Nombre y apellidos:

1. Iberdrola S.A. necesita un programa para controlar el suministro de luz de sus clientes. De cada cliente se desea saber el número de contrato, el CUPS del contador (String), dirección, ciudad, cuenta corriente y potencia contratada. Cuando un cliente se da de alta se puede hacer asignándole ya un CUPS o no, ya que puede no saberse el contador que le corresponde. Todos los demás datos son necesarios. Iberdrola necesita poder calcular el precio mensual a pagar de un cliente dados los Kw consumidos, la fórmula para ello es: precio=potencia contratada*Kw consumidos/impuesto. El impuesto, actualmente, es una cantidad de 15€ para todos los clientes por igual.

   Un usuario debe de poder cambiar la potencia contratada, antes de poder realizarlo debemos de darle la posibilidad de hacer una simulación de cuanto pagaría dada la nueva potencia contratada y unos Kw.

   Iberdrola desea poder imprimir todos los datos de un cliente.

   No se necesita E/S por parte del usuario. Se valorará el uso de la POO. (3 pto)

2. Cinechachi es un grupo que tiene salas de cine por toda España. Cinechachi desea poder llevar el control de sus salas, para ello necesita saber el nombre de la sala, ciudad, dirección y las películas que tiene en cartelera en cada sala.

   De cada película se desea saber su título, director, duración y género. **No puede haber dos películas con el mismo título.**

   Al crear una sala se puede hacer dando todos sus datos (nombre, ciudad, dirección y número películas en cartelera) o solo su ciudad y número máximo de películas en cartelera.

   Crea la clase Sala con las propiedades mínimas necesarias y los siguientes métodos:

   - **boolean anyadirPelicula(título,director,duración,género):** El método devuelve si se ha podido realizar o no.
   - **boolean eliminarPelicula(String título):** Método que elimine de la cartelera de la sala la película indicada. El método debe devolver true si se ha eliminado la película o false en caso contrario.
   - **int peliculasDeTipo(String genero):** El método debe devolver cuantas películas de un determinado genero hay.
   - **String toString():** Muestra la información de la sala y de todas las películas que hay actualmente en cartelera.
   - **boolean eliminarPeliculaGenero(String genero):** Método que elimine de la cartelera de la sala todas las películas del genero indicado. El método debe devolver true si se ha eliminado la película o false en caso contrario.
   - setNombre y getNombre

   Todos los métodos deben de controlar los posibles errores
   Crea una clase main para probar la clase Sala (Para crear la clase Sala hay que pensar si necesitamos alguna otra clase más).
   No se necesita E/S por parte del usuario
   Se valorará el uso de la POO, las estructuras adecuadas, así como el control de posibles errores. (5 pto)

3. Dado el siguiente código indica si compila o no. **Razona tu respuesta**. En caso de que creas que si indica cuál sería la salida por pantalla. (2pto). **Se valorará la concreción**

```
public class Prueba
{
   public static final double METROS_POR_PIE = 0.3048;

   private double longitud;
   private double anchura;

   Prueba(double la_longitud, double la_anchura, int cambio)
   {
       longitud=la_longitud*cambio;
       anchura=la_anchura*cambio;

   }

   Prueba(double la_longitud, double la_anchura)
   {
       longitud=la_longitud;
       anchura=la_anchura;

   }
   Prueba(){
         longitud=0;
        anchura=0;
   }
   public void setMetros(double m){
         METROS_POR_PIE=m;
   }

   public double perimetro()
   {
       double l=longitud*METROS_POR_PIE;
       double w=anchura*METROS_POR_PIE;
       return (2*l)+(2*w);
   }

   public double area()
   {
       double l=longitud*METROS_POR_PIE;
       double w=anchura*METROS_POR_PIE;
       return w*l;
   }
   public double diagonal()
   {
       double l=longitud*METROS_POR_PIE;
       double w=anchura*METROS_POR_PIE;
       return Math.sqrt(Math.pow(w,2)+Math.pow(l,2));
   }
}


public class pruebaTest {
      public static void main (String[] args) {
             Prueba p1=new Prueba(3,4);
             System.out.println(p1.area());
             System.out.println(p1.perimetro());
             System.out.println(p1.diagonal());
             //System.out.println(p1);

      }
}
```

API

- **Scanner**
  - Instanciación: Scanner sc=new Scanner(System.in)
  - Uso:
    - nextInt()----->enteros;
    - nextDouble->decimales
    - nextLine()--->String

- **Random**
  - Instanciación: Random rnd=new Random();
  - Uso:
    - nextInt(a): genera números aleatorios entre 0 y a-1

- **String**
  - lentgth(): devuelve la longitus del String
  - charAt(i): devuelve el carácter en la posición i
  - String.valueOf(x): devuelve el número x como String

- **Integer.parseInt(s):** devuelve el String s como int

Algoritmo de la burbuja

```
for(int i = 1; i < vec.length; i++) {
        for(int j = 0; j < (vec.length - i); j++) {
                if(vec[j] > vec[j+1]){
                        aux = vec[j];
                        vec[j] = vec[j+1];
                        vec[j+1] = aux;
                }
        }
}
```

Modificadores de acceso
private (solo accesible desde dentro de la clase) y public (accesible por todos)

Otros:
static------->no instanciable, común a todos los objetos
final-------->no modificable