AOOP ASSIGNMENT

NAMES ID’S

ALUGALA VINAY KUMAR 2110030350

THERALA SANDEEP 2110030196

REDDAMALLA NAGAMEESHWAR 2110030344

GATTU SAICHARAN 2110030437

**1.Choose any 2 Structural Design pattern of your choice.**

* Decorator design pattern
* Bridge design pattern

**2. Similarities and difference between Decorator design patter and Bridge design pattern**

**Similarities:** Both patterns try to simplified complex inherit class structures, so they can look similar in intent, but the problem they try to solve and the way they do is different. Bridge tries to decouple an abstraction from its implementation, and make both independent to vary when needed

**Differences:** Bridge will have two different type hierarchies. One for the super class abstraction and the other for the implementation. The super class abstraction has a reference to the implementation. On the other hand, decorator structure will have one common type at the top

**3. Identify participants**

Here are the participants of the Decorator Design pattern:

* **Component** – this is the wrapper which can have additional responsibilities associated with it at runtime.
* **Concrete component**– is the original object to which the additional responsibilities are added in program.
* **Decorator**-this is an abstract class which contains a reference to the component object and also implements the component interface.
* **Concrete decorator**-they extend the decorator and builds additional functionality on top of the Component class.

Here are the participants of the bridge Design pattern:

* **Abstraction** – core of the bridge design pattern and defines the crux. Contains a reference to the implementer.
* **Refined Abstraction** – Extends the abstraction takes the finer detail one level below. Hides the finer elements from implemetors.
* **Implementer** – It defines the interface for implementation classes. This interface does not need to correspond directly to the abstraction interface and can be very different. Abstraction imp provides an implementation in terms of operations provided by the Implementer interface.
* **Concrete Implementation** – Implements the above implementer by providing the concrete implementation.

**4. Identify one scenario and develop the sample code.**

Code for Decorator design pattern

public class CarDecorator implements Car {

protected Car car;

public CarDecorator(Car c){

this.car=c;

}

@Override

public void assemble() {

this.car.assemble();

}

}

Code for Bridge design pattern

abstract class Vehicle {

protected Workshop workShop1;

protected Workshop workShop2;

protected Vehicle(Workshop workShop1, Workshop workShop2)

{

this.workShop1 = workShop1;

this.workShop2 = workShop2;

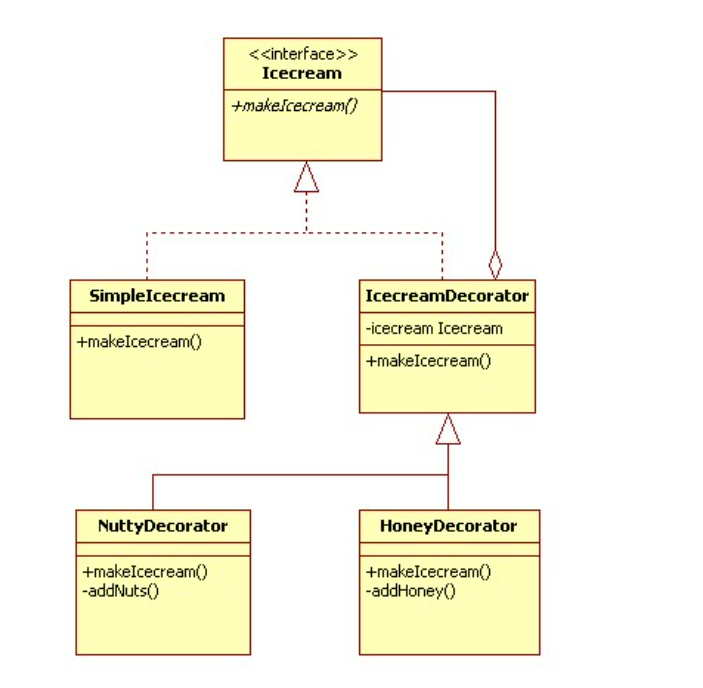
}

abstract public void manufacture();

}

**5. Give any real time example.**

* Ice cream is a classic example for decorator design pattern. You create a basic ice cream and then add toppings to it as you prefer.



* A household switch controlling lights, ceiling fans, etc. is an example for bridge design pattern

