

classes i objectes

1. Clasifica las siguientes variables en diferentes clases. Luego un objeto de cada clase para asignarle los valores. Luego imprime los valores.

a)

```
public class Formula1 {
    public static void main(String[] args) {

        String nombrePiloto = "Lewis Hamilton";
        int numeroPiloto = 44;
        String equipoPiloto = "Mercedes-AMG Petronas";
        int podiosTotales = 191;
        String nombreCircuito = "Circuit de Monaco";
        int vueltasTotales = 78;
        float tiempoVueltaRapida = 1.10f;
        int totalCarreras = 23;
        int posicionActualPiloto = 2;

        System.out.println("Piloto: " + nombrePiloto);
        System.out.println("Número: " + numeroPiloto);
        System.out.println("Equipo: " + equipoPiloto);
        System.out.println("Circuito: " + nombreCircuito);
        System.out.println("Vueltas totales: " + vueltasTotales);
        System.out.println("Tiempo vuelta rápida: " + tiempoVueltaRapida + " min");
        System.out.println("Carreras en la temporada: " + totalCarreras);
        System.out.println("Posición actual del piloto: " + posicionActualPiloto);

    }
}
```

b)

```
public class Biblioteca {
    public static void main(String[] args) {

        String tituloLibro1 = "Cien años de soledad";
        String autorLibro1 = "Gabriel García Márquez";
        int anioPublicacionLibro1 = 1967;
        String tituloLibro2 = "1984";
        String autorLibro2 = "George Orwell";
        int anioPublicacionLibro2 = 1949;
        String nombreBiblioteca = "Biblioteca Central";
        String direccionBiblioteca = "Calle Principal, 123";

        System.out.println("Biblioteca: " + nombreBiblioteca);
        System.out.println("Dirección: " + direccionBiblioteca);
        System.out.println("Libros disponibles:");
        System.out.println("- " + tituloLibro1 + " (Autor: " + autorLibro1 + ", Año: " + anioPublicacionLibro1 + ")");
        System.out.println("- " + tituloLibro2 + " (Autor: " + autorLibro2 + ", Año: " + anioPublicacionLibro2 + ")");

    }
}
```

c)

```

public class Torneo {
    public static void main(String[] args) {
        String nombreTorneo = "Copa Mundial";
        String paisOrganizador = "España";
        int anioTorneo = 2025;
        String estadio = "Estadio Nacional";
        int espectadores = 50000;
        int golesEquipo1 = 2;
        int golesEquipo2 = 0;
        boolean partidoConcluido = true;
        String nombreEquipo1 = "Tigres Rojos";
        String entrenadorEquipo1 = "Carlos Martínez";
        int puntosEquipo1 = 3;
        String nombreEquipo2 = "Águilas Azules";
        String entrenadorEquipo2 = "Ana López";
        int puntosEquipo2 = 0;

        System.out.println("Torneo: " + nombreTorneo + " (" + anioTorneo + ")");
        System.out.println("Partido: " + nombreEquipo1 + " vs. " + nombreEquipo2);
        System.out.println("Goles " + nombreEquipo1 + ": " + golesEquipo1);
        System.out.println("Goles " + nombreEquipo2 + ": " + golesEquipo2);
        System.out.println("Estadio: " + estadio + " (Espectadores: " + espectadores + ")");
    }
}

```

d)

```

public class ZeldaGame {
    public static void main(String[] args) {
        String nombreMundo = "Hyrule";
        int numeroRegiones = 5;
        boolean tieneCalabozos = true;

        String nombrePersonaje = "Link";
        int salud = 100;
        String armaPrincipal = "Espada Maestra";
        String nombreEnemigo = "Goblin";
        int saludEnemigo = 50;
        String armaEnemigo = "Porra";
        String nombreObjeto = "Frasco de poción";
        String tipoObjeto = "Curación";
        int cantidadEfecto = 50;

        System.out.println("Mundo: " + nombreMundo + " (Regiones: " + numeroRegiones + ")");
        System.out.println("Personaje: " + nombrePersonaje + " (Salud: " + salud + ", Arma: " + armaPrincipal + ")");
        System.out.println("Enemigo: " + nombreEnemigo + " (Salud: " + saludEnemigo + ", Arma: " + armaEnemigo + ")");
        System.out.println("Objeto: " + nombreObjeto + " (Tipo: " + tipoObjeto + ", Efecto: +" + cantidadEfecto + " salud)");
    }
}

```

2. Para hacer un programa de visualización 🌌 del Sistema Solar, primero necesitas datos como, nombres de los planetas, tamaño en kilómetros de diámetro, distancia al sol en millones de

kilómetros. Para almacenar estos datos en tu programa necesitarás clases, y objetos. Inicializa los datos de cada objeto usando un método constructor. Por último, imprime los datos de cada objeto.

No te olvides del Sol ☀️.

3. Para implementar un juego de cartas sencillo, necesitarás almacenar los datos de cada carta en tu programa. En el juego hay dos tipos de cartas: cartas de poder y cartas de personaje. Cada clase de carta tiene unos atributos:

Cartas de poder:

- Nombre.
- Tipo de carta (por ejemplo, ataque, defensa, hechizo).
- Puntos de poder (número de puntos que otorga al jugador cuando se juega).

Cartas de personaje:

- Nombre del personaje.
- Salud inicial.
- Habilidad especial (una descripción breve de la habilidad).

Crea las clases con los atributos necesarios para almacenar las cartas, y añade un método llamado `print()` en cada clase para mostrar una carta de dicha clase.

En el main, crea al menos 3 cartas de poder y 2 de personajes en tu programa. Inicialízalas con un constructor. Utiliza el método `print()` de cada carta para mostrarlas. El programa debe mostrar algo parecido a esto:

```
Poderes:
- Nombre: Fuego Ardiente, Tipo: Hechizo, Puntos de poder: 30
- Nombre: Escudo Protector, Tipo: Defensa, Puntos de poder: 10
- Nombre: Rayo Relampagueante, Tipo: Ataque, Puntos de poder: 40

Personajes:
- Nombre: Guerrero, Salud inicial: 100, Habilidad especial: Ataque fuerte
- Nombre: Mago, Salud inicial: 80, Habilidad especial: Invocar criaturas
```

4. Crea una clase `Alumno` con los campos `nombre` y `nota`, y añade un *constructor*, *getters* y *setters*. Utiliza esta clase desde el main, creando al menos dos objetos.

5. Define las siguientes clases:

a)

```
public class Main {
```

```

public static void main(String[] args) {
    CompteCorrent cc = new CompteCorrent();

    cc.nomPropietari = "Jeff Bezos";
    cc.saldo = 999.9f;
    cc.bloquejada = false;

    System.out.println(cc.nomPropietari);
    System.out.println(cc.saldo);
    System.out.println(cc.bloquejada);
}
}

```

b)

```

public class Main {

    public static void main(String[] args) {
        Book book = new Book();

        book.title = "Through the looking glass";
        book.yearOfPublishing = 1871;
        book.isAvailable = true;

        System.out.println(book.title);
        System.out.println(book.yearOfPublishing);
        System.out.println(book.isAvailable);
    }
}

```

c)

```

Producte producte = new Producte("Corsair Vengeance DDR5", "6000MHz 32GB 2x16GB CL30", 109.99f, 43);

System.out.println("Nom: " + producte.nom);
System.out.println("Descripcio: " + producte.descripcio);
System.out.println("Preu: " + producte.preu);
System.out.println("Stock: " + producte.stock);

```

d)

```

Song song = new Song("Whole Lotta Love", "Led Zeppelin");

song.setRating(9.8f);
song.setFavorite(true);

System.out.println(song.getInfo()); // Whole Lotta Love - Led Zeppelin - 9.8 - true

```

6. 🎮 En los videojuegos, sobretodo los 2D, es muy importante la geometría. Entre otras muchas cosas es nos suele importar el tamaño y la posición de los elementos. Habitualmente, el tamaño se define como un rectángulo, y la posición como un punto en el eje de coordenadas. Por supuesto, en un videojuego necesitaremos manipular los objetos, redimensionándolos y desplazándolos. Además deberemos tener en cuentas las interacciones entre ellos (colisiones, distancia entre ellos).

Crea dos clases que sirvan para manejar estos aspectos. Añade los métodos que consideres necesarios.

Utiliza la siguiente clase **Lienzo** para dibujar los objetos y moverlos al pulsar teclas:

1. Copia y pega esta clase **Lienzo** en tu programa:

```
class Lienzo extends javax.swing.JPanel {
    class R {
        int x,y,w,h;
        java.awt.Color color;

        public R(int x, int y, int w, int h, java.awt.Color color) {
            this.x = x;
            this.y = y;
            this.w = w;
            this.h = h;
            this.color = color;
        }
    }

    interface KeyPressed {
        void Keypressed(int key);
    }
    KeyPressed k;
    private java.util.List<R> rects = new java.util.ArrayList<>();

    public Lienzo() {
        javax.swing.JFrame ventana = new javax.swing.JFrame("Mover Rectángulos con Teclado");
        ventana.setDefaultCloseOperation(javax.swing.JFrame.EXIT_ON_CLOSE);
        ventana.setSize(400, 400);
        ventana.add(this);
        ventana.setVisible(true);
        setFocusable(true);
        addKeyListener(new java.awt.event.KeyListener() {
            @Override
            public void keyTyped(java.awt.event.KeyEvent e) {}

            @Override
            public void keyPressed(java.awt.event.KeyEvent e) {
                k.Keypressed(e.getKeyCode());
                repaint();
            }

            @Override
            public void keyReleased(java.awt.event.KeyEvent e) {}
        });
    }

    public void draw(int x, int y, int base, int altura, java.awt.Color color) {
        rects.add(new R(x,y,base,altura,color));
    }

    @Override
    protected void paintComponent(java.awt.Graphics g) {
        super.paintComponent(g);
        rects.forEach(r -> {
            g.setColor(r.color);
            g.fillRect(r.x, r.y, r.w, r.h);
        });
        rects.clear();
    }

    void onKeyPressed(KeyPressed k) {
        this.k = k;
        k.Keypressed(-1);
    }
}
```

2. Utiliza la clase **Lienzo** de la siguiente forma:

```
public class Main {
    public static void main(String[] args) {

        // crea tus objetos

        Lienzo lienzo = new Lienzo();

        lienzo.onKeyPressed(key -> {
            if (key == java.awt.event.KeyEvent.VK_LEFT) {
                // se ha pulsado la flecha LEFT, mueve un rectangulo a la izquierda
            }
        });
    }
}
```

```

    } else if (key == java.awt.event.KeyEvent.VK_PLUS) {
        // se ha pulsado la tecla '+', haz un rectangulo mas grande
    }
    // detecta la pulsacion de más teclas, y actua en consecuencia

    // en lugar de poner estos valores, pon los valores de tus objetos
    lienzo.draw(150, 150, 100, 60, java.awt.Color.BLUE);
    lienzo.draw(250, 50, 30, 40, java.awt.Color.RED);
    });
}
}

```

7. Para gestionar un inventario necesitamos guardar la información de los productos, y manipular sus cantidades. Crea una clase **Producto** para almacenar el nombre, el precio y la cantidad. Implementa los siguientes métodos:

- Constructor
- **boolean vender(int unidades)**: Reduce la cantidad disponible. Si no hay suficiente cantidad debe retornar `false`.
- **void reponer(int unidades)**: Aumenta la cantidad disponible. Debe asegurarse, de que las unidades a aumentar sean positivas
- **String mostrarInformacion()**: Devuelve un resumen del producto.
- **Setters** para el precio y la cantidad, que aseguren que los valores no son negativos.

8. Crea la clase **Estudiante** con los campos, **nombre** (String) y **notas** (array de floats). Añade los siguientes métodos:

- Constructor: recibe el nombre del estudiante y el número de asignaturas. Inicializa el nombre, y crea el array para almacenar las notas.
- **boolean actualizarNota(float nota, int numeroDeAsignatura)**: actualiza la nota de la asignatura indicada. Si la nota no está entre 0 y 10, o el `numeroDeAsignatura` no está en los límites del array de notas, devuelve `false`;
- **String obtenerDetalles()**: devuelve el nombre, las notas y la nota media.

Producto: Crea la clase Producto con los campos nombre (String) y precios (array de doubles). Añade métodos para actualizar un precio específico y obtener los detalles del producto con el precio medio.

Libro: Define la clase Libro con los campos título (String) y calificaciones (array de enteros). Implementa métodos para actualizar calificaciones y obtener detalles del libro con su calificación promedio.

Empleado: Diseña la clase Empleado con nombre (String) y salarios (array de doubles). Crea métodos para actualizar un salario específico y obtener un resumen del empleado con el salario promedio.

Película: Implementa la clase Película con título (String) y puntuaciones (array de floats). Métodos: actualizar puntuación y obtener detalles de la película con su puntuación media.

Equipo: Crea la clase Equipo con nombre (String) y puntuaciones (array de enteros). Añade métodos para actualizar una puntuación y obtener información del equipo con su puntuación promedio.

Curso: Desarrolla la clase Curso con nombre (String) y duraciones (array de enteros). Métodos: actualizar duración de un módulo y obtener información del curso con la duración total.

Vehículo: Define la clase Vehículo con modelo (String) y consumos (array de doubles). Métodos para actualizar un consumo específico y obtener información del vehículo con el consumo medio.

Jugador: Crea la clase Jugador con nombre (String) y estadísticas (array de floats). Métodos: actualizar estadísticas y obtener detalles del jugador con su rendimiento promedio.

Paciente: Implementa la clase Paciente con nombre (String) y temperaturas (array de doubles). Métodos para actualizar una temperatura y obtener un historial con la temperatura promedio.

CuentaBancaria: Diseña la clase CuentaBancaria con titular (String) y movimientos (array de doubles). Métodos para actualizar un movimiento y obtener el saldo promedio.

Ciudad: Define la clase Ciudad con nombre (String) y temperaturas (array de floats). Métodos para actualizar temperaturas y obtener detalles con la temperatura promedio.

Sensor: Crea la clase Sensor con id (String) y lecturas (array de doubles). Métodos para actualizar una lectura y obtener el promedio de todas.

Concierto: Diseña la clase Concierto con banda (String) y ventas (array de enteros). Métodos para actualizar una venta y obtener el total de ingresos promedio.

Hotel: Implementa la clase Hotel con nombre (String) y ocupaciones (array de enteros). Métodos para actualizar la ocupación y obtener la ocupación promedio.

Receta: Crea la clase Receta con nombre (String) y calorías (array de enteros). Métodos para actualizar una cantidad de calorías y obtener el promedio.

Aeropuerto: Diseña la clase Aeropuerto con nombre (String) y vuelos diarios (array de enteros). Métodos para actualizar vuelos y obtener el promedio diario.

Viaje: Implementa la clase Viaje con destino (String) y distancias (array de doubles). Métodos para actualizar una distancia y obtener la distancia promedio.

Parque: Crea la clase Parque con nombre (String) y visitantes (array de enteros). Métodos para actualizar visitantes y obtener el promedio diario.

Dispositivo: Diseña la clase Dispositivo con modelo (String) y consumos de energía (array de floats). Métodos para actualizar consumo y obtener consumo promedio.

Restaurante: Implementa la clase Restaurante con nombre (String) y valoraciones (array de doubles). Métodos para actualizar una valoración y obtener la media de valoraciones.

Atleta: Crea la clase Atleta con nombre (String) y tiempos de carrera (array de floats). Métodos para actualizar tiempos y obtener el tiempo promedio.

Clima: Diseña la clase Clima con ciudad (String) y precipitaciones (array de floats). Métodos para actualizar precipitaciones y obtener el promedio mensual.

Casa: Implementa la clase Casa con dirección (String) y consumos de agua (array de doubles). Métodos para actualizar consumos y obtener el promedio.

Proyecto: Crea la clase Proyecto con nombre (String) y horas trabajadas (array de enteros). Métodos para actualizar horas y obtener el promedio de horas.

Zoológico: Diseña la clase Zoológico con nombre (String) y poblaciones de especies (array de enteros). Métodos para actualizar la población y obtener el promedio.

9. Implementa la clase **Vehiculo** para un juego de carreras.

Campos:


- **velocidad**
- **numeroDeMarchas**
- **marchaActual**

Métodos:

- Constructor: `numeroDeMarchas`
 - `void acelerar(int incremento)`
 - `void frenar(int decremento)`
 - `void subirMarcha()`: debe asegurar que no se pasa del número de marchas del vehículo
 - `void bajarMarcha()`: debe asegurar que no se pasa de -1 (marcha atrás)
 - `String obtenerEstado()`: devuelve la velocidad y la marcha actual.
-

10. Cuenta bancaria. Campos: titular y saldo. Métodos: `depositar()`, `retirar()` y `obtenerSaldo()`.

array de objetos

11.  **BiblioApp**: Array de libros. Libro: `titulo`, `autor`, `disponible`, `prestar()`, `devolver()`, `getInfo()`

Haz un programa principal funcional completando el siguiente código:

```
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        while(true) {

            System.out.println("""
            ✧🌟🌟🌟☆BiblioApp☆🌟🌟🌟✧
            ①✚ Añadir Libro
            ②📖 Ver libros
            ③🗑️ Eliminar libro
            ④👤 Prestar libro
            ⑤👤 Devolver libro
            ⑥👋 Salir
            """);

            String op = scanner.nextLine();

            switch (op) {
                case "1" -> {
                    System.out.print("Titulo: ");
                    System.out.print("Autor: ");
                }

                case "2" -> {
                    System.out.println("Lista de libros");
                }

                case "3" -> {
                    System.out.print("Número del libro a eliminar: ");
                }

                case "4" -> {
                    System.out.print("Número del libro a prestar: ");
                }
            }
        }
    }
}
```


Lope de Vega
Federico García Lorca
Miguel de Cervantes
Literatura
¿En qué año comenzó la Segunda Guerra Mundial?
1939
1914
1945
1929
1939
Historia
¿Cuál es el símbolo químico del oro?
Au
Ag
Fe
O
Au
Ciencia
¿Qué país ganó la Copa del Mundo de la FIFA en 2018?
Francia
Brasil
Alemania
Argentina
Francia
Deportes
¿Cuál es la capital de Australia?
Canberra
Sídney
Melbourne
Brisbane
Canberra
Geografía
¿Quién pintó la Capilla Sixtina?
Miguel Ángel
Leonardo da Vinci
Rafael
Donatello
Miguel Ángel
Arte

static

14. 📁 Crea una clase `Producto` con:

- Los campos: `descripción`, `precio`, `descuento` e `IVA`.

La descripción, el precio y el descuento son particulares para cada producto, aunque por defecto, el descuento es del 15%.

El IVA es comun a todos los productos, de forma que **cuando se cambia el IVA, se ha de cambiar para todos.**

- Constructor
- Setters/getters para los cuatro campos.
- Los métodos `float getPrecioConDescuento()`, `float getPrecioMasIva()` y `float getPrecioConDescuentoMasIva()`
- Un método `String toString()` que devuelva toda la información de un producto: descripcion, precio, descuento, precio con descuento, IVA, precio mas IVA y precio con descuento mas IVA.

Valora si algún metodo/campo debe ser static.

En el método main, crea varios productos y imprime su información, luego cambia el IVA y vuelve a imprimir su información.

15. 🧑 Crea una clase `Usuario` con:

- Campos: `nombreUsuario`, `sesionesActivas`, y `sesionesTotales`
`sesionesActivas` se refiere a las sesiones que tiene abiertas cada usuario.
`sesionesTotales` cuenta el total de sesiones activas de todos los usuarios.
- Métodos: `iniciarSesion()` y `cerrarSesion()`, que incrementan o decrementan tanto `sesionesActivas` como `sesionesTotales`.
- `obtenerSesionesTotales()` para consultar el número total de sesiones activas.

En el main, simula el inicio y cierre de sesiones para varios usuarios y muestra el total de sesiones activas.

16. 🎓 Crea una clase `Alumno` con:

- Atributos de instancia: `nombre`, `apellido1`, `apellido2`, y `email`.
- Atributo estático: `dominio` (por defecto: `@alu.edu.gva.es`)
- Un método **estático** `generarEmail` que devuelva un email formado por: primeras tres letras del nombre + tres letras del apellido + tres letras del segundo apellido + un número aleatorio del 100 al 999 + `dominio`.
 Todo en minúsculas.
- Un método **estático** `generarNumeroAleatorio(int min, int max)` que devuelva un número aleatorio en ese rango.
- Un método `toString()` que retorne la info del alumno.

En el main, crea varios usuarios e imprímelos. Ejemplo:

```
public class Main {
    public static void main(String[] args) {
```

```
Alumno alumno = new Alumno("Gerard", "Falco", "Perez");
System.out.println(alumno); // Gerard : Falco : Perez : gerfalper616@alu.edu.gva.es

}
```

17. 🏦 Crea una clase **CuentaBancaria** para:

- ☐ Gestionar las cuentas bancarias de los clientes.
- ☐ Llevar un **registro global del total de dinero** en el banco.
- ☐ Asignar automáticamente un **número de cuenta único** a cada nueva cuenta creada.

Atributos de instancia:

- **titular**: el nombre del titular de la cuenta.
- **saldo**: el saldo de la cuenta.
- **numeroCuenta**: el número único de la cuenta asignado automáticamente.

Atributos estáticos:

- **contadorCuentas**: un contador que se incrementa con cada nueva cuenta, y se utiliza para generar números de cuenta únicos.
- **totalDineroEnBanco**: el total de dinero que existe en el banco considerando todas las cuentas.

Constructor:

- Debe aceptar **titular** y un depósito inicial como parámetros.
- Asignar automáticamente un número de cuenta usando **contadorCuentas**.
- Incrementar **totalDineroEnBanco** con el depósito inicial.

Métodos de instancia:

- **depositar(double cantidad)**: suma la cantidad al saldo de la cuenta y actualiza **totalDineroEnBanco**.
- **retirar(double cantidad)**: resta la cantidad del saldo (si hay suficiente saldo) y actualiza **totalDineroEnBanco**.
- **mostrarInformacion()**: muestra el titular, número de cuenta y saldo actual.

Métodos estáticos:

- **obtenerTotalDineroEnBanco()**: devuelve el total de dinero en el banco.
- **obtenerNumeroTotalCuentas()**: devuelve el número total de cuentas creadas.

Haz un main para probar que la clase funciona correctamente.

18. 🏊 Crea una clase **Deportista** con:

Atributos de instancia:

- **nombre**: el nombre del deportista.
- **deporte**: el deporte que practica.
- **puntos**: los puntos acumulados en su carrera.

Atributo estático:

- **puntosTotales**: el total de puntos acumulados por todos los deportistas juntos.

Métodos:

- **sumarPuntos(int cantidad)**: aumenta los puntos del deportista y actualiza los puntosTotales.
- **obtenerPuntosTotales()**: devuelve el total de puntos de todos los deportistas.
- **toString()**: muestra el nombre, el deporte y los puntos actuales.

En el **main**, crea varios deportistas, suma puntos y muestra los puntos individuales y totales.

19. **ID** Crea una clase **Producto** con:

- **Atributos de instancia**: **nombre**, **categoria**, **codigoProducto**.
- **Atributo estático**: **codigoBase** (valor inicial: 1000).
- **Un método estático generarCodigoProducto** que:
 - Genere el código de producto con el formato:
 - Inicial de la categoría + tres primeras letras del nombre + códigoBase.
 - Todo en mayúsculas.
 - Incremente **codigoBase** en 1 tras cada generación.
- **Constructor**: **Producto(String nombre, String categoria)**, que asigne **codigoProducto** llamando a **generarCodigoProducto**.
- **Un método toString()** que retorne la info del producto.

En el **main**, crea varios productos e imprímelos.

¿Cómo afectaría a los viejos y nuevos productos un cambio en el campo **codigoBase**?

variables primitivas y de referencia

20. Qué imprimen los siguientes programas?

a)

```
class ArrayUtils {
    static void crearArray(int[] array) {
        array = new int[]{1,2,3,4};
    }
}
```

```

static String toString(int[] array){
    String result = "";
    for (int i = 0; i < array.length; i++) {
        result += array[i];
    }
    return result;
}

public class Main {
    public static void main(String[] args) {

        int[] array = new int[]{6,7,8,9};

        ArrayUtils.crearArray(array);

        System.out.println(ArrayUtils.toString(array));
    }
}

```

b)

```

class Alumno {
    String nombre;

    Alumno(String nombre) {
        this.nombre = nombre;
    }

    static Alumno crearAlumno(String nombre){
        return new Alumno(nombre);
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno alumno = Alumno.crearAlumno("Juan");

        System.out.println(alumno.nombre);
    }
}

```

c)

```

class Alumno {
    String nombre;
    String id;
    static int lastId = 0;

    Alumno(String nombre) {
        this.nombre = nombre;
        asignarId(this);
    }

    static void asignarId(Alumno alumno){
        alumno.id = "alu" + lastId++;
    }
}

public class Main {
    public static void main(String[] args) {

```

```

        Alumno a = new Alumno("Juan");
        Alumno b = new Alumno("Lola");

        System.out.println(a.id);
        System.out.println(b.id);
    }
}

```

d)

```

class Alumno {
    String nombre;

    static void asignarNombre(Alumno a){
        a = new Alumno();
        a.nombre = "Juan";
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno alumno = new Alumno();
        Alumno.asignarNombre(alumno);

        System.out.println(alumno.nombre);
    }
}

```

e)

```

class Alumno {
    String nombre;

    public Alumno(String nombre) {
        this.nombre = nombre;
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno a = new Alumno("Lola");

        Alumno b = a;

        a.nombre = "Pepa";

        System.out.println(b.nombre);
    }
}

```

f)

```

class Alumno {
    String nombre;

    public Alumno(String nombre) {
        this.nombre = nombre;
    }

    static void cambiarNombre(Alumno alumno){

```

```

        alumno.nombre = "Pepa";
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno a = new Alumno("Lola");

        Alumno b = a;

        Alumno.cambiarNombre(b);

        System.out.println(b.nombre);
    }
}

```

g)

```

class Alumno {
    String nombre;

    public Alumno(String nombre) {
        this.nombre = nombre;
    }

    static void cambiarNombre(String nombre){
        nombre = "Pepa";
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno a = new Alumno("Lola");

        Alumno b = a;

        Alumno.cambiarNombre(b.nombre);

        System.out.println(b.nombre);
    }
}

```

h)

```

class Alumno {
    String nombre;

    public Alumno(String nombre) {
        this.nombre = nombre;
    }

    static void cambiarNombre(String nombre, String nuevoNombre){
        nombre = nuevoNombre;
    }
}

public class Main {
    public static void main(String[] args) {
        Alumno a = new Alumno("Lola");

        Alumno b = a;
    }
}

```



```

        Alumno.cambiarNombre(b.nombre, "Pepa");

        System.out.println(b.nombre);
    }
}

```

21. Implementa los siguientes métodos:

- `intercambiar(int a, int b)`: recibe dos enteros e intercambia sus valores:

```

public class Main {
    public static void main(String[] args) {
        int x = 5, y = 10;
        intercambiar(x, y);
        System.out.println("x: " + x + ", y: " + y);    // x: 10, y: 5
    }
}

```

- `modificarCaja(Caja caja)`: recibe un objeto `Caja` y modifica su `contenido`:

```

public class Main {
    public static void main(String[] args) {
        Caja caja = new Caja("Manzanas");
        modificarContenido(caja);
        System.out.println(caja.contenido);    // Peras
    }
}

```

- `intercambiarArrays(Entero a, Entero b)`: recibe dos objetos `Entero` e intercambia su `valor`:

```

public class Main {
    public static void main(String[] args) {
        Entero x = new Entero(5);
        Entero y = new Entero(10);

        intercambiar(x, y);

        System.out.println("x: " + x.valor + ", y: " + y.valor);    // x: 10, y: 5
    }
}

```

- Crea una clase `Persona` con un atributo `nombre`.
Implementa un método `cambiarPersona(Persona p)` que intente asignarle al parámetro `p` una nueva instancia de `Persona` con otro nombre.
Implementa otro método `cambiarNombre(Persona p)` que solo modifique el atributo `nombre` del parámetro `p`.
Comprueba qué cambios se mantienen después de llamar a los métodos.

```

public class Main {
    public static void main(String[] args) {

```

```

Persona persona = new Persona("Anna");

cambiarPersona(persona);
System.out.println(persona.nombre);
// Debe seguir mostrando "Anna" porque la referencia original no cambia.

cambiarNombre(persona);
System.out.println(persona.nombre);
// Debe mostrar el nuevo nombre porque solo se ha modificado un campo del objeto.
}
}

```

composición

22. 🚗

Crea una clase **Motor** con atributos como **tipo** (String) y **caballos** (int).

Crea una clase **Coche** que tenga un objeto **Motor** como atributo, además de otros como **marca** y **modelo**.

Implementa un método en **Coche** para obtener toda su información, incluida la del motor.

En el **main**, crea un **Coche** y muestra sus datos.

23. 💻

Crea una clase **Pantalla** con atributos **tamaño** y **resolucion**.

Crea una clase **Procesador** con atributos **modelo** y **frecuencia**.

Crea una clase **Ordenador** que tenga un objeto **Pantalla** y un objeto **Procesador**.

En el **main**, crea un **Ordenador**, asigne una pantalla y un procesador, y muestra sus características.

24. 👷

Crea una clase **Empleado** con atributos como **nombre**, **puesto** y **nomina**.

Crea una clase **Empresa** con una **lista** de empleados.

Implementa métodos en **Empresa** para añadir empleados, eliminar empleados y calcular la nómina total.

En el **main**, añada empleados y muestra la nómina total de la empresa.

25. 🧶

Crea las clases necesarias para que estos programas funcionen:

a)

```
System.out.println(personaje.name);           // Eldran
System.out.println(personaje.especie.name);    // Hadron
System.out.println(personaje.especie.color.name); // Gris
System.out.println(personaje.especie.color.code); // #323232
System.out.println(personaje.especie.esMitica); // true
```

b)

```
System.out.println(animal.name);           // Tigre
System.out.println(animal.habitat.name);    // Selva
System.out.println(animal.habitat.clima.name); // Cálido
System.out.println(animal.habitat.clima.temperatura); // 30
System.out.println(animal.habitat.esNatural); // true
```

c)

```
System.out.println(reserva.cliente.name);    // Ana
System.out.println(reserva.hotel.name);       // Grand Palace
System.out.println(reserva.hotel.habitacion.numero); // 203
System.out.println(reserva.hotel.habitacion.tipo); // Doble
```

d)

```
System.out.println(torneo.name);           // Liga Nacional
System.out.println(torneo.equipos[0].name); // Tigres
System.out.println(torneo.equipos[1].name); // Leones
System.out.println(torneo.equipos[0].jugadores[0].name); // Carlos
System.out.println(torneo.equipos[0].jugadores[0].dorsal); // 10
System.out.println(torneo.equipos[0].jugadores[1].name); // Luis
System.out.println(torneo.equipos[0].jugadores[1].dorsal); // 11
System.out.println(torneo.equipos[1].jugadores[0].name); // Miguel
System.out.println(torneo.equipos[1].jugadores[0].dorsal); // 10
```

e)

```
System.out.println(pelicula.name);           // Avengers
System.out.println(pelicula.director.name);  // Joss Whedon
System.out.println(pelicula.director.edad + " años"); // 50 años

System.out.println("Actores:");
for (Actor actor : pelicula.actores) {
    System.out.println(actor.name + " - " + actor.personaje);
}
// Robert Downey Jr. - Iron Man
// Chris Evans - Capitán América
// Scarlett Johansson - Black Widow
```

f)

```

Alumno[] alumnos = {
    new Alumno("Carlos", 15),
    new Alumno("Laura", 14),
    new Alumno("Diego", 16)
};

Clase clase = new Clase("Matemáticas", alumnos);
Escuela escuela = new Escuela("Instituto Central", clase);

System.out.println(escuela.name); // Instituto Central
System.out.println(escuela.clase.name); // Matemáticas
for (Alumno alumno : escuela.clase.alumnos) {
    System.out.println(alumno.name + " - " + alumno.edad + " años");
}
// Carlos - 15 años
// Laura - 14 años
// Diego - 16 años

```

26. Crea las clases necesarias para guardar la siguiente información:

```

{
  "empresa": {
    "nombre": "TechCorp",
    "ubicacion": {
      "pais": "España",
      "ciudad": "Madrid",
      "direccion": {
        "calle": "Gran Vía",
        "numero": 25,
        "oficina": {
          "planta": 3,
          "departamento": {
            "nombre": "Desarrollo",
            "responsable": {
              "nombre": "Carlos Pérez",
              "email": "cperez@techcorp.com"
            }
          }
        }
      }
    }
  }
}

```

JSON