输出语法：

Document.write(‘要输出的内容’)；

向body内输出内容

如果输出的内容写的是标签，也会被解析成网页元素

Alert(‘要输出的内容’);

网页弹出警告对话框

Console.log(‘控制台打印’)

控制台输出语法，程序员调试使用

输入语法：

Promp(‘提示用户输入文字’)

显示一个对话框，对话框中包含一条文字信息，用来提示用户输入汉字

字面量：

‘ ’ 字符串字面量

[ ] 数组字面量

{ } 对象字面量

变量：存储数据的容器

命名规范规则：

规则：

不能用关键字；只能用下划线，字母，数字，$,且不能用数字开头

字母严格区分大小写

规范：

起名要有意义

第一个单词首字母小写，后面每个单词首字母大写。例：userName

数组:按顺序保存多个数据

let arr = [ ]

计算机中的编号是从0开始的。

数据类型：  
 基本数据类型：

number 数字型 所有数字

string 字符串型 单引号(‘’) 双引号(“”) 或反引号(`)包裹的数据都叫字符串

单引号双引号可以互相嵌套，（外单内双，外双内单）

必要时可以使用转义符\输出单双引号

字符串拼接：document.write(‘ ’+’ ’)

document.write(变量1+变量2)

模板字符串：利用反引号``

内容拼接变量时，用${ }包住变量

例：document.wirte(`大家好，我叫${name},今年${age}岁了`)

boolean 布尔型 表示肯定或者否定只有两个值：true false

undefined 未定义型 只声明不赋值

null 空类型 表示已赋值但值为空

引用数据类型：object 对象 function 函数 array 数组

typeof关键字检测数据类型

类型转换

隐式转换：

规则：+号两边只要有一个是字符串，都会把另外一个转换成字符串

除了+以外的算数运算符比如-\*/等都会把数据转换成数字类型

+号作为正号解析可以转换成number比如+1 +2 +3

显式转换：

Number（数据）

转换成数字类型

如果字符串内容有非数字，转换失败

NaN也是number类型的数据，代表非数字

parselnt（数据）

只保留整数

parseFloat（数据）经常用于过滤单位例如console.log(parseFloat(’10.5kg’) ) 输出为10.5

可以保留小数

String(数据)

变量.toString(进制) 转换为任意进制输出

运算符：

+ - \* / %取余数判断某个数字是否被整除

赋值运算符： = 左边必须是一个容器

num+= 1 num变量加一

一元运算符

自增 ++ 让变量的值加一

++num 前置自增

num++ 后置自增

1.前置性递加运算符的意思就是：先运算，再赋值

2.后置型递加运算符的意思就是：先赋值，再运算

自减 -- 让变量的值减一

经常用来计数使用。

比较运算符

>

<

>=

<=

== 左右两边是否相等 只要求值相等，不要求 数据类型

=== 左右两边是否类型和值都相等

!== 左右两边是否不全等

比较结果为boolean类型，只会得到true或false

字符串比较

比较字符对应的ASCII码

从左往右以此比较

如果第一位一样再比较第二位

NaN不等于任何值，包括它本身

尽量不要比较小数，因为小数有精度问题

不同类型之间的比较会发生隐式转换

最终把数据隐式转换成number类型再比较

所以开发中，如果进行准确的比较我们更喜欢===或者!==

分支语句

if分支

if（）{

} 小括号会转换为布尔型 false 0 ‘’ undefined null NaN

多分支if语法：

判断顺次判断。

三元运算符

条件？满足条件执行的代码：不满足条件执行的代码

一般用于取值

switch语句：找到与小括号里数据全等的case值，并执行里面对应的代码

若没有全等===的执行default里的代码

没有break会造成分支穿透即执行后续所有分支

switch（数据）{

case 值1：

代码1

break

case 值2：

代码2

break

default：

代码n

break

}

数组添加新的数据

arr.push(新增内容)

arr.unshift(新增内容)

删除数组中数据

arr.pop()

arr.shift()

arr.splice(操作的下标，删除的个数)