WEEK 11

Question 1: Create an interface Account having methods- deposit(), withdraw() and aboutBank() (abountBank() is a static method). Create two classes Saving and Current which implement the Account interface. Call the methods of Saving and Current classes in main method.

Code:

Main.java

```
package com.morgan;
public class Main {
    public static void main(String[] args) {
        Account.aboutBank();
        System.out.println();
        SavingsAccount sa = new SavingsAccount("Bhanu Varhsney", "SA515", 5000);
        sa.displayInfo();
        sa.deposit(2000);
        sa.withdraw(4000);
        sa.withdraw(3000);
        System.out.println();


        CurrentAccount ca = new CurrentAccount("Chandu", "CA562", 2000);
        ca.displayInfo();
        ca.deposit(500);
        ca.withdraw(2500);
        ca.withdraw(2000);
    }
}
```

Account.java

```
package com.morgan;


public interface Account {
    String bankName = "Lalu Varshney Bank";


    void  deposit(double amount);
    void withdraw(double amount);
    static void aboutBank(){
        System.out.println("Bank Name: " + bankName);
    }
}
```

Code:

SavingsAccount.java

```
package com.morgan;
public class SavingsAccount implements Account{
    private String accountHolderName;
    private String accountNumber;
    private static final double MIN_BALANCE = 1000;
    private double balance;
    SavingsAccount(String accountHolderName, String accountNumber, double balance) {
        this.accountHolderName = accountHolderName;
        this.accountNumber = accountNumber;
        this.balance = balance;
    }
    @Override
    public void deposit(double amount){
        if (amount <= 0) {
            System.out.println("Invalid amount");
            return;
        }
        this.balance += amount;
        System.out.println("Deposited " + amount + " to SavingsAccount. New balance: " +
balance);
    }
    @Override
    public void withdraw(double amount){
        if ( (balance - amount) >= MIN_BALANCE && amount > 0 ){
            this.balance -= amount;
            System.out.println("Withdrawn " + amount + " from SavingsAccount. New
balance: " + balance);
        }
        else {
            System.out.println("Invalid amount");
        }
    }
    public void displayInfo(){
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: " + balance);
        System.out.println("Minimum Balance: " + MIN_BALANCE);
    }
}
```

Code:

CurrentAccount.java

```
package com.morgan;
public class CurrentAccount implements Account {
    private String accountHolderName;
    private String accountNumber;
    private double balance;
    private static final double OVERDRAFT_LIMIT = 1000;
    public CurrentAccount(String accountHolderName, String accountNumber, double
balance) {
        this.accountHolderName = accountHolderName;
        this.accountNumber = accountNumber;
        this.balance = balance;
    }
    @Override
    public void deposit(double amount){
        if(amount <= 0){
            System.out.println("Invalid deposit amount");
            return;
        }
        balance += amount;
        System.out.println("Deposited " + amount + " to CurrentAccount. New balance: " +
balance);
    }
    @Override
    public void withdraw(double amount){
        if(amount <= 0){
            System.out.println("Invalid withdraw amount");
            return;
        }
        if(balance - amount >= -OVERDRAFT_LIMIT){
            balance -= amount;
            System.out.println("Withdrawn " + amount + " from CurrentAccount. New
balance: " + balance);
        } else {
            System.out.println("Withdrawal exceeds overdraft limit. Transaction denied.");
        }
    }
    public void displayInfo(){
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: " + balance);
        System.out.println("Overdraft Limit: " + OVERDRAFT_LIMIT);
    }
}
```

Output:

```
"C:\Program Files\Java\OpenJDK\jdk-25\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=56425" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath "D:\Uni Material\LAB\sem 3\Week 11\Question1\target\classes" com.morgan.Main
Bank Name: Lalu Varshney Bank

Account Holder Name: Bhanu Varhsney
Account Number: SA515
Balance: 5000.0
Minimum Balance: 1000.0
Deposited 2000.0 to SavingsAccount. New balance: 7000.0
Withdrawn 4000.0 from SavingsAccount. New balance: 3000.0
Invalid amount

Account Holder Name: Chandu
Account Number: CA562
Balance: 2000.0
Overdraft Limit: 1000.0
Deposited 500.0 to CurrentAccount. New balance: 2500.0
Withdrawn 2500.0 from CurrentAccount. New balance: 0.0
Withdrawal exceeds overdraft limit. Transaction denied.

Process finished with exit code 0
```

Question 2: In the previous question, create a new method in Account interfacetakeLoan() (takeLoan() is a default method). takeLoan() method would be implemented by Saving class only. Call the methods of Saving and Current classes in main method

Code:

Main.java

```
package com.morgan;
public class Main {
    public static void main(String[] args) {

        Account.aboutBank();
        System.out.println();

        SavingsAccount sa = new SavingsAccount("Bhanu Varhsney", "SA515", 5000);
        sa.displayInfo();
        sa.deposit(2000);
        sa.withdraw(4000);
        sa.withdraw(3000);
        sa.takeLoan(15000);

        System.out.println("----------");

        CurrentAccount ca = new CurrentAccount("Chandu", "CA562", 2000);
        ca.displayInfo();
        ca.deposit(500);
        ca.withdraw(2500);
        ca.withdraw(2000);
    }
}
```

Account.java

```
package com.morgan;

public interface Account {

    String bankName = "Lalu Varshney Bank";

    void  deposit(double amount);
    void withdraw(double amount);

    static void aboutBank(){
        System.out.println("Bank Name: " + bankName);
    }
    default void takeLoan(double amount) {
        System.out.println("Loan feature is not available for this account.");
    }
}
```

Code: *SavingsAccount.java*

```
package com.morgan;
public class SavingsAccount implements Account{

    private String accountHolderName, accountNumber;
    private static final double MIN_BALANCE = 1000;
    private double balance;

    SavingsAccount(String accountHolderName, String accountNumber, double balance) {
        this.accountHolderName = accountHolderName;
        this.accountNumber = accountNumber; this.balance = balance;
    }
    @Override
    public void deposit(double amount){
        if (amount <= 0) {
            System.out.println("Invalid amount"); return;
        }
        this.balance += amount;
        System.out.println("Deposited " + amount + " to SavingsAccount. New balance: " +
balance);
    }
    @Override
    public void withdraw(double amount){
        if ( (balance - amount) >= MIN_BALANCE && amount > 0 ){
            this.balance -= amount;
            System.out.println("Withdrawn " + amount + " from SavingsAccount. New
balance: " + balance);
        }
        else System.out.println("Invalid amount");
    }
    @Override
    public void takeLoan(double amount) {
        if (amount <= 0) {
            System.out.println("Invalid loan amount."); return;
        }
        System.out.println("Loan approved for " + accountHolderName + "
(SavingsAccount): " + amount);
    }
    public void displayInfo(){
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: " + balance);
        System.out.println("Minimum Balance: " + MIN_BALANCE);
    }
}
```

Code: *CurrentAccount.java*

```
package com.morgan;
public class CurrentAccount implements Account {

    private String accountHolderName;
    private String accountNumber;
    private double balance;
    private static final double OVERDRAFT_LIMIT = 1000;

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

    @Override
    public void deposit(double amount){
        if(amount <= 0){
            System.out.println("Invalid deposit amount"); return;
        }
        balance += amount;
        System.out.println("Deposited " + amount + " to CurrentAccount. New balance: " +
balance);
    }

    @Override
    public void withdraw(double amount){
        if(amount <= 0){
            System.out.println("Invalid withdraw amount"); return;
        }
        if(balance - amount >= -OVERDRAFT_LIMIT){
            balance -= amount;
            System.out.println("Withdrawn " + amount + " from CurrentAccount. New
balance: " + balance);
        }else {
            System.out.println("Withdrawal exceeds overdraft limit. Transaction denied.");
        }
    }
    public void displayInfo(){
        System.out.println("Account Holder Name: " + accountHolderName);
        System.out.println("Account Number: " + accountNumber);
        System.out.println("Balance: " + balance);
        System.out.println("Overdraft Limit: " + OVERDRAFT_LIMIT);
    }
}
```

Output:

```
"C:\Program Files\Java\OpenJDK\jdk-25\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=62512" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath "D:\Uni Material\LAB\sem 3\Week 11\Question2\target\classes" com.morgan.Main
Bank Name: Lalu Varshney Bank

Account Holder Name: Bhanu Varhsney
Account Number: SA515
Balance: 5000.0
Minimum Balance: 1000.0
Deposited 2000.0 to SavingsAccount. New balance: 7000.0
Withdrawn 4000.0 from SavingsAccount. New balance: 3000.0
Invalid amount
Loan approved for Bhanu Varhsney (SavingsAccount): 15000.0
----------
Account Holder Name: Chandu
Account Number: CA562
Balance: 2000.0
Overdraft Limit: 1000.0
Deposited 500.0 to CurrentAccount. New balance: 2500.0
Withdrawn 2500.0 from CurrentAccount. New balance: 0.0
Withdrawal exceeds overdraft limit. Transaction denied.

Process finished with exit code 0
```

Question 3: Create interfaces Bike and Scooty, both of which have two methodsoffer() and details() (details() is default method). Create a new class BuySomething which implements both interfaces. To remove ambiguity, create a method details() in BuySomething class as well in which call the details() method of both interfaces. Call the methods of BuySomething class in main method.

Code:

Bike.java

```
package mart.chandu;
public interface Bike {
   void offer();
   default void details() {
      System.out.println("Bike details: Sports design, 650cc engine.");
   }
}
```

Scooty.java

```
package mart.chandu;
public interface Scooty {
   void offer();
   default void details() {
      System.out.println("Scooty details: Lightweight, 200cc engine.");
   }
}
```

BuySomething.java

```
package mart.chandu;
public class BuySomething implements Bike, Scooty {
   @Override
   public void offer() {
      System.out.println("Exclusive Offer: Get upto ₹5000 off on your purchase!");
   }
   @Override
   public void details() {
      Bike.super.details();
      Scooty.super.details();
      System.out.println("BuySomething: Compare both and choose what fits you best!");
   }
}
```

Main.java

```
package mart.chandu;
public class Main {
   public static void main(String[] args) {
      BuySomething bs = new BuySomething();
      bs.offer();
      bs.details();
   }
}
```

Output:

```
"C:\Program Files\Java\OpenJDK\jdk-25\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=49769" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath "D:\Uni Material\LAB\sem 3\Week 11\Question3\target\classes" mart.chandu.Main
Exclusive Offer: Get upto ₹5000 off on your purchase!
Bike details: Sports design, 200cc engine.
Scooty details: Lightweight, 110cc engine.
BuySomething: Compare both and choose what fits you best!

Process finished with exit code 0
```

OPTIONAL

Question 4: Create two interfaces Printer and Scanner, both having methods connect() and details() (details() is a default method). Create a class MultiFunctionMachine that implements both interfaces. In MultiFunctionMachine, override the details() method to resolve ambiguity and call the details() methods of both interfaces. Call all methods of MultiFunctionMachine in the main() method.

Code:

Printer.java

```
package ino.sam;
public interface Printer {
    void connect();
    default void details() {
        System.out.println("Printer details: Supports color printing.");
    }
}
```

Scanner.java

```
package ino.sam;
public interface Scanner {
    void connect();
    default void details() {
        System.out.println("Scanner details: Supports PDF scanning.");
    }
}
```

Code:

MultiFunctionMachine.java

```
package ino.sam;

public class MultiFunctionMachine implements Printer, Scanner {

    @Override
    public void connect() {
        System.out.println("Connected to MultiFunctionMachines.");
    }

    @Override
    public void details() {
        Printer.super.details();
        Scanner.super.details();
        System.out.println("MultiFunctionMachine: Combines printer and
scanner in one device.");
    }

}
```

Main.java

```
package ino.sam;
public class Main {
    public static void main(String[] args) {
        MultiFunctionMachine mfm = new MultiFunctionMachine();
        mfm.connect();
        mfm.details();
    }
}
```

Output:

```
"C:\Program Files\Java\OpenJDK\jdk-25\bin\java.exe" "-javaagent:C:\Program
Files\JetBrains\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=51771" -Dfile.encoding=UTF-8 -
Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath "D:\Uni
Material\LAB\sem 3\Week 11\Question4\target\classes" ino.sam.Main
Connected to MultiFunctionMachines.
Printer details: Supports color printing.
Scanner details: Supports PDF scanning.
MultiFunctionMachine: Combines printer and scanner in one device.


Process finished with exit code 0
```

Question 5: Create an interface Device with a method powerOn(). Create another interface SmartDevice that extends Device and adds a method connectWiFi(). Create a class SmartPhone that implements SmartDevice. Demonstrate calling both powerOn() and connectWiFi() using a SmartPhone object in the main() method.

Code:

Device.java

```
package blast.nord;
public interface Device {
    void powerOn();
}
```

SmartDevice.java

```
package blast.nord;
public interface SmartDevice extends Device {
    void connectWiFi();
}
```

SmartPhone.java

```
package blast.nord;
public class SmartPhone implements SmartDevice {
    @Override
    public void powerOn() { System.out.println("SmartPhone is powering on..."); }
    @Override
    public void connectWiFi() { System.out.println("SmartPhone connected to Wi-Fi."); }
}
```

Main.java

```
package blast.nord;
public class Main {
    public static void main(String[] args) {
        SmartPhone phone = new SmartPhone();
        phone.powerOn();
        phone.connectWiFi();
    }
}
```

Output:

```
"C:\Program Files\Java\OpenJDK\jdk-25\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2025.2.1\lib\idea_rt.jar=57104" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath "D:\Uni Material\LAB\sem 3\Week 11\Question5\target\classes" blast.nord.Main

SmartPhone is powering on...

SmartPhone connected to Wi-Fi.


Process finished with exit code 0
```