



TS clases_01.ts X

typescript > TS clases_01.ts > ...

```
1  // Clase
2  class Vehiculo{
3      marca:string
4      anio:number
5      modelo:string
6      precio:number
7      velocidad:number
8      constructor(_marca:string,_anio:number,_modelo:string,_precio:number) {
9          this.marca = _marca
10         this.anio = _anio
11         this.modelo = _modelo
12         this.precio = _precio
13         this.velocidad = 0
14     }
15     acelerar():void{
16         if (this.velocidad < 200){
17             this.velocidad = this.velocidad + 10
18             console.log(`El vehiculo va a una velocidad de ${this.velocidad} km/hr`)
19         }else{
20             this.velocidad = 200
21             console.log(`El vehiculo va a su velocidad maxima que es ${this.velocidad} km/hr`)
22         }
23     }
24
25     frenar():void{
26         if (this.velocidad > 10){
27             this.velocidad = this.velocidad - 10
28             console.log(`El vehiculo va a una velocidad de ${this.velocidad} km/hr`)
29         }else{
30             this.velocidad = 0
31             console.log(`El vehiculo se encuentra parado su velocidad es ${this.velocidad} km/hr`)
32         }
33     }
34
35 }
```

TS clases_01.ts X

typescript > TS clases_01.ts > ...

```
36
37  const auto_01 = new Vehiculo("Ford",2023,"Lobo",1300000)
38  auto_01.acelerar();
39  auto_01.acelerar();
40  auto_01.acelerar();
41  auto_01.acelerar();
42
43  auto_01.frenar();
44  auto_01.frenar();
45  auto_01.frenar();
46  auto_01.frenar();
47  auto_01.frenar();
48  auto_01.frenar();
49
```



UANL

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN



FIME

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

TS clases_02.ts X

typescript > TS clases_02.ts > ...

```
1 // Herencia
2 // Hereda propiedades y metodos de clase Padre
3 class Padre{
4     nombre:string
5     edad:number
6     constructor(_nombre:string, _edad:number){
7         this.nombre=_nombre
8         this.edad=_edad
9     }
10    mostrarNombre():void{
11        console.log(`el nombre es: ${this.nombre}`)
12    }
13 }
14
15 class Hijo extends Padre{
16     apellido:string
17     constructor(_nombre:string, _edad:number, _apellido:string){
18         // super ejecuta el constructor de la clase padre
19         // que recibe dos parametros de acuerdo al tipo de dato
20         super(_nombre, _edad)
21         this.apellido = _apellido
22     }
23
24     mostrarDatosHijo():void{
25         console.log(`Los datos de la pesona son:`)
26         console.log(`nombre: ${this.nombre}, apellido: ${this.apellido} y edad ${this.edad}`)
27     }
28 }
29
30 const persona = new Hijo("Jose", 25, "Munoz")
31
32 persona.mostrarNombre()
33 persona.mostrarDatosHijo()
```

TS clases_03.ts X

typescript > TS clases_03.ts > ...

```
1 // Modificadores Publicos
2 // Por default las clases, propiedades y metodos son publicos
3 class Animal01{
4     nombre:string
5     patas:number
6     constructor(_nombre:string, _patas:number){
7         this.nombre = _nombre
8         this.patas = _patas
9     }
10    moverse(){
11        console.log(`El ${this.nombre} se esta moviendo`)
12    }
13 }
14
15
16 const animal01 = new Animal01("perro",4)
17 animal01.moverse();
18 animal01.nombre = "gato"
19 animal01.moverse()
```

TS clases_04.ts X

typescript > TS clases_04.ts > ...

```
1 // Modificadores Privados
2 // Por default las clases, propiedades y metodos son publicos
3 class Animal{
4     private nombre:string
5     private patas:number
6     public constructor(_nombre:string, _patas:number){
7         this.nombre = _nombre
8         this.patas = _patas
9     }
10    public moverse(){
11        console.log(`El ${this.nombre} se esta moviendo`)
12    }
13
14 }
15
16 const animal02 = new Animal("cocodrilo",4)
17 animal02.moverse();
18 // No se puede asignar el valor de una propiedad privada
19 //animal02.nombre = "pato"
20 animal02.moverse()
```



UANL

EN NUEVO LEÓN



FIME

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

TS clases_05.ts X

typescript > TS clases_05.ts > ...

```
1  // Modificadores Protegidos
2  // Son accesibles desde la misma clase
3  // 0 accesibles de la clase heredada
4
5  class Padre02{
6      private nombre:string
7      private apellido:string
8      protected edad:number
9      public constructor(_nombre:string, _apellido:string){
10         this.nombre = _nombre
11         this.apellido = _apellido
12     }
13     mostrarDatosPadre(){
14         console.log(`Los datos del hijo son: nombre ${this.nombre}, apellido${this.apellido} `)
15     }
16 }
17 class Hijo02 extends Padre02{
18     constructor(_nombre:string,_apellido:string, _edad){
19         super(_nombre,_apellido)
20         this.edad = _edad
21     }
22     mostrarDatosHijo(){
23         console.log(`La edad es: ${this.edad}`)
24     }
25 }
26
27 const hijo02 = new Hijo02("Juan", "Sin Miedo", 6)
28 hijo02.mostrarDatosPadre();
29 hijo02.mostrarDatosHijo();
30
```

TS clases_06.ts X

typescript > TS clases_06.ts > ...

```
1  // Metodos get y set
2
3  class Persona{
4      private nombre:string
5      private edad:number
6      constructor(_nombre:string, _edad:number){
7          this.nombre = _nombre
8          this.edad = _edad
9      }
10     getNombre():string{
11         return this.nombre
12     }
13     getEdad():number{
14         return this.edad
15     }
16     setNombre(_nombre:string):void{
17         this.nombre = _nombre
18     }
19     setEdad(_edad:number):void{
20         this.edad = _edad
21     }
22     mostrarDatosPersona(){
23         console.log(`Los datos de la persona son: nombre ${this.nombre} y edad ${this.edad}`)
24     }
25
26 }
27
28 let persona01 = new Persona("Jose",25)
29 persona01.mostrarDatosPersona()
30
31 persona01.setNombre("Pepe")
32 persona01.setEdad(26)
33
34 persona01.mostrarDatosPersona()
```


TS clases_07.ts X

typescript > TS clases_07.ts > ...

```
1 // la clase abstracta puede declarar metodos
2 // que se implementan en los hijos
3 // debe tener por lo menos un metodo abstracto
4 abstract class Animal02{
5     abstract ruido():void;
6
7     camina():void{
8         console.log(`El animal esta caminando`)
9     }
10 }
11
12 class gato extends Animal02{
13     ruido(){
14         console.log(`Miau`);
15     }
16 }
17
18 class perro extends Animal02{
19     ruido(){
20         console.log("Guau")
21     }
22 }
23
24 let perro01 = new perro()
25 perro01.ruido()
26
27 let gato01 = new gato()
28 gato01.ruido()
```

TS interfaz_01.ts X

typescript > TS interfaz_01.ts > Persona

```
1 interface Persona{
2     // Atributos
3     nombre:string
4     apellido: string
5 }
6
7 let nueva_persona = {
8     nombre:'Jose',apellido:'Munoz'
9 }
10
11 function caminar (persona:Persona):void {
12     console.log(`la persona ${persona.nombre} ${persona.apellido} esta caminando`)
13 }
14
15
16 caminar(nueva_persona)
17
```

TS interfaz_02.ts X

typescript > TS interfaz_02.ts > • Figura

```
1 interface Figura{
2     figura:string
3     valor_01:number
4     valor_02?:number
5 }
6
7 function MostrarAreaFigura(areafigura:Figura):number{
8     let area:number
9     if(areafigura.valor_02){
10         area = areafigura.valor_01 * areafigura.valor_02
11         return area
12     }
13     else{
14         if(areafigura.figura == "circulo"){
15             area = 3.14 * areafigura.valor_01 * areafigura.valor_01
16             return area
17         }else{
18             area = areafigura.valor_01 * areafigura.valor_01
19             return area
20         }
21     }
22 }
23
24 let figura_01 = {figura:"circulo",valor_01:10}
25 let figura_02 = {figura:"cuadrado",valor_01:10}
26 let figura_03 = {figura:"rectangulo",valor_01:10,valor_02:5}
27 let area:number
28
29 area = MostrarAreaFigura(figura_01)
30 console.log(`la figura ${figura_01.figura} tiene un area de ${area}`)
31
32 area = MostrarAreaFigura(figura_02)
33 console.log(`la figura ${figura_02.figura} tiene un area de ${area}`)
34
35 area = MostrarAreaFigura(figura_03)
36 console.log(`la figura ${figura_03.figura} tiene un area de ${area}`)
37
```