1. SRP

**package** com.faith.app;

**import** com.faith.model.BillCalculation;

**import** com.faith.model.Customer;

**import** com.faith.model.DeliveryApp;

**import** com.faith.model.Order;

**public** **class** SRP\_Principle {

**public** **static** **void** main(String[] args) {

Customer customer1 = **new** Customer("Rahi", "Nagercoil");

Order order1 = **new** Order();

order1.setItemName("Pizza");

order1.setQuantity(2);

order1.setCustomer(customer1);

order1.prepareOrder();

BillCalculation billCalculation = **new** BillCalculation(order1);

billCalculation.calculateBill();

DeliveryApp deliveryApp = **new** DeliveryApp(order1);

deliveryApp.delivery();

}

}

**package** com.faith.model;

**public** **class** Customer {

//instance variables

**private** String name;

**private** String address;

//parameterized constructor

**public** Customer(String name, String address) {

**super**();

**this**.name = name;

**this**.address = address;

}

//getters and setters

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getAddress() {

**return** address;

}

**public** **void** setAddress(String address)

{

**this**.address = address;

}

}

**package** com.faith.model;

**import** java.util.Random;

**public** **class** Order {

//instance variables

**private** Customer customer;

**private** String orderId;

**private** String itemName;

**private** **int** quantity;

**private** **int** totalBillAmt;

//getters and setters

**public** Customer getCustomer() {

**return** customer;

}

**public** **void** setCustomer(Customer customer)

{

**this**.customer = customer;

}

**public** String getOrderId() {

**return** orderId;

}

**public** **void** setOrderId(String itemName)

{

Random random = **new** Random();

**this**.orderId = itemName + "-" + random.nextInt(500);

}

**public** String getItemName() {

**return** itemName;

}

**public** **void** setItemName(String itemName)

{

**this**.itemName = itemName;

setOrderId(itemName);

}

**public** **int** getQuantity() {

**return** quantity;

}

**public** **void** setQuantity(**int** quantity)

{

**this**.quantity = quantity;

}

**public** **int** getTotalBillAmt() {

**return** totalBillAmt;

}

**public** **void** setTotalBillAmt(**int** totalBillAmt)

{

**this**.totalBillAmt = totalBillAmt;

}

**public** **void** prepareOrder()

{

System.***out***.println("Preparing order for customer -"

+ **this**.getCustomer().getName()

+ " who has ordered "

+ **this**.getItemName());

}

}

**package** com.faith.model;

**import** java.util.Random;

**public** **class** BillCalculation {

**private** Order order;

**public** BillCalculation(Order order)

{

**this**.order = order;

}

**public** **void** calculateBill()

{

Random rand = **new** Random();

**int** totalAmt

= rand.nextInt(200) \* **this**.order.getQuantity();

**this**.order.setTotalBillAmt(totalAmt);

System.***out***.println("Order with order id "

+ **this**.order.getOrderId()

+ " has a total bill amount of "

+ **this**.order.getTotalBillAmt());

}

}

**package** com.faith.model;

**public** **class** DeliveryApp {

**private** Order order;

**public** DeliveryApp(Order order) {

**this**.order = order;

}

**public** **void** delivery() {

System.***out***.println("Delivering the order");

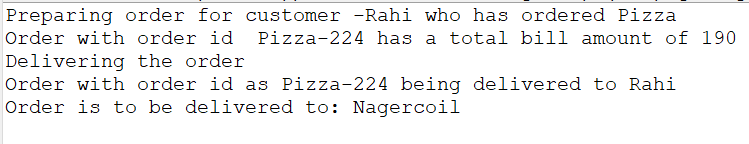
System.***out***.println("Order with order id as " + **this**.order.getOrderId() + " being delivered to "

+ **this**.order.getCustomer().getName());

System.***out***.println("Order is to be delivered to: " + **this**.order.getCustomer().getAddress());

}

}



2. OCP

**package** com.faith.model;

//create an abstract class so that the class can be extended instead of modifying the original class

**public** **abstract** **class** Geo\_objects {

**public** **abstract** **double** get\_volume();

}

**package** com.faith.model;

**public** **class** Cuboid **extends** Geo\_objects{

//instance variables

**public** **double** length;

**public** **double** breadth;

**public** **double** height;

//parameterized constructor

**public** Cuboid(**double** length, **double** breadth, **double** height) {

**super**();

**this**.length = length;

**this**.breadth = breadth;

**this**.height = height;

}

//getters and setters

**public** **double** getLength() {

**return** length;

}

**public** **void** setLength(**double** length) {

**this**.length = length;

}

**public** **double** getBreadth() {

**return** breadth;

}

**public** **void** setBreadth(**double** breadth) {

**this**.breadth = breadth;

}

**public** **double** getHeight() {

**return** height;

}

**public** **void** setHeight(**double** height) {

**this**.height = height;

}

@Override

**public** **double** get\_volume() {

**return** **this**.length \* **this**.breadth \* **this**.height;

}

}

**package** com.faith.model;

**public** **class** Sphere **extends** Geo\_objects{

//instance variables

**public** **double** radius;

//parameterized constructor

**public** Sphere(**double** radius) {

**super**();

**this**.radius = radius;

}

//getters and setters

**public** **double** getRadius() {

**return** radius;

}

**public** **void** setRadius(**double** radius) {

**this**.radius = radius;

}

@Override

**public** **double** get\_volume() {

**return** (4 / 3) \* Math.***PI*** \* radius \* radius \* radius;

}

}

**package** com.faith.model;

**public** **class** Application {

**public** **double** get\_total\_volume(Geo\_objects[] geo\_objects) {

**double** vol\_sum = 0;

//to get the total volume of all objects

**for** (Geo\_objects geo\_obj : geo\_objects) {

vol\_sum += geo\_obj.get\_volume();

}

**return** vol\_sum;

}

}

**package** com.faith.app;

**import** com.faith.model.Application;

**import** com.faith.model.Cuboid;

**import** com.faith.model.Geo\_objects;

**import** com.faith.model.Sphere;

**public** **class** OCP\_Principle {

**public** **static** **void** main(String[] args) {

//Initializing 3 cuboids

Cuboid cuboid1 = **new** Cuboid(5, 10, 15);

Cuboid cuboid2 = **new** Cuboid(2, 4, 6);

Cuboid cuboid3 = **new** Cuboid(3, 12, 15);

//Initializing 3 spheres

Sphere sphere1 = **new** Sphere(5);

Sphere sphere2 = **new** Sphere(2);

Sphere sphere3 = **new** Sphere(3);

//initializing and declaring an array of Geo\_objects

Geo\_objects[] g\_arr = **new** Geo\_objects[6];

// Setting Geo\_objects to cuboid class

g\_arr[0] = cuboid1;

g\_arr[1] = cuboid2;

g\_arr[2] = cuboid3;

// Setting Geo\_objects to sphere class

g\_arr[3] = sphere1;

g\_arr[4] = sphere2;

g\_arr[5] = sphere3;

// Initializing the Application class

Application app = **new** Application();

// Getting the total volume

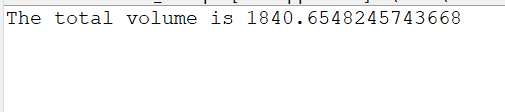
**double** vol = app.get\_total\_volume(g\_arr);

// Printing total volume

System.***out***.println("The total volume is " + vol);

}

}



3. LSP

**package** com.faith.model;

**public** **class** MediaPlayer {

// Play audio implementation

**public** **void** playAudio() {

System.***out***.println("Playing audio...");

}

}

**package** com.faith.model;

**public** **class** VideoMediaPlayer **extends** MediaPlayer{

// Play video implementation

**public** **void** playVideo() {

System.***out***.println("Playing video...");

}

}

**package** com.faith.model;

**public** **class** DivMediaPlayer **extends** VideoMediaPlayer{

**public** **void** output() {

System.***out***.println("------------DivMediaPlayer-----------");

**super**.playAudio();

**super**.playVideo();

}

}

**package** com.faith.model;

**public** **class** VlcMediaPlayer **extends** VideoMediaPlayer{

**public** **void** output() {

System.***out***.println("--------------VlcMediaPlayer---------------");

**super**.playAudio();

**super**.playVideo();

}

}

**package** com.faith.model;

**public** **class** WinampMediaPlayer **extends** MediaPlayer{

**public** **void** output() {

System.***out***.println("--------------WinampMediaPlayer-------------");

**super**.playAudio();

}

}

**package** com.faith.app;

**import** com.faith.model.DivMediaPlayer;

**import** com.faith.model.VlcMediaPlayer;

**import** com.faith.model.WinampMediaPlayer;

**public** **class** LSP\_Principle {

**public** **static** **void** main(String[] args) {

//creating object for each classes

VlcMediaPlayer vlc = **new** VlcMediaPlayer();

vlc.output();

DivMediaPlayer divMedia = **new** DivMediaPlayer();

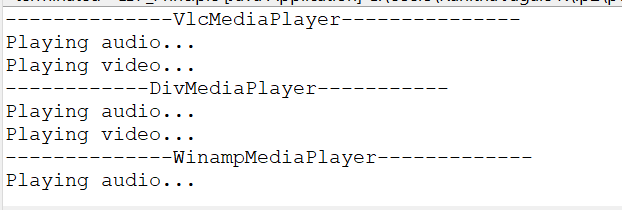
divMedia.output();

WinampMediaPlayer winamp = **new** WinampMediaPlayer();

winamp.output();

}

}



4. ISP

**package** com.faith.model;

//create two other interfaces — one for Jumping athletes and one for Swimming athletes

//instead of putting all athletes into one interface

**public** **interface** Athlete {

**void** compete();

}

**package** com.faith.model;

**public** **interface** SwimmingAthlete **extends** Athlete {

**void** swim();

}

**package** com.faith.model;

**public** **interface** JumpingAthlete **extends** Athlete {

**void** highJump();

**void** longJump();

}

**package** com.faith.model;

**public** **class** SwimmingAthleteImpl **implements** SwimmingAthlete {

@Override

**public** **void** compete() {

System.***out***.println("Athelete started competing for swimming");

}

@Override

**public** **void** swim() {

System.***out***.println("Athelete started swimming");

}

}

**package** com.faith.model;

**public** **class** JumpingAthleteImpl **implements** JumpingAthlete {

@Override

**public** **void** compete() {

System.***out***.println("Athlete started competing for jumping competitions");

}

@Override

**public** **void** highJump() {

System.***out***.println("Athlete starting high Jump");

}

@Override

**public** **void** longJump() {

System.***out***.println("Athlete starting long Jump");

}

}

**package** com.faith.app;

**import** com.faith.model.JumpingAthlete;

**import** com.faith.model.JumpingAthleteImpl;

**import** com.faith.model.SwimmingAthlete;

**import** com.faith.model.SwimmingAthleteImpl;

**public** **class** ISP\_Principle {

**public** **static** **void** main(String[] args) {

//instantiating objects for Swimming and jumping classes

JumpingAthlete jumpingPlayer = **new** JumpingAthleteImpl();

jumpingPlayer.compete();

jumpingPlayer.highJump();

jumpingPlayer.longJump();

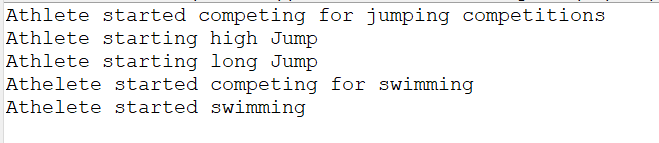
SwimmingAthlete swimmingPlayer = **new** SwimmingAthleteImpl();

swimmingPlayer.compete();

swimmingPlayer.swim();

}

}



2. Cohesion and coupling

Same example as SRP example. If all methods from all classes are put together in a single class, it is called low cohesion as one class is responsible to execute a lot of unrelated actions. SRP is an example for high cohesion.

4. Factory design pattern

**package** com.faith.model;

**public** **interface** Notification {

**void** notifyUser();

}

**package** com.faith.model;

**public** **class** SMSNotification **implements** Notification {

@Override

**public** **void** notifyUser() {

System.***out***.println("Sending an SMS notification");

}

}

**package** com.faith.model;

**public** **class** EmailNotification **implements** Notification {

@Override

**public** **void** notifyUser() {

System.***out***.println("Sending an e-mail notification");

}

}

**package** com.faith.model;

**public** **class** NotificationFactory {

**public** Notification createNotification(String channel)

{

**if** (channel == **null** || channel.isEmpty())

**return** **null**;

**switch** (channel) {

**case** "SMS":

**return** **new** SMSNotification();

**case** "EMAIL":

**return** **new** EmailNotification();

**default**:

**throw** **new** IllegalArgumentException("Unknown channel "+channel);

}

}

}

**package** com.faith.app;

**import** com.faith.model.Notification;

**import** com.faith.model.NotificationFactory;

**public** **class** NotificationService {

**public** **static** **void** main(String[] args) {

NotificationFactory notificationFactory = **new** NotificationFactory();

Notification notification = notificationFactory.createNotification("SMS");

notification.notifyUser();

}

}

