Nota: es todos estos ejercicios la interfaz debe ser lo más sencilla posible, en el sentido de que lo que importa es que el *script* introduzca las entradas indicadas y las salidas, pero sin preocuparse de detalles de presentación.

Jerarquía de clases y polimorfismo

1. CASO DE ESTUDIO.

Este ejercicio tiene como objetivo profundizar en la relación entre:

- **jerarquía de clases**, **sobreescritura de métodos** (method overriding)
- inicialización de objetos usando el **super** (para inicializar correctamente el objeto, teniendo en cuenta que hereda de otro(s) que necesitan ser inicializados)
- **polimorfismo**
- **encapsulación** (ocultación)
- datos `static` y ejemplo de su uso para contar instancias creadas de una clase

Paso 1: estudia detenidamente la **jerarquía de clases** que aparece en el `DWEC-Core-JS-Boletin-11\E1` (en ejemplos de código), en el fichero `clases.js`.

Paso 2: ejecuta en modo depuración el código de `script.js` (que hace uso de las clases de `clases.js`) tratando de entender qué hace cada elemento del código.

2. Crea un fichero `clases.js` JavaScript que sea una plantilla resumen de los elementos que pueden aparecer dentro de una declaración de clase en JavaScript, que te permita recordar fácilmente qué elementos PUEDEN aparecer en la declaración de una clase.

3. Partiendo de las ideas del CASO DE ESTUDIO del ejercicio 1, crea una aplicación en JavaScript que permita representar trabajadores de una empresa (por ejemplo, gerente/s, técnicos, comerciales, dependientes, limpieza, ..., lo que tu veas).

1) Crea un fichero `clases.js` con una jerarquía de clases que:

- modele los atributos principales de cada trabajador (nombre, apellidos, edad, sueldo, antiguedad en empresa, ... lo que veas)
- métodos setter, getter para modificar los atributos (utiliza ocultación de datos para los atributos)
- métodos accessor setter y getter

2) En otro fichero `script.js` crearás un array de elementos y los procesarás polimórficamente.

CONSEJO: empieza de menos a más; es decir, crea la jerarquía más sencilla de clases básicamente con un constructor y un método miembro, luego creas un `script.js` que pruebe ese código y luego vas añadiendo gradualmente funcionalidad a tus clases codificando nuevos métodos.

CONSEJO: NUNCA INTENTES HACER TODO DE GOLPE BIEN A LA PRIMERA: EMPIEZA CON UNA "MAQUETA" (ESQUELETO DE JERARQUÍA DE CLASES Y CÓDIGO DE PRUEBAS) Y VETE AÑADIENDO FUNCIONALIDAD, PROBANDO (DEPURANDO) CADA MEJORA.