学习一门编程语言路线图：

        （1）连接语言的背景、历史、特点、应用领域，现状

        （2）搭建开发环境，编写 hello world

        （3）声明变量和常量

        （4 ）数据类型

        （5）运算符

        （6）逻辑结构

        （7）通用小程序

        （8）函数和对象

        （9）第三方的类库，插件，组件，框架

        （10）项目开发

## 1.JS的执行环境

    （1）运行在客户端，使用浏览器所自带的JAvaScript解释器。本身浏览器过多，存在兼容性问题。

    （2）运行在服务器端，使用NpdeJS解释器，基于谷歌的V8引擎，在服务器端不存在兼容性。

        nodejs下载：http://nodejs.org   下载8.11.3 LTS

        查看nodejs是否安装成功

        node -v 显示所安装的nodejs的版本

        运行方式：node  js文件路径

2. js语法规范

    （1）区分大小写

    （2）每一行结束，可以不加英文的分号结束，但是习惯上加上分号。

    （3）单行注释（//……）多行注释（/\*……\*/）

## 4.变量

    变量是用于存储信息的容器，

    （1）变量的声明——var     variable

        var x=1; var 声明变量的关键字    X变量名称  x=1表示把1赋值给变量x  
        说明：如果声明了变量，但是没有赋值。默认是undefined。

    （2）变量的命名规范：可以放数字，字母，下划线（\_），美元符号（$）。

        不能以数字开头

        不能使用关键字和保留字作为变量名。

        保留字（class、int、float）

        变量名称习惯上有明确的语义，例如用户名：userName，密码 userPwd。命名分为驼峰命名法和下划线命名法。

        要求：使用变量来储存员工的信息，员工的编号，姓名，性别，生日，工资，并初始化具体的信息。

## 5.常量

    常量是不能改变的值。

    使用const关键字来声明

    习惯上常量名称使用大写： const PI=3.14；

6.数据类型。

    JS分为原始类型和引用类型

    （1）原始类型

        分为 数值型、字符串型、布尔型、未定义型(undefined)、null 共5类

        数值型又分为整数和浮点数。

            1)整数型

                分为八进制（以0开头），十六进制（以0x开头），十进制

            2）浮点型

                使用小数点记录的数据，还可以使用指数记录数据  如555.88   5.5588e2

课后练习：计算圆的周长和面积

                把半径5使用变量存储，把圆周率使用常量存储。

（2）字符串型

常用于表示一些文本字符数据，例如姓名，邮箱，电话等。

特点：使用引号（单引号双引号都可以）把数据包含

查看汉字的Unicode编码

‘一’.charCodeAt(); 19968

查看汉字Unicode十六进制形式

‘一’charCodeAt().toString(16); //4e00

查看16进制码对应得汉字

Console.log(‘\u4e00’) 检验汉字范围 4e00 ~ 9fa5

（3）布尔型

在程序中表示真或者假的结果

取值 true/false

常用于表示是否的结果 如：是否注册，是否是会员。

（4）未定义型

只有一个值undefined 。当声明了一个变量，未赋值，此时变量的值为undefined。

（5）null

用于释放一个变量。

检测数据类型

typeof（参数）参数表示要检测的数据

结果有number，string，boolean，undefined。

## 2.数据类型转换

（1）隐式转换

①数字+字符串：数字会被转换成字符串

②数字+布尔型：布尔值会被转换成数字。True->1 false->0.

③布尔值+布尔值：布尔值会被转换成数值。

④布尔值+字符串：结果为字符串。

练习：

var num1=3,num2='st',num3=true;

console.log(num1+num3+num2);//4st

console.log(num2+num3+num1);//sttrue3

（2）强制转换

①强制转换成整型

parseInt（）

parseInt(’23.21sf’); //23

从第一位开始找数字，遇到非数字或者小数点结束，如果第一位是非数字，显示NaN。

②强制转换成浮点型

parseFloat();

parseFloat(23.55t) //23.55

从第一位开始找数字，遇到数字、小数点继续往后找，遇到字符串停止。

③强制转换整数值型；

Var num3=’25.55’;

Number(‘23.55’) //23.55

Number(‘23.55t’) //NaN

Console.log(Number(num3));

如果被转换的值中有非数字，则结果为NaN

④转换成字符串型

toString();

20.toString //”20”

toString() 可以将数值型和布尔型的值转换为字符串型。

3.两个运行于客户端的函数

alert()警示窗口

prompt()提示窗口 提示窗口填入的数为字符串。

## 4.运算符和表达式

由运算符连接的操作数据，这种形式称为表达式。

1. 算术运算符 + - \* / % ++ --

%取余数

++ 自增 ，在原来的基础上加1 --自减

Console.log(num++) 先打印num，再执行＋1

Console.log(++num) 先执行+1，再打印num。

1. 比较（关系）运算符

> >= < <= == != ===（全等于） !==（不全等于）

产生的结果是一个布尔型的值（true/false）

等于（==）：只比较值。

全等于（===）：比较值和类型，两个都满足结果是true。

比较运算符的两端，只有一个数值型，另一个隐式转换成数值。

console.log(‘3’>’10’);//true 比较两端Unicode编码值得大小。

Console.log(‘3m’>10); ‘3m’会转换成数值型，采用Number（）转换，结果为NaN，

NaN和任何值比较（不分运算符）结果都是false。

1. 逻辑运算符

&& 并且 || 或者 !非（非真为假，非假为真）

逻辑中断（短路逻辑）

&& 如果第一个条件为false，整个结果就是false，第二个条件就没有执行的必要。

|| 如果第一个条件为true，则整个结果为true，第二个条件就没有执行的必要。

1. 位运算符（了解即可）

在执行位运算的时候，十进制的值会转成二进制。

按位与 & console.log(3&5) //1

按位或 | 上下两位只要有一个1，则为1.否则为0.

按位异或^ 上下两位相同，结果为0, 不同为1

按位右移>> （3>>1） 结果为1 11向右移动一位，删除最右一位。

按位左移<< （3<<1）结果为6 11向左移一位，最左加0，为110.

1. 赋值运算符

(++) (--) (=)

(+=) 在原来基础上＋3 (-=) (\*=) (/=) (%=)

练习：判断年份是否为闰年。4年 && !100年整除 && 400年整除

2018.8.8

1.三目（条件）运算符

一目运算符就是只有一个操作数或者表达式 a++ a-- !false

二目运算符就是有两个操作数或者表达式 1+2 3>2 a=1 +1\*/ >= && 等

三目运算符是含有三个操作数或者表达式：

条件表达式？表达式1：表达式2

如果条件表达式结果是true，执行表达式1，否则执行表达式2.

今日目标： 流程控制 循环语句

## 1.流程控制

程序：数据加算法

任何程序的算法可以分为三类 “顺序执行”，“选择执行”，“循环执行”。

选择执行：程序可以选择执行某一段代码，也可以选择不执行。

1. if语句

|  |
| --- |
| if(逻辑表达式){  语句1;  语句2;  } |

如果逻辑表示是结果是true，执行语句1&语句2.如果结果是false，跳过语句1&2。

练习：通过弹出提示框输入商品的总额，如果总额>=500,则打八折。

注意：①if后的大括号语句只有一条，可以省略大括号②逻辑表达式可以直接写true/false.

False : undefined null 0 ‘’ null NaN

1. if…else语句

|  |
| --- |
| 语句1；  If(逻辑表达式){  语句2;  }else{  语句3；  }  语句4； |

执行流程：

①执行语句1；

②逻辑表达式值，如果结果是true，执行语句2. 如果结果是false，执行语句三。

③执行语句4；

1. if…else的嵌套

|  |
| --- |
| 语句0；  if(逻辑表达式1){  语句1  }else if(逻辑表达式2){  语句2  }else if（逻辑表达式n）{  语句n  } else {  语句n+1//以上n中情况都不满足的情况才执行。  } |

执行流程：

①执行语句0

②执行逻辑表达式1，如果结果是true，执行语句1，否则执行逻辑表达式2……

③如果以上所有逻辑表达式都是false，则执行语句n+1.

1. switch…case语句

|  |
| --- |
| Switch(表达式){  case 1： //如果表达式的值是1，选择进入这个入口执行程序  语句1；  ………  Break;//终止程序的进行。  Case n：  语句n  ………  Break;  default: //以上所有值都不匹配，选择这个路口。  语句n+1  } |

在switch语句中，比较所执行的语句是全等于（===）

If…else嵌套语句和switch…case语句的区别

If…else可以判断所有的情况，如大于，小于，等于。

Switch…case只能用于全等于（===）的比较，局限性比较大；但是其执行效率更快。

1. 循环语句 while循环

循环就是一遍又一遍执行相同或者相似的代码

例如：循环产生商品列表，打印100次hello。

|  |
| --- |
| While(布尔型的表达式){  //要循环的代码。  } |

2018.08.09

1.循环中的break

用于终止循环，出现break之后，循环到此结束。

2.do…while循环

|  |
| --- |
| do  {//循环体  }  while (循环条件); |

执行过程：

先执行循环体，若判断条件为true，则继续执行循环体。

While于do…while的区别

①while先判断后执行

②do while先执行后判断

③如果条件为false，while立即终止，do…while先执行，在终止。

For循环

|  |
| --- |
| for(表达式1;表达式2;表达式3) {  //循环体  } |

表达式1：循环的初始值,例如：var i=1;

表达式2：循环的条件，例如：i<10;

表达式3：循环的变换，例如：i++

以上表达式位置固定。

执行顺序：

for ( var I =0 ; I <=9 ; i++ )

{console.log(i)

}

①初始化i的值

②判断i的值是否满足小于10

③打印i的值

④执行自增i++

⑤再执行②

练习：打印一行\*\*\*\*\*。

Break he continue在循环中的应用

Break 结束循环

Continue 结束这一次的循环，还执行自增自减等。

练习：打印2000~2100年之间所有的闰年

练习：交换两个变量的值。

作业：斐波那契数列

有一对兔子C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B014579.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B0709CA.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B074E65.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B078A16.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B07A36B.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B07D112.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B07EB70.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B082906.png C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\10344\2B088B98.pngC:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\9232\49E1AFBF.png，兔子每到第三个月，开始繁殖一对；所繁殖的兔子，到第三个月，又继续繁殖。

1 1 **2 3 5 8 13 21 34 55**

**使用for循环计算第12个数是多少。**

5.循环嵌套

外层循环控制行数

内层循环控制列数

作业

1. 整理思维导图
2. 预习js自定义函数
3. 完成99乘法表。

复习：

While（）{ }

Do{ }while( )

For(; ;){ }

2018.8.10

## 函数

函数 变量的作用域 函数的作用域 递归。

1. 函数

分为两种：①JS提供的，例如typeof，parseInt，可以直接使用。

②自定义函数（function），也称为方法（method）；一段已经预定义好的代码块，并且可重复执行。

1. 定义函数和使用函数
2. 普通函数的声明和调用

|  |
| --- |
| Function 函数名称（）{} |

函数只是声明，里面的代码不会执行，只有调用函数，里面的代码才执行。

1. 带有参数的函数声明和调用

|  |
| --- |
| Function 函数名称（参数列表）{函数体  }  声明时的参数称为‘形参’，调用时的参数称为“实参”，形参用于占位，在调用的时候实参的值会付给形参。 |

1. 带有返回值的函数的声明和调用。

|  |
| --- |
| 声明  Function 函数名称（）{  //函数体  Return 值；  }  调用  函数名称（） |

说明：①函数中没有添加return，返回的结果是undefined；

②函数中有return，返回值为空，返回结果为undefined。

③return后的所有语句都不能被执行。Return就是函数的结束。

练习：声明函数getMax，传递两个参数，返回最大值。

练习：声明函数getMax，返回最大值。

练习：声明函数声明闰年；

1. 变量的作用域

变量的作用域指变量的可访问范围。

分为全局作用域 和 局部（函数）作用域

全局作用域下的变量可以在任何位置下访问。

局部作用域下的变量只能在局部（函数内部）访问到。

注意：函数内部声明变量没有使用var关键字，声明的变量也是全局变量。

**变量提升**

JS程序在执行前，声明的变量都会加载到程序的最前边，他只是提升声明，但是赋值还是在原来的位置。函数内部也存在变量的提升。

1. 函数的作用域

函数和变量相同，也分为全局作用域和局部作用域。

局部作用域（在函数内部）下声明的函数只能在函数的内部访问。

全局作用域下声明的函数可以在任何位置访问。

在程序执行前，声明的函数会提升到程序的最前面。

1. 递归

在函数的内部，调用自身。

作业：

1. 整理思维导图
2. 使用递归计算斐波那契数列1 1 2 3 5 8
3. 预习匿名函数，回调函数，对象。

复习

## 对象

1、函数内部访问函数名

arguments.callee 访问当前函数名称。经常使用于函数内部。会自动访问当前的函数名称。

2、声明函数的方式，还可以使用匿名函数的方式

3、回调函数

匿名函数以实参形式传递，这种写法叫回调函数。

getSum（function（）{return 1}）

匿名函数的字调用

(Function(){//函数体})();//匿名函数自调用

4、全局函数

EncodeURI 对一个URI进行编码（针对里面的中文）

DecodeURI 对已经编码的URI进行解码

isNaN 判断一个值是否为NaN，是返回true

isFinite 检测一个值是否为有限值，是返回true。

parseInt

parseFloat

eval 执行字符串表达式的值，。、

1. 对象

Js中的数据类型：原始数据类型和引用数据类型。

应用数据类型：对象，函数。

对象如何定义？

任何的事物只要有属性，有功能，都可以叫对象。

万物皆对象。

对象：具体到某一个事物。

JS中的对象是一组属性和方法的集合。

JS中的对象分为①内置对象（原生对象）；②宿主对象；③自定义对象。

1. 自定义对象：

对象直接量

内置构造函数

使用自定义的构造函数。

1. 使用对象直接量创建对象

①使用大括号创建对象{}

②多个属性之间用空格隔开

③属性名和属性值之间用冒号隔开

如果对象中出现了空格，-，属性名必须添加引号，其他情况可添加可不添加。

对象中属性的访问o[a]

对象.属性名

对象[‘属性名’]o= { a:50}

属性名使用了‘’引号，访问形式只能使用中括号形式。

添加或者修改对象的属性

对象.属性名=属性值

对象[‘属性名’]=属性值

1. 使用内置构造函数

Var o=new Object();

需要单独添加属性，为了便于使用这种方式——工厂函数。

练习：创建电脑对象，品牌brand，大小size，颜色 color

1. 遍历属性

获取所有的属性及对应得属性值。

|  |
| --- |
| for(var key in 对象){//如果要获取属性值只能使用——随想[key]  } |

注意：只能遍历自定义属性，不能遍历预定义属性。

var car={

brand:'BYD',

type:'皮卡',

color:'white',

country:'ningbo',

}

for (var key in car )

{console.log(key+'----'+car[key])

}

1. 对象中的方法

|  |
| --- |
| var person ={  say:function(){}//成员方法  }  成员方法的调用 person.say() |

作业：创建一个圆对象，添加属性（r），添加方法getLength，获取面积（getArea）

预习数组的创建和方法

复习

Arguments.callee()函数内部访问函数名称

匿名函数

Var fn=function(a,b){ } fn(1,2)

全局函数

EncodeURI isFinite eval 执行字符串表达式

decodeURI

对象：属性和方法

内置构造函数 var dog=new Object();

封装：把重复的代码放到函数里（函数封装）

把重复的代码放在对象中称为对象封装

遍历 for（var key in dog）{key +dog[key] }

8.14.2018

1. 对象中的方法
2. 对象.hasOwnProperty(‘属性名’) //true 存在
3. 属性 in 对象 //true 存在
4. 对象.属性名 ===undefined //true 不存在
5. 数据在内存中的存储方式

原始数据类型储存在栈内存中，储存的就是数据的值。

引用数据类型储存在在堆内存中，栈内存储存地址。

1. 数组的创建

数组也是一种数据类型，是数据的结合。可以储存任何形式的数据类型。

1. 数组直接量

|  |
| --- |
| var arr=[元素1，元素2] |

1. 内置构造函数

var arr=new Array(元素1，元素2)

var arr=new Array(3)//创建数组，长度是3

1. 数组的访问形式

数组名称[下标 ]

在数组中添加新的元素 arr[arr.length]=新的元素

练习：信息录入

弹出提示框，每次输入一个姓名，把输入的姓名放到数组中

1. undefined在哪里出现过

①声明里变量未赋值

②实参的数量小于形参的数量，未赋值的形参为undefined。

③访问对象中不存在的属性，结果为undefined。

④创建数组，未初始化数据，结果为undefined

1. 数组遍历

使用for循环遍历数组的下标

|  |
| --- |
| var arr=[‘a’,’b’,c,d]  for(i=0;i<=arr.length;i++){console.log(arr[i])} |

1. 数组的分类

数组分为索引数组和关联数组

索引数组：下标以数字的形式 可用for循环，也有用for in 遍历

关联数组：下标以字符串形式 可用for in来遍历

创建数组，保存学生的成绩，使用遍历数组来获取总分以及平均分。

练习：使用函数，传递一个参数，格式是数组，返回数组中的最小值。

练习：查找数组中指定元素的位置下标。

1. 数组中的方法

js中提供的方法称为API

JS中提供的数组方法称为数组API

toString（）将数组转为字符串

join（[参数]）（带”[ ]”的参数可有可无）参数表示元素之间的分隔符，默认是逗号

concat（）用于拼接多个数组 arr.concat(arr1,arr2)

slice（）截取数组元素

console.log(arr3.slice(2,4));//从2开始，包含本身。在4结束，不含本身。包含开始不包含结束

console.log(arr3.slice(-3));//倒数第三个。

splice（start,count,value）删除数组元素。start开始。count表示删除数量，value表示删除后替换的元素。如果count为空，表示删除到最后。

reserve（）翻转数组元素

sort([参数]) 对数组元素进行排序，

默认按照Unicode码，从小到大排列。

|  |
| --- |
| Sort(function(a,b){return a-b}) //从小到大排序 |

8.15.2018

push(参数) 在数组最后添加元素

pop() 删除最后一个元素

unshift（参数）在数组最前边添加元素

shift（）删除数组第一个元素

## 二维数组

1. 二维数组

在数组的元素也是数组。

1. 字符串对象

把字符串、数值、布尔型包装成对象。都称为包装对象

|  |
| --- |
| var str1='heLlo'; //直接量  var str2=new String('WoRld'); // 构造函数  var str3=String('TOm'); |

## 字符串API

toUpperCase 转为大写

toLowerCase 转为小写

charCodeAt() 查找下标字符对应得Unicode码

转义字符

\n ：换行

\’ ：将引号转换成普通引号

\t ：转换成制表符

### indexOf(‘a’，5)查找某个字符对应下标

lastIndexOf（a）最后一次出现位置的下标

练习，检测一个邮箱是否合法

//通过身份证获取年月日

练习：通过邮箱地址获取有户名和域名

练习：把英文单词首字母大写，其余小写

split（‘|’）将一个字符串按括号内指定字符切割字符串 ‘’

## 匹配模式

replace（参数1，参数2）查找某个字符串并替换；参数1表示要查找的字符串，参数2表示要替换的字符串

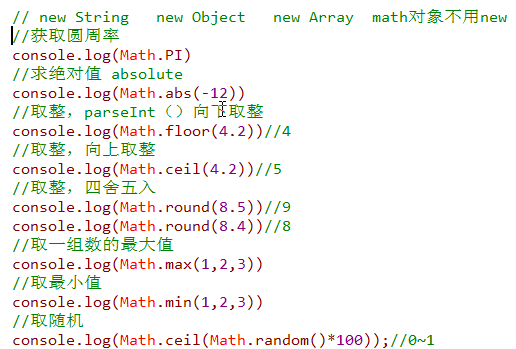
str.replace(/china/ig) ignore global

match(参数) 查找某个字符串，返回的是数组，可以用ig

search（参数）查找某个字符串，返回查找到的第一个字符串的下标值。

## math对象

不需要new来创建，直接使用math对象



课后练习：

使用数组保存10个人名，每次取随机一个

预习date对象及ES6

复习

数组API push() pop()删除最后一个 unshift() shift()删除第一个

二维数组

字符串API

str.charAt()/charCodeAt() str.indexOf()字符对应下标

lastindexOf()最后一个的下标

截取slice(2,5)

截取 substring()

splite(‘ ’)切割字符串变成数组。

join（‘|’）将数组变成字符串

匹配模式 replace（/CHina/ig, ‘ ’）

match 查找，返回一个数组 search 返回下标

8.16.2018

## date 对象

1、date对象

1. 获取date对象

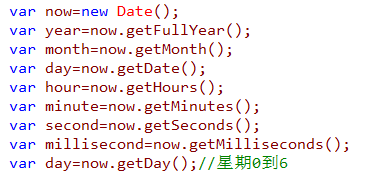
new Date()获取系统对象

new Date(2018,11,20,10,20,30)获取某个时间（月份：0~11）

new Date (‘2018-11-20 10:20:30’)

new Date(24\*60\*60\*1000) 获取距离计算机元年的对象

1. 获取date对象下时间信息



1. 设置date对象下时间信息

setFullYear（）/setMonth() setDate()/setHours()/setTime

setTime() 设置距离计算机元年的毫秒数

练习：计算合同的到期时间

1. 本地化date对象

toLocaleString() 年-月-日 时：分：秒

toLocalDateString() 年-月-日

toLocaleTimeString 时：分：秒

2.Number对象

console.log(Number.MAX\_VALUE);//1.7976931348623157e+308

console.log(Number.MIN\_VALUE);//5e-324

console.log((0.1+0.2).toFixed(2))

3.Boolean 对象

将数据转为Boolean的值

new Boolean() /Boolean()

!!值 （推荐写法）

4.错误处理

（1）语法错误 Uncaught Syntax Error: Invalid or unexpected token

只要出现语法错误，整个程序不执行

(2)引用错误 Uncaught Reference Error: hello is not defined

使用了不存在的变量，会影响后续代码的执行

1. 类型错误Uncaught TypeError: a is not a function 错误使用了括号，会影响后续代码的执行。
2. 范围错误 Uncaught RangeError: Invalid array length 参数超出了范围，影响后续代码的执行

因为引用错误，类型错误，范围错误会影响后续代码的执行

try

## ES6新特征

5.ES6新特征

ECMAScript

（1）块级作用域

在{ }之间使用let声明的变量，只能在{ }内部访问。这个{ }所在的作用域称为块级作用域。

1. 箭头函数C:\Users\web\AppData\Local\Temp\SGPicFaceTpBq\4848\4F3A5A1A.png

在回调函数中传递匿名函数可以使用箭头函数

|  |
| --- |
| arr.sort(a,b)=>{ return a-b} |
| arr.sort((a,b)=>a-b) 如果只有1行，并且含有return可这样使用 |

（3），模板字符串

使用反引号`` 所包含的字符串就是模板字符串，在模板字符串之间可以放任意的字符，

····· 例如：引号， 括号，运算符

在模板字符串中使用变量或者其他的值----$[变量名称]

课后作业

获取自己的生日，查看是否为周末，如果是周末提前到上一个周六

预习nodejs 中的模块---概念