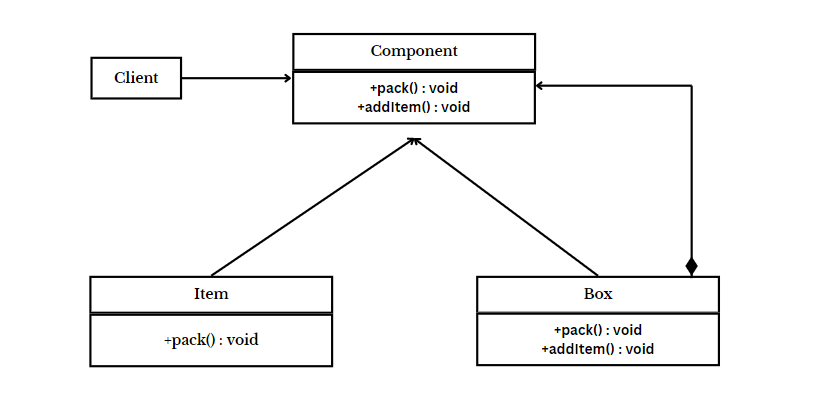
**Composite Design Pattern**

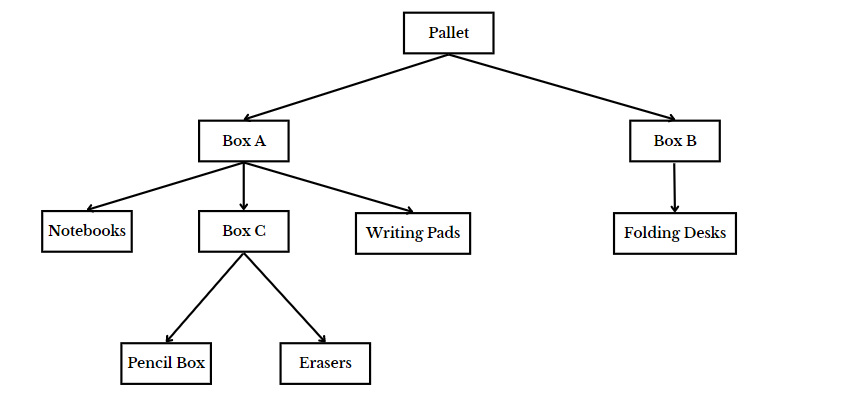
**Assignment - 2**

Name - Heet Dobariya Roll No. - 22BCP177 Group - G5

* **Program :** Implement composite design pattern for the packaging as composite component.
* **UML Diagram :**



* **Tree Diagram :**



* **Code :**

import java.util.ArrayList;

import java.util.List;

// Component interface

interface PackagingComponent

{

void pack();

}

// Leaf class (individual item)

class Item implements PackagingComponent

{

private String itemName;

public Item(String itemName)

{

this.itemName = itemName;

}

@Override

public void pack()

{

System.out.println("Packing individual item (Leaf): " + itemName);

}

}

// Composite class (box or pallet)

class Box implements PackagingComponent

{

private List<PackagingComponent> items = new ArrayList<>();

private String boxName;

public Box(String boxName)

{

this.boxName = boxName;

}

public void addItem(PackagingComponent item)

{

items.add(item);

}

@Override

public void pack()

{

System.out.println("Packing box (Composite): " + boxName);

// Pack individual items in the box

for (PackagingComponent item : items)

{

item.pack();

}

}

}

// Client code

public class PackagingFactoryExample

{

public static void main(String[] args)

{

// Creating individual items

Item item1 = new Item("Notebooks");

Item item2 = new Item("Writing Pads");

Item item3 = new Item("Pencil Box");

Item item4 = new Item("Erasers");

Item item5 = new Item("Foldable Desk");

// Creating composite boxes

Box box1 = new Box("Box C");

Box box2 = new Box("Box A");

Box box3 = new Box("Box B");

// Adding individual items to boxes

box2.addItem(item1);

box2.addItem(item2);

box1.addItem(item3);

box1.addItem(item4);

box2.addItem(box1);

box3.addItem(item5);

// Creating a pallet as a composite of boxes

Box pallet = new Box("Pallet");

pallet.addItem(box2);

pallet.addItem(box3);

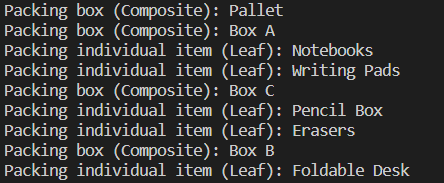
// Packing the entire structure

pallet.pack();

}

}

* **Output :**

****