Exemple Principis S.O.L.I.D.

1. Es vol tenir la utilitat de calcular l'àrea total d'un conjunt de formes (Shapes). Es tenen tres tipus de formes: Quadrats, Cercles i Triangles Equilàters i un programador ha fet la següent classe. Quin/s principi/s de disseny viola aquesta classe? Per què? Com canviaries el codi per a que es compleixin?

```
public class Shape {
        //TODO add more shapes if needed
        public static final  int SQUARE = 1;
        public static final  int CIRCLE = 2;
        public static final  int EQUILATERAL_TRIANGLE = 3;
        private double width;
        public int type = -1;

        public Shape(int type, double width)
        {
            this.type = type;
            this.width = width;
        }

        public double getArea() throws Exception
        {
            switch (type)
            {
                case SQUARE:
                    return width * width;
                case CIRCLE:
                    return Math.PI * (width / 2) * (width / 2);
                case EQUILATERAL_TRIANGLE:
                    return (Math.sqrt(3) / 4) * width * width;
            }
            throw new Exception("Can`t compute area of unknown shape");
        }

        public static double CalculateTotalArea(ArrayList<Shape> shapes)
throws Exception
        {
            double totalArea = 0;
            Shape shape;

            for (int i = 0; i < shapes.size(); i++)
```

```
                {
                    if (shapes.get(i).type == SQUARE)
                    {
                        totalArea += shapes.get(i).getArea();
                    }
                    if (shapes.get(i).type == CIRCLE)
                    {
                        totalArea += shapes.get(i).getArea();
                    }
                    if (shapes.get(i).type == EQUILATERAL_TRIANGLE)
                    {
                        totalArea += shapes.get(i).getArea();
                    }
                }
            return totalArea;
        }
}
```

2. Ara es vol afegir un tipus de forma 3D, enlloc de les formes 2D que es tenien en l'exercici 1, com es el cas del Cub. Es vol tenir també la utilitat de calcular els volums totals enlloc de les àrees. Com canvia el codi? Caldrà canviar el Quadrat, el Cercle i el Triangle per a que suportin aquest nou mètode? Mira el codi següent que proposa un programador. Quin/s principi/s vulnera? Com faries que el codi cumplís els principis S.O.L.I.D.?

Interficie IShape:

```
public interface IShape {
    double getArea();
    double getVolume();
}
```

Classe Cub:

```
public class Cube implements IShape {

    private int width;

    public Cube(int w) {
        width = w;
    }

    @Override
    public double getArea()
```

```
    {
        return 6 * width * width;
    }

    @Override
    public double getVolume()
    {
            return width * width * width;
    }
}
```

Classe Square:

```
public class Square implements IShape {

    private int width;

    public Square (int w) {
        width = w;
    }
    @Override
    public double getArea() {
        double area = width * width;
            return area;
    }

    @Override
    public double getVolume() {
        throw new UnsupportedOperationException("Not supported yet.");
//To change body of generated methods, choose Tools | Templates.
    }
}
```

Interficie IVolumeCalculator:

```
public interface IVolumeCalculator {
    double CalculateVolume(ArrayList<IShape> shapes);
}
```

Classe VolumeCalculator:

```
public abstract class VolumeCalculator implements IVolumeCalculator {
    @Override
    public double CalculateVolume(ArrayList<IShape> shapes)
        {
```

```
            double totalVolume = 0;
            for (int i = 0; i < shapes.size(); i++)
            {
                totalVolume += shapes.get(i).getVolume();
            }
            return totalVolume;
        }
}
```

3. Suposa ara que la classe VolumeCalculator vol mostrar el volum a la vegada que el calcula. Es dissenya el següent codi. Quin/s principi/s vulnera? Com canviaries el codi per a que els cumplís?

Classe Messenger:

```
public class Messenger {
    public void Message(double volume)
        {
            System.out.println("The total Volume is " + volume);
        }
}
```

Classe VolumeCalculator:

```
public  class VolumeCalculator implements IVolumeCalculator {

    Messenger message;

    public VolumeCalculator() {
        message = new Messenger();
    }
    @Override
    public double CalculateVolume(ArrayList<IShape> shapes)
        {
            double totalVolume = 0;
            for (int i = 0; i < shapes.size(); i++)
            {
                totalVolume += shapes.get(i).getVolume();
            }
            message.Message(totalVolume);

            return totalVolume;
        }
}
```