DAY 25 AUG 7TH

NAME : Aravind Kasanagottu

Id:mvsnarav

**Task 01:**

Builder

package com.BuilderMethodDP;

public class BuilderDPDriver {

public static void main(String[] args) {

GadgetBuilder builder = new MobileGadgetBuilder();

GadgetDirector director = new GadgetDirector(builder);

Mobile mobile = director.constructGadget();

System.out.println("Mobile Configuration: " + mobile);

}

}

**Task 02:**

Adaptor

// Assuming this is the adapter class

public class Iphone16Adapter implements Iphone {

private Charger charger; // The adaptee

public Iphone16Adapter() {

this.charger = new Charger();

}

@Override

public void onCharge() {

charger.charge();

// The output "I am charging Iphone 16" and "Stopped charging Iphone 16"

// indicates these methods are probably within Charger.

}

@Override

public void offCharge() {

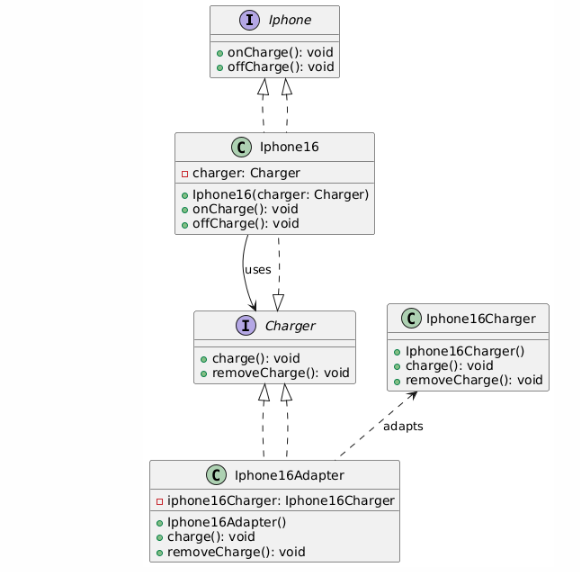
charger.removeCharge();

}

}

**Task03:**

Create a class diagram in uml.



**Task 04: Commonly Used Visibility Notations in Class Diagrams**

In UML class diagrams, visibility notations define the accessibility of a class member (attribute or operation). Here are the most common ones:

* **+** (Public): Accessible from anywhere.
* **-** (Private): Accessible only within the class.
* **#** (Protected): Accessible within the class and its subclasses.
* **~** (Package): Accessible to classes within the same package.

**Task 05: Parameter Directionality**

Parameter directionality in UML specifies the flow of data for an operation's parameters. They are:

* in: The parameter is an **input**; data flows into the operation. This is the default.
* out: The parameter is an **output**; data is passed back to the caller.
* inout: The parameter is both an **input and output**; data is passed in, modified, and returned.
* return: The **return value** of the operation.

**Task 06: Class Relationships**

UML diagrams use various relationships to show how classes interact.

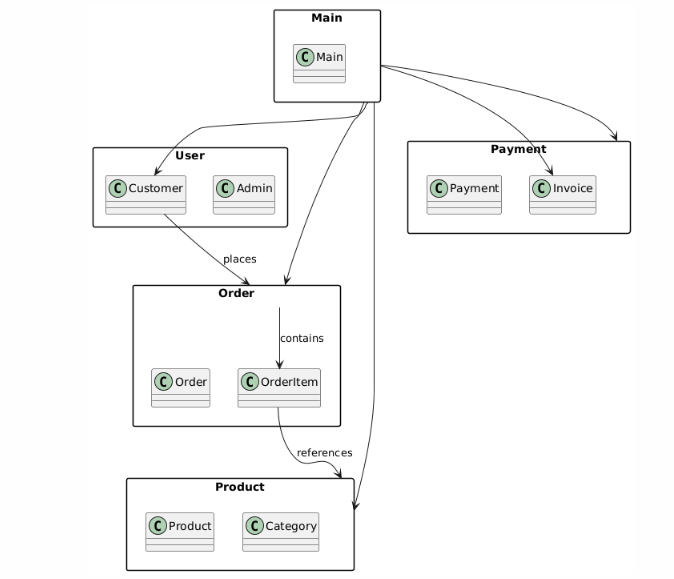
* **Association:** A general link between two independent classes, where one class knows about or uses the other. For example, a student is associated with a teacher.
* **Aggregation:** A "part-of" relationship where the parts can exist independently of the whole. This is a weak "has-a" relationship, represented by a hollow diamond. For example, a Department has Professors; if the department closes, the professors still exist.
* **Composition:** A strong "part-of" relationship where the parts cannot exist without the whole. This is a strong "has-a" relationship, represented by a filled diamond. For example, a House contains Rooms; if the house is destroyed, the rooms are too.
* **Inheritance:** An "is-a" relationship where a subclass inherits properties and behaviors from a superclass. This represents a hierarchical structure. For example, a Dog is an Animal.

**Task 07:**

Package notation

Answer:

**Package Notation is used in UML (Unified Modeling Language) to group related classes, interfaces, diagrams, or other UML elements under a common namespace. It's a way to organize large systems into manageable parts.**

****

**Task 08**

### State Notations in UML

State notation is used in **UML State Machine Diagrams** to represent the different conditions or modes an object can be in.

* **Initial State (●):** Represented by a solid black circle, this is the starting point of the state machine. An object begins its lifecycle in this state.
* **Final State (◎):** Represented by a black filled circle surrounded by a white ring, this indicates the end of a state machine's lifecycle. An object that reaches this state has completed its activity.