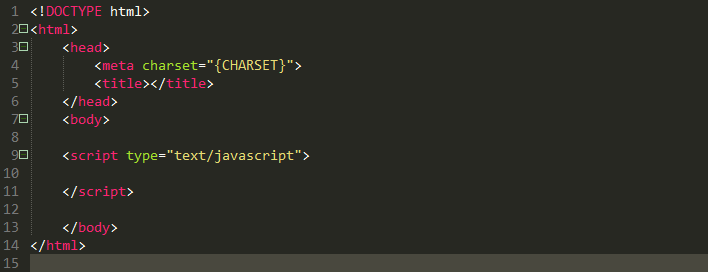
# JS第一天笔记

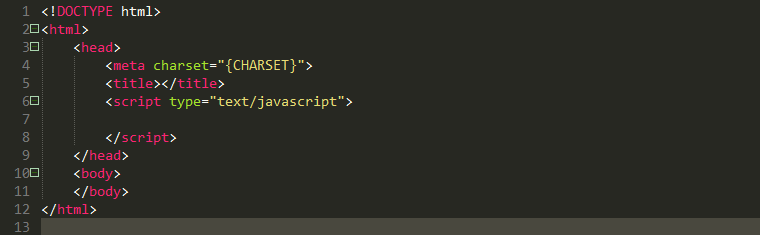
# 一、认识JavaScript

## 1.1 JS书写的位置

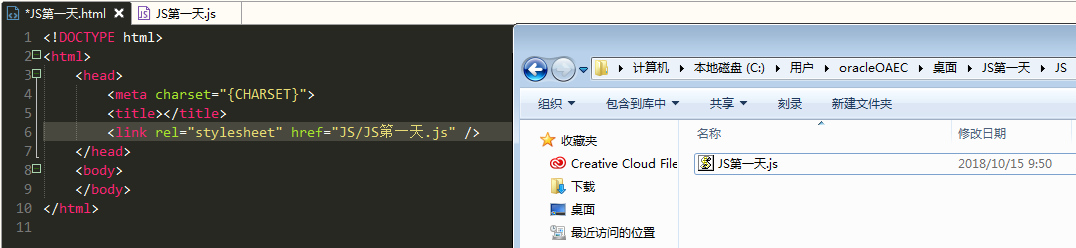
1.1.1 书写在HTML body标签中 （第一种方法是最常见的方法）



1.1.2 书写在HTML head标签中



1..1.3 外链式（在项目制作中常用）



## 1.2 JavaScript基本语句

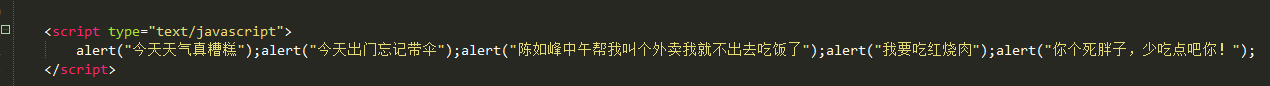
### 1.2.1 alert语句

alert 是警告的意思。它用途就是在网页中弹出一个警告对话框

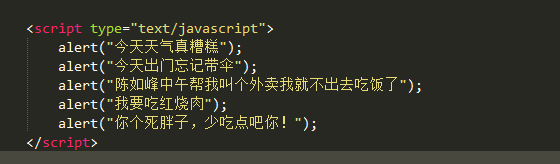


那么，请注意：

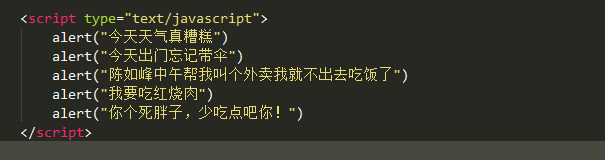
alert语句是可以多行执行的语句，就是可以显示多次弹框，只要每一个语句用分号隔开就可以去执行。



等价于



也等价于

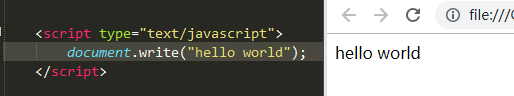


**但是这种书写方式，是违法的！是一种不严谨的程序书写方式，虽然可以运行，但是你不能保证是否会出错！**

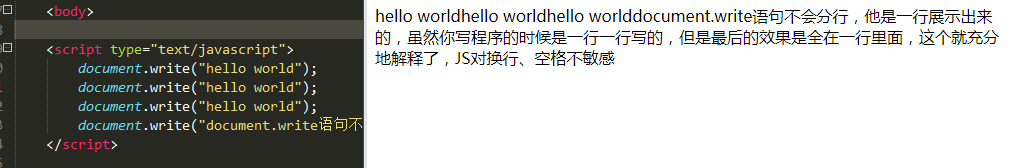
### 1.2.2 页面书写语句 document.write

document 就是文档的意思

write 书写的意思



document.write的书写方式和alert语句差不多，同样可以多行书写，它的注意点和上面也是一样的。



**可以多行书写，但是效果：并排显示！**

### 1.2.3控制台输出语句 console.log()

console 面板 就是控制面板的意思，这是很多前端工程师回去查看的一个工具



**console语句不像alert 他没有弹窗出现**

**也不想document 他在页面中也不会出现**

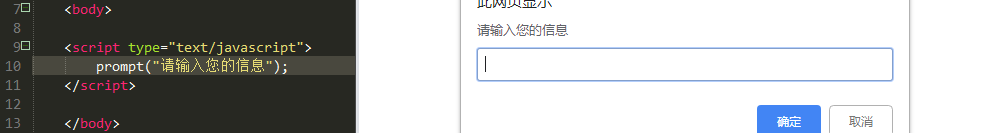
**console 语句只会在控制台输出书写的内容**

书写语法：



### 1.2.4 用户输入语句 prompt

输入框的意思

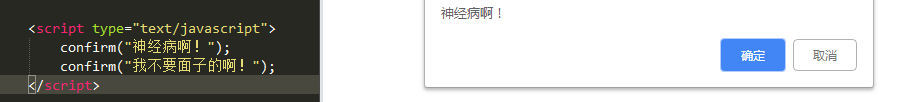


请注意：小括号里面输入的文字是一个提示性的文字。

同样可以做到多行输入，输入完毕一次再运行下一行



### 1.2.5 confirm 在页面中弹出一个对话框 常用来配合if条件判断的



这个语句长的非常像alert 都是弹出的警告框

但是confirm是以后用来配合条件判断

alert也配合条件判断，但是alert是在早期JS当中使用的

### 1.2.6 总结

**语句书写规范：**

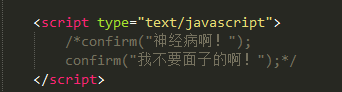
**1、语句中记得把东西写在（）小括号里，里面有引号的代表是字符串、没有引号的代表是数值。（这句话现在不懂没关系）。**

**2、程序是一局一局来执行的。因此记得不能漏下分号！（执行完上一句才执行下一句）。**

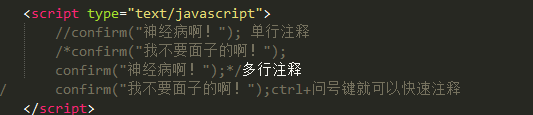
**3、JS对换行、缩进、空格不敏感（这个相当于HTML中的空白折叠现象）**

## 1.3 JS的注释方法

1.3.1单行注释 // 

1.3.2多行注释 /\* \*/

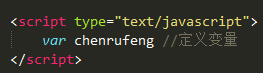
1.3.3注释的快捷键：ctrl+/



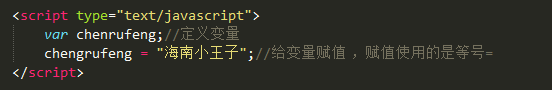
## 1.4 变量

什么叫做变量？会变化的量，变量是用来储存数据的容器。

### 1.4.1 定义变量 var



## 1.4.2 给变量赋值



赋值使用的是等号= 在这里必须要强调，在程序中，我们把“=”叫做赋值运算，“=”它是一个非常特殊的含义

### 1.4.3 定义并赋值



等价于



请注意：var 后面的空格不能省略！ 在定义变量的时候，var后面一定要打上一个空格！

变量的名字是程序员自定义，可以随便取

语法：var+空格+变量名

## 1.5 变量的命名规则

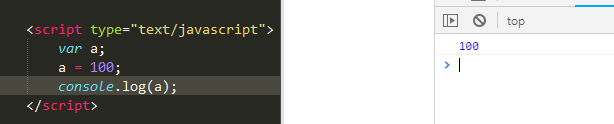
1、只能由英文字母、数字、下划线、美元符号构成

2、不能以数字开头 ~~（var 1ab 这是一个错误的写法）~~

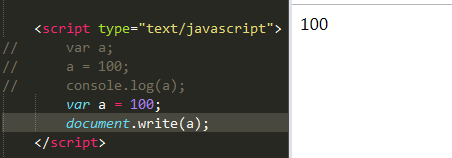
3、严格区分大小写 （A a 这是两个变量）

4、不能是JS里面的保留文字 （谁背谁傻逼）

例子：变量从定义到赋值



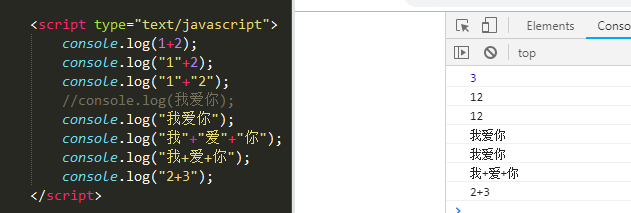
等价



## 1.6 变量的类型

直接量：也被叫为字面量。也就是说你看见什么，他就是什么。所见即所得

简单的直接量有两种：数值（字）、字符串



数值和字符串是非常好辨别的，只要看谁在引号里面，谁就是字符串，只要是在引号里的东西，就像打印机一样，直接输出来

因此 1 就是代表的是数值 数字类型 number（数值型），也就是在JS中，只要是个数字，那么它就是数值型的变量，无论**整浮**、无论大小、无论正负、都是Number类型。

整浮：整就是整数，浮就是小数 术语：浮点数

“1” 就是代表的是字符串 string

空字符串 就是一个空格 

在JS查看变量的类型：typeof



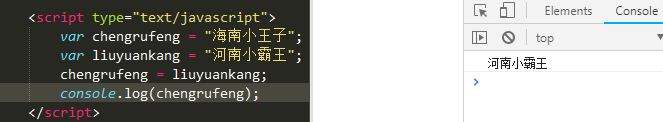
语法：typeof +空格+变量的名字

## 1.7 变量的传递

举个例子：比如现在陈如峰 是典型的海南小王子

胖点的刘远康 也想成为陈少般的偏偏少男

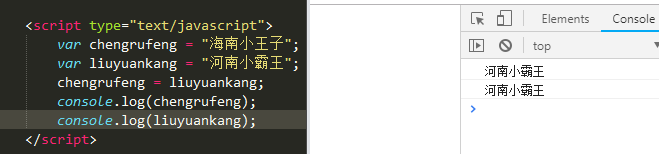
这个时候 我们就需要变量之间的传递



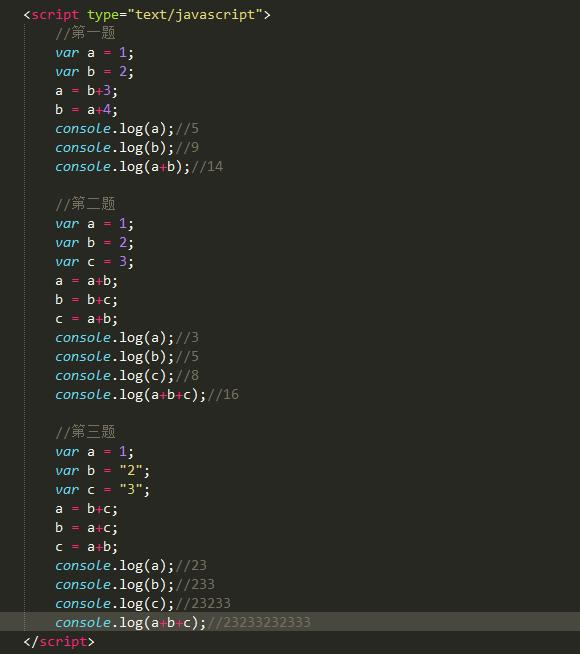
变量传递，就是使用如同 a=b 这样的写法，将等号右边的值，赋给了左边的变量；

等号右边的变量是不变的！

陈如峰已经嫁给了河南，然而刘远康依旧还是自己



简单的小练习：



注意：把引号内的东西看成是一个文字，不要看成数字，文字之间只有拼接没有运算

第三题中，变量的传递，将数值型转换为了字符串类型，因此，所有的运算都变成登录拼接！

## 1.8 变量扩充

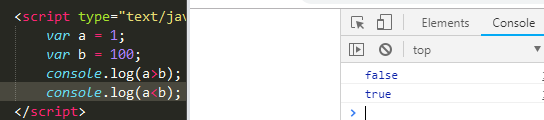
### 1.8.1 布尔值 boolean

布尔值 只是一个判断值

ture

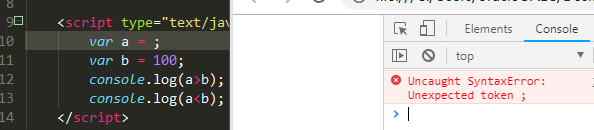
false

在我们的实际运算中。 ture=1 false=0



### 1.8.2 undefined 变量未初始化

也就是说，定义了变量，但是没有给变量赋值，那么这个变量就没有初始化。此时会报错！



### 1.8.3 null 变量未引用 值为空 object



JS有复杂的数据类型

object 对象

array 数组

# 二、JS的运算法则

**程序员眼中 算法为王！**

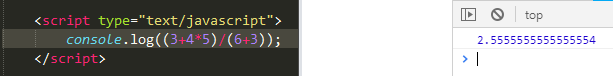
## 2.1 数学运算符

整体感知什么是数学运算



这是在上学的时候最常见的数学公式

表达式：



在这里 + \* / （） 都叫运算符

那么这个写法叫做表达式

在JS中有很多的运算符

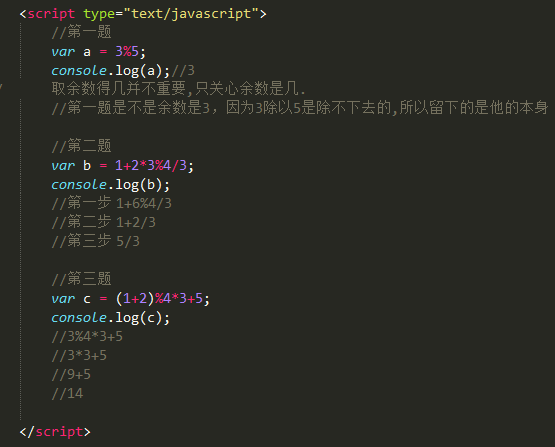
数学运算符、逻辑运算符、自增运算符

|  |  |
| --- | --- |
| 符号 |  |
| + | 加号（1、表达是相加 2、表达拼接） |
| - | 减号 |
| **~~=~~** | **~~等号不是数学运算符它是赋值~~** |
| \* | 乘法 |
| / | 除法 |
| % | 取余数 |
| （） | 运算的法则 |

先乘除 后加减 有括号先算括号里的

注意：在程序中没有中括号和大括号，中括号和大括号是用来执行程序的，小括号才是用来计算的

练习题：



在这里强调下，取余数，注意除不下去就是取它本身

在第一题中，3%5，因此取余数就是3

思考题：今天是礼拜一，第1000天后是礼拜几？ 1000%7 = 6

请用我们的算法思维

1000/7 = 142.8

就是相当于1000内有142个礼拜

142周有多少天？

142\*7=994

994那天是礼拜几？ 礼拜一

994+7 礼拜一

所以减1

1000天以后是礼拜天

## 2.2 乘方与开方（Math对象）object

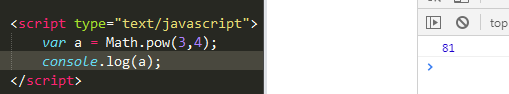
=3\*3\*3\*3=81

语法：

Math.pow(a,b)

a是底数

b是指数





Math.pow(3,4\*5)



先算4的5次方

这个案例中就出现了次方套次方

Math.pow(3,Math.pow(4,5))



并不是程序报错，而是无法计算

infinity是无限的意思





请注意：这个写法中，经常会出现括号套错的现象！

## 2.3 开方

=9

Math.sqrt()



其实呢，在我们的数学运算中，还有很多高级的写法，比如sin cos 三角函数

log 对数

这些写法呢，暂时不做介绍，大家可以去找下

打开一个W3C

## 2.3 变量的格式转换

prompt 用户输入 请注意，在程序中默认的用户输入的东西 全部都是字符串

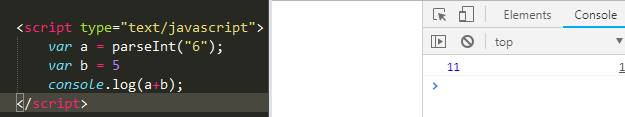
我们需要一个媒介，把用户输入的字符串转换成为数值

### 2.3.1 字符串转数值

parse**I**nt 把字符串转换为数字

parse 转换的意思

Int 整数的意思





注意：这个东西有两个特征

第一个特征，就是它会去取整，并且没有四舍五入这种说法！**直接保留的是小数点前面的整数！**

**记住在JS中四舍五入是要单独去运算的！round(x)**

第二个特征，自动净化功能。



就是JS中要么读取字符串，要么就是读取数值，假如你的字符串中，出现了数值，那么。它将会把第一个数值保留，并且自动删除后面的文字！



**自动净化，一定是数字在最前面！**



练习题：



答案：10



答案：9

parse**Float()** 保留小数



### 2.3.2 数值转换为字符串

你特么的就在数字上加上一个狗逼引号不就好了么！~

# 运算练习题：

第一题：小学生级别

世界上有两种表示气温的方式，第一个叫摄氏度，第二个叫华氏度。

两者之间是可以转换的

转换的公式：华氏度=（9/5）\*摄氏度+32

那么，现在请编写一个程序，让用户输入一个摄氏度，然后向用户提示相应的华氏度。

解题：

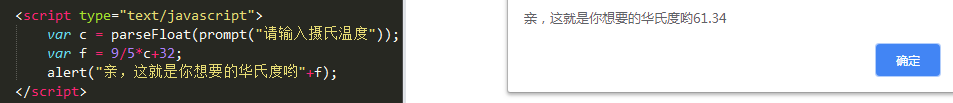
原则：编程呢就是开开心心的

步骤：第一步：用户输入一个数值 prompt

第二步：将用户输入的字符串转换为数值（保留小数）parseFloat

第三步：进行公式的计算 算法

第四步：弹出计算结果 alert



第二题：初中生级别

请用户输入一个六边形的边长，然后程序显示出该六边形的面积

计算六边形面积的公式：

面积=\*边长2

第三题：高中生级别

用户输入一个三位数，用程序计算该三位数每个位数的总和

输入123 输出1+2+3=6

输入101 输出 1+0+1=2

比如让用户输入 345 3 4 5

第一步：取百位

345/100=3.45 取整 之后 就剩3 int(a/100)

第二步：取十位

345%100 45 45/10 4.5 取整 4

第三步：取个位

345%10 5

第四步：计算

**作业：如果今天是礼拜六，那么1000天后是礼拜几。用一条语句书写！**

# JS第二天笔记

# 一、布尔值和关系运算符、逻辑运算符

## 1.1 布尔值（布尔类型）

数值型：里面的值有无穷多个，因为世界上所有的数字，都是数值类型。

字符串类型：里面的值有无穷多个，因为世界上所有的文字、语句都是字符串类型。

布尔类型：只有两个，一个叫true，一个叫false。



**那么请注意，千万不要给布尔值打上引号！**



## 1.2 关系运算符

整体感知下：什么叫关系

5和6 之间 它们是怎样的关系 5<6

|  |  |
| --- | --- |
| 等于 | == 陈如峰=陈铁峰（双胞胎） |
| 全等于 | === 5===5 |
| 小于 | < |
| 大于 | > |
| 小于等于 | <= |
| 大于等于 | >= |
| 不等于 | != |
| 不全等于 | !== |

在这里，等于和不等于，全等和不全等他们是相互对立的关系，跟其它的关系符不一样

55==55 t

55!=55 f

55===55 t

55!==55 f

3=8 f

3!=8 t

3===8 f

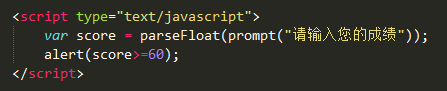
3!===8 t

**总结下：**

**!= 就是 == 的反面**

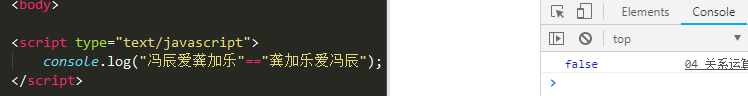
**!== 就是 === 的反面**

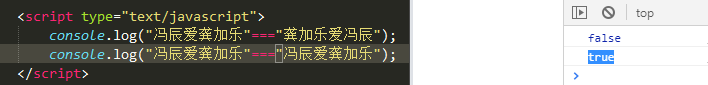
编写一个案例：判断用户输入的成绩是否合格 >=60



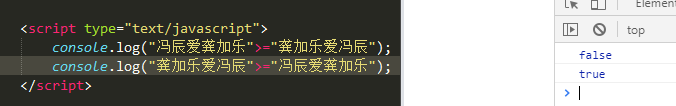
**关系运算的重难点：**

**重点：关系运算符还可以去判断字符串类型，它可以做到判断里面的内容！**





字符串的判断一样可以用>==或者是<=



**难点：**但是注意，通过我们的实验发现：当a>=b的时候，false

但是当a<=b的时候，ture

**龚加乐（b）比任何一个人都爱你！**



等号他是一个非常不严谨的家伙，因此它会把数据转换成相同的类型来比较！字符串就被转换成了数值，因此56=56 T

如果，想判断数据和类型都一样的话，就必须要用全等



因此全等是非常严谨的

## 1.3 逻辑运算符

什么是逻辑？

有效性

标准和规范

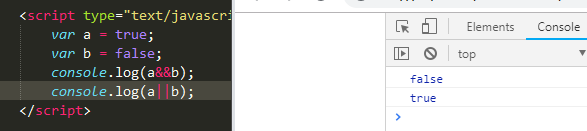
执行 --- 讲究是与否

逻辑是讲究一个规律性的（思维上的规律，客观事实上的规律）

|  |  |
| --- | --- |
| 与（且） | && |
| 或 | || |
| 非 | ! |

那么请注意，参与逻辑运算的，都是布尔值！也就是说，只有ture 和 false才能参与逻辑运算，而且这个运算的结果它也是一个布尔值

举个例子



与：只有一个是错误的，那么输出的结果全是错误的

就比如 1+1=2 true

冯辰是男的 true

输出就是 true

再比如 1+1=2 true

冯辰是女的 false

输出的结果 false

因此只要一个条件不满足，就会输出错误结果！

或：只要满足一个条件，那么输出的结果就是正确的

比如： 1+1=3 false

冯辰是男的 true

输出就是 ture

再比如： 1+1=3 false

冯辰是女的 false

输出的结果 false

非：但凡只有一个条件满足，它的对立面就是错误的



## 1.4 连比的写法

举个例子：

大家试着写下：3<2<15



为什么？

ture=1

false=0

首先，你们要明白，程序是一行一行来执行的，也就是它一行从头读到尾。

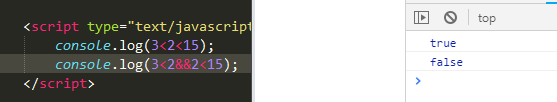
因此，计算机先会去判断3<2 这件事情是否成立，显然这是false

第三步就变成了false<15 此时false=0

因此，就变成了0<15，那么事实上 这个是true

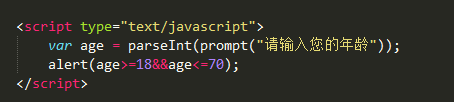
那么正确的写法怎么写？我们需要把它拆开

3<2&&2<15



练习题：

判断一个人的年龄是否能够考取驾照，交通法规规定的年龄是18-70周岁



# 二if语句

整体感知：如果。。。。。那么。。。。。否则。。。。。

if(明天下雨的话){

我就要出去玩

}else{

我就安安心心在家里写作业

}

### 2.1语法结构：

if(写上条件表达式){

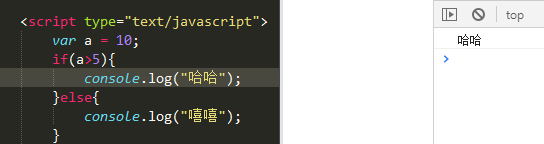
条件满足时（条件为真时）去执行命令

}else{

条件未满足时（条件为假时）执行的命令

}

举个例子：



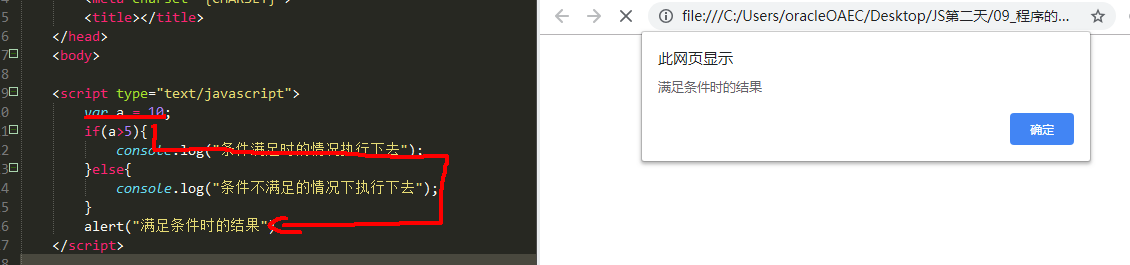
因此，请注意：

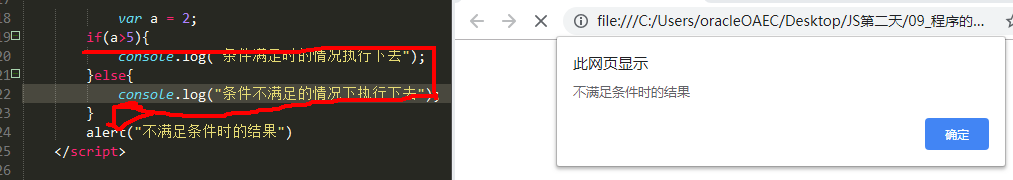
1、表达式，要么是true，要么是false。在一个计算当中，这是一个的绝对的情况，只有真或者假，不会出现模棱两可的情况！

2、程序执行必定会产生结果，除非你的程序写错了。

3、程序是可以有选择的。

什么叫程序的选择。





殊途同归：也就是程序回去选择一个条件来进行。那么这条路是程序员事先设定好的！

而且它的结果也是程序员事先设计好的！

案例：

用户输入一个成绩，提示用户是否及格。如果及格了，那么弹出警告框“恭喜你，合格了！”，

如果没有及格，那么弹出“兄弟，别着急，稳住，下次再来！”，但是不管结局如何，我们都要去满足用户体验，因此在最后都要弹出一句话：“我是冯辰，爱你么么哒！”

75分

原则：做程序员呢，就是要想开点！写程序呢一定要整整奇奇。

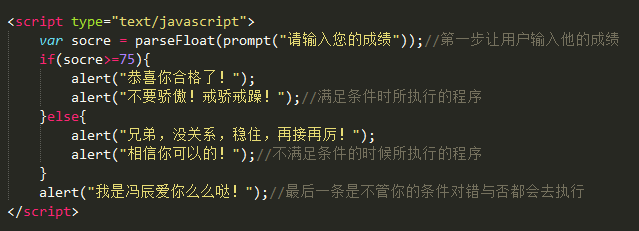
步骤：第一步：先让用户输入一个成绩，而且我们的成绩是可以带小数点的

第二步：执行if语句（条件） 成绩是否大于等于75

else否则

第三步：执行程序，并且弹出合格与否的结果

第四步：满足用户体验，我是冯辰爱你么么哒



练习题：

判断一个人的年龄是否能够考取驾照，交通法规规定的年龄是18-70周岁

如果满足年龄要求，那么提示他 恭喜你可以考取驾照了

如果没有满足年龄要求，那么提示他 你个马路杀手 滚！

但是最后不管怎么样

都要输出

我是冯辰 爱你么么哒！

## 2.2 if语句的多分支结构

也就是执行多次的条件判断

比如说：

用户输入成绩

如果成绩>=80 那么提示优秀

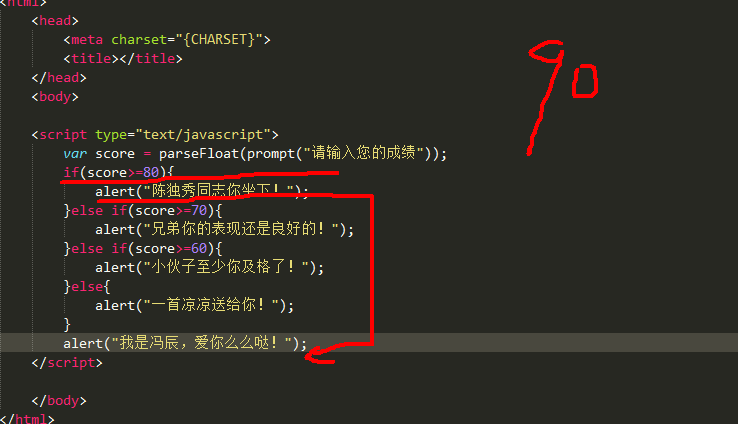
如果成绩>=70<80 那么提示良好

如果成绩>=60<70 那么提示合格

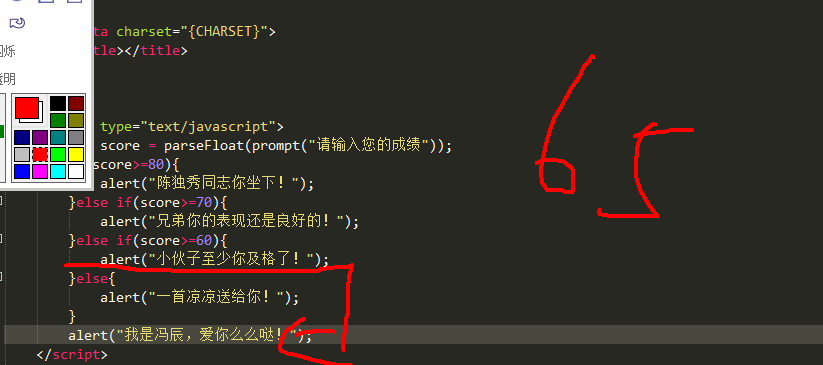
如果成绩<60 那么提示不及格



## 2.3 if语句的跳楼现象

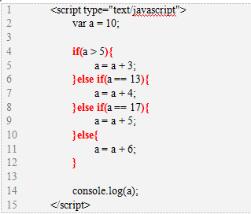


就是下一层已经暗示了上面的条件已经被满足，所以直接跳过下面的条件，直接输出结果。



跳楼现象就是直接跳过不满足的条件，执行下一行，如果下一行也不满足，那么继续跳过，直到程序执行结束

网易的面试题：



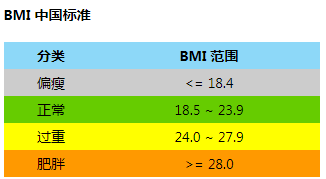
请回答，最后程序执行的结果是多少！

这道看上去很复杂，就是只管去考察跳楼现象

世界上有个标准 BMI 指数 去计算一个人的身体体型

BMI=体重/身高的平方

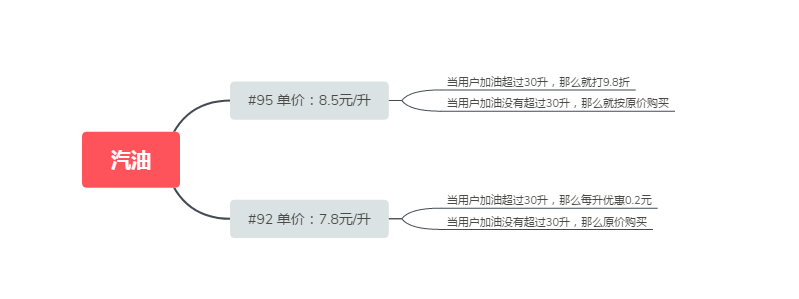
编写一个程序，让用户输入他的身高单位是cm；输入他的体重单位是kg，然后输出他的身体体型





## 2.4 if语句的嵌套

if else 再去添加 if else



原则：

步骤：第一步 首先输入汽油的种类，然后去判断是92还是95

第二步 如果是92号汽油

2.1步 去判断是否加满30升

2.2步 计算油价

2.3步 否则全价购买

第三步 如果是95号汽油

3.1步 判断是否加满30升

3.2步 计算油价

3.3步 否则全价购买

第四步 输出购物价格，然后祝您旅途愉快

但是！你试着去猜想一下，有没有傻逼会输入94号汽油！



今天晚上的作业：

某个公司要给员工发年终奖，为了奖励老员工，所以工作时间越长，发的越多，规则如下：

工作满0年             发月薪的1倍月薪年终奖，如果月薪大于8000，那么就是发1.2倍

工作满1年             发月薪的1.5倍月薪年终奖，如果月薪大于10000，那么就是发1.7倍

工作满2年，甚至更多 发月薪的3倍月薪年终奖，如果月薪大于12000，那么就是发3.2倍

用JS编写程序，让用户输入工作了几年，可以输入0，然后输入月薪。然后算年终奖。

# 三、循环

## 3.1 for循环

for循环就是去执行一些相似的代码，让这些代码在编写时显得简单一点

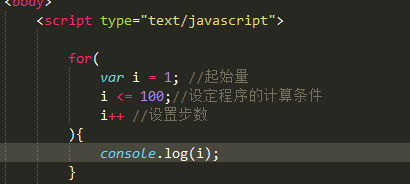
举个例子 ：请在控制台输出1-100

console.log(1);

console.log(2);

...

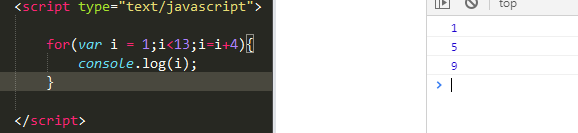
console.log(100);



补充一个知识点：

注意：i++ 自增算法 等价于 i=i+1 表达意思自增1步

案例一：

第一步： 起始值=1 判断1<13 成立，因此执行输出1；

第二步： 执行i=i+4 此时i=5 判断5<13 成立，因此执行输出5；

第三步： 继续执行i=i+4 但是此时i已经是5 因此i=5+4=9 判断9<13 成立，输出9；

第四步： 继续执行i=i+4 但是此时i已经是9 因此i=9+4=13 判断13<13 不成立，程序终止；

案例二：



这段程序是没有错的，只是条件没有满足，因此不输出任何的结果

第一步：起始值i=50 它同时会执行判断50<40 条件不成立，因此结束程序，没有输出任何结果

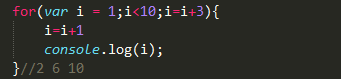
案例三：



1-10

注意：条件中10<=10 这个是成立的

案例四：



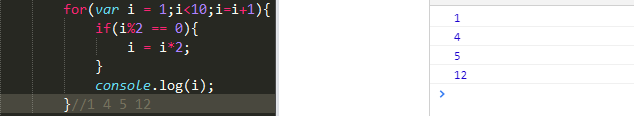
i=1 1<10条件成立 因此要执行i=i+1 因此输出i=2;

i=2 2<10 条件成立 同时执行 i=i+3 此时i=5 再次执行 i=i+1 因此输出i=6;

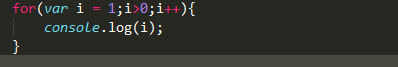
i=6 6<10 条件成立 同时执行 i=i+3 此时i=9 再次执行 i=i+1 因此输出i=10;

i=10 10<10 条件不成立，程序结束。

案例五：



案例六：

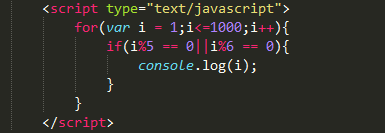


这个叫做死循环

练习：

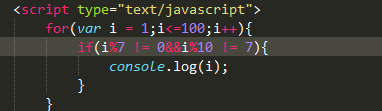
请寻找1-1000以内所有能被5整除、或者能被6整除的数字

i%5 == 0||i%6 == 0



踩地雷游戏 如果报到能够被7整除的数字，或者是尾数为7的数字，就算是踩到地雷。

请大家在控制台输出1-100以内所有安全的数字



思考题：

水仙花数：每个数位的立方和等于它本身

153

13+53+33=153

1-1000之内有4个水仙花数 153 370 371 407

思考下如何把它找出来。

for(var i=0;i < 10;i++){

代码段

}

var i=0;

for(;i<10;){

i++;

}

3.2 while 语句

var i=0;

while(i < 10)

{

代码段;

i++;

}

while循环和for循环的区别在于，while的定义是放在代码块的外部，++或者--是放在代码体中，是在我们的代码块执行完成之后

3.3 do while 语句

var i=0;

do{

代码段;

i++;

}while(i<10);

do while和while语句的区别

1、while是先判断条件语句，当条件语句为真时，才进入代码段，如果条件为假，则不进入循环语句

2、do while循环，是无论如何都会先执行一次代码段，再进入判断条件语句，当条件语句为真时，再次进入代码段，如果条件为假，则不进入循环语句

for( 1;2;3){

4

}

1 2 4 3 2 4 3 2 4 3

3.4 break语句

用于跳出当前循环，紧接着后面的代码段语句也不执行，只适用于循环语句和判断语句

3.5 continue语句

用于语句中断循环中的迭代，如果出现了指定的条件，继续循环，循环中的下一个迭代

四 switch 语句

是用来选择多个要执行的代码块之一的一个

switch(初始值)

{

case 1:

执行代码块1；

break;

case 2:

执行代码块2；

break;

default:

当初始值不满足case当中的条件时，执行的代码块

}

switch和if的区别

switch只能使用整型或者单字符型的参数,中有分支的话，是使用case关键字做为条件的代码块

if 可以使用任意表达式( < > <= >= != == && || )，是使用else关键字做为条件的代码块

五 数组

5.1 数组定义

var 数组名称 = new Array();

三种方式：第一种，不固定长度，第二种，固定长度，第三种，在初始化时赋值

5.2 赋值

赋值从0开始 数组名称[下标],例如：数组名称[0] = 值;

5.3 数组的初始化和遍历

固定长度的，在初始化时，会开辟固定长度值的内存空间

遍历使用就是循环语句来遍历数组

5.4数组数据的个数 length属性

数组名.length属性

5.5 数组合并 concat

第一种：可以直接拼接值 数组名.concat(第一个值,第二个值....)

第二种：可以直接拼接另一个数组 数组名.concat(另一个数组名)

第三种：可以拼接多个数组

5.6 join方法 返回一个字符串

该方法是将数组中所有的元素转换成字符串的形式连接起来返回，如果没有设置分隔符的话，默认是以逗号分隔，分隔符可以按照自定义规则，做为分隔符。

六 函数

6.1 函数的定义与方法

6.2 返回值