IES ILIBERIS
Avda. Andalucía
Atarfe - Granada

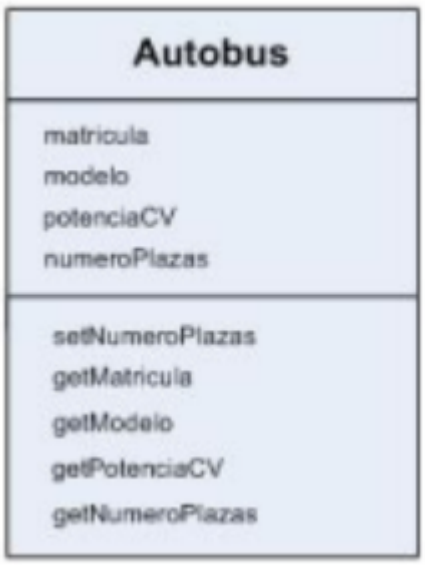
Departamento de Informática
1º DAW
Programación

RELACIÓN DE EJERCICIOS 9 - Clases

Resolver los siguientes problemas escribiendo el algoritmo con lenguaje Java.

Crear en GitHub un repositorio llamado **UD5-Clases** para subir cada uno de los ejercicios de la relación.

1. Crea una clase llamada Vivienda para representar los atributos que consideres necesarios
2. Piensa en el mundial de fútbol ¿Qué 5 clases se te ocurren para representar los elementos que intervienen?
3. Crea las clases Animal, Mamifero, Ave, Gato, Perro. Crea, al menos, tres métodos específicos de cada clase. Prueba las clases creadas en un programa en el que se instancien objetos y se les apliquen métodos.
4. Implementa la clase autobús según el siguiente modelo:



5. Crear la clase bicicleta y posteriormente un objeto con los valores indicados:

RELACIÓN DE EJERCICIOS 9 - Clases

6. Crea la clase Fracción. Los atributos serán numerador y denominador. Y algunos de los métodos pueden ser invierte, simplifica, multiplica, divide. Crea un programa para probar los distintos métodos.

   NOTA: para simplificar una fracción hay que dividir numerador y denominador por el MCD de ambos números. Para calcular el MCD usa el siguiente método (algoritmo de Euclides):

   Se empieza dividiendo el número mayor entre el menor. Si se obtiene de resto 0, entonces el número menor es el m.c.d. de ambos.

   Si no se obtiene de resto 0, se divide el divisor de la división anterior entre el resto obtenido y se repite el mismo razonamiento hasta obtener una división exacta.

   Una vez que hayamos obtenido una división exacta, el divisor de la última división es el máximo común divisor de los dos números iniciales.

   Ejemplo. Vamos a calcular el m.c.d. de los números 1344 y 360.

   Se realizan las divisiones descritas anteriormente hasta obtener una división exacta:

   $$\begin{array}{r|l} 1344 & 360 \\ 264 & 3 \end{array} \qquad \begin{array}{r|l} 360 & 264 \\ 96 & 1 \end{array} \qquad \begin{array}{r|l} 264 & 96 \\ 72 & 2 \end{array} \qquad \begin{array}{r|l} 96 & 72 \\ 24 & 1 \end{array} \qquad \begin{array}{r|l} 72 & 24 \\ 0 & 3 \end{array}$$

   El m.c.d. es el divisor de la última división:

   $$\text{m.c.d.}(1344,360) = 24.$$

7. Crea la clase Pizza con los atributos y métodos necesarios. Sobre cada pizza se necesita saber el tamaño (mediana o familiar), el tipo (margarita, cuatro quesos o funghi) y su estado (pedida o servida). La clase debe almacenar información sobre el número total de pizzas que se han pedido y que se han servido. Siempre que se crea una pizza nueva, su estado es “pedida”. El siguiente código del programa principal debe dar la salida que se muestra:

```java
public class PedidosPizza {
  public static void main(String[] args) {
    Pizza p1 = new Pizza("margarita", "mediana");
    Pizza p2 = new Pizza("funghi", "familiar");
    p2.sirve();
    Pizza p3 = new Pizza("cuatro quesos", "mediana");
    System.out.println(p1);
    System.out.println(p2);
    System.out.println(p3);
    p2.sirve();
    System.out.println("pedidas: " + Pizza.getTotalPedidas());
    System.out.println("servidas: " + Pizza.getTotalServidas());
  }
}
```

```
pizza margarita mediana, pedida
pizza funghi familiar, servida
pizza cuatro quesos mediana, pedida
esa pizza ya se ha servido
pedidas: 3
servidas: 1
```

RELACIÓN DE EJERCICIOS 9 - Clases

8. Queremos gestionar la venta de entradas (no numeradas) de Expocoches Atarfe que tiene 3 zonas, la sala principal con 1000 entradas disponibles, la zona de compra-venta con 200 entradas disponibles y la zona vip con 25 entradas disponibles. Hay que controlar que existen entradas antes de venderlas. La clase Zona con sus atributos y métodos se muestra a continuación:

```
public class Zona {

  private int entradasPorVender;

  public Zona(int n){
    entradasPorVender = n;
  }

  public int getEntradasPorVender() {
    return entradasPorVender;
  }

  /**
   * Vende un número de entradas.
   * Comprueba si quedan entradas libres antes de realizar la venta.
   *
   * @param n número de entradas a vender
   */
  public void vender(int n) {

    if (this.entradasPorVender == 0) {
      System.out.println("Lo siento, las entradas para esa zona están agotadas.");
    } else if (this.entradasPorVender < n) {
      System.out.println("Sólo me quedan " + this.entradasPorVender
                         + " entradas para esa zona.");
    }

    if (this.entradasPorVender >= n) {
      entradasPorVender -= n;
      System.out.println("Aquí tiene sus " + n + " entradas, gracias.");
    }
  }
}
```

El menú del programa debe ser el que se muestra a continuación. Cuando elegimos la opción 2, se nos debe preguntar para qué zona queremos las entradas y cuántas queremos. Lógicamente, el programa debe controlar que no se puedan vender más entradas de la cuenta.

```
1. Mostrar número de entradas libres
2. Vender entradas
3. Salir
```

9. Crea la clase Tiempo con los métodos suma y resta. Los objetos de la clase Tiempo son intervalos de tiempo y se crean de la forma Tiempo t = new Tiempo(1, 20, 30) donde los parámetros que se le pasan al constructor son las horas, los minutos y los segundos respectivamente. Crea el método toString para ver los intervalos de tiempo de la forma 10h 35m 5s. Si se suman por

ejemplo 30m 40s y 35m 20s el resultado debería ser 1h 6m 0s. Realiza un programa de prueba para comprobar que la clase funciona bien.

10. Una empresa quiere implementar un programa que lleve el control de las incidencias que se producen en sus ordenadores. Cada incidencia tiene un código: 1, 2, 3, 4, etc. Cuando se crea una nueva incidencia, se le asigna un código de forma automática y se pone el estado como "pendiente". Al crear una incidencia hay que indicar también el número de puesto (un número entero). Cuando se resuelve una incidencia, hay que proporcionar información sobre cómo se ha resuelto o qué es lo que fallaba, además, el estado pasa a "resuelta". El siguiente trozo de código que va dentro del main genera la salida que se muestra a continuación.

Programa principal:

```
Incidencia inc1 = new Incidencia(105, "No tiene acceso a internet");
Incidencia inc2 = new Incidencia(14, "No arranca");
Incidencia inc3 = new Incidencia(5, "La pantalla se ve rosa");
Incidencia inc4 = new Incidencia(237, "Hace un ruido extraño");
Incidencia inc5 = new Incidencia(111, "Se cuelga al abrir 3 ventanas");
inc2.resuelve("El equipo no estaba enchufado");
inc3.resuelve("Cambio del cable VGA");
System.out.println(inc1);
System.out.println(inc2);
System.out.println(inc3);
System.out.println(inc4);
System.out.println(inc5);
System.out.println("Incidencias pendientes: " + Incidencia.getPendientes());
```

Salida:

```
Incidencia 1 - Puesto: 105 - No tiene acceso a internet - Pendiente
Incidencia 2 - Puesto: 14 - No arranca - Resuelta - El equipo no estaba enchufado
Incidencia 3 - Puesto: 5 - La pantalla se ve rosa - Resuelta - Cambio del cable VGA
Incidencia 4 - Puesto: 237 - Hace un ruido extraño - Pendiente
Incidencia 5 - Puesto: 111 - Se queda colgado al abrir 3 ventanas - Pendiente
Incidencias pendientes: 3
```

11. Implementa las clases Piramide y Rectangulo. Sobre una pirámide se debe saber su altura y sobre un rectángulo se debe saber tanto la base como la altura. Cada una de las clases debe tener un atributo de clase (static) que lleve la cuenta de las pirámides y de los rectángulos creados respectivamente. El siguiente código que va dentro del main genera la salida que se indica.

Programa principal:

```
Piramide p = new Piramide(4);
Rectangulo r1 = new Rectangulo(4, 3);
Rectangulo r2 = new Rectangulo(6, 2);
System.out.println(p);
System.out.println(r1);
System.out.println(r2);
System.out.println("Pirámides creadas: " + Piramide.getPiramidesCreadas());
System.out.println("Rectángulos creados: " + Rectangulo.getRectangulosCreados());
```

RELACIÓN DE EJERCICIOS 9 - Clases

Salida:

```
   *
  ***
 *****
*******

****
****
****

******
******
```

12. Crea las clases Punto y Linea. De un punto se tienen que saber sus coordenadas x e y, mientras que una línea está definida por dos puntos. Define lasclases y los métodos necesarios para que el siguiente código muestre la salida que se indica.

Programa principal:

```
Punto p1 = new Punto(4.21, 7.3);
Punto p2 = new Punto(-2, 1.66);
Linea l = new Linea(p1, p2);
System.out.println(l);
```

Salida:

```
Línea formada por los puntos (4.21, 7.3) y (-2.0, 1.66)
```

13. Vamos a crear la clase Cubo. Un cubo se fabrica con el propósito de contener líquido; por tanto una característica es la cantidad de litros de líquido que contiene en un momento determinado. Por ahora, solo nos interesa saber la capacidad máxima y los litros que contiene el cubo en cada momento, así que esos serán los atributos que tendremos en cuenta. Diseñar los métodos necesarios para obtener la información de la capacidad y de los litros que contiene el cubo en un momento dado. También los métodos para actualizar el contenido del cubo y para pasar líquido de un cubo a otro (obviamente habrá que tener en cuenta las capacidades de ambos cubos, si ya tenían líquido, etc)