一、函数

1） 作用

封装一段待执行的代码

2）语法

```
//函数声明
function 函数名(参数列表){
  函数体
  return 返回值;
}
//函数调用
函数名(参数列表);
```

3）使用

函数名自定义，见名知意，命名规范参照变量的命名规范。普通函数以小写字母开头，用于区分构造函数(构造函数使用大写字母开头，定义类)

4）匿名函数

匿名函数：省略函数名的函数。语法为：

- 匿名函数自执行

```
(function (形参){

})(实参);
```

- 定义变量接收匿名函数

```
var fn = function (){};
fn(); //函数调用
```

```
<script>
        //匿名函数的使用,自执行: (匿名函数的声明)(实参)
        (function (){
            //构建一个局部作用域,主要从内存释放的角度,比如某个变量只使用一次
            document.write("匿名函数自执行");
        })();
 </script>
```

```
//定义一个变量接受匿名函数
var fn=function(){
document.write("fn被调用"+ "<br>");
}
fn();
document.write(fn);
//fn被调用
//function(){ document.write("fn被调用"+ "
//"); }
```

```
//函数作为参数传递
function show(f)
{
document.write("show:",f);
document.write("<br>")
f();
}
show(fn);//直接传递函数,实际上是用变量来传输函数地址
//匿名函数作为参数传递
show(function ()
{
document.write("<br>")
document.write("直接传递匿名函数");
})
```

5）作用域

JS中作用域分为全局作用域和函数作用域，以函数的{ }作为划分作用域的依据

1. 全局变量和全局函数

- 只要在函数外部使用var关键字定义的变量,或函数都是全局变量和全局函数,在任何地方都可以访问
- 所有省略var关键字定义的变量,一律是全局变量

2. 局部变量/局部函数
   - 在函数内部使用var关键字定义的变量为局部变量,函数内部定义的函数也为局部函数,只能在当前作用域中使用,外界无法访问
3. 作用域链 局部作用域中访问变量或函数,首先从当前作用域中查找,当前作用域中没有的话,向上级作用域中查找,直至全局作用域

二、 内置对象

1） 对象

对象是由属性和方法组成的,使用点语法访问

2） Array 数组

1. 创建

```
<script>
        //1.数组的创建
        var arr1 = ["超哥哥",30,true];
        console.log(arr1);
        //使用new关键字创建
        var arr2 = new Array("小泽",31,false);
        console.log(arr2);
        //特殊
        var arr3 = [5];
        //使用new关键字传递一个整数参数创建数组,代表初始化数组长度
        var arr4 = new Array(5);
        console.log(arr3,arr4);
        //
        //2.操作数组元素(根据元素索引)
        console.log(arr3[0]);
        arr4[0] = "老祁";
        arr4[2] = "大旭";
        arr4[8] = "AI";
        console.log(arr4);
        //3.数组对象的属性:length
        console.log(arr4.length);
        //4.遍历数组元素
        //创建数组,包含若干元素,实现数组元素的正序遍历和倒序遍历
        var arr5 = [1,2,3,4,5];
        //普通for循环
        for(var i = 0;i < arr5.length;i++){
            console.log(arr5[i]);
        }
        for(var i = arr5.length-1;i >= 0;i--){
            console.log(arr5[i]);
        }
        console.log("-------------------------");
```

```
        //for-in循环
        for( var i in arr5){
            console.log(i,arr5[i]);
        }
        //数组对象提供的遍历方法,forEach(function (){})
        arr5.forEach(function (elem,index){
            console.log("-----:",elem);
        });
        /*
        1.循环接收用户输入,将数据存储在数组中,直至输入"exit"结束,
        控制台打印数组
        2.声明包含若干元素的数组,打印数组中的最大值

        */
```

2. 特点

- 数组用于存储若干数据,自动为每位数据分配下标,从0开始
- 数组中的元素不限数据类型,长度可以动态调整
- 动态操作数组元素： 根据元素下标读取或修改数组元素， arr[index]

```
<script>
        function saveData(){
            var arr = [];
            while(true){
                var input = prompt("请输入");
                if(input == "exit"){
                    break;
                }
                //添加至数组,数组长度表示数组中下一个元素的索引
                arr[arr.length] = input;
            }
            console.log(arr);
        }
        //遍历数组取最大值
        function showMax(){
            //数据源
            var arr = [10,5,66,78,9,56,3,-90,0];
            var max = arr[0];
            for(var i = 0;i < arr.length;i++){
                if(arr[i] > max){
                    max = arr[i];
                }
            }
            console.log(max);
        }
        //showMax();
        /*定义数组,接收用户输入的内容,查询数组中是否存在相应元素
        如果存在返回对应元素的下标,不存在返回-1*/
        function findIndex(){
            //数据源
```

```
            var arr = [10,5,66,10,78,9,56,3,-90,0,10];
            //var arrInd = [];
            var data = prompt("请输入要查找的数值");
            var index = -1;
            /*
            //查找元素第一次出现的下标
            for(var i = 0;i < arr.length;i++){
                if(arr[i] == data){
                    index = i;
                    break;
                }
            }
            */
            //查找元素最后一次出现的下标
            for(var i = arr.length-1;i >= 0;i--){
                if(arr[i] == data){
                    index = i;
                    break;
                }
            }
            console.log(data,index,arr);
        }
        findIndex();

    </script>
```

3. 属性和方法

1. 属性 : length 表示数组长度,可读可写
2. 方法 :
    - push(data) 在数组的末尾添加一个或多个元素,多个元素之间使用逗号隔开 返回添加之后的数组长度
    - pop() 移除末尾元素 返回被移除的元素
    - unshift(data) 在数组的头部添加一个或多个元素 返回添加之后的数组长度
    - shift() 移除数组的第一个元素 返回被移除的元素
    - toString() 将数组转换成字符串类型 返回字符串结果
    - join(param) 将数组转换成字符串,可以指定元素之间的连接符,如果参数省略,默认按照逗号连接 返回字符串
    - reverse() 反转数组,倒序重排 返回重排的数组,注意该方法直接修改原数组的结构
    - sort() 对数组中元素排序,默认按照Unicode编码升序排列 返回重排后的数组,直接修改原有数组 参数 : 可选,自定义排序算法 例:

        ```
        //自定义升序
        function sortASC(a,b){
          return a-b;
        }
        ```

作用：作为参数传递到sort()中,会自动传入两个元素进行比较,如果a-b>0,交换元素的值,自定义升序排列

```
//自定义降序
function sortDESC(a,b){
  return b-a;
}
//如果返回值>0,交换元素的值,b-a表示降序排列
```

```
var array=[
         {uname:"daxu",age:35},
         {uname:"weichao",age:33},
            {uname:"xiaozi",age:31}
        ]
        console.log(array)
        array.sort
        (
            function (a,b)
            {
                //a,b代表数组中相邻的两个元素,如果a-b>0,则交换元素的位置
                return a.age-b.age;
            }
        );
        console.log(array)
```

- forEach(param) 遍历数组元素 参数为函数 例：

```
arr.forEach(function (elem,index){
    //forEach()方法会自动调用匿名函数,依次传入元素及下标
});
```

```
var arr5=[1,2,3,4];
n = arr5.push(5,6,7);

document.write(n +"<br>")
/*
//在数组的末尾添加一个或多个元素,多个元素之间使用逗号隔开
//返回添加之后的数组长度
//数组对象提供的遍历方法forEach(function (){}),回调函数
arr5.forEach(function (elem,index){
document.write(elem,index,"<br>");
});
```

```
<script>
        //1.操作数组的头尾元素
        var arr = ["hello","world"];
        var r1 = arr.push(1,2,3);
        console.log(r1,arr);
        var r2 = arr.pop();
```

```
console.log(r2,arr);
var r3 = arr.unshift(10);
r3 = arr.unshift(20,30);
console.log(r3,arr);
var r4 = arr.shift();
console.log(r4,arr);
//2.转换字符串
var r5 = arr.toString();
var r6 = arr.join("");
console.log(r5);
console.log(r6);
console.log(arr);
//3.数组反转(倒序重排)
arr.reverse();//调整原有数组的结构
console.log(arr);
//4.数组排序(对数组原有结构的调整)
//默认按照字符的Unicode编码升序排列
arr.sort();
console.log(arr);
//对number值进行大小排列
arr.sort(function (a,b){
    /*a,b代表数组中相邻的两个元素,如果a-b>0,则交换
    元素的位置*/
    //自定义升序
    return a-b;
})
console.log(arr);
//自定义降序
function sortDESC(a,b){
    //如果b-a>0,则交换两个元素的位置
    return b-a;
}
arr.sort(sortDESC);
console.log(arr);
/*复杂数组*/
var obj = {
    uid:001,
    uname:"大旭",
    play:function (){
        console.log("play");
    }
};
console.log(obj.uid);
obj.play();
var array = [
{uname:"大旭",age:35},
{uname:"超哥哥",age:33},
{uname:"小泽",age:31}
];
console.log(array[0].age);
//将数组元素按照年龄升序排列
array.sort(function (e1,e2){
    return e1.age-e2.age;
```

```
})
console.log(array);
/*代码实现十进制转二进制:
除2取余,直至商为0;余数倒序排列
6 / 2 = 3 ... 0
3 / 2 = 1 ... 1
1 / 2 = 0 ... 1
*/
function decode(){
    var num = prompt("请输入数字");
    var arr = [];
    while(num != 0){
        arr.unshift(num % 2);
        num = parseInt(num / 2);
    }
    console.log(arr.join(""));
}
decode();

//二维数组
var arr = [
[1],
[2,3],
[4,5,6]
];
console.log(arr[2][1]);
//二维数组的遍历
for(var i in arr){
    for(var j in arr[i]){
        console.log(arr[i][j]);
    }
}
```

4. 二维数组

数组中的每个元素又是数组

```
var arr1 = [1,2,3];
var arr2 = [[1,2],[3,4],[5,6,7]];
//操作数组元素
var r1 = arr2[0] //内层数组
var num = r1[0]; //值 1
//简写
var num2 = arr2[1][0];
```

3）String 对象

1. 创建

```
    var str = "100";
    var str2 = new String("hello");
```

2. 特点

字符串采用数组结构存储每位字符,自动为字符分配下标,从0开始

3. 属性

length： 获取字符串长度

4. 方法

- 转换字母大小写 toUpperCase() 转大写字母 toLowerCase() 转小写字母 返回转换后的字符串,不影响原始字符串
- 获取字符或字符编码 charAt(index) 获取指定下标的字符 charCodeAt(index) 获取指定下标的字符编码 参数为指定的下标,可以省略,默认为0
- 获取指定字符的下标
    - indexOf(str,fromIndex) 作用 : 获取指定字符的下标,从前向后查询,找到即返回 参数 : str 表示要查找的字符串,必填 fromIndex 表示起始下标,默认为0 返回 : 返回指定字符的下标,查找失败返回-1
    - lastIndexOf(str,fromIndex) 作用 : 获取指定字符最后一次出现的下标,从后向前查找,找到即返回 参数 : str 必填,表示要查找的内容 fromIndex 选填,指定起始下标
- 截取字符串 substring(startIndex,endIndex) 作用 : 根据指定的下标范围截取字符串,startIndex ~ endIndex-1 参数 : startIndex 表示起始下标 endIndex 表示结束下标,可以省略,省略表示截止末尾
- 分割字符串 split(param) 作用 : 将字符串按照指定的字符进行分割,以数组形式返回分割结果 参数 : 指定分隔符,必须是字符串中存在的字符,如果字符串中不存在,分割失败,仍然返回数组
- 模式匹配 作用 : 借助正则表达式实现字符串中固定格式内容的查找和替换 正则表达式 : var reg1 = /字符模式/修饰符; 修饰符 : i : ignorecase 忽略大小写 g : global 全局范围 字符串方法 :
    - match(regExp/subStr) 作用 : 查找字符串中满足正则格式或满足指定字符串的内容 返回 : 数组,存放查找结果
    - replace(regExp/subStr,newStr) 作用 : 根据正则表达式或字符串查找相关内容并进行替换 返回 : 替换后的字符串,不影响原始字符串

```
<script>
        //1.创建
        var s1 = "小泽";
        var s2 = new String("大旭老师");
        console.log(s1,s2);
        //2.字符串按照数组结构存储,可以使用数组方法操作字符
        console.log(s1[1],s2[0]);
        //遍历字符串
        for(var i = 0;i < s2.length;i++){
            console.log(s2[i]);
        }
        //3.字符串方法
        //3.1转换大小写字母
        var s3 = "Maria";
        var r1 = s3.toUpperCase();
        var r2 = s3.toLowerCase();
```

```
        console.log(r1,r2,s3);
        //3.2获取指定下标对应的字符或字符编码
        var r3 = s3.charAt();
        var r4 = s3.charCodeAt();
        console.log(r3,r4);
        //3.3获取指定字符的下标
        //"Maria"
        var r5 = s3.indexOf("a",2);
        var r6 = s3.lastIndexOf("b",2);
        console.log(r5,r6);
        //3.4截取字符串substring(start,end)
        //截取范围start~end-1
        var r7 = s3.substring(0,2);
        console.log(r7);
        //3.5分割字符串
        var r8 = s3.split("/");
        console.log(r8);
        //3.6模式匹配
        var str = "上知乎,搜知乎,问知乎,答知乎";
        //match(param),返回结果数组
        var res1 = str.match("知乎");
        console.log(res1);
        //定义正则,表示查找内容
        var regexp = /\d{6,10}/ig;
        var reg2 = /知乎/g;
        var res2 = str.match(reg2);
        console.log(res2);
        //replace()查找并替换,返回替换后的字符串结果
        var res3 = str.replace(reg2,"新浪");
        console.log(res3,str);
```

```
/*
        1.接收用户输入的邮箱,提取用户名和服务商
            "zhangsan@163.com"
        2.从身份证号中提取年月日信息
            "100100190012120123"
        3."101_5&201_7&301_9"
            商品id为101,数量为5
        */
        //1.提取用户名和服务商
        var mail = "lvze@tedu.cn";
        //方法1:截取
        var index = mail.indexOf("@");
        var username = mail.substring(0,index);
        //省略结束下标,表示截取至末尾
        var servername = mail.substring(index+1);
        console.log(username,servername);

        //方法2:使用@符号分割字符串
        var res = mail.split("@");
        console.log(res[0],res[1]);

        //2.提取身份证号中的年月日
```

```
var uid = "1001001900121000123";
var year = uid.substring(6,10);
var month = uid.substring(10,12);
var day = uid.substring(12,14);
console.log(year,month,day);

//3.分割商品信息
var str = "101_5&201_7&301_9";
var arr = str.split('&');//["101_5","",""]
for(var i=0; i<arr.length;i++){
    var arr2 = arr[i].split('_');//["101","5"]
    console.log("商品id为:",arr2[0],"商品数量为:",arr2[1]);
}
```