TASK's ATM

TEODOR SEREDIUC
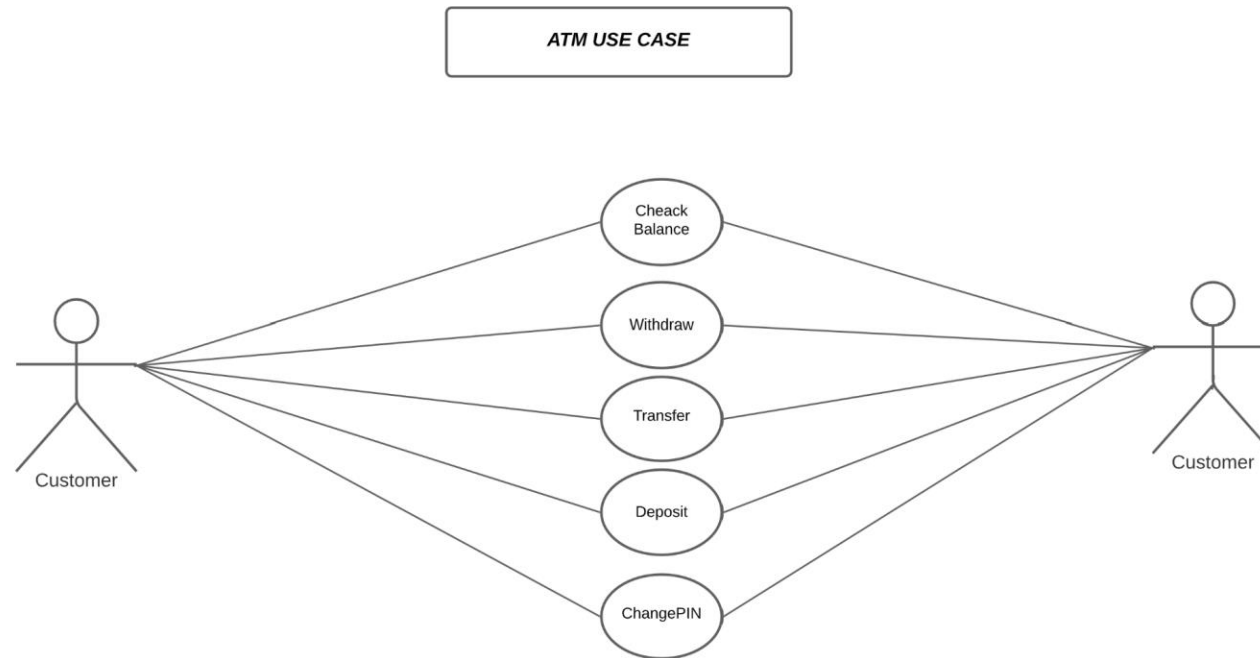
ANASTASSIA TITOVA

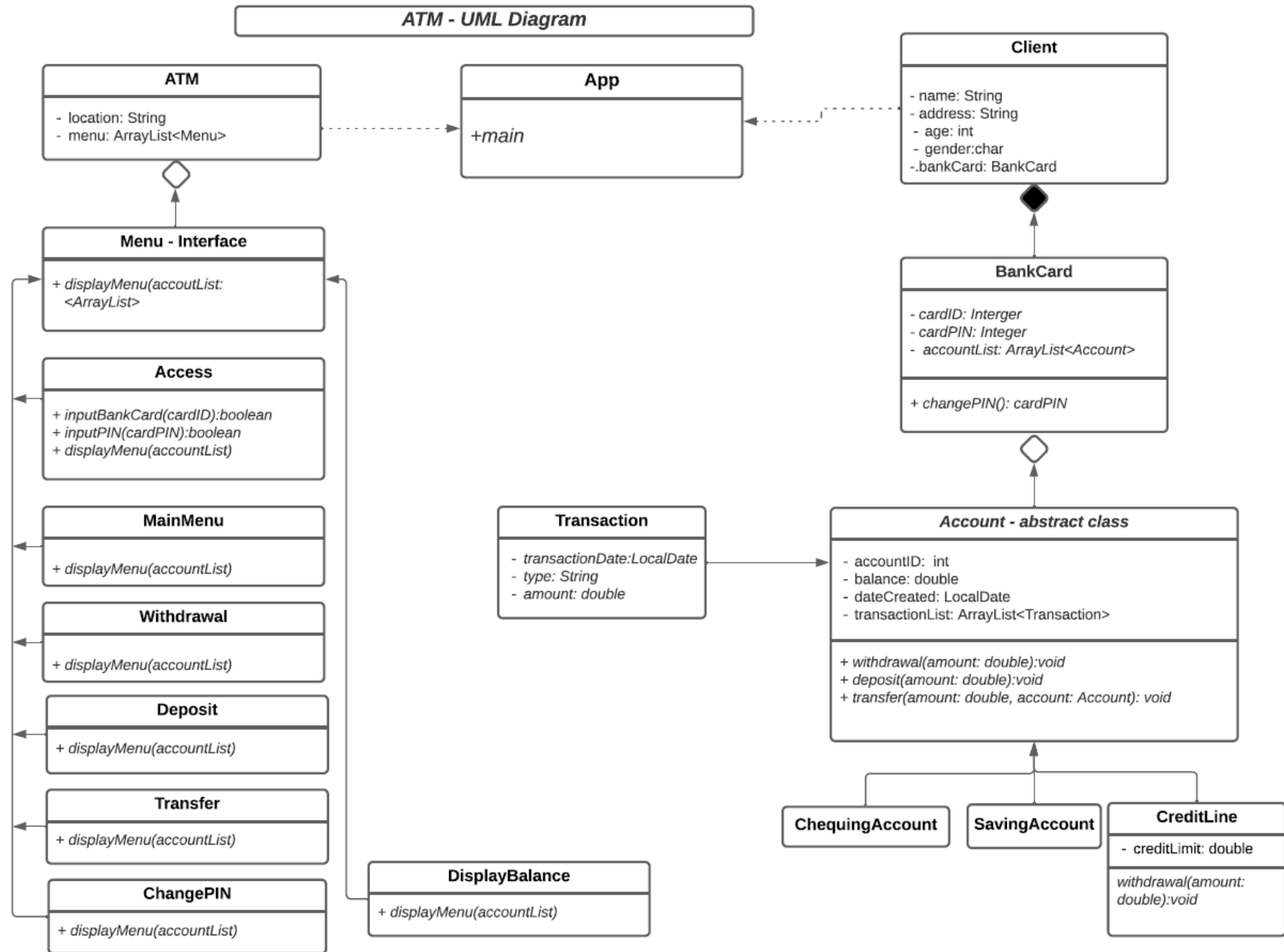**S**OMAYEH KARAMI

KHALED GHAMRAOUI

Three elements that a use case must contain:
1.Actor, which is the user, which can be a single person or a group of people, interacting with a process
2.System, which is the process that's required to reach the outcome
3.Goal, which is the successful user outcome

ATM USE CASE

Cheack Balance

Withdraw

Transfer

Deposit

ChangePIN

Customer

Customer

ATM - UML Diagram

ATM
- location: String
- menu: ArrayList<Menu>

App
+*main*

Client
- name: String
- address: String
- age: int
- gender:char
-.bankCard: BankCard

Menu - Interface
+ *displayMenu(accoutList: <ArrayList>*

Access
+ *inputBankCard(cardID):boolean*
+ *inputPIN(cardPIN):boolean*
+ *displayMenu(accountList)*

MainMenu
+ *displayMenu(accountList)*

Withdrawal
+ *displayMenu(accountList)*

Deposit
+ *displayMenu(accountList)*

Transfer
+ *displayMenu(accountList)*

ChangePIN
+ *displayMenu(accountList)*

DisplayBalance
+ *displayMenu(accountList)*

BankCard
- *cardID: Interger*
- *cardPIN: Integer*
- *accountList: ArrayList<Account>*

+ *changePIN(): cardPIN*

Transaction
- *transactionDate:LocalDate*
- *type: String*
- *amount: double*

Account - abstract class
- *accountID: int*
- *balance: double*
- *dateCreated: LocalDate*
- *transactionList: ArrayList<Transaction>*

+ *withdrawal(amount: double):void*
+ *deposit(amount: double):void*
+ *transfer(amount: double, account: Account): void*

ChequingAccount

SavingAccount

CreditLine
- creditLimit: double

withdrawal(amount: double):void

CHECK THE WEB

OPINION OF THE MAJORITY OF JAVA PROGRAMMERS:

Java is OOP language

The only way to design a program is bottom-up.

TOP-DOWN FOR ME

I realized I do not know what should be at the bottom of the logic chain

I know quite well who is on the top of my design – THE CLIENT

MY FIRST PICTURE OF THE CLIENT SIDE

DONE

Outline of classes

LIST OF QUESTIONS:

Where to put transactions

What variable to declare and where

CLIENT

BANK CARD

ACCOUNT

CHEQUING ACCOUNT

CHEQUING ACCOUNT		SAVING ACCOUNT		CREDIT LINE
Account Number		Account Number		Account Number
Account Balance		Account Balance		Account Balance
Account Transactions		Account Transactions		Account Transactions
Overdraft Limit		Overdraft Limit		Credit Limit
Fee		Fee		Fee
Account Restriction: withdrawal limit		Account Restriction: number of transactions		Account Restriction:

BOTTOM-UP

Design concrete classes

Values that are the same are declared in Parent class

Chain up to the Client class

QUESTION LEFT:

Transaction?

DECISION

Transaction class has an aggregation relationship with Account class
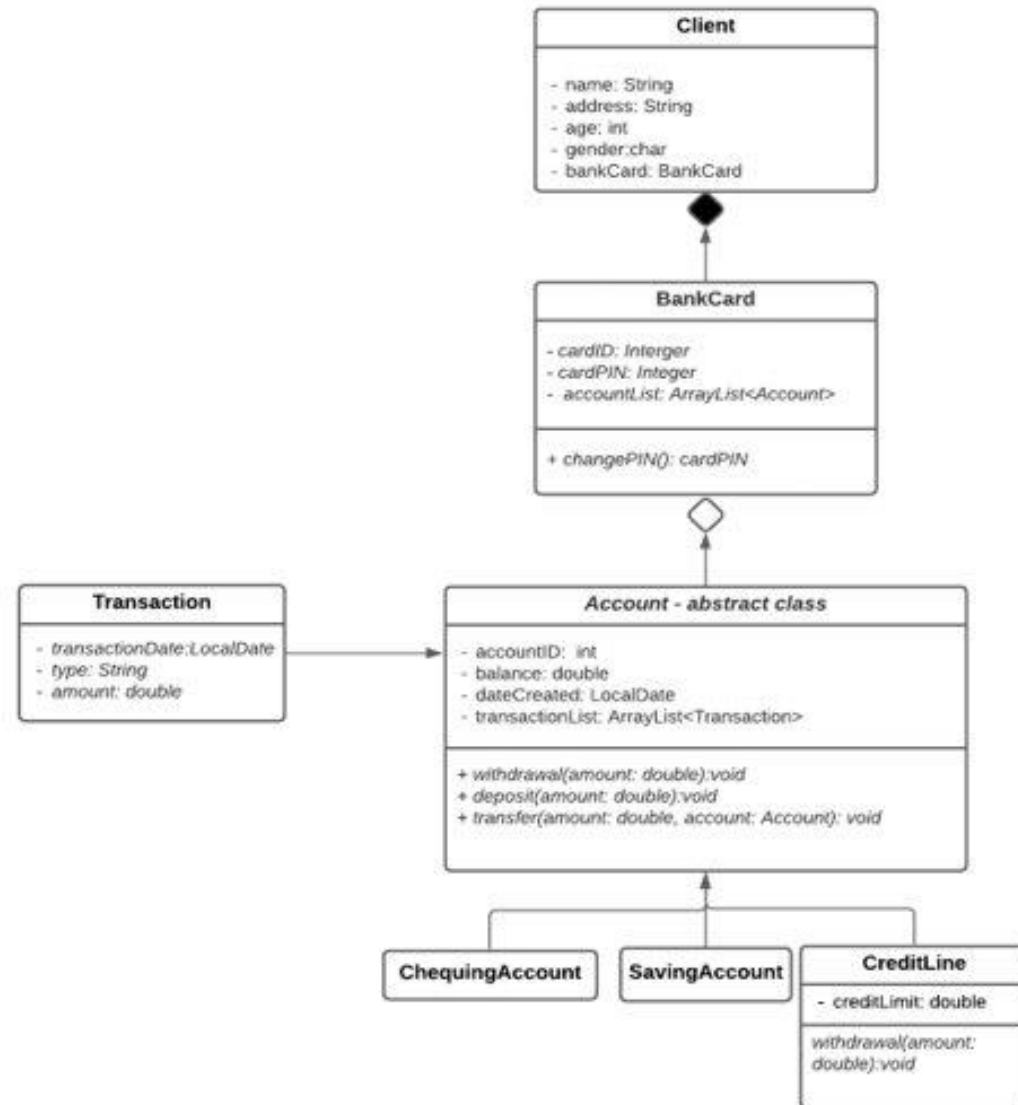
AND THE WINNER IS?

In reality, almost all programming is done with a combination of approaches. in object oriented programming, you commonly subdivide the problem by identifying domain objects (which is a top down step), and refining those, then recombining those into the final program — a bottom up step.

Charlie Martin from StackOverflow

ATM

```java
package domain;

import java.util.ArrayList;

public class ATM {
    private String location;
    List<Menu> menuList;

    public ATM(String location, List<Menu> menuList) {
        this.location = location;
        this.menuList = menuList;
    }

    public String getLocation() {
        return location;
    }

    public void setLocation(String location) {
        this.location = location;
    }

    public Menu getMenuList(int i) {
        return menuList.get(i);
    }

}
```

Menu

```java
package domain;

import java.util.ArrayList;

public interface Menu {

    public int displayMenu(ArrayList<Account> accountList);

}
```

DisplayBalance

```java
package domain;

import java.util.ArrayList;

public class DisplayBalance implements Menu  {

    public DisplayBalance() {
        }

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        System.out.println("Your accounts balance:");
        for (int i = 0; i < accountList.size(); i++) {
            System.out.print("Your account " + accountList.get(i).getAccountID(
            System.out.printf("        Balance: %.2f", accountList.get(i).getBal
            System.out.println();
        }
        System.out.println();
        System.out.println("Press Enter key to continue...");
        try
        {
            System.in.read();
        }
        catch(Exception e)
        {}
        return 1;
    }
}
```

Access

```java
package domain;

import java.util.ArrayList;

public class Access implements Menu {

    static Scanner input = new Scanner(System.in);

    public Access() {
    }

    public int displayAccessMenu(BankCard bankcard) {
        System.out.println("Welcome to our Bank.");
        System.out.println("Open 24 hours a day, 7 days a week.");
        System.out.println();

        while (!inputPIN(bankcard)) {
            System.out.println("Wrong PIN try again");
            System.out.println();
        }
        return 1;
    }

    public boolean inputPIN(BankCard bankcard) {
        System.out.print("Enter your PIN: ");
        int temp = input.nextInt();
        input.nextLine();

        if (temp == bankcard.getCardPIN()) {
            return true;

        } else
            return false;
    }

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        return 1;
    }

}
```

MainMenu

```java
package domain;

import java.util.ArrayList;

public class MainMenu implements Menu {
    static Scanner input = new Scanner(System.in);

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        int choice = 0;
        do {
            System.out.println();
            System.out.println("\t\tWelcome to our Bank");
            System.out.println();
            System.out.println("1. MainMenu\t\t4. Transfer");
            System.out.println();
            System.out.println("2. Withdrawal\t\t5. Display Balance");
            System.out.println();
            System.out.println("3. Deposit\t\t6. Change PIN ");
            System.out.println();
            System.out.println("0 - To exit");
            System.out.print("Enter your choice: ");
            choice = input.nextInt();
            if (choice >= 0 && choice < 7) {
                return choice;
            } else {
                System.out.println("Error: Invalid choice. Try again");
                System.out.println();
            }
        } while (true);
    }

}
```

Withdrawal

```java
package domain;
import java.util.ArrayList;

public class Withdrawal implements Menu{
    static Scanner input = new Scanner(System.in);

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        double amount = 0;
        boolean check = false;
        do {
            System.out.println();
            System.out.print("Enter the amount of withdrawal:");
            amount = input.nextDouble();
            amount = Math.round(amount * 100) / 100.0;
            if (amount > 0) {
                check = true;
            } else {
                System.out.println("Wrong amount. Try again");
            }
        } while (!check);
        System.out.println("Your accounts:");
        for (Account n : accountList) {
            System.out.println(n.getAccountID());
        }
        check = false;
        do {
            System.out.print("Choose your account:");
            int acc = input.nextInt();
            for (int i = 0; i < accountList.size(); i++) {
                if (acc == accountList.get(i).getAccountID()) {
                    accountList.get(i).withdrawal(amount);
                    System.out.println();
                    System.out.println(accountList.get(i).toString());
                    System.out.println("Account balance is " + accountList.get(i).getBalance());
                    check = true;
                }
            }
        } while (!check);
        System.out.println();
        System.out.println("Press Enter key to continue...");
        try {
            System.in.read();
        } catch (Exception e) {
        }
        return 1;
    }

}
```

Deposit

```java
package domain;

import java.util.ArrayList;

public class Deposit implements Menu {
    static Scanner input = new Scanner(System.in);

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        double amount = 0;
        boolean check = false;
        do {
            System.out.print("Enter the amount of deposit:");
            amount = input.nextDouble();
            amount = Math.round(amount * 100) / 100.0;
            if (amount > 0) {
                check = true;
            } else {
                System.out.println("Wrong amount. Try again");
            }
        } while (!check);
        System.out.println("Your accounts:");
        for (Account n : accountList) {
            System.out.println(n.getAccountID());
        }
        check = false;
        do {
            System.out.println();
            System.out.print("Choose your account:");
            int acc = input.nextInt();
            for (int i = 0; i < accountList.size(); i++) {
                if (acc == accountList.get(i).getAccountID()) {
                    accountList.get(i).deposit(amount);
                    System.out.println();
                    System.out.println(accountList.get(i).toString());
                    System.out.println("Account balance is " + accountList.get(i).getBalance());
                    check = true;
                }
            }
        } while (!check);
        System.out.println();
        System.out.println("Press Enter key to continue...");
        try {
            System.in.read();
        } catch (Exception e) {
        }
        return 1;
    }

}
```

Transfer (Part 1)

```java
package domain;

import java.util.ArrayList;

public class Transfer implements Menu {
    static Scanner input = new Scanner(System.in);

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        double amount = 0;
        boolean check = false;
        do {
            System.out.print("Enter the amount of transfer:");
            amount = input.nextDouble();
            amount = Math.round(amount * 100) / 100.0;
            if (amount > 0) {
                check = true;
            } else {
                System.out.println("Wrong amount. Try again");
            }
        } while (!check);
        System.out.println("Your accounts:");
        for (Account n : accountList) {
            System.out.println(n.getAccountID());
        }

        check = false;
        Account receivingAccount = new ChequingAccount();

        do {
            System.out.print("Choose an account where you want to deposit funds:");
            int acc = input.nextInt();
            for (int i = 0; i < accountList.size(); i++) {
                if (acc == accountList.get(i).getAccountID()) {
                    receivingAccount = accountList.get(i);
                    check = true;
                }
            }
        }
```

Transfer (Part 2)

```java
        } while (!check);
        check = false;
        do {
            System.out.print("Choose an account where you want to withdraw funds:");
            int acc = input.nextInt();
            for (int i = 0; i < accountList.size(); i++) {
                if (acc == accountList.get(i).getAccountID()) {
                    accountList.get(i).transfer(amount, receivingAccount);
                    System.out.println(accountList.get(i).toString());
                    System.out.println("Account" + accountList.get(i).getAccountID() + "     Balance: " + accountList.get(i).getBalance());
                    System.out.println("Account" + receivingAccount.getAccountID() + "     Balance: " + receivingAccount.getBalance());
                    check = true;
                }
            }
        } while (!check);
        System.out.println();

        System.out.println("Press Enter key to continue...");
        try {
            System.in.read();
        } catch (Exception e) {
        }
        return 1;
    }

}
```

ChangePIN

```java
package domain;

import java.util.ArrayList;

public class ChangePIN implements Menu {

    static Scanner input = new Scanner(System.in);

    @Override
    public int displayMenu(ArrayList<Account> accountList) {
        return 1;
    }

    public int changePin(Client client) {
        boolean check = true;
        while (check) {
            System.out.print("Enter your 4 digit pin : ");
            int pin = input.nextInt();

            if (pin != client.getBankCard().getCardPIN()) {
                System.out.println("Wrong PIN! Try again!");
            } else if

            (pin == client.getBankCard().getCardPIN()) {
                System.out.println("Enter your NEW PIN");
                int newPIN1 = input.nextInt();
                System.out.println("Please re-enter your new PIN");
                int newPIN2 = input.nextInt();
                if (newPIN1 != newPIN2) {
                    System.out.println("Your new PIN does not match");
                } else {
                    client.getBankCard().setCardPIN(newPIN1);
                    System.out.println("Your PIN has been changed successfully");
                    check = false;
                }
            }
        }
//      bankCard.changePIN();
        System.out.println("Press Enter key to continue...");
        try {
            System.in.read();
        } catch (Exception e) {
        }
        return 1;
    }
}
```

App (Part 1)

```java
package domain;

import java.util.ArrayList;

public class App {
    static Scanner input = new Scanner(System.in);

    public static void main(String[] args) {
        Account ac1101 = new ChequingAccount(1101, 100.0);
        Account ac1201 = new SavingAccount(1201, 100.0);
        Account ac1301 = new CreditLine(1301, 0.0, 1000.00);

        Account ac1102 = new ChequingAccount(1102, 100.0);
        Account ac1202 = new SavingAccount(1202, 100.0);

        Account ac1103 = new ChequingAccount(1102, 100.0);
        Account ac1303 = new CreditLine(1302, 0.0, 1000.00);

        Account ac1104 = new ChequingAccount(1102, 100.0);
        Account ac1204 = new SavingAccount(1202, 100.0);
        Account ac1304 = new CreditLine(1302, 0.0, 1000.00);

        BankCard bc1 = new BankCard(123456789, 1234);
        bc1.accountList.add(ac1101);
        bc1.accountList.add(ac1201);
        bc1.accountList.add(ac1301);

        BankCard bc2 = new BankCard(987654321, 1155);
        bc2.accountList.add(ac1102);
        bc2.accountList.add(ac1202);

        BankCard bc3 = new BankCard(147258369, 4523);
        bc3.accountList.add(ac1103);
        bc3.accountList.add(ac1303);

        BankCard bc4 = new BankCard(963258741, 4517);
        bc4.accountList.add(ac1104);
        bc4.accountList.add(ac1204);
        bc4.accountList.add(ac1304);


        List<Client> clientsList = new ArrayList<Client>();
        clientsList.add(new Client("Khaled", 25, 'M', new Address("123", "Prince", "Montreal", "3H3 Z8J"), bc1));
        clientsList.add(new Client("Anastassia", 25, 'F', new Address("173", "Prince", "Montreal", "3H3 L2J"), bc2));
        clientsList.add(new Client("Somayeh", 25, 'F', new Address("193", "Prince", "Montreal", "5H3 Z2J"), bc3));
        clientsList.add(new Client("Teodor", 25, 'M', new Address("121", "Prince", "Montreal", "3H4 Z2J"), bc4));
```

App (Part 2)

```java
List<Menu> menuList = new ArrayList<>();
menuList.add(new Access());
menuList.add(new MainMenu());
menuList.add(new Withdrawal());
menuList.add(new Deposit());
menuList.add(new Transfer());
menuList.add(new DisplayBalance());
menuList.add(new ChangePIN());

ATM atm1 = new ATM("West Island, Montreal", menuList);

Client mainClient = clientsList.get(0); // Card Scanner device - load the client

System.out.println("We are glad to see you today, " + mainClient.getName());
System.out.println();
int choice = ((Access)atm1.getMenuList(0)).displayAccessMenu(mainClient.getBankCard());
while(choice != 0) {
    switch(choice) {

        case 1:  choice = ((MainMenu)atm1.getMenuList(choice)).displayMenu(mainClient.getBankCard().getAccountList());break;

        case 2: choice = ((Withdrawal)atm1.getMenuList(choice)).displayMenu(mainClient.getBankCard().getAccountList());break;

        case 3: choice = ((Deposit)atm1.getMenuList(choice)).displayMenu(mainClient.getBankCard().getAccountList());break;

        case 4: choice = ((Transfer)atm1.getMenuList(choice)).displayMenu(mainClient.getBankCard().getAccountList());break;

        case 5: choice = ((DisplayBalance)atm1.getMenuList(choice)).displayMenu(mainClient.getBankCard().getAccountList());break;

        case 6: choice = ((ChangePIN)atm1.getMenuList(choice)).changePin(mainClient);break;
    }
  }
 }
}
```

Client (Part 1)

```java
package domain;

public class Client {

    private String name;
    private int age;
    private char gender;
    private Address address;
    private BankCard bankCard;

    public Client(String name, int age, char gender, Address address, BankCard bankCard) {
        this.name = name;
        this.age = age;
        this.gender = gender;
        this.address = address;
        this.bankCard = bankCard;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

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

    public char getGender() {
        return gender;
    }

    public void setGender(char gender) {
        this.gender = gender;
    }

    public Address getAddress() {
        return address;
    }
```

Client (Part 2)

```java
    public void setAddress(Address address) {
        this.address = address;
    }

    public BankCard getBankCard() {
        return bankCard;
    }

    public void setBankCard(BankCard bankCard) {
        this.bankCard = bankCard;
    }

    @Override
    public String toString() {
        return "Client: name=" + name + ", age=" + age + ", gender=" + gender + ", address=" + address + ", bankCard="
                + bankCard;
    }

}
```

BankCard (Part 1)

```java
package domain;

import java.util.ArrayList;

public class BankCard {

    private long cardID;
    private int cardPIN;
    List<Account> accountList;

    public BankCard(long cardID, int cardPIN) {
        this.cardID = cardID;
        this.cardPIN = cardPIN;
        this.accountList = new ArrayList<Account>();
    }

    public long getCardID() {
        return cardID;
    }

    public void setCardID(long cardID) {
        this.cardID = cardID;
    }

    public int getCardPIN() {
        return cardPIN;
    }

    public void setCardPIN(int cardPIN) {
        this.cardPIN = cardPIN;

    }

    public ArrayList<Account> getAccountList(){
        return (ArrayList)accountList;
    }

    @Override
    public String toString() {
        return "BankCard [cardID=" + cardID + ", cardPIN=" + cardPIN + "]";
    }
```

BankCard (Part 2)

```java
public void pinValidation() {
    Scanner input = new Scanner(System.in);

    while (true) {
        int yourPinInput = input.nextInt();

        if (cardPIN != yourPinInput) {
            System.out.println("Wrong Password, try again!");
            continue;
        } else {
            System.out.println("Thank You!");
            break;
        }
    }

}

public void changePIN() {
    Scanner input = new Scanner(System.in);

    while (true) {
        int oldPIN = input.nextInt();
        if (oldPIN == cardPIN) {
            while (true) {
                System.out.println("What is your new pin?");
                int newPIN = input.nextInt();
                if (newPIN == oldPIN) {
                    System.out.println("Cannot choose the same PIN. Please select a new one");
                    continue;
                } else if (newPIN != oldPIN) {
                    newPIN = cardPIN;
                    System.out.println("Your new PIN has been set.");
                    break;
                }
            }

        } else {
            System.out.println("You have entered the wrong PIN. Please try again.");
            continue;
        }
    }
}
```

Account (Part 1)

```java
package domain;

import java.util.ArrayList;

public abstract class Account{
    private int accountID;
    private double balance;
    private LocalDate dateCreated;
    protected ArrayList<Transaction> transactionList = new ArrayList<>();

    public Account() {
    }

    protected Account(int accountID, double balance) {
        this.accountID = accountID;
        this.balance = balance;
        this.dateCreated = LocalDate.now();
    }

    public int getAccountID() {
        return accountID;
    }

    public double getBalance() {
        return balance;
    }

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

    public LocalDate getDateCreated() {
        return dateCreated;
    }
    public Transaction getTransaction(int index) {
        return transactionList.get(index);
    }

    public void withdrawal(double amount) {
        if (balance - amount < 0) {
            System.out.println("Insufficien Funds.");
        } else {
            transactionList.add(new Transaction("Withdrawal", amount));
            balance -= amount;
        }
    }
}
```

Account (Part 2)

```java
public void deposit(double amount) {
    transactionList.add(new Transaction("Deposit", amount));
    balance += amount;
}

public void transfer(double amount, Account account) {
    if (balance - amount < 0) {
        System.out.println("Insufficient Funds.");
    } else {
        transactionList.add(new Transaction("Transfer", amount));
        account.transactionList.add(new Transaction("Transfer", amount));
        balance -= amount;
        account.setBalance(account.getBalance() + amount);
    }
}

@Override
public String toString() {
    return transactionList + " ";
}

}
```

Transaction

```java
package domain;

import java.time.LocalDate;

public class Transaction {
    private LocalDate transactionDate = LocalDate.now();
    private String type;
    private double amount;

    public Transaction(String type, double amount) {
        this.type = type;
        this.amount = amount;
    }

    public LocalDate getTransactionDate() {
        return transactionDate;
    }

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public double getAmount() {
        return amount;
    }

    public void setAmount(double amount) {
        this.amount = amount;
    }

    @Override
    public String toString() {
        return "Transaction Date: " + transactionDate + "     Type: " + type + ""
                + "     Amount of transaction: " + amount + "\n";
    }
}
```

CheckingAccount

```java
package domain;

import java.util.ArrayList;

public class ChequingAccount extends Account {

    public ChequingAccount() {
    }

    public ChequingAccount(int newId, double newBalance) {
        super(newId, newBalance);
    }
}
```

SavingAccount

```java
package domain;

public class SavingAccount extends Account {

    public SavingAccount(int newId, double newBalance) {
        super(newId, newBalance);
    }
}
```

CreditLine

```java
package domain;

import java.util.ArrayList;

public class CreditLine extends Account {
    private double creditLimit;

    public CreditLine(int newId, double newBalance, double creditLimit) {
        super(newId, newBalance);
        this.creditLimit = creditLimit;
    }

    @Override
    public void withdrawal(double amount) {
        if (creditLimit - amount < 0) {
            System.out.println("Insufficient Funds.");
        } else {
            transactionList.add(new Transaction("Withdrawal", amount));
            super.setBalance(super.getBalance() - amount);
        }
    }

}
```