CLASE FIGURA:

```
import java.awt.Color;
public abstract class Figura
{
	//ATRIBUTOS DE INSTANCIA

	//int Lado; NO TODAS LAS FIGURAS COMPARTEN ESTE PARAMETRO
	int PosX;
	int PosY;
	Color Col;
	boolean Relleno = false;

	//ATRIBUTOS DE CLASE

	//static boolean RELLENO = false; AHORA ESTO ES PROPIO DE CADA CLASE HIJA



	//MÉTODOS DE INSTANCIA

	void SetPosX(int X)
	{
		if(X>0 && X<800)
			this.PosX = X;
	}

	int getX()
	{
		return PosX;
	}



	void SetPosY(int Y)
	{
		if(Y>0 && Y<600)
			this.PosY = Y;
	}

	int getY()
	{
		return PosY;
	}


/*
	void SetLado(int L)
	{
		if(L<600 && L>1)
```

```
                this.Lado = L;
        }

        int getLado()
        {
                return Lado;
        }
*/

        //YA NO SON MÉTODOS DE CLASE
        void setRelleno(boolean relleno)
        {
                Relleno = relleno;
        }

        boolean isRelleno()
        {
                return Relleno;
        }


        void setColor(Color C)
        {
                Col = C;
        }

        Color getColor()
        {
                return Col;
        }


        String getInfo()
        {
                return "Posicion X: " + this.getX() + "\nPosicion Y: " +
this.getY() + "\nColor: " + this.getColor() + "\nRelleno? =>" +
this.isRelleno();
        }

        abstract void setLado(int L);
        abstract int getLado();
        abstract int getRadio();
        abstract void setRadio(int R);

        //MÉTODOS DE CLASE => NO HAY
/*
        static void setRELLENO(boolean relleno)
        {
                RELLENO = relleno;
        }
```

```
	static boolean isRELLENO()
	{
		return RELLENO;
	}
*/


	//CONSTRUCTOR => SIGUE HABIENDO PARA HACER EL super(X,Y, R, C) EN EL CONSTRUCTOR DE CADA CLASE HIJA

	Figura(int X, int Y, boolean R, Color C)  //POR DEFECTO RELLENO ES FALSE
	{
		this.SetPosX(X);
		this.SetPosY(Y);
		this.setRelleno(R);
		this.setColor(C);
	}
}
```

CLASE CUADRADO:

```
import java.awt.Color;
public class Cuadrado extends Figura
{

	//ATRIBUTOS DE INSTANCIA

	int Lado;

	//ATRIBUTOS DE CLASE

	//static boolean RELLENO = false; NO ES DE CLASE Y SE HEREDA DE FIGURA



	//MÉTODOS DE INSTANCIA SE HEREDAN DE FIGURA EXCEPTO LOS DE LADO
/*
	void SetPosX(int X)
	{
		if(X>0 && X<800)
			this.PosX = X;
	}

	int getX()
	{
		return PosX;
	}



	void SetPosY(int Y)
	{
		if(Y>0 && Y<600)
			this.PosY = Y;
	}

	int getY()
	{
		return PosY;
	}

*/

	@Override
	void setLado(int L)
	{
		if(L<600 && L>1)
			this.Lado = L;
		else
```

```
                this.Lado = 200;
	}

	@Override
	int getLado()
	{
		return Lado;
	}


	@Override
	String getInfo()
	{
		return "Clase: " + this.getClass() + "\n" + super.getInfo() +
"\nLado: " + this.getLado();
	}

	int getRadio(){return 0;} //CHAPUZA => SI NO PONGO ESTO, LIENZO NO VA
(COMO SE HARIA?)

	void setRadio(int R){}; //CHAPUZA => SI NO PONGO ESTO, LIENZO NO VA (COMO
SE HARIA?)
/*

	//MÉTODOS DE CLASE

	static void setRELLENO(boolean relleno)
	{
		RELLENO = relleno;
	}

	static boolean isRELLENO()
	{
		return RELLENO;
	}


*/
	//CONSTRUCTOR

	Cuadrado(int X, int Y, boolean R, Color C, int Lado)
	{
		super(X,Y,R,C);
		this.setLado(Lado);
	}
}
```

CLASE CIRCULO:

```
import java.awt.Color;
public class Circulo extends Figura
{
	int Radio;

	void setRadio(int Rad)
	{
		if(Rad>1 && Rad<600)
			this.Radio = Rad;
		else
			this.Radio = 100;
	}

	int getRadio()
	{
		return Radio;
	}

	@Override
	String getInfo()
	{
		return "Clase: " + this.getClass() + "\n" + super.getInfo() +
"\nRadio: " + this.getRadio();
	}

	void setLado(int L){}; //CHAPUZA => SI NO PONGO ESTO, LIENZO NO VA (COMO
SE HARIA?)
	int getLado(){return 0;} //CHAPUZA => SI NO PONGO ESTO, LIENZO NO VA
(COMO SE HARIA?)

	Circulo(int X, int Y, boolean R, Color C, int Rad)
	{
		super(X,Y,R,C);
		this.setRadio(Rad);
	}
}
```

CLASE APPDIBUJO01:

```
import java.awt.Color;
public class AppDibujo01
{
	public static void main(String[] args)
	{
		//Util util = new Util; NO HACE FALTA INSTANCIAR UTIL =>
Util.wait(s) ES METODO DE CLASE
		Dibujo dibujo = new Dibujo();

		Figura figuras[] = new Figura[10];

		figuras[0] = new Circulo(150,150, true, Color.blue, 100);
		figuras[1] = new Cuadrado(350,100, true, Color.red, 150);
		figuras[2] = new Cuadrado(50,400, false, Color.red, 150);
		figuras[3] = new Circulo(500,400, true, Color.green, 100);

		for(int i=0;i<figuras.length;i++)
			if(figuras[i] != null)
			{
				//System.out.println(figuras[i].getInfo() +
"\n\n--------------------\n");
				dibujo.pintar(figuras[i]);
			}
	}
}
```

CLASE APPDIBUJO02:

```
import java.awt.Color;
public class AppDibujo02
{
	public static void main(String[] args)
	{
		//Util util = new Util; NO HACE FALTA INSTANCIAR UTIL =>
Util.wait(s) ES METODO DE CLASE
		Dibujo dibujo = new Dibujo();

		Figura figuras[] = new Figura[10];

		for(int i=0; i<figuras.length;i++)
		{
			figuras[i] = new
Cuadrado(50+25*i,50+25*i,false,Color.blue,120);
		}


		for(int i=0;i<figuras.length;i++)
			if(figuras[i] != null)
			{
				//System.out.println(figuras[i].getInfo() +
"\n\n--------------------\n");
				dibujo.pintar(figuras[i]);
				Util.wait(1);
			}
	}
}
```

CLASE LIENZO:

```
import java.awt.*;        //AQUÍ HE HECHO COSAS RARAS CON LAS LLAVES
import javax.swing.JFrame;   //PARA QUE ME QUEPA LA CLASE EN UNA SOLA PAGINA
public class Lienzo extends Canvas  //DEL PDF (QUEDABAN COMO 4 LINEAS POR FUERA)
{
	Figura figuras[] = new Figura[10];
	static int NUM = 0;

	void addFigura(Figura fig)
		for(int i=0;i<figuras.length;i++)
			if(figuras[i]==null){
				figuras[i] = fig;
				i=figuras.length;} // ESTO FUNCIONA A MODO DE BREAK

	public void paint(Graphics g)
	{
		int i = 0;
		NUM++;
		//g.setColor(Color.RED);  => AHORA SE ELIGE EN LA INSTANCIA
		for(Figura figura:figuras)
		{
			if(figura!=null)
			{
				System.out.println("N: " + NUM + "-" + ++i);
				g.setColor(figura.getColor());

				if(figura instanceof Cuadrado)
				{
					if(figura.isRelleno())
						g.fillRect(figura.getX(), figura.getY(),
figura.getLado(), figura.getLado());
					else
						g.drawRect(figura.getX(), figura.getY(),
figura.getLado(), figura.getLado());
				}

				if(figura instanceof Circulo)
				{
					if(figura.isRelleno())
						g.fillOval(figura.getX(), figura.getY(),
figura.getRadio(), figura.getRadio());
					else
						g.drawOval(figura.getX(), figura.getY(),
figura.getRadio(), figura.getRadio());
				}
			}
		}
	}
}
```

CLASE DIBUJO:

```
import java.awt.*;
import javax.swing.JFrame;

/**
	Facilita la representación gráfica de objetos creados por el alumno mediante una ventana gráfica y un lienzo
*/
public class Dibujo extends JFrame
{
	Lienzo lienzo;

	public Dibujo()
	{
		super("Dibujo");
		lienzo = new Lienzo();
		lienzo.setSize(800,600);
		this.add(lienzo);
		this.pack();
		this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		this.setVisible(true);
	}

	/**
		Pinta el figura recibido por el App y actualiza el lienzo (canvas)
		@param figura figura a pintar
	*/
	public void pintar(Figura figura)
	{
		lienzo.addFigura(figura);
		lienzo.repaint();
	}

}
```

CLASE UTIL:

```
public class Util
{
	/**
		Detiene el programa el tiempo especificado
		@param segundos número de segundos a esperar
	*/
	static void wait(int segundos)  //static y ya? no public static? quién puede acceder a métodos de clase con el modificador por defecto?
	{
		try
		{
			Thread.sleep(segundos*1000);
		}
		catch(Exception e)
		{

		}
	}
}
```