[day01 3](#_Toc10049761)

[1.JS概述 4](#_Toc10049762)

[2.搭建JS开发环境 4](#_Toc10049763)

[3.JS代码规范 5](#_Toc10049764)

[4.变量 5](#_Toc10049765)

[5.常量 6](#_Toc10049766)

[6.数据类型 6](#_Toc10049767)

[day02 7](#_Toc10049768)

[1.数据类型 8](#_Toc10049769)

[1.原始类型 8](#_Toc10049770)

[2.数据类型转换 9](#_Toc10049771)

[3.运算符 12](#_Toc10049772)

[day03 15](#_Toc10049773)

[1.位运算符 16](#_Toc10049774)

[2.赋值运算符 16](#_Toc10049775)

[3.三目运算符 16](#_Toc10049776)

[4.浏览器端函数 17](#_Toc10049777)

[5.流程控制——选择执行 17](#_Toc10049778)

[day04 20](#_Toc10049779)

[1.循环执行 21](#_Toc10049780)

[1.while循环 21](#_Toc10049781)

[2.do-while循环 22](#_Toc10049782)

[3.for循环 22](#_Toc10049783)

[4.break和continue 23](#_Toc10049784)

[5.循环嵌套 23](#_Toc10049785)

[day05 23](#_Toc10049786)

[1.函数 24](#_Toc10049787)

[2.变量的作用域 26](#_Toc10049788)

[3.函数作用域 27](#_Toc10049789)

[4.递归 27](#_Toc10049790)

[day06 28](#_Toc10049791)

[1.匿名函数 28](#_Toc10049792)

[2.全局函数 30](#_Toc10049793)

[3.对象 30](#_Toc10049794)

[day07 33](#_Toc10049795)

[1.判断是否含有某个属性 33](#_Toc10049796)

[2.数组 33](#_Toc10049797)

[3.创建数组 33](#_Toc10049798)

[4.数组的长度 34](#_Toc10049799)

[5.数组的分类 34](#_Toc10049800)

[6.遍历数组中的元素 35](#_Toc10049801)

[7.数组中的方法(api) 36](#_Toc10049802)

[day08 38](#_Toc10049803)

[1.对象和数组之间的关系 38](#_Toc10049804)

[2.二维数组 39](#_Toc10049805)

[3.字符串对象 39](#_Toc10049806)

[4.匹配模式 42](#_Toc10049807)

[5.Math对象 42](#_Toc10049808)

[day09 43](#_Toc10049809)

[1.Date对象 43](#_Toc10049810)

[2.Number对象 45](#_Toc10049811)

[3.Boolean对象 45](#_Toc10049812)

[4.错误处理 45](#_Toc10049813)

[5.ES6 46](#_Toc10049814)

# day01

学习一门编程语言的基本步骤

(1)了解背景知识：历史、现状、特点、应用场景

(2)搭建开发环境，编写hello world

(3)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)第三方库、框架

(10)实用的项目

## 1.JS概述

(1)历史

1995年，JS最早出现在Netscape公司的浏览器中;

1997年，出现JS的标准规范ECMAScript;

2009年，JS遵循CommonJS规范，开始向服务器端发展，出现nodejs。

(2)现状

既可以在客户端浏览器，也可以运行在服务器端

(3)特点

解释型语言，编译一行执行一行

弱类型语言

基于对象

跨平台

(4)应用场景

用于浏览器端的交互效果

用于服务器端的创建web服务器、操作数据库、操作文件...

## 2.搭建JS开发环境

(1)浏览器自带的JS解释器

(2)nodejs下的JS解释器(服务器端)

http://nodejs.org 下载地址，选左侧

在命令行cmd下 node -v 查看当前计算机的nodejs版本

(3)执行JS代码

浏览器：

创建01.js和01.html

在html文件中引入js脚本

<script src='01.js'></script>

nodejs:

node C:/xampp/.../01.js 回车

## 3.JS代码规范

(1)区分大小写

(2)每行代码结束的分号可加可不加，建议都加

(3)分为单行注释(//...)和多行注释(/\*...\*/)

## 4.变量

用于存储数据的容器

(1)声明变量

var a=1;

练习：声明多个变量，分别保存员工的编号，姓名，性别，年龄，工资，所属部门编号。

(2)声明多个变量

var x=1,y=2;

声明多个变量用逗号隔开

(3)变量的命名规则

变量名称由字母、数字、下划线、美元符号，组成，数字不能出现在开头

不能使用关键字和保留字作为变量名

username user\_name userName

(4)变量声明未赋值

结果是undefined 未定义

## 5.常量

一旦声明，就不允许重新赋值。

使用关键字const

const pi=3.14;

## 6.数据类型

分为原始类型和引用类型

原始类型分为数值型、字符串型、布尔型、未定义型、空

(1)数值型

分为整形和浮点型

整形占4个字节，浮点型占8个字节

8进制，以0开头，例如：012->10

1 2 3 ... 7 10 11 12

16进制

1 2 3 ... 9 a b c d e f 10

0xf 0xE 0xF0 0xFF

10 20 30 F0 FF

16 2\*16 3\*16 15\*16=240 240+15=255

课后任务：

(1)复习今天内容，整理思维导图

(2)练习

声明常量保存圆周率，声明变量保存圆的半径，声明变量计算圆

的面积和周长。

# day02

复习

复杂查询

聚合查询/分组查询

COUNT()/SUM()/MAX()/MIN()/AVG()

YEAR()/MONTH()

GROUP BY

子查询

多表查询

查询条件

内连接 t1 INNER JOIN t2 ON...

外连接 t1 LEFT OUTER JOIN t2 ON...

t1 RIGHT OUTER JOIN t2 ON...

全连接 UNION/UNION ALL

浏览器/服务器 nodejs

变量 var $\_a1=1;

常量 const pi=3.14;

日期时间存储的是 距离计算机元年的毫秒数

1秒=1000毫秒

1970年1月1日 0:0:0

2019-5-5 9:13:20

## 1.数据类型

原始类型和引用类型

### 1.原始类型

(1)数值型

整形和浮点型

8 16 10

314.15 31.415\*10 3.1414\*10^2

3.1514e2

(2)字符串型

数据被引号所包含就是字符串型，不区分单双引号

查看某个字符的Unicode码

'a'.charCodeAt() //97

(3)布尔型

在程序中表示真或者假的结果

true/false

常用语一些是否的结果，例如是否登录，是否注册，是否为会员，是否在售

var isLogin=false;

(4)未定义型

声明了变量未赋值，结果是undefined

(5)空

null，常用于引用类型

检测数据类型

typeof 数据

五种类型：

数值型：number

字符串型：string

布尔型：boolean

未定义型：undefined

空：object

### 2.数据类型转换

(1)隐式转换

①数值型+字符串型 数值型转成字符串型

1+'2' //'12'

②数值型+布尔型 布尔型转换成数值型 true->a false->0

1+true //2

③字符串型+布尔型 布尔型转成字符串型

'hello'+true //'hellotrue'

JS中加号(+)作用

执行加法运算

字符串的拼接

练习：声明变量保存姓名name，性别sex，年龄age；使用字符串拼接打印‘我的姓名(变量中的值),性别(变量中的值),我的年龄为(变量中的值)’。

练习：查看一下程序的执行结果

var a=3,b=true,c='tedu';

console.log(a+b+c);//'4tedu'

console.log(b+c+a);//'truetedu3'

console.log(c+a+b);//'tedu3true'

减法、乘法、除法隐式转换

将运算符两端的数据转成数值型，自动调用Number来转，如果字符串中含有非数字返回NaN(Not a Number),任何值和NaN执行减乘除都返回NaN

(2)强制转换

①将数据转为数值型

Number()

Number('9') //9

Number('2a') //NaN

Number('true') //1

Number(undefined) //NaN

Number(null) //0

②将数据转为整型

parseInt()

parseInt(2.5);//2

parseInt('2.9a');//2

parseInt('a2');//NaN

③将数据转为浮点型

parseFloat()

parseFloat(2.9a);//2.9

parseFloat('a2');//NaN

练习：将布尔型、未定义型、空转换为整型或者浮点型

parseInt和parseFloat在将布尔型、未定义型、空转数值的时候，都返回NaN

④将数值型和布尔型转成字符串型

toString()

var n=5;

n.toString() //'5'

### 3.运算符

由运算符连接操作的数据，所组成的形式就是表达式

(1)算数运算符

+ - \* / % ++ --

% 取余

++ 自增：在当前的基础之上加1

-- 自减：在当前的基础之上减1

console.log(num++ 先打印num的值，然后再执行自增)

console.log(++num) 先执行自增，然后打印num的值

练习：

var a=1;

console.log(a++ + ++a);

1 3

var b=2;

console.log(--b + b--);

1 1

(2)比较运算符

>= <= > < ==(等于) != ===(全等于) !===(不全等于)

返回布尔型(true/false)的值

== 只是比较两个值是否相等

=== 不仅比较值，还会比较数据类型

3>'10' //false

数字和字符串比较，字符串转为数值型

'3'>'10' //true

两个字符串比较，比较的是首个字符的Unicode码

3>NaN //false

3<NaN //false

3==NaN //false

NaN==NaN //false

NaN和任何值比较(> < >= <= == ===)都是false,包括自身

(3)逻辑运算符

并且&& 关联的两个条件都是true,结果才是true,否则false

或者|| 关联的两个条件都是false,结果才是false,否则true

非(反向)! !true->false !false->true

逻辑短路

&& 当第一个条件为false的时候，就不需要再执行第二个条件

|| 当第一个条件为true的时候，就不需要再执行第二个条件

注意：逻辑短路无需关注最终的结果是true还是false，重点看第二个表达式是否执行。

练习：查看以下程序是否会报错

var num=3;

num>5 && console.log(a);

num<1 || console.log(a);

(4)位运算符

在执行运算的时候，计算机会将数据转成二进制进行运算

1 10 11 100 101 110 111 1000 1001 1010 1110 1111

2 4 8

101101=100000+1000+100+1

32 8 4 1

0 1 2 3 4 5 6 7

1 2 4 8 16 32 64 128

25=16+8+1

10000+1000+1=11001

按位与(&):上下两位都是1，结果是1，否则为0

按位或(|):上下两位含有1，结果是1，否则为0

课后任务：

(1)复习，整理思维导图

(2)练习：声明变量保存一个年份，判断是否为闰年，结合着逻辑短路，如果是闰年，打印“闰年”。(普通闰年:能被4整除但不能被100整除,世纪闰年:能被400整除的为世纪闰年)。

var year=2012;

((year%4===0&&year%100!==0)||year%400===0) && console.log('闰年');

# day03

复习：

原始类型

数值型、字符串型、布尔型、未定义型undefined、空null

数据类型转换

隐式转换 字符串中加法运算 拼接

字符串+数值

数值+布尔型

字符串+布尔型

强制转换

Number()

parseInt('3a') //3

parseFloat('3.4a') //3.4

toString()

数据类型

1.算术运算符 + - \* / % ++ --

2.比较运算符 >= <= > < == != === !==

3.逻辑运算符 &&(逻辑与) ||(逻辑或) !(逻辑非)

4.位运算符 &(按位与) |(按位或)

程序员必做的50题

https://wenku.baidu.com/view/af66e2f14afe04a1b071de42.html

## 1.位运算符

按位与(&):上下两位都是1，结果是1，否则为0

按位或(|):上下两位含有1，结果是1，否则为0

按位异或(^):上下两位相同为0，不同为1

按位右移(>>):删除2进制的最后的相应位数

按位左移(<<):在2进制的末尾补相应的0,每移动一位增大一倍

## 2.赋值运算符

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

练习：声明一个变量为空字符串，声明多个变量分别保存员工的姓名，使用+=往后拼接员工的姓名。

练习：打八折，声明变量保存商品的价格，使用\*=实现打八折。

## 3.三目运算符

单目运算符：有一个数据和表达式组成

a++ a-- !a

双目运算符：有两个数据和运算符组成

1+1 3>2 && ||...

三目运算符：有三个数据和运算符组成

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

## 4.浏览器端函数

alert() 弹出警示框

prompt() 弹出提示框，需要使用变量来保存输入的值，这个值的数据类型是字符串型

练习：两次弹出提示框，输入数字，计算两次输入的和，并打印

程序=数据+算法

程序分为顺序执行、选择执行、循环执行

## 5.流程控制——选择执行

(1)if语句

if(条件表达式){

语句1;

}

语句2;

说明：如果if后的语句只有一行，则可以省略大括号

在条件表达式中，有一些值默认转成false

Undefined null 0 NaN

判断一个值是否为NaN

isNaN() 是->true 否->false

(2)if-else 语句

if(条件表达式){

语句1;

}

else{

语句2;

}

(3)if-else嵌套

if(条件表达式1){

语句1;

}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

}

(4)switch-case语句

是一种特殊的分置于句，可以根据一个表达式的不同值，来选择执行不同的语句

switch(表达式){

case 值1: //如果表达式的值为值1

语句1;

break;//结束，不会往后执行其它的语句

case 值2: //如果表达式的值为值2

语句2;

...

default:

语句n;//如果表达式的值和所有的值比较都是false

}

表达式在和case后的值比较的时候，使用的是全等于，要求值和类型都相等

对比if-else嵌套和switch-case的区别

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等的情况，适用范围更广泛；switch-case只能用于全等的比较，结构上更为清晰合理，执行效率相对高。

课后任务

(1)复习，整理思维导图

(2)练习：使用弹出提示框，输入商品的价格和数量并计算总价，如果满500打9折；使用变量保存余额,如果足以支付，打印"pay success",否则打印"pay error"。

var price=prompt("请输入价格：");

var number=prompt("请输入数量：");

total=price\*number;

var yue=500;

if(yue>=total){

console.log('pay success');

}else{

console.log('pay error');

}

# day04

复习

运算符

位运算符 & | ^ >> <<

赋值运算符 = += -= \*= /= %=

三目运算符 条件表达式?表达式1:表达式2

选择执行

if(条件表达式){语句1;}

if(条件表达式){语句1;}else{语句2;}

if(条件1){语句1;}else...if(条件n){语句n;}else{语句n+1;}

switch(表达式){

case 1:

语句1;

break;

default:

语句n;

}

## 1.循环执行

循环：一遍又一遍的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：重新执行的相同或者香色代码

### 1.while循环

(1)while(循环条件){//是一个布尔型的值

循环体

}

(2)break

while(true){//是一个布尔型的值

循环体

break;

}

在循环中，可以结束任何形式的循环

练习：使用无限循环计算1~10之间所有整数的乘积

练习：声明变量保存任意一个数字，使用无限循环的循环体中弹出提示框输入数字，如果输入的数字大于保存的数字，使用警示框弹出'big';如果小于保存的数字，弹出'small';否则弹出'right'，并结束循环。

### 2.do-while循环

do{

循环体

}while(循环条件);

无限循环

do{

循环体

}while(true);

练习：声明变量保存密码123456，(无限)循环弹出提示框输入密码，如果输入正确结束循环。

### 3.for循环

for(表达式1;表达式2;表达式3){

循环体;

}

表达式1:初始值

表达式2:循环条件

表达式3:增量

### 4.break和continue

break:结束循环，后续不会再执行其他的循环

continue:跳过本次循环，继续下一次循环

### 5.循环嵌套

任意循环之间可以相互嵌套

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

课后任务

(1)复习，整理思维导图

(2)练习：打印九九乘法表

打印本世纪(2000-2100)前10个闰年

计算1~100之间的和，当和大于4000的时候，提前结束循环，并打印当前的和是多少

# day05

复习

while(循环条件){ 循环体 };

do{ 循环体 }while(循环条件);

for(初始值;循环条件;增量){ 循环体 };

循环嵌套

break

continue

var i=0,sum=0;

do{

i++;

if(i%2===0) continue;

if(i%5===0) break;

sum+=i;

}while(i<10);

console.log(sum);

## 1.函数

parseInt()/parseFloat()/alert()/prompt()/isNaN...

函数分为系统函数和自定义函数

function:功能体，函数(方法)，可以接受若干个数据，返回处理的结果。用于封装反复执行的代码——饺子机。

(1)创建普通函数

function 函数名称(){

函数体——封装的代码

}

调用：

函数名称();

练习：创建函数，封装计算1-100之间所有整数的和并打印出现；调用3次。

练习：创建函数，封装两个数字相加的和；调用多次

(2)创建带有参数的函数

function 函数名称(参数列表){//用于接收传递的数据

函数体

}

调用：

函数名称(参数列表);//实际要传递的数据

参数：创建函数时的参数称为形参，调用时的参数称为实参，

同时实参会赋值给形参，多个参数之间用逗号隔开。参数可以有0个或者多个，如果形参未被赋值，值为undefined。

练习：创建函数，计算1~任意数字之间所有整数的和，调用多

次。

练习：创建函数，计算任意两个年份之间的闰年个数并打印，

调用多次。

(3)创建带有返回值的函数

function 函数名称(参数列表){

函数体

return 值;//返回值，就是函数调用后的结果

}

调用:

函数名称(参数列表);

注意事项：如果函数中没有return或者return后没有加任何的值，返回undefined。return 后的代码不会再执行。

练习：创建函数，传递两个参数，返回任意两个数字的最大值

练习：创建函数，传递订单的状态码这个参数，返回对应的中文状态。

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

练习：创建函数，返回任意三个数字的最大值 3 4 5

练习：创建函数，传递一个年份，要求返回是否为闰年，是->true,否则返回false

练习：创建函数，传递一个年份，如果是闰年返回366，不是闰年返回365

练习：计算1-任意数字之间所有整数阶乘的和

## 2.变量的作用域

(1)作用域

变量或者函数的可访问范围，分为两种

函数作用域：在函数中使用var声明的变量，只能在函数内部访问

全局作用域：在全局使用var声明的变量，可以再任意合法位置访问

注意：在函数内不适用var声明的变量，是全局变量，可以在函数外访问到。

(2)变量声明提升

JS程序在执行前，会将使用var声明的变量提升到所在作用域的最前边，但赋值还是在原来的位置。

## 3.函数作用域

函数的可访问范围，分为全局作用域和函数作用域;在全局作用域下创建的函数可以在任何作用域下调用，函数作用域下创建的函数，只能在这个函数内部调用。

函数提升

和变量提升一样，JS在执行前，会将使用function关键字创建的

函数提升到所在作用域的最前边。在调用的位置执行函数体中的代码。

## 4.递归

在函数内部调用自身，本身是一个死循环。

递归的正确用法：要有结束的条件，结合着return。

练习：使用递归计算1-任意数字之间所有整数的和。

斐波那契数列

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

从第3个数开始，每个数的值为前两项相加的和

课后任务：

(1)复习

(2)练习：使用递归计算斐波那契数列的第n项

使用循环计算斐波那契数列的第n项

12/30/50/100

# day06

\*\*\*《JavaScript高级程序设计》(第3版)

## 1.匿名函数

没有名称的函数 function(){ 函数体; }

(1)创建函数

函数声明创建

function fun(){ 函数体; }

函数表达式创建

var fn=function(形参){ 函数体; }

函数名称和函数名称()的区别

例如：fun/fn fun()/fn()

函数名称就是一个变量，对应的是函数的结构。

函数名称()是函数的调用，所得到的是函数的返回结果。

对比函数声明和函数表达式的区别：

函数声明创建的函数存在提升，可以在任意的合法位置调用。

函数表达式创建的函数只存在变量提升，必须先创建再调用。

练习：使用函数表达式来创建函数，计算任意两个数字之间所有整数的和，并返回结果。

(2)匿名函数的自调用

目的：创建局部作用域，防治污染全局

(function(形参列表){

//函数体中的变量和函数都不能被外部访问

})(实参列表);

(3)回调函数

将匿名函数以实参来传递，此时的形参就是匿名函数的函数名称

function fn(a){

//调用fn的时候，匿名函数赋给了a,a就是函数名称

a() //执行传递的匿名函数体中的代码

}

fn(function(){...});

练习：创建函数，传递两个参数，都是使用匿名函数的形式传递，在匿名函数的函数体重分别返回一个数字；计算两个数字相加的和。

## 2.全局函数

parseInt() //将数据转为整型

parseFloat() //将数据转为浮点型

encodeURI() //对URI中的汉字进行编码

decodeURI() //对已编码的URI进行解码

isNaN() //检测一个值是否为NaN，是返回true,不是返回false

isFinite() //检测一个值是否为有限值，是返回true,不是返回false

1/0 ->Infinity 无限值

'1+3' 'typeof 3'

eval() //执行字符串中的表达式

练习：弹出提示框，输入一组运算，使用eval来执行字符串中的表达式

## 3.对象

属于引用类型数据

对象：是一组属性(property)和方法(method)的集合

电脑：属性有屏幕尺寸、颜色、品牌、内存大小...方法有看视频、玩游戏、敲代码...

一个杯子：属性有颜色、品牌、容量、方法有喝水

万物皆对象

(1)JS中的对象

内置对象：JS提供的

宿主对象：根据不同的执行环境划分

自定义对象：自己创建的对象

(2)自定义对象

对象字面量

内置构造函数

自定义构造函数

(3)使用对象字面量来创建对象

使用大括号{}创建空对象

属性名和属性值之间用冒号隔开

多组属性之间用逗号隔开

属性名中引号可加可不加，如果含有特殊字符必须添加

练习：创建一个员工对象，包含编号、姓名、性别、生日、工资

练习：创建一个手机对象，包含颜色、品牌、尺寸、内存大小

(4)访问对象中的属性

对象.属性名

对象['属性名'] 如果属性名不加引号，会认为是变量

如果属性不存在，返回undefined

练习：创建图书的对象，包含图书的编号、标题、作者、价格、出版时间；修改图书的价格，访问图书的作者，添加图书的出版社属性

(5)使用内置构造函数创建对象

new Object(); 创建一个空对象

需要通过访问对象中属性来添加每一个属性

练习:创建一个汽车对象，包含品牌、型号、颜色、长度、宽度

(6)遍历对象中的属性

依次访问对象中的每一个属性

for(var key in 对象){

key 代表对象中的每一个属性名,

对象[key] 获取每个属性值

}

练习：创建对象，包含有若干个成绩，遍历对象中属性，获取总成绩

(7)对象中的方法

var person={

name:'Tom',

say:function(){

this.name;//指代当前所在的对象

}

}

person.say();

练习：创建一个手机对象，含有属性颜色、品牌、价格；添加打电话、发短信方法。

课后任务：

(1)复习，整理思维导图

(2)练习：创建一个圆对象，含有半径和圆周率属性，添加计算面积，计算周长的方法。

# day07

## 1.判断是否含有某个属性

对象.属性名===undefined 存在->false,不存在->true

对象.hasOwnProperty('属性名') 存在->true,不存在->false

'属性名' in 对象 存在->true,不存在->false

## 2.数组

数组是由多个元素组成的集合，每个元素就是一个数据

## 3.创建数组

(1)数组字面量

[元素1,元素2...]

(2)数组元素的访问

数组[下标] 下标是从0开始

练习：创建数组，包含若干个城市名称，修改某个元素，添加某个元素，最后打印结果。

(3)内置构造函数

new Array(元素1,元素2..)

new Array(3) 初始化数组的长度为3，可以添加更多个元素

练习：创建数组，包含若干个手机品牌名称

练习：创建数组，初始化长度为5，添加篮球场上的五个位置

## 4.数组的长度

数组.length 获取数组元素的个数

在数组的末尾添加新元素

数组[ 数组.length ]=值;

练习：创建一个空数组，使用数组的长度来添加多个国家的名称。

## 5.数组的分类

分为索引数组和关联数组

索引数组以0开始的整数作为下标

关联数组以字符串作为下标，关联数组中的元素只能单独添加。

练习：创建一个关联数组，包含有图书的编号、标题、作者以及价格

## 6.遍历数组中的元素

(1)for-in

for(var key in 数组){

key //要遍历的每个元素的下标

数组[key] //每个下标对应的元素

}

练习：创建索引数组，包含若干个学生的成绩，计算总成绩和平均成绩

(2)循环

for(var i=0;i<数组.length;i++){

i 表示每个元素的下标

数组[i] 表示下标对应的元素

}

只能遍历索引数组

练习：创建数组，包含多个名字，把姓名中所有的tom替换成'汤姆'

练习：创建数组，包含有多个姓名，查看tom出现的次数

练习：创建函数，传递一个参数(一组工资)，返回平均工资

练习：创建函数，传递两个参数(要查找的数组，要查找的值)，如果找到了返回该元素的下标，如果没有找到返回-1

findName(['kate','tom','jerry'],'tom')

练习：创建函数， 传递一个参数(数组中含有多个数字)，返回数字中的最大值

## 7.数组中的方法(api)

api 应用程序编程接口，预定义好的一些方法或者函数。

数组.toString();将数组中的元素按照逗号转为字符串。

join('-');将数组中的元素按照指定的字符来分割转为字符串。

数组1.concat(数组2,数组3,...);拼接多个数组。

数组.slice(start,end);截取数组中的元素，start是开始的下标 ，end是结尾的下标，不包含end本身，如果end为空会截取到最后，如果是负数，表示倒数。

练习1：创建数组a~g,每个字母是一个元素，分别截取cd,f,b,组成一个新数组。

数组.splice(start,count,val1,val2,...);删除数组中的元素 ，start是开始的下标，count是删除的长度，value1是删除后添加的元素，如果count为空删除到最后，start表示倒数。

练习2：创建数组a~h，每个字母是一个元素，删除d、e替换f为m,在下标为2的位置添加字母z。

数组.reverse();翻转数组中的元素。

数组.sort();对数组中的元素进行排序，默认按照Unicode码从

小到大排序。

对数字排序

sort(function(a,b){

return a-b;//从小到大排序

//return b-a;//从大到小

});

数组.push();在数组的末尾添加元素，返回数组的长度。

数组.pop();删除数组的最后一个元素，返回被删除得元素。

数组.unshift();在数组的开头添加元素，返回数组的长度。

数组.shift();删除数组的第一个元素，返回被删除的元素。

课后任务：

(1)复习，整理思维导图

(2)练习：使用冒泡排序对数组中的数字从小到大排序

使用数组遍历，将数组中的元素翻转

# day08

复习：

数组

数据的集合

创建 [] 内置构造函数new Array(3)

访问数组元素 下标 数组[下标]

添加元素 数组长度 数组[数组.length]

关联数组和索引数组

arr['name']='tom';

遍历

for-in 循环

数组的API

toString()/join/concat/slice/splice(start,count,v1,v2...)/reverse/sort(function(a,b){ return a-b; })

push/pop/unshift/shift

## 1.对象和数组之间的关系

练习：创建数组，包含若干个员工(姓名，性别，年龄)

练习：创建对象(某个学员)，学号、姓名、爱好(敲代码、看电影、旅游)

## 2.二维数组

数组中的每个元素也是数组

var arr=[[数组0],[数组1],[数组2]];

访问二维数组中的元素

arr[下标][下标]

## 3.字符串对象

包装对象：目的是为了让原始类型数据可以像引用类型数据一样，具有属性和方法。本质上还是原始类型。

JS包装对象有三种：String/Number/Boolean

将任意的类型转为字符串 String()

new String(true) 转为字符串，返回对象

String(true) 转为字符串，返回字符串

练习1：把数组转为字符串

字符串、数值、布尔型

对象有属性和方法

(1)转义字符 \

转换字符本身的意义

\n 将字符n转义成换行符号

\' 将具有特殊意义的单引号转成普通字符

练习2：在控制台下打印welcome to chi\na!

(2)api

字符串.toUpperCase();将英文字母转为大写。

字符串.toLowerCase();将英文字母转为小写。

字符串.length;查看字符长度。

字符串[下标];查看某个下标对应的字符。

字符串.charAt();获取下标对应的字符，等价于 字符串[下标]

字符串.indexOf(val,start);查找字符出现的下标，val要查找的

字符(串),start开始查找的下标，默认为0；如果找不到返回-1。

字符串.lastIndexOf(val);查找字符最后一次出现的下标，val要

查找的字符(串)，找不到返回-1。

练习：声明变量保存字符串，检测该字符串是否为邮箱格式，如果是打印true,否则打印false(hello@123.com)(04\_api.js)

练习：声明变量保存验证码(英文字母)，无限循环弹出提示框，出入验证码(不区分大小写)，如果输入正确结束循环(04\_api.js)

slice(start,end);截取字符串，start开始的下标,end结束的下标,不包含end本身，如果end为空，截取到最后，如果为负数，表示倒

数。

练习：声明变量保存邮箱，截取邮箱中的用户名和域名,先查找 @ 的

下标，再截取(hello1995@sina.com)(04\_api.js)

substr(start,count);截取字符串，start开始的下标，count截

取的长度，如果count为空，截取到最后，start为负数表示倒数。

练习：声明变量保存身份证号，截取身份证号中年月日

1997年01月02日 性别女 532222199701022122

substring(start,end);截取字符串，start开始的下标，end结束

的下标，如果end为空截取到最后，如果为负数，自动转成0，如果

start大于end，位置会自动交换

对比slice和substring

slice下标允许使用负数，sunstring下标为负数自动转成0；

slice中开始的下标小于结束的下标，substring下标不分顺序。

练习：声明变量保存字符串'heLLO',要求吧首字母转为大写，其余字

母转为小写。'Hello'(04\_api.js)

split(str);将字符串按照指定的字符str分割成数组，和数组下

的join方法是相对的。

练习：使用split方法截取邮箱的用户名和域

(hello1995@sina.com)(04\_api.js)

## 4.匹配模式

作用：用于查找、替换字符串

(1)match(value);用于查找匹配的字符串，返回所有满足条件的字符串，组成数组。/wenhua/ig,i-ignore忽略大小写,g-global全局查找

(2)search(value);用于查找满足条件的第一个字符串的下标，如果找不到返回-1;功能和indexOf一致。

(3)replace(value1,value2);用于查找并替换，value1是要查找的字符串，value2是要替换的字符串。

## 5.Math对象

实例化，创建对象

new Object()/new Array()/new String()

Math对象不需要是要new创建，可以直接使用

Math.PI;圆周率

Math.abs(3-10);绝对值

Math.floor(3.8);向下取整

Math.ceil(3.8);向上取整

Math.round(3.4);四舍五入取整

Math.max(val1,val2,val3,,...);取一组数字的最大值

Math.min(val1,val2,val3,,...);取一组数字的最小值

Math.pow(x,y);取x的y次幂 x\*\*y

Math.random(val1,val2);取随机数 >=0 <1

练习：随机产生0~9之间的一个整数

课后任务：

(1)复习，整理思维导图

(2)练习：将一句英文中每个单词的首字母转大写，其余转小写

hoW aRe yOU -> How Are You

随机取a~z,A~Z,0~9之间的4个字符，把取到的结果放入到一个新数组，不能重复splice

# day09

复习：

字符串操作

包装对象 new String(true) String()

## 1.Date对象

用于对日期时间进行存储和计算

(1)创建Date对象

new Date('2019/5/17 9:52:30');

new Date(2019,4,17,9,52,30);//第二个参数月份，范围0~11

new Date() 存储当前所在系统的日期时间

new Date(1000\*60\*60\*24) 存储的是距离计算机元年的毫秒数

存储距离计算机元年时间 1970-1-1 0:0:0

(2)获取Date对象中的日期时间

getFullYear(年)/getMonth(月范围0~11)/getDate(日)/getHours(时)/getMinutes(分)/getSeconds(秒)/getMilliseconds(毫秒)/getDay(星期)/getTime(距离计算机元年毫秒数)

练习：创建Date对象，保存'2019/6/7 10:25:40',打印2019年6月7日 10点25分40秒 星期五

(3)获取本地字符串格式

toLocaleString() 2019-5-17 14:18:30

toLocaleDateString() 2019-5-17

toLocaleTimeString() 14:18:30

(4)设置日期时间

setFullYear/setMonth/setDate/setHours/setMinutes/setSeconds/setMilliseconds/setTime

setTime以后，所有的日期时间可能会发生变化

(5)复制Date对象

var d1=new Date();

var d2=new Date(d1);//复制一个d1对象，d2是一个新对象

练习：创建Date对象,保存员工的入职时间'2019-9-30';3年后合同到期,计算到期时间,合同到期前1个月续签合同，假如是周末，提前到周五，计算续签时间。

## 2.Number对象

new Number(true) 将数据转为数值，返回对象

Number(true) 将数据转为数值，返回数值

Number.MAX\_VALUE

Number.MIN\_VALUE

toFixed(n) 取小数点后n位

toString(n) 将数值转为字符串，同时可以设置进制,n就是要设置的进制，可以使2,8,16...

## 3.Boolean对象

new Boolean() 将数据转为布尔型，返回对象

Boolean() 将数据转为布尔型，返回布尔型

练习：0 NaN"undefined null [] {}

!!数据 将数据转为布尔型

## 4.错误处理

SyntaxError 语法错误，错误的使用了中文，缺少括号等;出现后所有的代码都不执行

ReferenceError 引用错误，使用了未声明的变量，影响后续代码的执行

TypeError 类型错误，错误的使用了数据类型，影响后续代码的执行

RangeError 范围错误，参数的使用超出了范围，影响后续代码的执行

try{

尝试执行的代码，可能出现错误

}catch(err){

err 捕获到的错误

具体处理错误的内容

}

练习：声明一个变量add没有赋值，尝试调用add函数，传递两个参数(数字);如果执行错误，使用函数表达式创建函数,赋给add一个匿名函数，然后再调用add

## 5.ES6

ECMAScript6

http://es6.ruanyifeng.com/ es6电子书

(1)块级作用域

使用let关键字声明的变量，只能在块级作用域下使用，不能被外部访问，不存在

变量的提升。

块级作用域：if if-else while do-while for switch 都是块级作用域

练习：打印1~10之间所有的整数，初始化变量使用let,在循环结束后，打印初始化的变量。

(2)箭头函数

是回调函数的另一种写法，和匿名函数不完全一样

sort( (a,b)=>{

return a-b;

} );

如果箭头函数的函数体中只有一行代码，并且是return形式，可以简化为 sort((a,b)=>a-b)

练习：创建函数add,传递两个参数，每个参数是回调函数，在回调函数的函数体中返回一个数字；在函数add中计算两个数字相加的结果并打印出来。

课后任务：

(1)复习，整理思维导图

(2)练习：计算2019/6/7 9:25:34 距离2019/10/1 19:52:48相差的天，小时，分钟，秒钟

双色球