2016.6.11

一:

<title>运算符测试.html</title>

<script type="text/javascript">

function test1()

{

var i=1;

var j=2;

var k=3;

var rs1= i+j;

alert(i+"+"+j+"="+rs1);

alert(i+"-"+j+"="+(i-j));

alert(i+"\*"+j+"="+(i\*j));

alert(i+"/"+j+"="+(i/j));

alert(k+"%"+j+"="+(k%j));

//字符拼接符 + 在表达式中若含有字符串,+将表示拼接符

alert("AAA"+i+j);

}

function test2()

{

var i=4;

var j=2;

var k="4";

var rs1=i>j;

alert(i+">"+j+":"+rs1);

alert(i+"<"+j+":"+(i<j));

alert(i+">="+j+":"+(i>=j));

alert(i+"<="+j+":"+(i<=j));

alert(i+"=="+j+":"+(i==j));

alert(i+"=="+k+":"+(i==k));

alert(i+"==="+k+":"+(i===k));

}

function test3()

{

var i=4;

var j;

//将变量i的值 赋给 变量 j

j=i;

alert("i="+i+" j="+j);

var k=1;

k+=22; //实质为 k= k+22;

alert("k="+k);

//同理 -= \*= /= %=

}

function test4()

{

var i=1;

var j=2;

var k=3;

//且: 前后两个条件必须都为true时,整体为 true. 否则为false.

var rs1= j>i && j>k;

alert("rs1:"+rs1);

var rs2= j>i && k>i;

alert("rs2:"+rs2);

//或: 前后两个条件至少有一个为true时,整体为 true. 否则为false.

var rs3= j>i || j>k;

alert("rs3:"+rs3);

var rs4= i>j || i>k;

alert("rs4:"+rs4);

//非: 当条件为true时,整体为false. 否则为true.

//i不大于 j

var rs5=!(i>j);

alert("rs5:"+rs5);

}

function test5()

{

var sex="女";

var result= sex=="男" ? "帅哥,你好":"美女,你好";

var result2= false ? "AAAA":"BBB";

alert(result2);

}

</script>

</head>

<body>

<!--

算术运算符: 主要用于计算两个数据的结果

常用的符号: + - \* / %[求余]

关系运算符: 用于比较两个数据的大小关系

常用的符号: > < >= <= ==[只比较字面值] != ===[比较字面值和类型] !==

赋值运算符: 用于将数据 传递给另一个变量

常见的符号: = += -= \*= /= %=

逻辑运算符: 用于表示两个条件的满足关系

常见的符号: &&[且] ||[或] !{非 取反}

三目运算符:

格式: 条件 ? 代码1:代码2;

原理: 当条件为true时,将执行代码1. 否则执行代码2.

-->

<input type="button" value="算术运算" onclick="test1()">

<input type="button" value="关系运算" onclick="test2()">

<input type="button" value="赋值运算" onclick="test3()">

<input type="button" value="逻辑运算" onclick="test4()">

<input type="button" value="三元运算" onclick="test5()">

</body>

</html>

二:

<script type="text/javascript">

function test1()

{

var all=new Array(4);

//查询数组: 数组名[索引]

//修改/设置数组: 数组名[索引]=值

all[0]=23;

all[1]="张三";

all[2]="男";

all[3]="上海";

//删除数组: 数组名[索引]=null;

all[3]=null;

alert(all[0]);

alert(all[1]);

alert(all[2]);

alert(all[3]);

//验证元素个数有效性

all[6]="新数据";

alert(all);

}

function test2()

{

var all=new Array();

all[0]="AAA";

all[1]="BBB";

all[2]="CCC";

alert(all);

}

function test3()

{

var all=new Array("zhangsan","lisi","xiaoli","tom");

alert(all);

}

function test4()

{

var all=[23,34,56,"AA"];

all[4]="BB";

alert(all);

}

</script>

</head>

<body>

<!--

为什么使用数组?

张三 男 郑州, 李四 男 大同, 王五 女 北京, 赵六 女 上海. 属于一个班级,请使用合适的容器存放他们.

以上需求需要同一个容器存放多个数据,JS中提供了数组解决.

数组: 可以同时存放多个不同类型的数据.

数组使用:

A. var 数组名=new Array(元素个数);

说明:

1.JS数组中元素个数的指定没有任何意义.

2.JS数组可以动态改变元素个数,若没有设置元素的值,新增的元素之为"".

B. var 数组名=new Array();

说明:

1.JS元素个数没有意义,JS支持动态增长.

C. var 数组名=new Array(元素1,元素2,..);

说明:

1.以上方式称为动态初始化.

D. var 数组名=[元素1,元素2,..];

-->

<!-- 普通方式: 指定元素个数 -->

<input type="button" value="确定1" onclick="test1()">

<!-- 普通方式: 不指定元素个数 -->

<input type="button" value="确定2" onclick="test2()">

<!-- 创建并指定各元素值 -->

<input type="button" value="确定3" onclick="test3()">

<!-- 数组直接量 -->

<input type="button" value="确定4" onclick="test4()">

</body>

</html>

三:

<title>数据传递方式</title>

<script type="text/javascript">

function test1(){

var i=1;

var j;

//将变量i的值 赋给变量 j

j=i;

alert("i="+i+" j="+j);

//修改变量j的值

j=-1;

alert("i="+i+" j="+j);

}

function test2()

{

var nums1=[23,45,68,21];

var nums2;

//将变量nums1 赋给 变量nums2

nums2=nums1;

alert("nums1: "+nums1+" nums2: "+nums2);

//修改nums的元素

nums2[0]=-1;

alert("nums1: "+nums1+" nums2: "+nums2);

}

</script>

</head>

<body>

<!--

JS的数据类型: 基本数据类型[number,string,boolean,undefined] 引用数据类型[object(数组,对象)]

基本数据类型都属于值传递.

引用数据类型都属于址传递.

址传递和值传递区别:

1.值传递传递的是变量含有的具体值[普通数据]

2.址传递传递的是变量含有的内存地址.

3.无论址传递或值传递,传递的都是一种数据.

-->

<!-- 测试值传递 -->

<input type="button" value="确定1" onclick="test1()">

<!-- 测试址传递 -->

<input type="button" value="确定2" onclick="test2()">

</body>

</html>