Final Exam

Name: Matthew Knox ID:R11804276

Problem-1

(Your Codes)

Part 1:

```
public abstract class Vehicle {
	String type;
	String brand;
	public Vehicle() {
		type = "car";
		brand = "Mazda";
	}

	public Vehicle(String t, String b) {
		type = t;
		brand = b;
	}

	public void setType(String t) {
		type = t;
	}

	public void setBrand(String b) {
		brand = b;
```

```
    }

    public abstract String getType();
    public abstract String getBrand();
}
```

Part 2:

```
public interface VehicleDetails {
    void setColor(String c);
    String getColor();
}
```

Part 3:

```
public class Car extends Vehicle implements VehicleDetails {
    private String color;
    private String wheels;

    public Car() {
        super();
        color = "black";
        wheels = "four";
    }

    public Car(String t, String b, String c, String w) {
        super(t, b);
        color = c;
        wheels = w;
```

```
}

@Override
public void setColor(String c) {
        color = c;
}

public void setWheels(String w) {
        wheels = w;
}

@Override
public String getColor() {
        return color;
}

public String getWheels() {
        return wheels;
}

@Override
public String getType() {
        return type;
}

@Override
public String getBrand() {
        return brand;
```

```
    }
}
```

Part 4:

```
public class CarDemo {
    public static void main(String[] args) {
        Car c1 = new Car();
        Car c2 = new Car("Car", "Mazda", "Red", "Four");

        c1.setType("Truck");
        c1.setBrand("Ford");
        c1.setColor("Blue");
        c1.setWheels("Four");

        System.out.println("Car1 type: " + c1.getType() + " - Brand: " +
c1.getBrand() + " - Color: " + c1.getColor() + " - Wheels: " + c1.getWheels());
        System.out.println("Car2 type: " + c2.getType() + " - Brand: " +
c2.getBrand() + " - Color: " + c2.getColor() + " - Wheels: " + c2.getWheels());
    }
}
```

Outputs:

Vehicle.java | VehicleDetails.java | Car.java | CarDemo.java | BankAccount.java | BankAccountDemo.java

```java
package FinalExamMatthewKnox;

public class CarDemo {
    public static void main(String[] args) {
        Car c1 = new Car();
        Car c2 = new Car("Car", "Mazda", "Red", "Four");

        c1.setType("Truck");
        c1.setBrand("Ford");
        c1.setColor("Blue");
        c1.setWheels("Four");

        System.out.println("Car1 type: " + c1.getType() + " - Brand: " + c1.getBrand() + " - Color: " + c1.getColor() + "
        System.out.println("Car2 type: " + c2.getType() + " - Brand: " + c2.getBrand() + " - Color: " + c2.getColor() + "
    }
}

```

Problems | Javadoc | Declaration | Console

<terminated> CarDemo (1) [Java Application] C:\Users\mknox\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0.2.v20240802-1626\jre\bin\javaw.exe (Dec 5, 2024, 2:30:16 PM – 2:30:16

```
Car1 type: Truck - Brand: Ford - Color: Blue - Wheels: Four
Car2 type: Car - Brand: Mazda - Color: Red - Wheels: Four
```

Problem-2

Part 1 and 2:

```
public class BankAccount {
	int bal = 50;

	public void withdraw(int w) throws IllegalArgumentException,
InsufficientFundsException {
		if (w <= 0) {
			throw new IllegalArgumentException("Withdraw amount
cannot be negative");
		}

		else if (w > bal) {
			throw new InsufficientFundsException("Withdraw amount is
higher than balance");
		}
	}
}

class InsufficientFundsException extends Exception {
	public InsufficientFundsException(String message) {
		super(message);
	}
}
```

Part 3:

```
public class BankAccountDemo {
```

```
    public static void main(String[] args) throws InsufficientFundsException {
        BankAccount b = new BankAccount();
        try {
            b.withdraw(-50);
        } catch(IllegalArgumentException e1) {
            System.out.println(e1.getMessage());
        }

        try {
            b.withdraw(100);
        } catch(InsufficientFundsException e2) {
            System.out.println(e2.getMessage());
        }
    }
}
```

Outputs:

Vehicle.java | VehicleDetails.java | Car.java | CarDemo.java | BankAccount.java | BankAccountDemo.java

```java
package FinalExamMatthewKnox;

public class BankAccountDemo {
    public static void main(String[] args) throws InsufficientFundsException {
        BankAccount b = new BankAccount();
        try {
            b.withdraw(-50);
        } catch(IllegalArgumentException e1) {
            System.out.println(e1.getMessage());
        }

        try {
            b.withdraw(100);
        } catch(InsufficientFundsException e2) {
            System.out.println(e2.getMessage());
        }
    }
}
```

Problems | Javadoc | Declaration | Console

<terminated> BankAccountDemo [Java Application] C:\Users\mknox\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_22.0

```
Withdraw amount cannot be negative
Withdraw amount is higher than balance
```