```
/*
  创建对象使用大括号，大括号中是一系列属性，用逗号隔开。
  属性名：属性表达式；
  属性名如果不是有效变量名或者数字，需要使用引号
  读取不存在的属性，返回undefined；
*/

var rabbit ={
  eat: false,
  "Go home": "nope",
  events:["work","eat carrot","slepping"]
};
console.log(rabbit.eat); //false
console.log(rabbit["Go home"]); //nope
console.log(rabbit.events[0]); //work
console.log(rabbit.go);//undefined

//change properties or add new properties
rabbit.eat = true;
console.log(rabbit.eat); //true (change property)
rabbit.go = "home";
console.log(rabbit.go);//home (add new property)

//delete properties And check propertiess
delete rabbit.eat;
console.log(rabbit.eat); //undefined (deleted)
console.log("eat" in rabbit); // false (deleted)
console.log("go" in rabbit); //true (go properties in rabbit object)

//array to save objects
var journal = [
  {
    events: [1,2,3,4],
    wolf: false
  },
  {
    events:["1","2","3"],
    wolf: true
  }
];

console.log(journal[1].events); //['1','2','3'];
console.log(journal[0].events.indexOf(3)); // 2  find index of the
value, if none, return -1
//compare two objects
var object1 = {n:3};
var object2 = {n:3};
var object3 = object1;
console.log(object1==object2); //false
```

```
console.log(object1==object3); //true
console.log(object2==object3); //false

//learn speical object: array
var array = [];
array.push(3);
array.push(2);
console.log(array); //[3,2] add at the back of an array
array.pop();
console.log(array); // [3] remove back
array.unshift(1);
console.log(array); // add in the front of an array
array.shift(); //remove front
array.lastIndexOf(3); // find index from last
array = [1,2,3,4,5];
console.log(array.slice(2,4)); //[3,4] return value from
index1(included) to index2(excluded)
console.log(array.concat([6,7])); //[1,2,3,4,5,6,7] link two arrays

//string has slice and indexOf method

/*
Constrcutor : create object 构造函数第一个字母大写
new object
调用函数之前添加关键字new则表示调其构造函数，有指针this
！对于构造函数来说（实际上，所有函数）都会自动获得一个名为prototype属性，来自于
Object.prototype空对象
*/
function Rabbit(type){
  this.type = type;
}

Rabbit.prototype.speak = function(line){
  console.log(this.type +" says: "+ line+"!");
}
var blankRabbit = new Rabbit("blank");
blankRabbit.speak("hahhaa...");

//class!!!
//Rabbit02 cannot have the same name as Rabbit.
class Rabbit02{
  constructor(type) {
    this.type = type;
  }
  speak(line) {
    console.log(`The ${this.type} rabbit says '${line}'`);
  }
}
let killerRabbit = new Rabbit02("killer");
```