复习

原始类型和引用类型

数值型number，字符串型string，布尔型boolean，未定义型undefined，空null null->object

数据类型转换

隐式转换1+’2’ true+2 false+’2’ -\*/

强制转换

Number(undefined)

pcarselnt(‘a3’)

parseFloat(3a)

toString(8)

运算符 比较运算符，逻辑运算符，位运算符

算术运算符 + - \* / % ++ --

比较运算符 > < >= <= == != === !=

3 > ‘10’ 1>true ‘3’>’10‘ ’3‘.charCodeAt()

逻辑运算符 && || !

逻辑短路 age>18 && console.log(’成年人‘)

位运算符 & | >> <<

声明一个变量保存年份，判断这个年份是否为闰年，如果是打印‘闰年’。

闰年：查看闰年的判断条件

能被4整除，并且不能被100整除，或者能被400整除

整除：取余为0

1. 运算符
2. 赋值运算符

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

练习：声明变量保存商品的价格为20，计算打八折的结果。

var price=20;

//计算打八折，在原来的基础之上乘以0.8

price\*=0.8;

console.log(price);

练习：假如本金10000，存储利率3.5%,计算一年的利息是多少？

var money=10000;

money\*=0.035;

console.log(money);

1. 三目运算符

需要有3个数据或者表达式

一个条件有两种结果

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

练习：声明两变量，分别保存数字，比较两个数字。打印最大值

练习：声明两个变量，分别保存用户名和密码，如果用户名是root,

并且密码是123456，打印“登录成功”，否则打印“登录失败”

1. 浏览器端函数

alsert() 弹出警示框(消息框)

prompt() 弹出提示框(输入框)，需要使用变量来接收输入的值，

数据类型是字符串

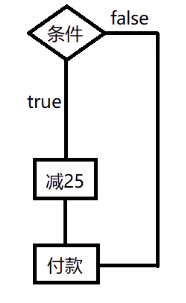
|  |
| --- |
| //1.弹出警示框  //ssalert('hello world');  //alert('达内');  //2.弹出提示框  //把输入的内容放入到变量中  var upwd=prompt('input password');  console.log(upwd);  //检测输入的内容数据类型  console.log(typeof(upwd)); |

1. 流程控制

程序=数据+算法

程序执行方式分为顺序执行，选择执行，循环执行。

1. if语句



;

|  |
| --- |
| 语句1  If(条件表达式){  语句2;  }  语句3; |

注意：如果if后的大括号中只有一个行语句，大括号是可以省略的。

|  |
| --- |
| 在if语句的条件表达式中，有一些值 默认就是false  undefined 0 null NaN ‘’ |

|  |
| --- |
| //练习:声明变量保存年龄，如果年龄超过50，在原来基础之上减5岁，打印年龄。  var age=52;  if(age>50){  age-=5;  }  console.log(age); |

|  |
| --- |
| 练习：声明变量保存个人签名，如果签名为空，打印‘这家伙很懒什么也没留下’，有签名就正常打印  var str='';  //如果签名为空，然后赋值  if (str==''){  str='这家伙很懒什么也没留下';  }console.log(str); |

1. if-else语句

|  |
| --- |
| 语句1  if(条件表达式){  语句2;  }else{  语句3;  } |

练习：使用弹出提示框分别输入商品的**单价和数量**，如果**总价**满500打八折，当前**余额**为1000，如果余额大于等于总价，打印支付成功，否则打印余额不足

|  |
| --- |
| 宵风:  //练习：使用弹出提示框分别输入商品的单价和数量，如果总价满500打八折，当前余额为1000，如果余额大于等于总价，打印支付成功，否则打印余额不足  //弹出提示框分别输入单价和数量  var price=prompt('input price');//strong  var num=prompt('input number');//strong  //计算总价  var total=price\*num;//隐式转换成数值型  //如果总价满500打八折  if(total>=500)  total\*=0.8;  console.log('money:'+total);  //余额  var money=1000;  //如果余额足够支付  if(money>=total){  console.log('pay success');  }else{  console.log('pay error');  } |

1. if-else嵌套

|  |
| --- |
| 语句0;  if(条件表达式1){  语句1;  }else if(条件表达式2){  语句2  }else…if(条件表达式n){  语句n;  }else{  语句n+1;//以上所有的条件表达式都为false  } |

练习：根据订单的状态来打印订单状态

1-等待付款 2-待发货 3-运输中 4-以签收 5-已取消 其它-无法追踪

1. switch-case

是一种特殊的多项分支语句，可以根据一个表达式的值，来执行不同的语句

|  |
| --- |
| 语句0;  Switch(表达式){  Case 1  语句 1  break; //阻止程序继续往后执行其它的语句  。。。  Case n;  语句 n  break;  default; //如果以上所有的条件都是false  语句n+1;  } |

注意;在case 中，表达式和值比较的时候使用的是全等于(===),

要求值和类型同时满足结果是true.;

练习：使用变量保存星期的值(0~6),根据值打印对应中文’星期’

。 对比if-else嵌套和switch-case语句

相同点：两者都可以用于多项分支语句

不同点：if-else可以判断相等或者不相等的情况，适用范围更广;switch-case 只能用于全等于的比较，结构更为清晰，执行效率相对较高

4循环执行

循环：就是一遍又一遍执行**相同或者相似**的代码。

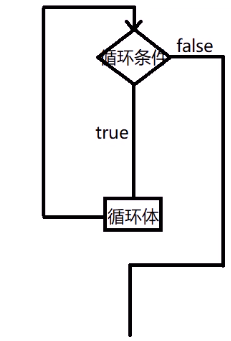
循环的两个要素：

循环条件：控制循环的次数

循环体：执行的相同或者相似代码

1. while 循环

|  |
| --- |
| while(循环条件){ //是一个布尔型的值  循环体  } |



|  |
| --- |
| //打印100~200之间所有的整数  //var i=100;  //while(i<=200){  // console.log(i);  //i++;  //} |

练习

使用switch-case 判断一个人的成绩标准

分数/10

80~90以下 8.0~8.9 取整8

70~80以下 7.0~7.9 取整7

使用while 循环打印1~100之间所有的奇数

使用while循环奇数1~100之间所有整除的和