArrayList<Book> books = (ArrayList<Book>) ois.readObject();
            for (Book book : books) {
                if (book.getName().equals(name)) {
                    return book;
                }
            }
            ois.close();
            fis.close();

        } catch (Exception e) {
            System.out.println(e);
        }
        return null;
    }
}
```

Output:



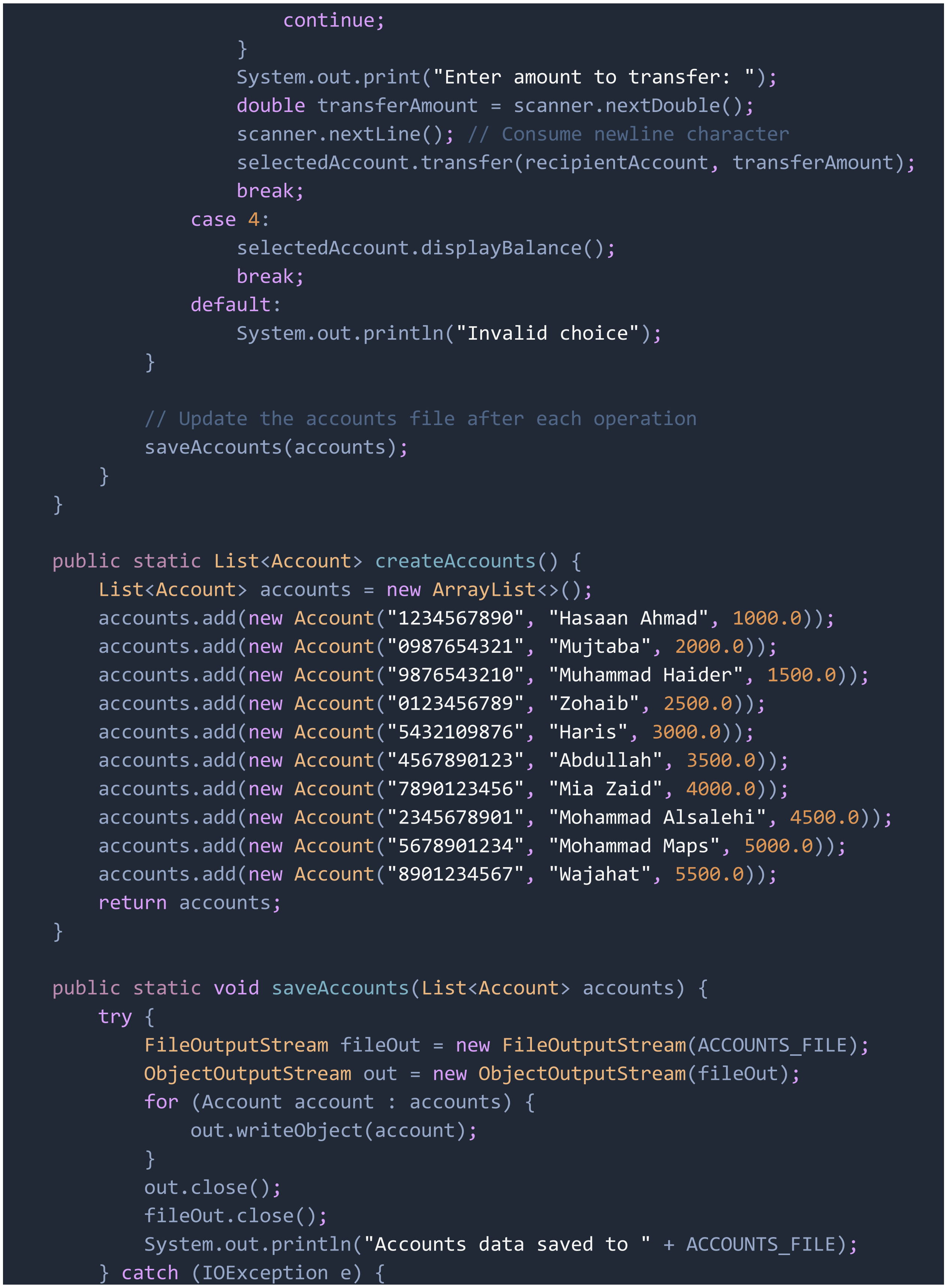
```
2\bin\java.exe' '-XX:+ShowCodeDe
Book Details:
Book1
Publisher1
Author1
Book not found
PS D:\Ishtudy Material\3rd Sem\C
```

GLT4:

Code:

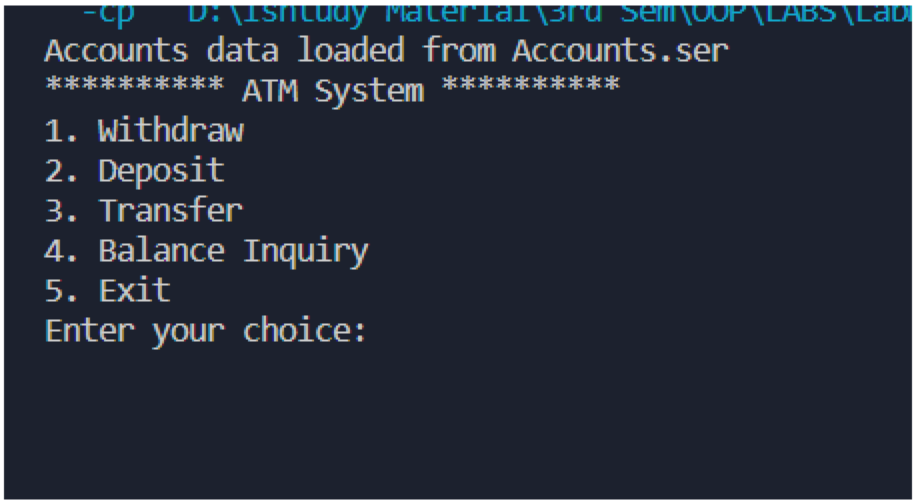
```
package LAB11;

import java.io.*;
import java.util.*;

class Account implements Serializable {
    private String accountNumber;
    private String accountHolderName;
    private double balance;

    public Account(String accountNumber, String accountHolderName, double
balance) {
        this.accountNumber = accountNumber;
        this.accountHolderName = accountHolderName;
        this.balance = balance;
    }

    public String getAccountNumber() {
        return accountNumber;
    }

    public String getAccountHolderName() {
        return accountHolderName;
    }

    public double getBalance() {
        return balance;
    }

    public void setBalance(double balance) {
        this.balance = balance;
    }
```

```java
    public void deposit(double amount) {
        balance += amount;
    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println("Insufficient balance");
        }
    }

    public void transfer(Account recipient, double amount) {
        if (balance >= amount) {
            balance -= amount;
            recipient.deposit(amount);
            System.out.println("Transfer successful");
        } else {
            System.out.println("Insufficient balance for transfer");
        }
    }

    public void displayBalance() {
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Account Holder: " + accountHolderName);
        System.out.println("Balance: $" + balance);
    }
}

public class GLT4 {
    private static final String ACCOUNTS_FILE = "Accounts.ser";

    public static void main(String[] args) {
        List<Account> accounts = new ArrayList<>();

        // Load existing accounts from the file or create new accounts
        File file = new File(ACCOUNTS_FILE);
        if (file.exists()) {
            accounts = loadAccounts();
        } else {
            accounts = createAccounts();
        }

        Scanner scanner = new Scanner(System.in);
        while (true) {
            System.out.println("********** ATM System **********");
            System.out.println("1. Withdraw");
            System.out.println("2. Deposit");
            System.out.println("3. Transfer");
            System.out.println("4. Balance Inquiry");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline character

            if (choice == 5) {
                break;
            }

            System.out.print("Enter account number: ");
            String accountNumber = scanner.nextLine();

            Account selectedAccount = findAccount(accounts, accountNumber);
            if (selectedAccount == null) {
                System.out.println("Account not found");
                continue;
            }

            switch (choice) {
                case 1:
                    System.out.print("Enter amount to withdraw: ");
                    double withdrawAmount = scanner.nextDouble();
                    scanner.nextLine(); // Consume newline character
                    selectedAccount.withdraw(withdrawAmount);
                    break;
                case 2:
                    System.out.print("Enter amount to deposit: ");
                    double depositAmount = scanner.nextDouble();
                    scanner.nextLine(); // Consume newline character
                    selectedAccount.deposit(depositAmount);
                    break;
                case 3:
                    System.out.print("Enter recipient account number: ");
                    String recipientAccountNumber = scanner.nextLine();
                    Account recipientAccount = findAccount(accounts,
recipientAccountNumber);
                    if (recipientAccount == null) {
                        System.out.println("Recipient account not found");
```

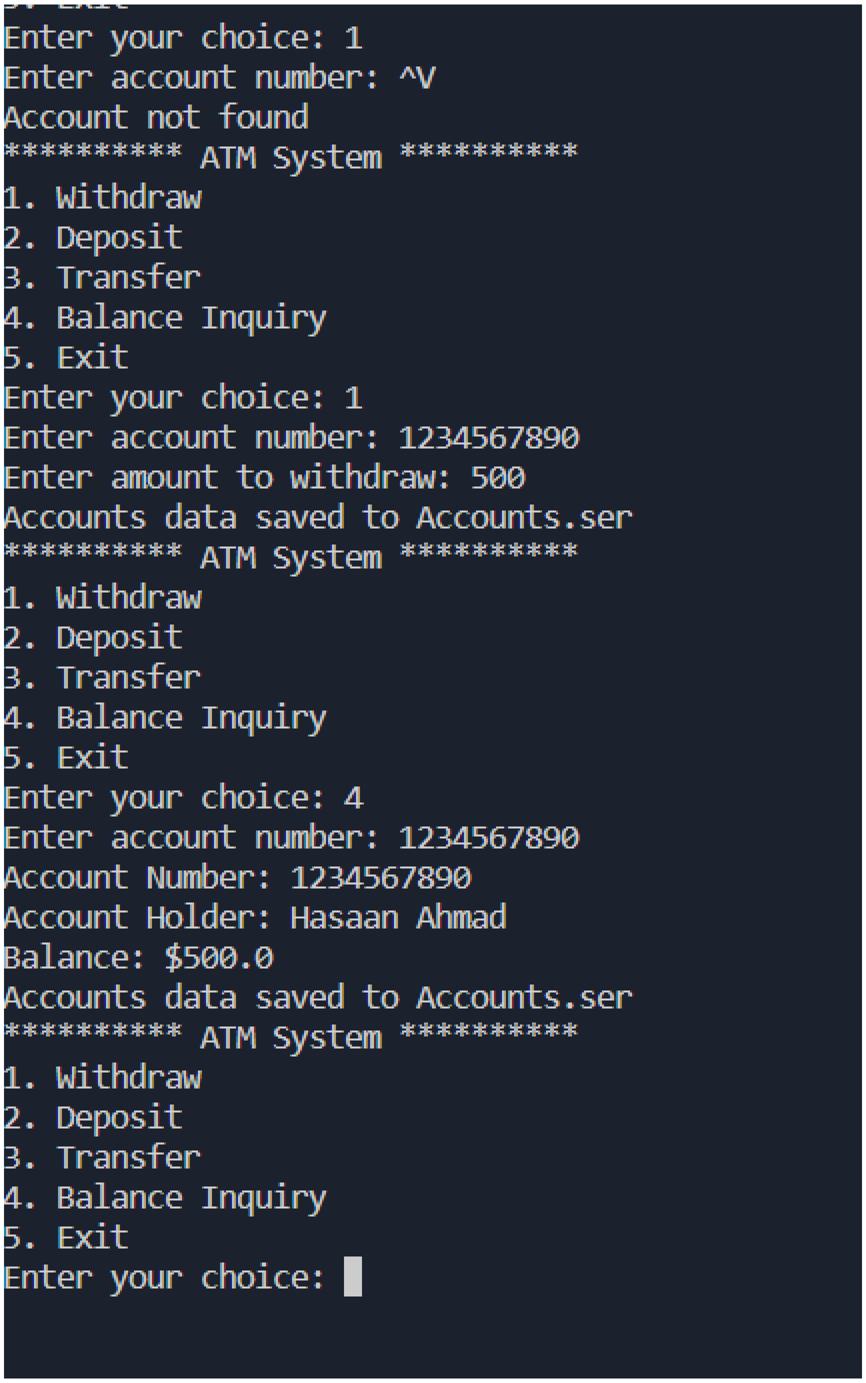


```java
                        continue;
                    }
                    System.out.print("Enter amount to transfer: ");
                    double transferAmount = scanner.nextDouble();
                    scanner.nextLine(); // Consume newline character
                    selectedAccount.transfer(recipientAccount, transferAmount);
                    break;
                case 4:
                    selectedAccount.displayBalance();
                    break;
                default:
                    System.out.println("Invalid choice");
            }

            // Update the accounts file after each operation
            saveAccounts(accounts);
        }
    }

    public static List<Account> createAccounts() {
        List<Account> accounts = new ArrayList<>();
        accounts.add(new Account("1234567890", "Hasaan Ahmad", 1000.0));
        accounts.add(new Account("0987654321", "Mujtaba", 2000.0));
        accounts.add(new Account("9876543210", "Muhammad Haider", 1500.0));
        accounts.add(new Account("0123456789", "Zohaib", 2500.0));
        accounts.add(new Account("5432109876", "Haris", 3000.0));
        accounts.add(new Account("4567890123", "Abdullah", 3500.0));
        accounts.add(new Account("7890123456", "Mia Zaid", 4000.0));
        accounts.add(new Account("2345678901", "Mohammad Alsalehi", 4500.0));
        accounts.add(new Account("5678901234", "Mohammad Maps", 5000.0));
        accounts.add(new Account("8901234567", "Wajahat", 5500.0));
        return accounts;
    }

    public static void saveAccounts(List<Account> accounts) {
        try {
            FileOutputStream fileOut = new FileOutputStream(ACCOUNTS_FILE);
            ObjectOutputStream out = new ObjectOutputStream(fileOut);
            for (Account account : accounts) {
                out.writeObject(account);
            }
            out.close();
            fileOut.close();
            System.out.println("Accounts data saved to " + ACCOUNTS_FILE);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static List<Account> loadAccounts() {
        List<Account> accounts = new ArrayList<>();
        try {
            FileInputStream fileIn = new FileInputStream(ACCOUNTS_FILE);
            ObjectInputStream in = new ObjectInputStream(fileIn);
            while (true) {
                try {
                    Account account = (Account) in.readObject();
                    accounts.add(account);
                } catch (EOFException e) {
                    break;
                }
            }
            in.close();
            fileIn.close();
            System.out.println("Accounts data loaded from " + ACCOUNTS_FILE);
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
        return accounts;
    }

    public static Account findAccount(List<Account> accounts, String
accountNumber) {
        for (Account account : accounts) {
            if (account.getAccountNumber().equals(accountNumber)) {
                return account;
            }
        }
        return null;
    }
}
```

Output:



```
   -cp   D:\1sStudy Material\3rd Sem\OOP\LABS\Lab
Accounts data loaded from Accounts.ser
********** ATM System **********
1. Withdraw
2. Deposit
3. Transfer
4. Balance Inquiry
5. Exit
Enter your choice:
```



```
Enter your choice: 1
Enter account number: ^V
Account not found
*********** ATM System ***********
1. Withdraw
2. Deposit
3. Transfer
4. Balance Inquiry
5. Exit
Enter your choice: 1
Enter account number: 1234567890
Enter amount to withdraw: 500
Accounts data saved to Accounts.ser
*********** ATM System ***********
1. Withdraw
2. Deposit
3. Transfer
4. Balance Inquiry
5. Exit
Enter your choice: 4
Enter account number: 1234567890
Account Number: 1234567890
Account Holder: Hasaan Ahmad
Balance: $500.0
Accounts data saved to Accounts.ser
*********** ATM System ***********
1. Withdraw
2. Deposit
3. Transfer
4. Balance Inquiry
5. Exit
Enter your choice:
```