Lab Assignment 08

HOMEWORK

Task 1

Implement the design of the Butterfly class and the Moth class so that these classes extend from the Caterpillar class and generate the output as follows.

- Butterfly loses 5 units of energy due to flying and gains 5 energy units per food amount
- Moth loses 2.5 units of energy due to nocturnal activity and gains 2.5 energy units per food amount

Given Code

```
public class Caterpillar_Tester {
  public static void main(String[] args) {
    Caterpillar c = new Caterpillar("Leaf", 5);
    c.showDetails();
    System.out.println("--------------1-------------");
    c.eat();
    c.eat(3);
    c.showDetails();
    System.out.println("--------------2-------------");
    Butterfly b = new Butterfly("Flower", 2);
    Moth m = new Moth("Cotton", 4);
    System.out.println("--------------3-------------");
    b.transform();
    m.transform();
    System.out.println("--------------4-------------");
    b.eat(2);
    m.eat(5);
    System.out.println("--------------5-------------");
    b.showDetails();
    m.showDetails();
  }
}

public class Caterpillar {
  public String food;
  public int age;
  public double energy;
  public Caterpillar(String food, int age) {
    this.food = food;
    this.age = age;
    this.energy = 0;
  }
  public void eat() {
    energy += 1;
    System.out.println("Caterpillar ate 1 unit of food");
  }
  public void eat(int amount) {
    energy += amount;
    System.out.println("Caterpillar ate " + amount + " units of food");
  }
  public void transform() {
    System.out.println("Caterpillar transforms");
  }
  public void showDetails() {
    System.out.println("Food: " + food);
    System.out.println("Age: " + age);
    System.out.println("Energy: " + energy);
  }
}
```

Expected Output

```
Food: Leaf
Age: 5
Energy: 0.0
--------------1-------------
Caterpillar ate 1 unit of food
Caterpillar ate 3 units of food
Food: Leaf
Age: 5
Energy: 4.0
--------------2-------------
--------------3-------------
Caterpillar transforms into Butterfly
Caterpillar transforms into Moth
--------------4-------------
Butterfly lost energy while flying and
absorbed nectar
Moth lost energy due to nocturnal activity
--------------5-------------
Food: Flower
Age: 2
Energy: 5.0 joules
Food: Cotton
Age: 4
Energy: 10.0 joules
```

Task 2

You are given the Audience class. You need to design the Concert (Parent) and VIPConcert (Child) classes so that the following tester code generates the desired output.
Hint: Assume a maximum of 5 artists can be added to a concert. Ticket Pricing:

- Zone A: 500 per ticket
- Zone B: 1,000 per ticket
- Zone VIP: 2,000 per ticket

Driver Code and Given Class	Output

```
public class Audience {
  private String name;
  public Audience(String name) {
    this.name = name;
  }
  public void buyConcertTicket(Concert concert, String zone, int quantity) {
    System.out.println(name + " bought " + quantity + " ticket(s) in Zone " + zone);
    concert.buyTicket(zone, quantity);
  }
  public void buyConcertTicket(Concert concert, String zone) {
    System.out.println(name + " bought 1 ticket(s) in Zone " + zone);
    concert.buyTicket(zone);
  }
  public void buyConcertTicket(VIPConcert concert) {
    System.out.println(name + " bought 1 ticket(s) in Zone VIP");
    concert.buyTicket();
  }
}

//Tester Class
public class ConcertTester{
  public static void main(String[] args) {
    Concert concert = new Concert("ABC Conference Center","7:00 PM");
    concert.showDetails();
    System.out.println("--------1--------");
    concert.addArtist("Tahsan Khan");
    concert.addArtist("Habib Wahid");
    concert.showDetails();
    System.out.println("--------2--------");
    Audience a1 = new Audience("Sarah");
    Audience a2 = new Audience("Alex");
    System.out.println("--------3--------");
    a1.buyConcertTicket(concert, "A", 3);
    a2.buyConcertTicket(concert, "B");
    System.out.println("--------4--------");
    concert.showDetails();
    System.out.println("--------5--------");
    VIPConcert vipConcert = new VIPConcert("Army Stadium","10:00 PM");
    vipConcert.addArtist("Atif Aslam");
    Audience a3 = new Audience("Emily");
    a3.buyConcertTicket(vipConcert);
    System.out.println("--------6--------");
    vipConcert.showDetails();
    System.out.println("--------7--------");;
    Concert.showTotalTicketsSold();
  }
}
```

```
Venue: ABC Conference Center
Showtime: 7:00 PM
Artist:
Tickets sold in this concert: 0
--------1--------
Venue: ABC Conference Center
Showtime: 7:00 PM
Artist:
-Tahsan Khan
-Habib Wahid
Tickets sold in this concert: 0
--------2--------
--------3--------
Sarah bought 3 ticket(s) in Zone A
Total price: 1500
Alex bought 1 ticket(s) in Zone B
Total price: 1000
--------4--------
Venue: ABC Conference Center
Showtime: 7:00 PM
Artist:
-Tahsan Khan
-Habib Wahid
Tickets sold in this concert: 4
--------5--------
Emily bought 1 ticket(s) in Zone VIP
Total price: 2000
--------6--------
Venue: Army Stadium
Showtime: 10:00 PM
Artist:
-Atif Aslam
Tickets sold in this concert: 1
--------7--------
Total tickets sold (all venues): 5
```

Task 3

Given the following classes, write the code for the **CarDriver** and the **BikeDriver** class so that the following output is printed.

Driver Code and Parent Class

```
import java.util.Arrays;
public class PathaoRideManager{
  public static void main(String[] args) {
    System.out.println("1.==============");
    GenericDriver d = new GenericDriver();
    System.out.println("2.==============");
    d.hasSafetyTraining();
    System.out.println("3.==============");
    BikeDriver d1 = new BikeDriver("John", "Not Premium");
    System.out.println("4.==============");
    System.out.println(d1);
    System.out.println("5.==============");
    System.out.println(d1.acceptRide(false));
    System.out.println("6.==============");
    d1.hasSafetyTraining();
    System.out.println("7.==============");
    CarDriver d2 = new CarDriver("Max");
    System.out.println("8.==============");
    System.out.println(d2);
    System.out.println("9.==============");
    d2.hasSafetyTraining();
    System.out.println("10.==============");
    System.out.println(d2.acceptRide(true));
    System.out.println("11.==============");
    System.out.println(d2);
    System.out.println("12.==============");
    BikeDriver.restrictedAreas("Airport Road");
    BikeDriver.restrictedAreas("Defense Area");
    BikeDriver.restrictedAreas("Navy Base");
    System.out.println(Arrays.toString(BikeDriver.restrictedAreas));
    System.out.println("13.==============");
    d1.fightRestriction(new String[]{"Defense Area", "Banani", "Uttara"});
    System.out.println("14.==============");
    CarDriver.restrictedAreas("Cantonment");
    CarDriver.restrictedAreas("Road 27");
    System.out.println(Arrays.toString(CarDriver.restrictedAreas));
    System.out.println("15.==============");
    d2.fightRestriction(new String[]{"Defense Area", "Road 27"});
  }
}

public class GenericDriver {
  private String name;
  public GenericDriver() {
    System.out.println("Welcome to Pathao!");
  }
  public GenericDriver(String name) {
    this.name = name;
    System.out.println("Welcome to Pathao Driver Panel!");
    System.out.println(this.name + " has been registered as a Pathao driver!");
  }
  public String getName() {
    return name;
  }
  public void hasSafetyTraining() {
    System.out.println("All drivers must have safety training.");
  }
  public String acceptRide(boolean hasVehicle) {
    String s = "All Pathao drivers can accept rides. ";
    if (!hasVehicle)
      s += "Driver does not have a verified vehicle. ";
    else
      s += "Driver has a verified vehicle. ";
    return s;
  }
}
```

Expected Output

```
1.==============
Welcome to Pathao!
2.==============
All drivers must have safety training.
3.==============
Welcome to Pathao Driver Panel!
John has been registered as a Pathao driver!
4.==============
John's driver profile is Not Premium
5.==============
All Pathao drivers can accept rides. Driver does not
have a verified vehicle.
John's driver profile is Not Premium
6.==============
All drivers must have safety training.
7.==============
Welcome to Pathao Driver Panel!
Max has been registered as a Pathao driver!
8.==============
Max's driver profile is a Premium
9.==============
All drivers must have safety training.
Premium drivers receive extra safety briefings.
10.==============
All Pathao drivers can accept rides. Driver has a
verified vehicle.
Max's driver profile is Premium
11.==============
Max's driver profile is a Premium
12.==============
[Airport Road, Defense Area, Navy Base]
13.==============
John cannot enter Defense Area
John can enter Banani
John can enter Uttara
14.==============
[Cantonment, Road 27]
15.==============
Max can enter Defense Area
Max cannot enter Road 27
```

Task 4

Design the Car and ElectricCar classes so that the following output is produced. The ElectricCar class and Car class should inherit from the Vehicle class.

Driver Code and Parent Class

```
public class VehicleShowroom {
  public static void main(String[] args) {
    Car c1 = new Car("Toyota Camry", 25000, 4);
    System.out.println("--------1---------");
    c1.vehicleDetail();
    System.out.println("--------2---------");
    Car.showAllAvailableCars();
    System.out.println("--------3---------");
    Car c2 = new Car("Honda Civic", 22000, 4);
    Car c3 = new Car("Ford Mustang", 35000, 2);
    Car.markAsSold(c1);
    Car.markAsSold(c2);
    System.out.println("--------4---------");
    c2.vehicleDetail();
    System.out.println("--------5---------");
    Car.showAllCars();
    System.out.println("--------6---------");
    ElectricCar e1 = new ElectricCar("Tesla Model 3", 45000, 75);
    ElectricCar e2 = new ElectricCar("Nissan Leaf", 32000, 60);
    ElectricCar e3 = new ElectricCar("Nissan Leaf", 32000, 60);
    System.out.println("--------7---------");
    e1.vehicleDetail();
    System.out.println("--------8---------");
    e3.vehicleDetail();
    System.out.println("--------9---------");
    ElectricCar.markAsSoldEV(e1);
    ElectricCar.markAsSoldEV(e2);
    ElectricCar.markAsSoldEV(e3);
  }
}

public class Vehicle {
  public String model;
  public int price;
  public boolean sold;
  public String vehicleId;
  public Vehicle(String model, int price) {
    this.model = model;
    this.price = price;
    this.sold = false;
    this.vehicleId = "";
  }
  public void vehicleDetail() {
    System.out.println("Model: " + model + ", Price: $" + price);
    System.out.print("Status: ");
    if(sold){
      System.out.print("Sold\n");
    }
    else{
      System.out.print("Available\n");
    }
  }
}
```

Output

```
Vehicle ID: CAR001 created
--------1---------
Model: Toyota Camry, Price: $25000
Status: Available
Type: Regular Car
Seats: 4
--------2---------
Total Car: 1
Available Cars:
CAR001 : Toyota Camry
--------3---------
Vehicle ID: CAR002 created
Vehicle ID: CAR003 created
--------4---------
Model: Honda Civic, Price: $22000
Status: Sold
Type: Regular Car
Seats: 4
--------5---------
Total Car: 3
CAR001 : Toyota Camry - sold
CAR002 : Honda Civic - sold
CAR003 : Ford Mustang - available
--------6---------
Vehicle ID: EV000 created
Vehicle ID: EV000 created
Vehicle ID: EV000 created
--------7---------
Model: Tesla Model 3, Price: $45000
Status: Available
Type: Electric Vehicle
Battery Capacity: 75 kWh
--------8---------
Model: Nissan Leaf, Price: $32000
Status: Available
Type: Electric Vehicle
Battery Capacity: 60 kWh
--------9---------
```

Task 5

Design the **Manager** and **Developer** class derived from the **Employee** class with appropriate attributes and properties so that the driver code can generate the output given below. [**Hint**:
Manager:

1. Adds a bonus to the base salary if the manager works more than 40 hours.
2. If the manager works more than 100 hours, the full amount is approved; if they work more than 80 hours, half the amount is approved. Otherwise, the increment is denied.

Developer:

1. Adds $700 to the base salary if the developer works with Java programming language.]

Driver Code and Parent Class

```
public class Employee {
    public String name;
    private double baseSalary;
    private int hoursWorked;

    public Employee(String name, double baseSalary, int hoursWorked){
        this.name = name;
        this.baseSalary = baseSalary;
        this.hoursWorked = hoursWorked;
    }
    public double getBaseSalary() {
        return baseSalary;
    }
    public void setBaseSalary(double baseSalary) {
        this.baseSalary = baseSalary;
    }
    public int getHoursWorked() {
        return hoursWorked;
    }
    public void setHoursWorked(int hoursWorked) {
        this.hoursWorked = hoursWorked;
    }
    public void displayInfo() {
        System.out.println("Name: " + name);
        System.out.println("Base Salary: $" + baseSalary);
        System.out.println("Work Hours: " + hoursWorked);
    }
}
public class EmployeeTester {
 public static void main(String[] args) {
  Manager neymar = new Manager("Neymar",1000, 45, 10);
  Developer messi = new Developer("Messi", 1000, 50, "Java");
  Developer chiesa = new Developer("Chiesa", 1000, 50, "Javascript");
  neymar.calculateSalary();
  System.out.println("1.==========");
  neymar.displayInfo();
  System.out.println("2.==========");
  neymar.requestIncrement(100);
  System.out.println("3.==========");
  neymar.setHoursWorked(85);
  neymar.requestIncrement(100);
  System.out.println("4.==========");
  neymar.calculateSalary();
  System.out.println("5.==========");
  neymar.displayInfo();
  System.out.println("6.==========");
  messi.calculateSalary();
  System.out.println("7.==========");
  messi.displayInfo();
  System.out.println("8.==========");
  chiesa.calculateSalary();
  System.out.println("9.==========");
  chiesa.displayInfo();
 }
}
```

Output

```
1.==========
Name: Neymar
Base Salary: $1000.0
Work Hours: 45
Bonus: 10.0 %
Final Salary: $1100.0
2.==========
Increment denied.
3.==========
$50 Increment approved.
4.==========
5.==========
Name: Neymar
Base Salary: $1050.0
Work Hours: 85
Bonus: 10.0 %
Final Salary: $1155.0
6.==========
7.==========
Name: Messi
Base Salary: $1000.0
Work Hours: 50
Language: Java
Final Salary: $1700.0
8.==========
9.==========
Name: Chiesa
Base Salary: $1000.0
Work Hours: 50
Language: Javascript
Final Salary: $1000.0
```

Task 6

```
1  public class A {
2  public static int temp = 4;
3  public static int x = -10;
4  public int sum, y;
5  public A() {
6    y = temp - 2;
7    sum = temp + 1 + this.x;
8    temp -= 2;
9  }
10 public A(int x){
11    this.methodA(3,5);
12 }
13 public void methodA(int m, int n) {
14   y = y + m + (temp++);
15   x = x + 1 + n;
16   sum = sum + this.x + y;
17   System.out.println(x + " " + y + " " + sum);
18  }
19 }
20 public class B extends A {
21 public static int x = 0;
22 public int sum = -6;
23 public B() {
24   sum = 0;
25   y = temp + 3;
26   super.sum = this.x + super.x + A.x;
27   B.x -= 2;
28 }
29 public B(B b) {
30   super(5);
31   sum = b.sum + super.sum;
32   x = b.x + 1;
33   b.methodB(2, 3);
34 }
35  public void methodA(int m, int n) {
36   y = this.y + n + sum;
37   x = x + 4 + n;
38   sum = super.sum + x + y;
39   System.out.println(x + " " + y + " " + sum);
```

```
40  }
41  public void methodB(int m, int n) {
42    x = y + 2 + (++temp);
43    sum = x + y + sum;
44    System.out.println(x + " " + y + " " + sum);
45    super.methodA(x, y);
46   }
47  }
```

```
B b1 = new B();
B b2 = new B(b1);
b1.methodA(2, 3);
```

x	y	sum