1. **Singleton Design Pattern. Class diagram**

**Client**

**Singleton**

**-instance: singleton**

**-Singleton ()**

**+getInstance: Singleton ()**

**….**

Working data and method of class

………

Public Synchronized static Singleton getInstance () {

If (instance == null) {

Instance = new Singleton ();

}

Return Instance;

}

1. **Factory method pattern class diagram**

**CarFactory**

+ buildCar(model : CarType) : Car

**SedanCar**

**~** SedanCar ()

Construct: void ()

**SmallCar**

**~** SmallCar ()

Construct: void ()

**LuxuryCar**

**~** LuxuryCar ()

Construct: void ()

**Car**

+Car (model: CarType)

Construct (): void

+ GetModel: CarType ()

+ SetModel (model: CarType): void

<<Enum >>

**Car Type**

+enum constant SMALL: CarType

+enum constant SEDAN: CarType

+enum constant LUXURY: CarType

1. **Adaptor Pattern class diagram**

**Adoptee**

+ MethodA ()

**Adapter**

+MethodA ()

**<<Interface>>**

**iTarget**

+MethodA ()

**Client**

+MakeRequest ()

1. **Decorator Design Pattern**

**ConcereteDecoratorA**

Operation ()

AddedBehaviour ()

**ConcereteDecoratorA**

Operation ()

AddedState ()

**Decorator**

Operation ()

**ConcereteComponent**

Operation ()

**Component**

Operation ()

**Client**