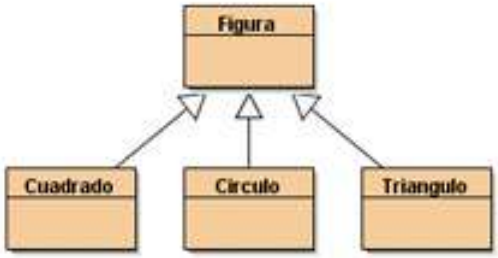
Ejercicio 1

Escribe un programa que implemente la siguiente jerarquía de clases



Implementar aquellos atributos y métodos necesarios para ue se pueda ejecutar el siguiente programa

```
public class Jerarquía {
      public static void main(String[] args) {
          Vector<Figura> figuras = new Vector<Figura>();
          figuras.add(new Circulo(10)); // Radio=10
          figuras.add(new Cuadrado(10)); // Lado=10
          figuras.add(new Triangulo(10,5)); // Base=10, Altura=5;
          for (Figura f: figuras)
              System.out.println("Área: "+f.area());
      }
}
```

Actividad 2

Define una clase abstracta Cuenta con los siguientes atributos:

- numerocuenta : entero largo
- saldo : double
- cliente : atributo de la clase Persona (que tiene nombre y apellidos, y NIF)

y con los siguientes métodos:

- Constructor parametrizado que recibe un cliente y un número de cuenta
- Accedentes para los tres atributos
- ingresar(double): permitirá ingresar una cantidad en la cuenta.
- abstract retirar(double): permitirá sacar una cantidad de la cuenta (si hay saldo). No se implementa, depende del tipo de cuenta
- actualizarSaldo(): actualizará el saldo de la cuenta, pero cada cuenta lo hace de una forma diferente

Define las subclases de Cuenta que se describen a continuación:

- CuentaCorriente: Cuenta normal con un interés fijo del 1.5%. Incluir constructor parametrizado y método toString().
- CuentaAhorro: Esta cuenta tiene como atributos el interés variable a lo largo del año y un saldo mínimo necesario. Al retirar dinero hay que tener en cuenta que no se sobrepase el saldo mínimo. Incluir constructor parametrizado, método toString() y método para cambiar el interés.

Crea un programa que cree varias cuentas y pruebe sus características.

Actividad 3

Se trata de crear una pequeña base de datos de personas de una universidad. De momento definiremos y probaremos las siguientes clases:

- Direccion: o atributos: calle, ciudad, código postal, pais o Constructores predeterminado y parametrizado.
- Persona: Clase ya creada (con nombre, apellidos y NIF, ver ejercicio anterior) a la que añadiremos el atributo dirección y sus métodos accedentes y mutadores. Esta clase implementa la interface Humano, con un método indentificate(), que muestra el tipo de la clase que lo implementa (el tipo de persona, en este caso).
- Estudiante: Subclase de Persona.
    - Atributos: ID de estudiante
    - Constructores : predeterminado y constructor parametrizado que admita el ID.
    - Métodos get y set y toString().
- Profesor: Subclase de Persona.
    - Atributos : despacho o Constructores: predeterminado y constructor parametrizado que admita el despacho.
    - Métodos get y set y toString()

Crea una lista de personas (con la clase Vector) y prueba a añadir varios alumnos y varios profesores a la lista y sus operaciones.