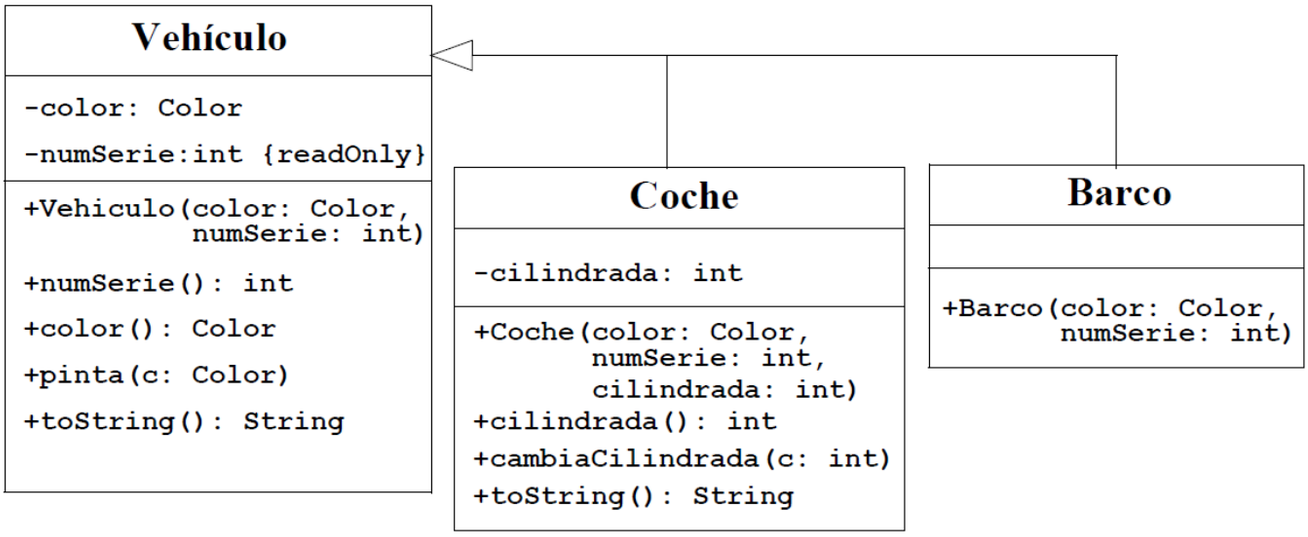
PRÁCTICO SOBRE HERENCIA, POLIMORFISMO E INTERFACES

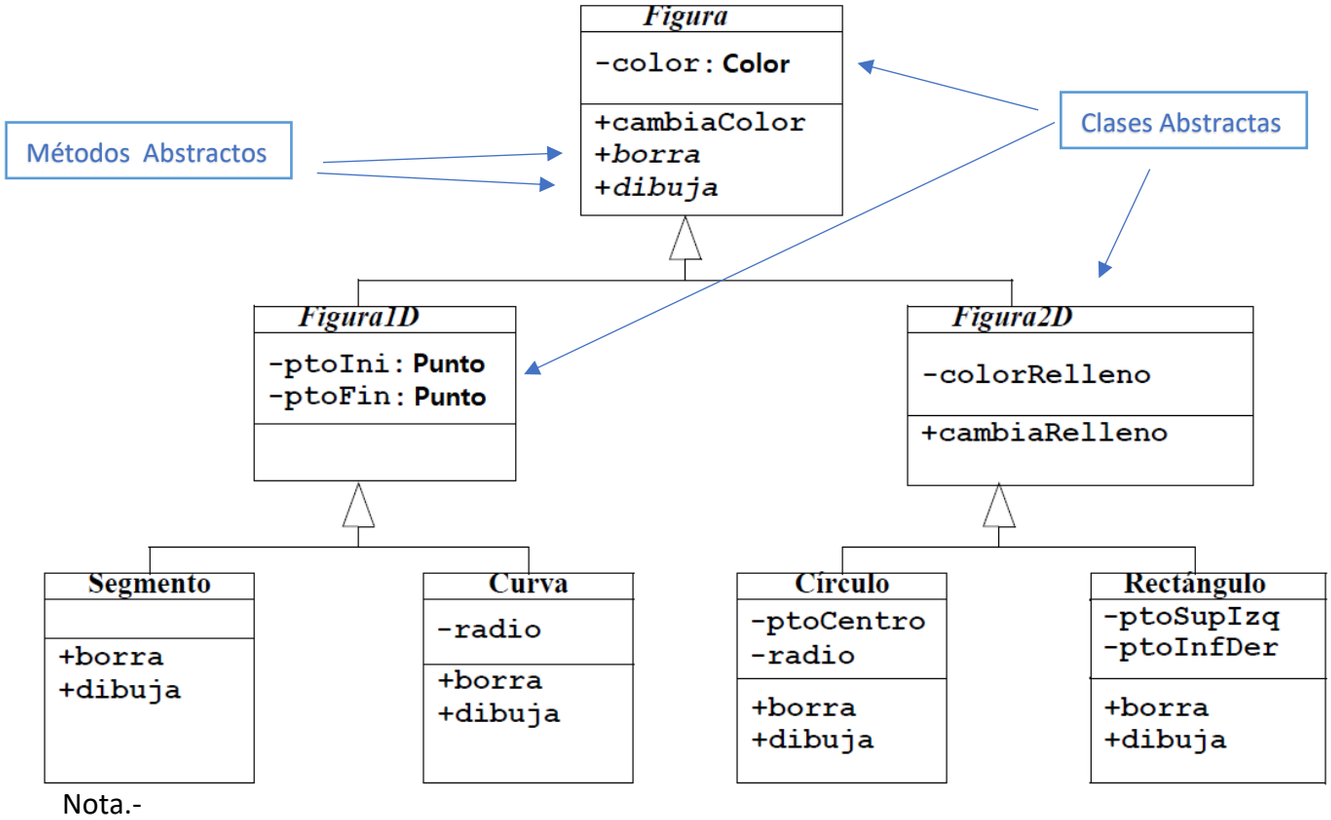
1. Utilizando herencia, diseñar e implementar la siguiente jerarquía de clases.



Nota.

- Investigar como implementar un atributo de sólo lectura (readonly).
- Investigar sobre la palabra reservada **enum** en Java, luego implementar el atributo color con **enum** y que contenga 3 colores ROJO, VERDE Y AZUL.

2. Implementar la siguiente jerarquía utilizando **Clases Abstractas y Métodos Abstractos.**



Nota.-

- Para implementar esta jerarquía deben tener implementada la clase **Punto** dentro del mismo paquete donde estarán definidas las clases.

3. Diseñe e implemente una clase llamada Reloj. Un objeto Reloj contiene una instancia de una hora como las 9:48 P.M.

   Debe tener al menos estos métodos públicos:

   - public Reloj()
     Constructor sin argumentos que inicializa la hora a medianoche.
   - public void setHora(int hora, int minuto, boolean am);
     Establece la hora actual del this de la clase Reloj
     Parámetros:
     hora - la hora a la que se debe configurar el Reloj
     minuto - el minuto en el que se ajusta el Reloj
     am - indicación de si la nueva hora es antes del mediodía

     Precondición: 1 <= hora <= 12, and 0 <= minuto <= 59.

     Postcondición:

     La hora de este reloj se ha ajustado a la hora y minuto dadas (usando la notación habitual de 12 horas). Si el tercer parámetro, am, es verdadero, entonces esta hora es de 12:00 de medianoche a 11:59 A.M. De lo contrario, esta hora es de 12:00 del mediodía a 11:59 P.M.
   - public void avanzar(int minutos)
     Método para avanzar la hora en un número determinado de minutos (que podría ser negativo para mover el reloj hacia atrás o positivo para adelantar el reloj).
   - int getHora()
     Obtiene la hora actual (siempre en el rango de 1 … 12).
   - int getMinutos()
     Obtiene los minutos actuales(siempre en el rango de 0 ... 59).
   - public boolean esManiana()
     Compruebe si la hora de este reloj es antes del mediodía.
     Si la hora de este reloj es desde las 12:00 de la medianoche hasta las 11:59 A.M.(inclusive), entonces el valor devuelto es verdadero; de lo contrario, el valor devuelto es false.
   - public static boolean masTemprano (Reloj c1, Reloj c2)
     Devuelve verdadero si la hora en c1 es anterior a la hora en c2 durante un día normal (comenzando a la medianoche y continuando hasta las 11:59 p.m.); de lo contrario, devuelve falso.

- Luego diseñe e implemente una clase derivada llamada RelojCucu que herede de la clase Reloj la cual tendrá los siguientes métodos:
  - public boolean estaCantando()
    Comprueba si está actualmente cantando
    Si el minuto actual de este RelojCucu es cero, entonces el valor devuelto es verdadero; de lo contrario, el valor devuelto es false.

- Luego diseñe e implemente una clase derivada llamada Reloj24 que herede de la clase Reloj la cual tendrá los siguientes métodos:

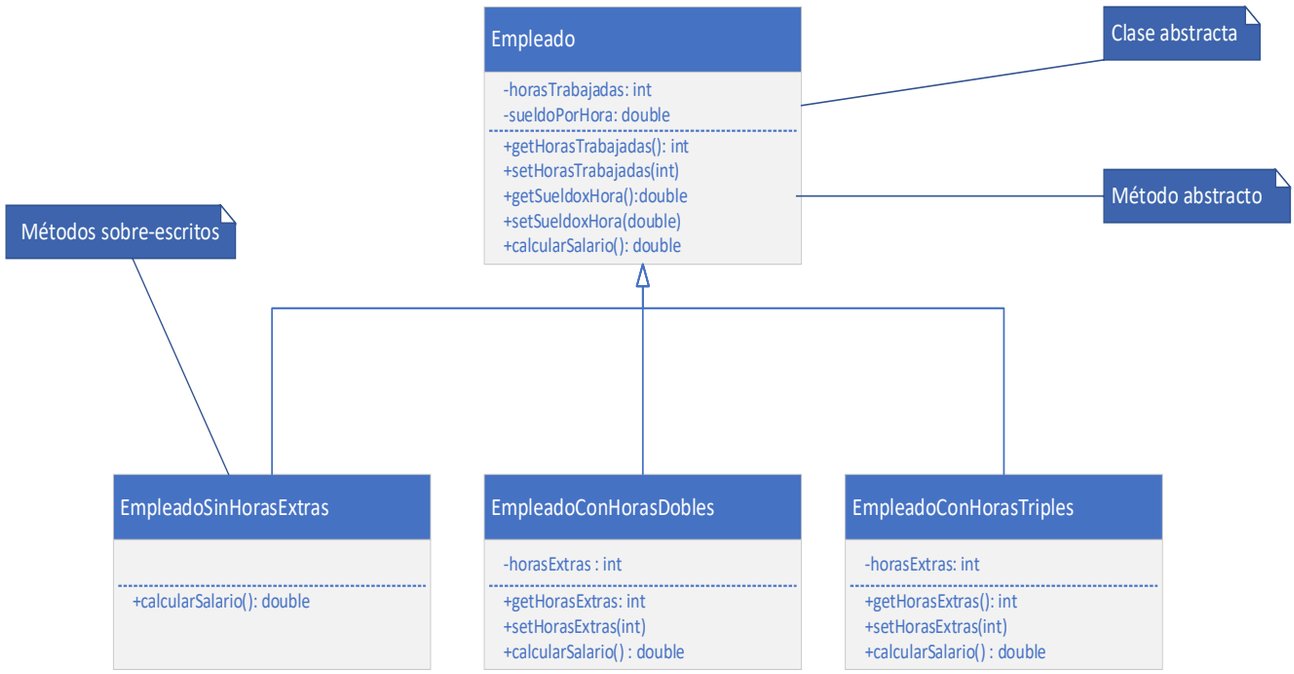
Un objeto Reloj24 es un Reloj con su hora en formato de 24 horas (0 a 23) en lugar de formato de 12 horas

Debe incluir un método:

- public int getHora()
  El cual debe estar sobre escrito (override) desde la super clase Reloj.
  Obtiene la hora actual de este Reloj24, en formato de 24 horas.
  La hora actual (siempre en el rango 0 ... 23)

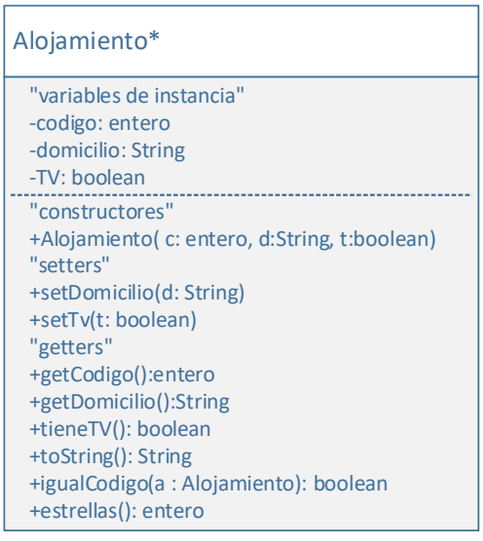
4. Una empresa desea un sistema capaz de calcular el salario semanal de sus empleados de acuerdo a la cantidad de horas trabajadas, el sueldo por hora y tomando en cuenta los siguientes criterios:
   a. Si las horas trabajadas son más de 40, entonces el excedente se considera hora extra.
   b. Si las horas trabajadas están entre el rango de 41 a 45, entonces cada hora extra se paga doble.
   c. Si las horas trabajadas son más de 45, entonces cada hora extra se paga triple.

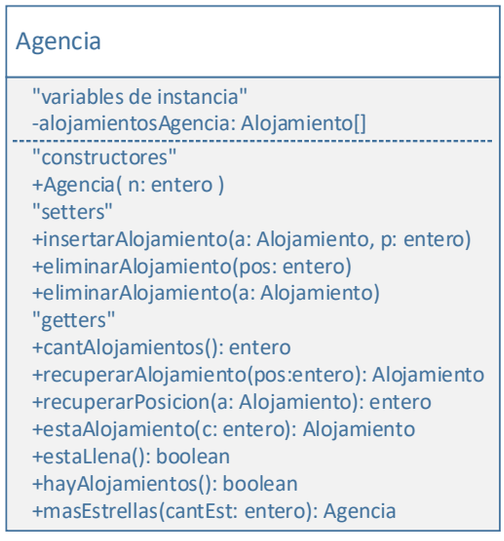
   Implementar el programa de acuerdo al siguiente diagrama UML

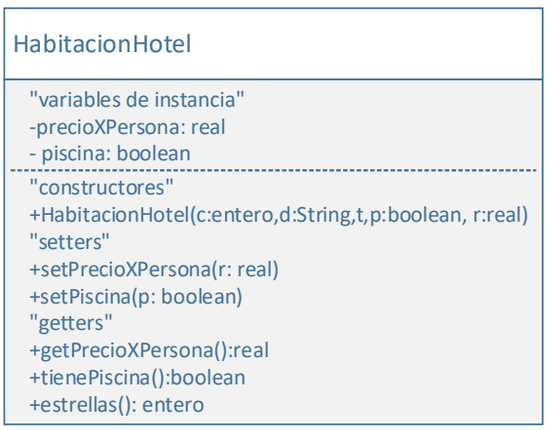


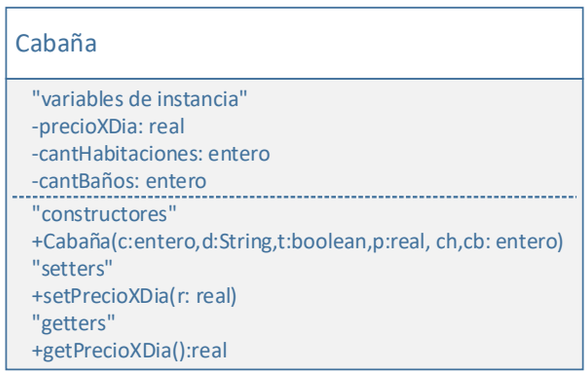
5. Una empresa de turismo almacena datos sobre los diferentes hoteles y cabañas que proporciona a sus clientes. Se crea la clase Alojamiento para centralizar tanto los atributos como las funciones relacionadas con estos establecimientos. En la aplicación todos los alojamientos son instancias de una de las clases derivadas: HabitaciónHotel o Cabaña (ambas clases heredan de alojamiento). La clase Agencia se implementa como un arreglo. La clase cliente tiene acceso a los alojamientos por posición. Algunas posiciones pueden ser nulas.

   a) Implemente el diagrama de clases.
   b) Implemente una clase tester que verifique los servicios provistos por la clase Agencia.









Donde:

- **estrellas(): entero** en Alojamiento retorna 2 si el alojamiento tiene TV, 1 en caso contrario.
- **estrellas(): entero** en HabitacionHotel retorna la cantidad de estrellas de un Alojamiento más 1 si tiene piscina.
- **recuperarPosicion(a: Alojamiento) : entero** busca por identidad.
- **estaAlojamiento(c: entero): Alojamiento** busca por código.
- **masEstrellas(cantEst: entero): Agencia** en Agencia retorna una nueva agencia sólo con los alojamientos que tengan más de cantEst estrellas.
- **hayAlojamientos(): boolean** en Agencia retorna true si y sólo si en el arreglo hay algún alojamiento.

6. Analice si las siguientes afirmaciones son correctas:
   a) Una interface en Java es una clase con todos los métodos abstractos.
   b) Una clase abstracta puede no tener métodos abstractos.
   c) Una clase que extiende a una clase abstracta debe implementar todos los métodos abstractos heredados.
   d) Un método p definido en una clase A se dice sobrecargado si existe una clase B que hereda de A y brinda un método p con el mismo número y tipo de parámetros que A.
   e) Cuando una clase D extiende a una clase C en Java, la clase D hereda todos los atributos de instancia (variables de clase) declarados tanto en la clase D como en la clase C. Esto significa que la clase D puede acceder a los atributos de instancia de ambas clases, formando así el conjunto completo de atributos disponibles dentro de la clase D.
   f) Los métodos privados no pueden ser redefinidos.

7. Identifique errores de compilación, ejecución y aplicación en el siguiente código. A medida que vaya detectando cada error elimine la instrucción o realice una modificación de modo pueda continuar con el análisis de lo que sigue.

```
abstract class Alfa{
  protected int i;
  protected boolean b;
  public Alfa () {
    i = 10;
    b= true;
  }
  abstract public int p(int j) ;
  public int q(int j) {
    return i+j;
  }
  public int r(int j) {
  //retorna la suma de todos los números entre i y j
  int s = 0;
  int m = 0;
  for (int m = i; m <= j; m++);
  s = s + m;
  return s;
  }
}
```

```
class Delta extends Alfa {
  private double k;
  public Delta () {
    super();
    k = 1.5;
  }
  public int p(int x){
    return (int)
    k * i + x;
  }
  public int q(int x,int y) {
    int z = (int) (x * k);
    return z+y;
  }
}
```

```
class Beta extends Alfa {
  protected int k;
  public Beta () {
    k = 2;
    super();
  }
  public int P(int j){
    return j * k;
  }
  public int Q(int j) {
    return i * j * k;
  }
}
```

```
class Gama extends Delta {
  protected int a;
  public Gama () {
    super();
    a = -1;
  }
  public double s(int x){
    return k * a * x;
  }
}
```

```
class Aplicacion {
  public static void main (String a []){
        Gama g1 = null;
        Gama g2 = null;
        Gama g3 = null;
        Delta d = null;
        Beta b = null;
        Alfa a;
        Alfa e1 = null;
        Alfa e3 = null;
        Alfa e2 = e3;
        int i;
        e2= new Alfa();
        i = e2.q(3);
```

```
        e3 = new Delta ();
        i = e3.q(3);
        i= e3.q(4,5);
        Alfa b = new Delta();
        i = b.q(3);
        b = d;
        d = new Delta();
        i = d.q(4,5);
        i = d.s(2);
        d = g1;
        double z;
        g1 = new Delta();
        z = g1.s(2);
        g2 = new Gama();
        z = g2.s(2);
        g1 = g3;
        g1.p(2);
   }
}
```

8. Construir una clase ArrayReales que declare un atributo de tipo double[] para asignar el tamaño del array. Tenga un método para rellenar con números aleatorios dicho array e implemente una interfaz llamada Estadisticas. El contenido de esta interfaz es el siguiente:

```
public interface Estadisticas{
   double minimo();
   double maximo();
   double sumatorio();
}
```

   Debes programar un método para imprimir los valores del array.
   Programa una clase que pruebe todos los métodos implementados de la interfaz.