1. null表示空的引用，虽然null使用typeof判断是是object，但是null不具有任何对象的特性，所以我们不能对其执行null.toString()等对象实例的默认调用。Null并不是以Object为原型创建出来的，所以null instanceof Object是false。
2. 循环语句用法：while、for用法：循环次数不限制、不确定用while；循环次数有限制、已确定用for
3. 原型链
   1. 原型对象（Person.prototype）是构造函数的一个实例。Object为函数，所以是Function的实例
   2. \_\_proto\_\_存在于实例与构造函数的原型对象之间，而不是存在于实例与构造函数之间。
   3. Function.prototype（相当于Function的实例）是function（函数对象）,所以Function.prototype没有prototype
   4. JS 在创建对象（不论是普通对象还是函数对象）的时候，都有一个叫做\_\_proto\_\_ 的内置属性，用于指向创建它的构造函数的原型对象。
   5. 可以创建对象的构造器不仅仅有 Object，也可以是 Array，Date，Function。他们的本质就是一个函数（函数对象）。
   6. Object.prototype对象也有\_\_proto\_\_属性，但它比较特殊，为null。因为null 处于原型链的顶端，
4. var c = 1

function c(c) {

console.log(c)

var c = 3 }

c(2)

该代码运行结果为”c不是一个函数”，因为变量提升函数提升后，执行了c=1赋值语句，所以c不是一个函数无法进行调用。

5、  var obj = {

    fn2: function () {

      console.log(fn2)

    }

  }

  obj.fn2()

该代码运行的结果是报错，因为在函数调用时，由于fn2自己的函数作用域没有fn2所以会自下往上查找全局作用域，而全局作用域也没有fn2，所以报错。

6、闭包：每次外部函数执行的时候,都会开辟一块内存空间,外部函数的地址不同，都会重新创建一个新的地址

function fun(n, o) {

console.log(o)

return {

fun: function (m) {

return fun(m, n)

 }

}

}

var a = fun(0)

a.fun(1)

a.fun(2)

 a.fun(3) //undefined,0,0,0

var b = fun(0).fun(1).fun(2).fun(3) //undefined,0,1,2

var c = fun(0).fun(1)

c.fun(2)

c.fun(3) //undefined,0,1,1

7、JS可以使用中文变量名

8、Unicode字符集中包括颜文字

9、JS原始数据类型：Null/Undefined/Number/Boolean/String/Symbol/Bigint

10、call()、bind()、apply()的用法，改变this的指向，区别在于  
f.call(obj, arg1, arg2...),  
f.bind(obj, arg1, arg2,...)(),  
f.apply(obj, [arg1, arg2, .])