/\*

\*JavaScript Objects:

You define (and create) a JavaScript object with an object literal:

 \*/

*const* car = { type: "tata", model: "hexa", color: "black" };

//type is the key

//tata is value

*const* person = {

  firstName: "Rohit",

  lastName: "Sharma",

  age: 34,

  city: "Mumbai",

};

console.log(person);

/\*

\*Accessing Object Properties:

You can access object properties in two ways:

1]objectName.propertyName

2]objectName["propertyName"]

 \*/

console.log(person.firstName); //dot notation.

console.log(person["lastName"]); //bracket notation.

/\*

 \*JavaScript objects are containers for named values called properties.

 \*/

//  \* Modifying Object Properties:

console.log((person.firstName = "Vamesh"));

console.log((person["lastName"] = "Patel"));

// \* delete keyword

delete person.age;

console.log(person);

console.log(person.height);

console.log("height" in person); // false , \*in operator gives the output in true or false

console.log("city" in person); //true

//For in Loop:

//for in loop used to iterate the objects.

// Description:

// The for...in statements combo iterates (loops) over the properties of an object.

// The code block inside the loop is executed once for each property.

// Note:

// Do not use for...in to iterate an array if the index order is important. Use a for loop instead.

for (*let* key in person) {

  console.log(`${key} : ${person[key]}`);

}

//console.table : display in the table.

console.table(person);

//Nested Object :

*const* person1 = {

  firstName: "Gaurav",

  lastName: "Patil",

  age: 25,

  city: "Pune",

  scores: {

    odi: 100,

    t20: 50,

  },

};

console.table(person1);

//Accessing object properties:

console.log(person1.firstName); //dot notation.

console.log(person1["lastName"]); //bracket notation.

console.log(person1.scores.odi);

console.log(person1.scores.t20);

// Object Methods:

*const* person2 = {

  firstName: "Gaurav",

  lastName: "Patil",

  age: 25,

  city: "Pune",

  sayhello() {

    //

    console.log("hello"); //2] way comment kar dena jab 1 way use karge tab

  }, //

  scores: {

    odi: 100, // nested object

    t20: 50,

  },

};

console.log(person2);

//two ways to create methods in object

//1] way

// person2.sayhello = function () {

//     console.log("Hello");

// }

person2.sayhello();

/\*

What is this?

In JavaScript, the this keyword refers to an object.

Which object depends on how this is being invoked (used or called).

The this keyword refers to different objects depending on how it is used:

In an object method, this refers to the object.

Alone, this refers to the global object.

In a function, this refers to the global object.

In a function, in strict mode, this is undefined.

In an event, this refers to the element that received the event.

Methods like call(), apply(), and bind() can refer this to any object.

Note

this is not a variable. It is a keyword. You cannot change the value of this

 \*/

*const* person3 = {

    firstName: "Gaurav",

    lastName: "Patil",

    age: 25,

    city: "Pune",

    sayhello() {

      //

      console.log(`My name is ${this.firstName}. I have ${car1.color} car.`); //2] way comment kar dena jab 1 way use karge tab

    }, //

    scores: {

      odi: 100, // nested object

      t20: 50,

    },

};

*const* car1 = {

    name: "maruti",

    color:"white"

}

//   console.log(person3);

person3.sayhello();