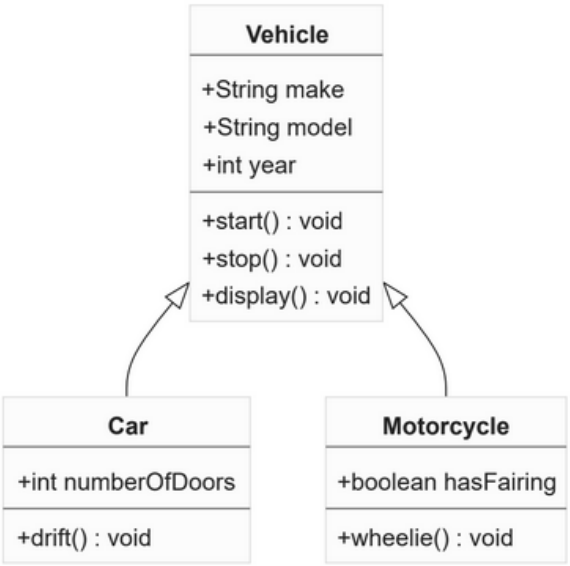
Title: Basics of Inheritance

Objectives:

- Understanding parent class and child class (Inheritance)
- Member access using super keyword
- Differentiating Overriding and Overloading
- Experimenting the effect of access specifier in inheritance
- Final classes and methods

Programs:

- Program to represent Car and Motocycle extending a vehicle class




```
import java.util.Random;

class MRandom {
    private static Random r = new Random();

    private MRandom() {
    }

    public static int getRandom(int rangeStart, int rangeEnd) {
        return r.nextInt(rangeEnd) + rangeStart;
    }
}

class Vehicle {
    protected String make;
    protected String model;
    protected int year;

    public Vehicle(String make, String model, int year) {
        this.make = make;
        this.model = model;
        this.year = year;
    }

    public void start() {
        System.out.println("The vehicle is starting");
    }

    public void stop() {
```

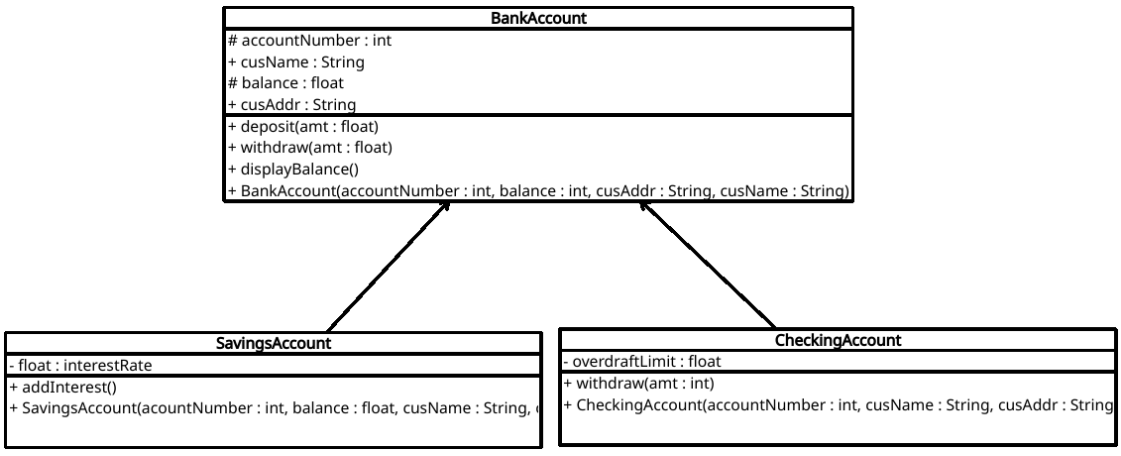
```
            System.out.println("Stopping vehicle");
        }

        public void display() {
            System.out.println("Make = " + make + " model = " + model + " year " + year);
        }
    }

    class Car extends Vehicle {
        private int numberOfDoors;

        public Car(String make, String model, int year, int numberOfDoors) {
            super(make, model, year);
            this.numberOfDoors = numberOfDoors;
        }

        @Override
        public void start() {
            System.out.println("Car Vehicle:");
            super.start();
        }

        @Override
        public void stop() {
            System.out.println("Car Vehicle:");
            super.stop();
        }

        public void drift() {
            int rand = MRandom.getRandom(0, 100);
            System.out.println("Your car drifted " + rand + " meters");
        }

        @Override
        public void display() {
            super.display();
            System.out.println("doors = " + numberOfDoors);
        }
    }

    class Motorcycle extends Vehicle {
        private boolean hasFairing;

        public Motorcycle(String make, String model, int year, boolean hasFairing) {
            super(make, model, year);
            this.hasFairing = hasFairing;
        }

        @Override
        public void start() {
            System.out.println("Motorcycle Vehicle:");
            super.start();
        }

        @Override
        public void stop() {
            System.out.println("Motocycle Vehicle:");
            super.stop();
        }

        public void wheelie() {
            int rand = MRandom.getRandom(0, 10);
            System.out.println("Your motorcycle wheelied " + rand + " meters");
        }
```

```
    @Override
    public void display() {
        super.display();
        System.out.println("fairing = " + hasFairing);
    }

}

public class VehicleApp {
    public static void main(String[] args) {
        Vehicle generic = new Vehicle("Unknown", "Unknown", 0);
        generic.start();
        generic.display();
        generic.stop();
        Car lambo = new Car("Lamborghini", "Gallardo", 2012, 2);
        lambo.start();
        lambo.drift();
        lambo.display();
        lambo.stop();
        Motorcycle unicorn = new Motorcycle("Honda", "Unicorn", 2010, false);
        unicorn.start();
        unicorn.wheelie();
        unicorn.display();
        unicorn.stop();
    }
}
```

Output:

```
The vehicle is starting
Make = Unknown model = Unknown year 0
Stopping vehicle
Car Vehicle:
The vehicle is starting
Your car drifted 55 meters
Make = Lamborghini model = Gallardo year 2012
doors = 2
Car Vehicle:
Stopping vehicle
Motorcycle Vehicle:
The vehicle is starting
Your motorcycle wheelied 9 meters
Make = Honda model = Unicorn year 2010
fairing = false
Motocycle Vehicle:
Stopping vehicle
```

- Program to implement SavingAccount and CheckingAccount inherited from generic BankAccount



```
class BankAccount{
    protected static int  accountCount = 0;
    protected int accountNumber;
    private String cusName;
    protected float balance;
    private String cusAddr;
    public BankAccount(String cusName, float balance, String cusAddr) {
        accountCount++;
        this.accountNumber = accountCount;
        this.cusName = cusName;
        this.balance = balance;
        this.cusAddr = cusAddr;
    }
    void deposit(float amt){
        if(amt<0){
            balance += amt;
            System.out.println(amt +" deposited, New Balance = "+balance);
        }else{
            System.out.println("Deposit amount cannot be negative");
        }
    }
    void withdraw(float amt){
        if(balance-amt > 0){
            balance = balance-amt;
            System.out.println(amt +" withdrawn, New Balance = "+balance);
        }
    }
    void displayBalance(){
        System.out.println("AC num = "+accountNumber);
        System.out.println("Customer name = "+cusName);
        System.out.println("Customer address = "+cusAddr);
        System.out.println("Balance = "+balance);
    }

}

class SavingAccount extends BankAccount{
    private float interestRate;
    public SavingAccount(String cusName, float balance, String cusAddr, float interestRate) {
        super(cusName, balance, cusAddr);
        this.interestRate = interestRate;
    }
    void addInterest(){
        float interest = balance * interestRate/100;
        balance += interest;
        System.out.println("Interest = "+interest+" New Balance = "+balance);
    }
    @Override
    void displayBalance(){
        super.displayBalance();
        System.out.println("Interest Rate = "+interestRate);
    }
}

class CheckingAccount extends BankAccount{
    private float overdraftlimit;
    public CheckingAccount(String cusName, float balance, String cusAddr, float overdraftlimit) {
        super(cusName, balance, cusAddr);
        this.overdraftlimit = overdraftlimit;
    }
    @Override
    void withdraw(float amt){
        if((balance-amt)>(0-overdraftlimit)){
```

```
            balance-=amt;
            System.out.println(amt +" withdrawn, New Balance = "+balance);
        }else{
            System.out.println("Overdraft limit exceeded");
        }
    }
    @Override
    void displayBalance(){
        super.displayBalance();
        System.out.println("Overdraft limt = "+overdraftlimit);
    }
}

public class Bank {
    private static void seperator(){
        System.out.println("-----------------");
    }
    public static void main(String[] args) {
        // Creating a Savings Account
        SavingAccount savingsAccount = new SavingAccount("Ayush Shrestha", 1000, "Mahendrapool", 2.5f)
        savingsAccount.displayBalance();
        seperator();
        savingsAccount.addInterest();
        seperator();
        savingsAccount.displayBalance();

        seperator();
        // Creating a Checking Account
        CheckingAccount checkingAccount = new CheckingAccount("Aayush Mulmi", 500, "Prithivi Chowk", 1
        checkingAccount.displayBalance();
        seperator();

        checkingAccount.withdraw(600); // Attempting to withdraw more than the balance and overdraft l
        seperator();
        checkingAccount.withdraw(50);  // Withdrawing within the overdraft limit
        seperator();
        checkingAccount.displayBalance();

        seperator();
        // Creating a regular Bank Account
        BankAccount bankAccount = new BankAccount("Ayush Pun", 2000, "Lamachaur");
        bankAccount.displayBalance();
        seperator();
        bankAccount.deposit(500);
        seperator();
        bankAccount.withdraw(300);
        seperator();
        bankAccount.displayBalance();
        seperator();
    }

}
```

Output

```
AC num = 1
Customer name = Ayush Shrestha
Customer address = Mahendrapool
Balance = 1000.0
Interest Rate = 2.5
-----------------
Interest = 25.0 New Balance = 1025.0
-----------------
AC num = 1
Customer name = Ayush Shrestha
```

```
Customer address = Mahendrapool
Balance = 1025.0
Interest Rate = 2.5
-----------------
AC num = 2
Customer name = Aayush Mulmi
Customer address = Prithivi Chowk
Balance = 500.0
Overdraft limt = 100.0
-----------------
Overdraft limit exceeded
-----------------
50.0 withdrawn, New Balance = 450.0
-----------------
AC num = 2
Customer name = Aayush Mulmi
Customer address = Prithivi Chowk
Balance = 450.0
Overdraft limt = 100.0
-----------------
AC num = 3
Customer name = Ayush Pun
Customer address = Lamachaur
Balance = 2000.0
-----------------
Deposit amount cannot be negative
-----------------
300.0 withdrawn, New Balance = 1700.0
-----------------
AC num = 3
Customer name = Ayush Pun
Customer address = Lamachaur
Balance = 1700.0
-----------------
```

Conclusion: We used inheritance with overriding to implement various methods in a concise way.