

public class Animal

private int id;
public string nombre;
protected int edad;

public Animal (int id, string nombre,
int edad)

{
this.id = id;
this.nombre = nombre;
this.edad = edad;

}

public int getId() {
return this.id;

}

public string GetNombre() {
return this.nombre;

}

public string edad () {
return this.edad;

}

public void setId (int nreid) {
this.id = nreid;

g

public void setNombre (string nreco
nomb)
this.nombre = nreco;

g

```
public void setEdad (int edad)
{
    this.edad = edad;
}
```

}

```
public void MostrarDatos()
{
    console.WriteLine ("id " + id + "Nombre " +
        nombre + "edad " + edad);
}
```

}

```
public void comer (String comida)
```

```
{
    console.WriteLine ("El animal " + nombre + " come " +
        comida);
}
```

}

```
public int Sumar (int edad1 + int edad2)
```

```

int suma = m.edad1 + edad2;
return suma;
```

}

```
public String toJSON()
```

{

```
return "{" + "id " + id + ", nombre: " + nombre +
    "edad: " + edad + "}"
```

}

}

public abstract class Persona
{
 public int id;
 public int edad;
 public bool genero;

public Persona (int id, int edad, bool genero)

{
 this.id = id;
 this.edad = edad;
 this.genero = genero;

} public int GetId () {
 return this.id; }

public int GetEdad () {
 return this.edad; }

}

public bool GetGenero ()

{

return this.genero;

}

public void SetId (int ncid) {
 this.id = ncid;

}

public void SetEdad (int ncedad) {
 this.edad = ncedad;

}

public void MostrarDatos () {
 console . writeLine ("Id: " + id + "edad: " + edad
 + Genero + "Genero? " "Masculino" "Femenino")

Public int Resta (int a int b)

{
 int resultado = a - b;
 return resultado;

5

Public String toString()

{
 return "id " + id + "edad: " + edad + "genero:
 genero" + genero + "f";

}

5

//creamos la clase persona
Public class Samuel : Persona, Persona constructor
{
 Public Samuel (int id, int edad, bool
 genero) : base (id, genero, edad);
};

5

Public class Persona : Persona

{
 Public Persona (int id, int edad, bool
 genero) : base (id, genero, edad);
};

Public class Maria : Persona

{
 Public Maria (int id, int edad, bool
 genero) : base (id, edad, genero);
};

```
public class Karol : persona  
& public Karol (int id, int edad, bool  
genero) : base (id, edad,  
genero) { }
```

```
class Main
```

```
& static void ...
```

```
& Animal leon = new Animal (2, "Pedro")  
Console.WriteLine ("El leon");  
leon. MostrarDatos ();
```

```
- Animal perro = new Animal (3, "David");  
perro. come ("Purina");
```

```
Console. WriteLine ("JSON de Animal");  
Console. WriteLine (leon. ToJSON));
```

```
Console. WriteLine ("Mostrar")
```

```
Calculadora cancion = new Calculadora  
(true, true, 2);
```

```
Calculadora canci = Calculadora (false,  
false);
```

```
cancion. MostrarDatos();  
canci. MostrarDatos();
```

Console.WriteLine ("Multiplicación + calculadora
resultado (60.2);

Console.WriteLine (compraToJson ());

Samuel Samuel = new Samuel (1, 22, true)
Samuel.mostrarDatos();

Console.WriteLine ("ID desde acceso" +
Samuel.id);

Mariola mariola = new Mariola (7, 11, false);
Console.WriteLine (mariolaToJson ());

Karol karol = new Karol (3, 33, false);
Console.WriteLine (karolToJson ());

Jean jean = new Jean (4, 22, true);
Console.WriteLine (jeanToJson ());

Pacora pacora = new Pacora (5, 12, false);
Console.WriteLine (pacoraToJson ());

T
S

class Animal {

#id;
race;
color;
age;
genero;
name;

// este es privado

(JS)

(Constr) Constructor C(id, name, race, color, age, gender) {

this.#id = id;
this.name = name;
this.race = race;
this.color = color;
this.age = age;
this.genero = genero;

5

mostrar datos c.d

console.log C'Animal \${this.name},
Raza: \${this.race},
Color: \${this.color},
ID: \${this.#id},
Edad: \${this.age},
Genero: \${this.genero});

5

AlimentoAnimal (comida) {

console.log C'\${this.name} esta comiendo
\${this.comida};');

5

toJSON() {

```
    return {  
        id: this.id, name: this.name, race: this.  
        race, color: this.color, age: this.age,  
        gender: this.gender  
    };
```

get id() {

```
    return this.id;
```

}

get name() { return this.name; }

get color() { return this.color; }

get race() { return this.race; }

get age() { return this.age; }

get gender() { return this.gender; }

set id(newid) { if (newid) { this.id = newid; }

}

}

Set name = (newname) {

this.name = newname; }

set race (newrace) { this.race = newrace; }

}

Set color (newcolor) { this.color = newcolor; }

}

set

age (newage) { this.age = newage; }

g

g

set gender (newgender) { this.gender = newgender; }

g

g

```
const get = new Animal C1, "Gato", "Gato"  
gat. mostrarDatos C1; "Gris", 3, "marrón";  
gat. alimentarAnimal C1, "nada");
```

```
console.log C JSON.stringify (gat));  
class Estudiante;
```

```
# id;  
- nombre;  
- apellido;  
- edad;  
- genero;
```

Constructor C id, nombre, apellido, edad, genero;

```
this. #id = id;  
this. - apellido = apellido;  
this. - nombre = nombre;  
this. - edad = edad;  
this. - genero = genero;
```

8

mostrarDatos C1;

```
console.log C'Persona ${this.id}, ${this.  
apellido}, ${this.apellido},  
${this.nombre}, ${this.edad},  
${this.genero});
```

8

Deberme (Darme) &

```
console.log C' ${this.nombre}, María d
```

3) Sobreme S,);

S

toJSON() {

```
return {  
    id: this.#id,  
    nombre: this._nombre,  
    apellido: this._apellido,  
    edad: this._edad,  
    genero: this._genero}
```

S;

get id() {

```
return this.#id;
```

S

get nombre() {

```
return this.apellido;
```

get apellido() { return this._nombre; }

S

get edad() { return this._edad; }

get genero() {

```
return this._genero;
```

S

Set id (newId) {

```
this.#id = newId;
```

S

Set nombre (newNombre) { this._nombre =

Set apellido (newApellido) { this._apellido =
newApellido; }

S

Set edad (newedad) { this._edad = newedad; }

3 set genero (Crea genero) &
3 this - genero = new genero;

3
3

const Maria = new Persona (1, "juan", "Rodríguez", 17,
"femenino");
console.log (JSON.stringify (Maria));

Maria.mostrarDatos();
Maria.dormir ("su ronrón")

class Hospital {

id;
nombre;
direccion;
cantidad_salas;
tiene_planta;

constructor (id, nombre, direccion, cantidad_salas,
tiene_planta) {

this.# id = id;
this.nombre =
this.cantidad_salas = cantidad_salas;
this.direccion = direccion;
this.tiene_planta = tiene_planta;

}

mostrarDatos () {

console.log ("Hospital, # this.HdS, nombre
this.nombre, cantidad de salas
this.cantidad_salas # direccion # this.direccion")

tiene panel (this. tiene_plantas);
planta (tiene_planta) &

if (tiene_planta) d
return "El " + this.nombre + " es el Hospital
Si tiene planta;

S else S

return &

Id: this.id,
nombre: this.nombre,
direccion: this.direccion,
cantidad_salas: this.cantidad_salas,
tiene_plantas: this.tiene_planta

T;

S
get id() &
return this.id;

S

get nombre () &
return this.nombre;

S
get direccion () &
return this.direccion;

S
get cantidad_salas () &
return this.cantidad_salas;

S
get tiene_planta () &
return this.tiene_planta;

```
set id (NuevoId) & this.necNombre;  
set nombre (NuevoNombre) & this.necNombre;  
set dirección (NuevoDireccion) &  
this.necDireccion;
```

```
set cantidad_sulas (NuevoCantidad) &  
this.necosCantidad;
```

```
set tiene_planta (NuevoPlanta) &  
this.necPlanta;
```

```
;
```

```
const canaima = new Hospital (9,  
canaima, "callao 43 sur # 13 -04",  
39, true);
```

```
const las_palmas = new Hospital  
(6, "cas palma", "correco, se lote 26",  
44, false);
```

```
canaima.mostrar();
```

```
console.log (las_palma.plata (las_palma,  
tiene_planta));
```

```
console.log (JSON.stringify (canaima, tiene_planta));
```

class Instrumento {

id;

costo;

nombre;

electrico;

constructor (id, nombre, electrico, costo) {

this. # id = id;

this. nombre = nombre;

this. electrico = electrico;

g

mostrarInformacion () {

console. log (ID: \${this. # id}, costos:
electrico \${this. electrico}, nombre:
\${this. nombre});

5

electrico (es_electrico)

IFC es_electrico)

return \${this. nombre};

Selse {

return \${this. nombre}.

T

g

return this

id: this. id

costo: this.costo

nombre: this.nombre

electrico: this.electrico

};
};

get id() { return this.id; }

get costo() { return this.costo; }

get nombre() { return this.nombre; }

get electrico() { return this.electrico; }

Guitarra = new Instrumento(3,

"Guitarra electrica", true, 5000);

const violin = new instrumento

(4, "Violin", false, 3000);

Guitarra.mostrarInformacion();

Violin.mostrarInformacion();

Console.log(Guitarra.electrico);

Console.log(Violin.electrico);

Console.log(JSON.stringify(Guitarra));

```
class Persona { /*
```

```
# id;  
nombre;  
apellido;  
edad;  
genero;
```

Constructor (id, nombre, apellido, edad, genero);

```
this.#id = id;  
this.apellido = apellido;  
this.nombre = nombre;  
this.edad = edad;  
this.genero = genero)
```

3

mostrar datos

```
console.log(CPersona.$(this.#id), $(this.  
apellido), $(this.nombre), $(this.edad),  
$(this.genero));
```

Dicirme (Dicirme) {

```
console.log(` ${this.nombre} Maria duerme.  
mas que ${Dicirme}`);
```

3

toJSON() {
return {

```
id: this.#id,  
nombre: this.nombre  
apellido: this.apellido  
edad: this.edad,  
genero: this.genero
```

```
8
8 get id () {
    return this.id;
}
8 get nombre () {
    return this.nombre;
}
8 get apellido () {
    return this.apellido;
}
8 get edad () {
    return this.edad;
}
8 get genero () {
    return this.genero;
}

9
set id (newId) {
    this.id = newId;
}
set nombre (newNombre) {
    this.nombre = newNombre;
}
set apellido (newApellido) {
    this.apellido = newApellido;
}
set edad (newedad) {
    this.edad = newedad;
}

9
9
const Maria = new Persona(1, 'Maria',
    'Rodriguez', 'F', 'femenina');
```

```
console.log(JSON.stringify(Maria));  
Maria.mostrarDatos();  
Maria.dormir("su paro")
```

(60)

```
type Animal struct {
```

```
    id int  
    nombre string  
    edad int
```

}

```
func NewAnimal (id int, nombre string,  
                edad int) Animal {  
    return Animal {id: id, nombre: nombre,  
                  edad: edad}}
```

```
func (a Animal) GetId () int {  
    return a.id}
```

5

```
func (a Animal) GetNombre () string {  
    return a.nombre}
```

8

```
func (a Animal) GetEdad () int {  
    return a.edad}
```

```
func (a Animal) setId (nuevoId int) {  
    a.id = nuevoId}
```

5

func (Ca. Animal) setedad (creceedad int)
a. edad = creceedad

func (Ca. Animal) mostralnotos () {
}

fmt. printn ("Id", a.id, "Nombre:", a.nombre,
"edad", a.edad)

3

func (a Animal) comer (comida string) {
}

fmt. printn ('C' Animal =)', a.Nombre,
"la gosta comer," comida)

5

func (a Animal) sonar (sonido int, duracion int)
return nombre + nombre

func (a animal) toJSON() string {
}

return fmt. sprintf ("id,edad,nombre")

type calcular struct

calcular bool
electricidad bool
edad int

5

Func RecalcularBool (calculadora bool, id int) * calculadora & electrica bool,
electrica bool, id int)

return & calculadora &

Calculadora: calculacion,
electrica: electrica;
id: id,

§

§

Func Cc * calculadora) Get calculadora () bool &
return C. calculux

§

Func Cc * calculadora) Get electrica () bool &
return C. electrica

§

Func Cc * calculadora) Get id () Int §
return C. id

§

Func Cc * calcula) set calcula (calculadora
calculadora bool) & C. calculadora = calcula
calculadora

§

Func Cc * calculadora) set electrica (
electrica bool) & C. electrica = calcula

§

Func Cc * calculadora) calcula (a int, b int)
int & resultado = a * b
return resultado.

§