1.运算符

(1)位运算符

模拟计算机底层的运算，先把数据转为2进制，然后进行运算；当运算完以后再把结果转回成10进制。

1 2 3 4 5 6 7 8

1 10 11 100 101 110 111 1000

& 按位与，上下两位都是1结果是1，否则是0

| 按位或，上下两位含有1结果是1，否则是0

^ 按位异或，上下两位不同是1，相同是0

>> 按位右移，删除最后的位数，在原来基础之上除以2再取整，例如7>>1 执行 parseInt(7/2)

<< 按位左移，在末尾补0，在原来的基础之上乘以2，例如3<<1 执行3\*2

3 & 7 7&11 12 | 8 6 ^ 15

011 0111 1100 0110

111 1011 1000 1111

——— ———— ———— ————

011 0011 1100 1001

(2)赋值运算符

= += -= \*= /= %= ...

运算赋值：先执行运算，再执行赋值

练习：声明变量保存商品的价格，让该商品执行打八折，最后打印价格

(3)三目运算符

一目运算符：由一个运算符连接的一个操作数据或者表达式 ! ++ --

二目运算符：由一个运算符连接的两个操作数据或者表达式

三目运算符：由两个运算符连接的三个操作数据或者表达式

条件表达式 ? 表达式1 : 表达式2

如果条件表达式为true，执行表达式1

如果条件表达式为false，执行表达式2

2.浏览器端函数

alert() 弹出警示框

prompt() 弹出提示框(输入框)，需要使用变量保存用户输入的值，类型是字符串型，如果点击取消返回null

练习：弹出两次提示框，分别输入数字，最后计算两个数字相加的和，并将结果以警示框形式弹出

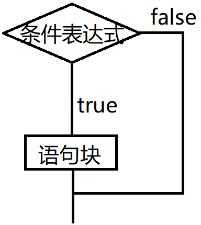
程序 = 数据 + 算法

程序的执行方式：顺序执行、选择执行、循环执行

3.流程控制

(1)if语句

满30减20

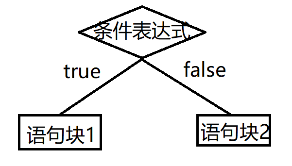


|  |
| --- |
| if(条件表达式){  语句块  } |

如果if后的语句块中只有一行代码，则大括号可以省略的

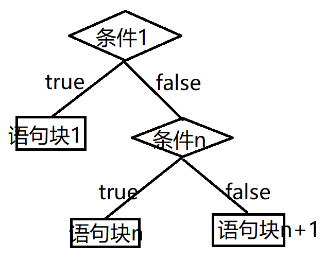
以下5种情况隐式转换为布尔型为false：0 NaN '' undefined null

(2)if-else语句



|  |
| --- |
| if(条件表达式){  语句块1  }else{  语句块2  } |

(3)if-else嵌套



|  |
| --- |
| if(条件表达式1){  语句块1  }else if(条件表达式n){  语句块n  }else{  语句块n+1 //以上所有的条件都是false  } |

(4)switch-case

是一种特殊的多项分支语句

|  |
| --- |
| switch(表达式){  case 值1: //如果表达式和case后的值相同，会执行对应的语句块  语句块1  break;  case 值n:  语句块n  break;  default:  语句块n+1  } |

表达式在和case后的值比较的时候，用的是全等于(===)

**对比if-else和switch-case**

if-else可以进行各种条件的比较，switch-case只能进行全等于的比较

switch-case的代码结构更为清晰，执行效率更高

课后任务

(1)复习今天内容，整理思维导图

(2)练习：弹出两次提示框，分别输入商品的单价和数量，计算出总价，假设总价满1000元打九折，当前会员卡内有余额1200，如果足以执行，警示框弹出'pay success'，否则警示框弹出'pay error'

html和js两个文件

(3)预习JS第4天

http://www.codece.com/archives/195