JavaScript 基础第三天

循环嵌套、数组（重点）

黑马程序员
www.itheima.com
传智教育旗下
高端IT教育品牌

今日单词

模块	说明	单词	作用
数组	数组长度	数组.length	得到数组的长度
	新增数组元素	数组.push(新增元素)	数组末尾新增元素
		数组.unshift(新增元素)	数组开头新增元素
	删除数组元素	数组.pop()	删除数组最后一个元素
		数组.shift()	删除第一个数组元素
	新增/删除数组元素	数组.splice()	从指定位置删除/新增数组元素
	数组排序	数组.sort()	对数组进行升序/降序排序

目录

Contents

- 循环嵌套
- 数组
- 综合案例

01 循环嵌套

1. 循环嵌套

- 循环嵌套：一个循环语句里面又包含另一个循环语句

```
// 要重复执行的代码
```

1. 循环嵌套

- 循环嵌套：一个循环语句里面又包含另一个循环语句

```
for（初始化；循环条件；变量计数）
{
    //  要重复执行的代码
}
```

1. 循环嵌套

- 循环嵌套：一个循环语句里面又包含另一个循环语句

```
for（初始化；循环条件;变量计数）{
        for（初始化；循环条件；变量计数）
{
        //  要重复执行的代码
    }
}
```

假如每天记住5个单词，连续记3天

代码如何实现？

```
第1天
记住第1个单词
记住第2个单词
记住第3个单词
记住第4个单词
记住第5个单词
第2天
记住第1个单词
记住第2个单词
记住第3个单词
记住第4个单词
记住第5个单词
第3天
记住第1个单词
记住第2个单词
记住第3个单词
记住第4个单词
记住第5个单词
```

1. 循环嵌套

- 循环嵌套: 一个循环语句里面又包含另一个循环语句，一般出现于for循环中

```
for（初始化；循环条件;变量计数）  {
        for（初始化；循环条件；变量计数）
{
      //  码要重复执行的代码
    }
 }
```

- **嵌套循环特点：**

➢ 外部循环每循环一次，内部循环执行所有次

练习

- **打印5行5列的星星**

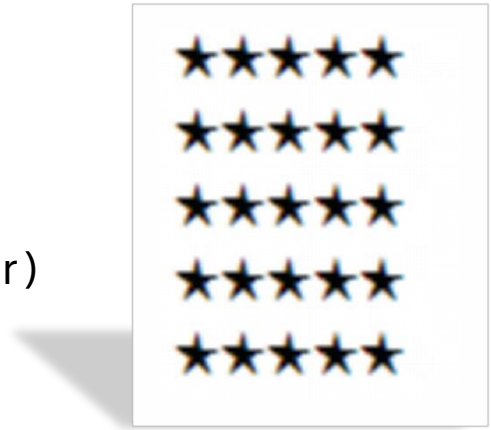
需求：　页面中打印出5行5列的星星

分析：

①：利用双重for循环来做

②：先利用for打印一行5个小星星，再把这个for重复5次（外层for）

总结：内层循环控制每行打印几个（列），外层循环打印行数



升级版本：

用户输入行数和列数，打印对应的星星！

练习

• 打印三角形星星

```
★
★★
★★★
★★★★
★★★★★
```

需求：如图所示

分析：

①：利用双重for循环来做

②：外层循环控制打印行，内层循环控制每行打印个数（列）

③：外层循环的行号和内层循环的个数是一一对应的(每列的个数等于行数)

外层(行) i	里层(个数) j
i = 1	j 打印1个星星
i = 2	j 打印2个星星
i = 3	j 打印3个星星
i = 4	j 打印4个星星
i = 5	j 打印5个星星

练习

- **九九乘法表**

需求：如图所示

分析：

①：只需要把刚才三角形星星做改动

②： ★ 换成 1 x 2 = 2 格式

1×1=1								
1×2=2	2×2=4							
1×3=3	2×3=6	3×3=9						
1×4=4	2×4=8	3×4=12	4×4=16					
1×5=5	2×5=10	3×5=15	4×5=20	5×5=25				
1×6=6	2×6=12	3×6=18	4×6=24	5×6=30	6×6=36			
1×7=7	2×7=14	3×7=21	4×7=28	5×7=35	6×7=42	7×7=49		
1×8=8	2×8=16	3×8=24	4×8=32	5×8=40	6×8=48	7×8=56	8×8=64	
1×9=9	2×9=18	3×9=27	4×9=36	5×9=45	6×9=54	7×9=63	8×9=72	9×9=81

★
★★
★★★
★★★★
★★★★★

目录

Contents

- 循环嵌套
- 数组
- 综合案例

02

数组

- 数组的基本使用
- 操作数组
- 数组案例

2. 数组(Array) - 重点

数组：(Array) 是一种**数据类型**，属于引用数据类型

作用：在单个变量名下存储多个数据

例如我要保存咱们班同学的姓名：

```
let name1 = '小明'
let name2 = '小刚'
let name3 = '小红'
let name4 = '小丽'
let name5 = '小米'
```

```
let names = ['小明', '小刚', '小红', '小丽', '小米']
```

2.1 数组的基本使用

1. 声明语法

```
let 数组名 = [数据1, 数据2, ..., 数据n]
```

```
let names = ['小明', '小刚', '小红', '小丽', '小米']
```

- **注意事项：**
  - 数组是按顺序保存（有序），所以每个数据都有自己的编号
  - 编号从 0 开始，数据的编号经常称为 **索引或下标**
  - 数组可以存储任意类型的数据

2.1 数组的基本使用

2. 取值语法

```
数组名[下标]
```

```
let names = ['小明', '小刚', '小红', '小丽', '小米']
names[0]  // 小明
names[1]  // 小刚
```

总结

1. 数组的作用是什么？
   - 在单个变量名下存储多个数据
   - 数组也是一种数据类型，属于引用数据类型
2. 数组如何声明的？

```
let names = ['小明', '小刚', '小红', '小丽', '小米']
```

   - 数组有序的，索引号从 0 开始
   - 数组里面可以存放任意数据类型
3. 数组如何取值？
   - 数组名[索引]

```
let names = ['小明', '小刚', '小红', '小丽', '小米']
names[0]  // 小明
names[1]  // 小刚
```

练习

数组练习

需求：

①：请声明一个数组，里面包含4个数据：'迪丽热巴','古力娜扎','佟丽丫丫','玛尔扎哈'

②：请依次将4个数据控制台打印输出

2.2 数组的基本使用

3. 遍历数组（重点）

- **遍历数组：**把数组中每个数据都访问到
- **使用场景：**

```
let arr = ['迪丽热巴', '古力娜扎', '佟丽丫丫', '玛尔扎哈']
console.log(arr[0])
console.log(arr[1])
console.log(arr[2])
console.log(arr[3])
```

- 我们可以使用for循环来遍历数组，其实遍历数组也是for循环重要的使用场景

2.2 数组的基本使用

3. 遍历数组（重点）

数组长度	数组中数据的个数，通过数组.length属性获得
数组元素	数组中数据，也称为数组元素

```
let arr = ['迪丽热巴', '古力娜扎', '佟丽丫丫', '玛尔扎哈']
for (let i = 0; i < arr.length; i++) {
  console.log(arr[i])
}
```

长度

案例

数组求和

需求：求数组 (arr) [2, 6, 1, 7, 4] 里面所有元素的和以及平均值

核心：

2	+	6	+	1	+	7	+	4
arr[0]		arr[1]		arr[2]		arr[3]		arr[4]

案例

数组求和

需求：求数组 [2, 6, 1, 7, 4] 里面所有元素的和以及平均值

分析：

①：声明一个求和变量 sum，一个平均值变量 average

②：遍历这个数组，把里面每个数组元素加到 sum 里面

③：用求和变量 sum 除以数组的长度就可以得到数组的平均值 average

案例

数组求最大值

需求：求数组 [2, 6, 1，25, 7] 中的最大值

分析：

①：声明一个保存最大元素的变量 max，默认最大值可以取数组中的第一个元素

②：遍历这个数组，把里面每个数组元素和 max 相比较，把比较的最大的放入max变量里面

③：最后输出这个 max 就是最大值

拓展：

自己求一下最小值

02

数组

- 数组的基本使用
- 操作数组
- 数组案例

2.2 操作数组

- 数组本质是数据集合，操作数据无非就是 增 删 改 查 语法：

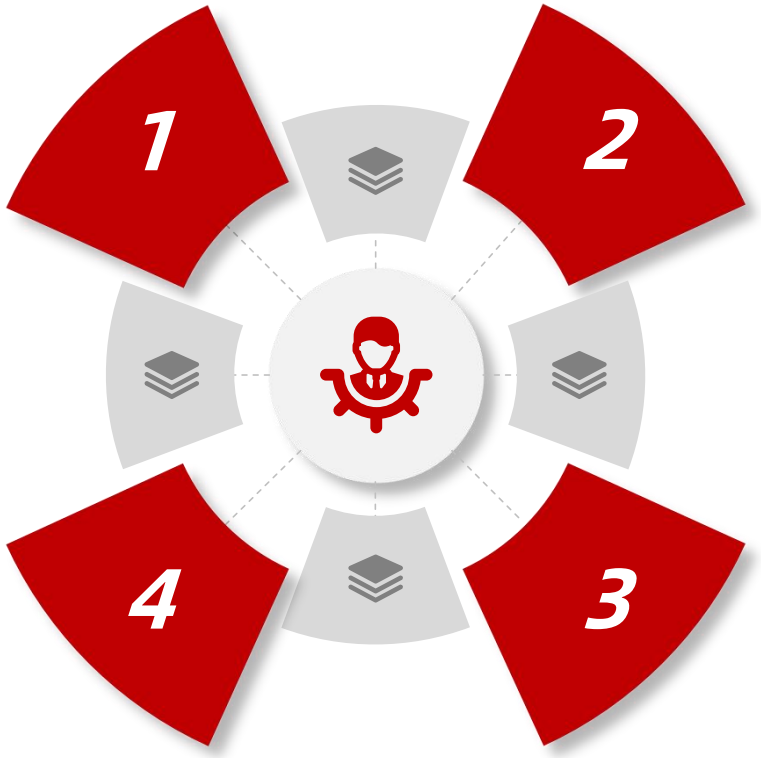
查询数组元素

数组[下标]

或者我们称为访问数组元素

修改数组元素

数组[下标] = 新值



增加数组元素

arr.push(新增的内容)

arr.unshift(新增的内容)

删除数组元素

arr.pop()

arr.shift()

2.2 操作数组 - 查和改

- **查询语法：**
  - 数组[索引]
  - 返回值：如果查询不到则返回 undefined

- **修改语法：**
  - 数组[索引] = 新值
  - 返回值：如果下标不存在，则是新增一个数组元素，并修改了数组长度（尽量避免）

2.2 操作数组 - 新增元素

新增

数组.push(新增数据)

作用：将一个或多个元素添加到数组的末尾**（重点）**

返回值：该数组的新长度，会修改原数组

```
let arr = ['red', 'green']
arr.push('pink')
console.log(arr)  //  ['red', 'green', 'pink']
```

数组.unshift(新增数据)

作用：将一个或多个元素添加到数组的开头

返回值：该数组的新长度，会修改原数组

```
let arr = ['red', 'green']
arr.unshift('pink')
console.log(arr)  //  ['pink', 'red', 'green']
```

总结

1. 想要数组末尾增加数据元素利用哪个方法？
   - arr.push(新增内容)
   - 可以添加一个或者多个数组元素
   - 返回的是数组新长度
2. 想要数组开头增加数据元素利用哪个方法？
   - arr.unshift(新增内容)
   - 可以添加一个或者多个数组元素
   - 返回的是数组长度
3. 两种方法会修改原数组吗？
   - 会的

案例

数组筛选

需求：将数组 [2, 0, 6, 1, 77, 0, 52, 0, 25, 7] 中大于等于 10 的元素选出来，放入新数组中

分析：

①：声明一个新的数组用于存放新数据 newArr

②：遍历原来的旧数组， 找出大于等于 10 的元素,依次追加push给新数组 newArr

案例

数组去0案例

需求：将数组 [2, 0, 6, 1, 77, 0, 52, 0, 25, 7] 筛选，形成一个不包含 0 的新数组

分析：

①：声明一个新的数组用于存放新数据 newArr

②：遍历原来的旧数组， 找出不等于0的元素,依次追加push给新数组 newArr

2.2 操作数组 - 删除元素

删除

数组.pop()

作用：从数组中删除最后一个元素，修改原数组

返回值：该元素的值

```
let arr = ['red', 'green']
arr.pop()
console.log(arr)  //  ['red']
```

数组.shift()

作用：从数组中删除第一个元素，修改原数组

返回值：该元素的值

```
let arr = ['red', 'green']
arr.shift()
console.log(arr)  //  ['green']
```

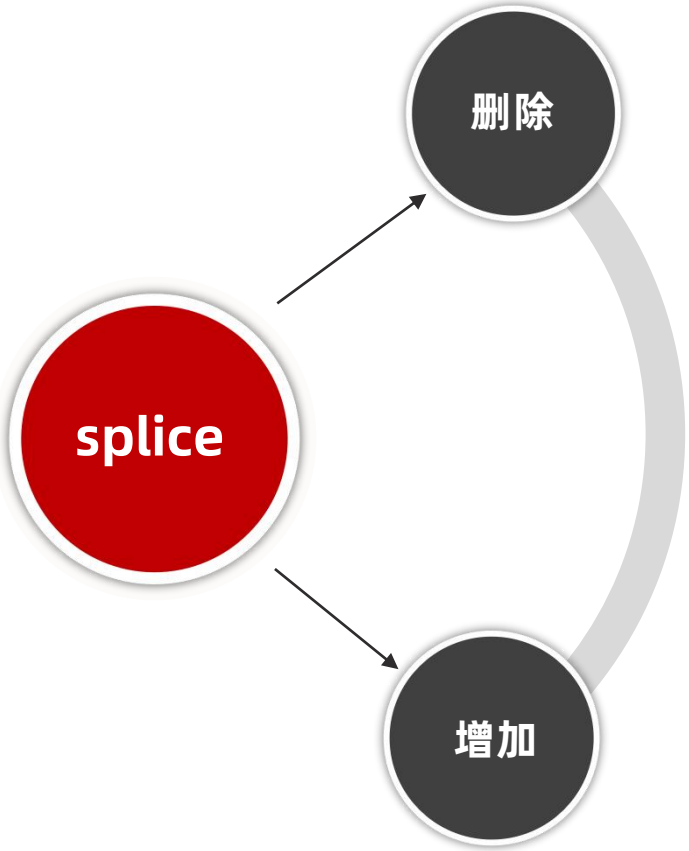
总结

1. 想要数组末尾删除一个数据元素利用哪个方法？
   - arr.pop()
   - 返回的是该数组元素
2. 想要数组开头删除一个数据元素利用哪个方法？
   - arr.shift()
   - 返回的是该数组元素
3. 两种方法会修改原数组吗？
   - 会的

2.2 操作数组

数组. splice() 方法　可以添加也可以删除数组元素

说明：　可以从指定位置删除或者增加的数组元素，注意它修改原数组



splice(start, deleteCount)

- start 起始位置:
  - 指定修改的开始位置（从0计数）
- deleteCount 删除个数:
  - 表示要移除的数组元素的个数
  - 可选的。 如果省略则默认从指定的起始位置删除到最后

splice(start, deleteCount, item1...)

目录

Contents

- 循环-for
- 数组
- 综合案例

案例

手风琴案例

需求：利用循环生成图片

案例

手风琴案例

需求： 利用循环生成图片

数组数据：

```
let arr = [
        './images/1.jpg',
        './images/2.jpg',
        './images/3.jpg',
        './images/4.jpg',
        './images/5.jpg',
        './images/6.jpg',
        './images/7.jpg',
]
```

案例

手风琴案例

```
<div><img src="./images/1.jpg" alt=""></div>
```

小技巧： 利用循环拼接字符串（原理跟累加和类似）

①：声明一个空的字符串 str = ''

②：循环里面利用 **+=** 进行拼接

③：把拼接完毕的字符串放入容器中

案例

手风琴案例

```
<div><img src="./images/1.jpg" alt=""></div>
<div><img src="./images/2.jpg" alt=""></div>
```

小技巧： 利用循环拼接字符串（原理跟累加和类似）

①：声明一个空的字符串 str = ''

②：循环里面利用 **+=** 进行拼接

③：把拼接完毕的字符串放入容器中

案例

手风琴案例

小技巧：利用循环拼接字符串（原理跟累加和类似）

①：声明一个空的字符串

②：循环里面利用 **+=** 进行拼接

③：把拼接完毕的字符串放入容器中

```
<div><img src="./images/1.jpg" alt=""></div>
<div><img src="./images/2.jpg" alt=""></div>
<div><img src="./images/3.jpg" alt=""></div>
<div><img src="./images/4.jpg" alt=""></div>
<div><img src="./images/5.jpg" alt=""></div>
<div><img src="./images/6.jpg" alt=""></div>
<div><img src="./images/7.jpg" alt=""></div>
```

```
<div class="box">

</div>
```

案例

根据数据生成柱形图

需求：用户输入四个季度的数据，可以生成柱形图

案例

根据数据生成柱形图

需求： 用户输入四个季度的数据，可以生成柱形图

分析：

①：需要输入4次，所以可以把4个数据放到一个数组里面

- 声明一个空数组
- 利用循环，弹出4次框，同时把数据存到数组里面

②：遍历该数组，根据数据生成4个柱形图，渲染打印到页面中

- 声明一个空字符串 str
- 根据数组利用循环拼接字符串，生成多个柱子
- 把生成的柱子添加到 box 盒子中

今日单词

模块	说明	单词	作用
数组	数组长度	数组.length	得到数组的长度
	新增数组元素	数组.push(新增元素)	数组末尾新增元素
		数组.unshift(新增元素)	数组开头新增元素
	删除数组元素	数组.pop()	删除数组最后一个元素
		数组.shift()	删除第一个数组元素
	新增/删除数组元素	数组.splice()	从指定位置删除/新增数组元素
	数组排序	数组.sort()	对数组进行升序/降序排序

拓展 - 数组排序

- 数组. sort() 方法可以排序

- **语法：**
- 数组.sort() 会修改原数组

```
[88, 78, 100, 34, 99]
```

```
[34, 78, 88, 99, 100]
```

升序排序

```
[100, 99, 88, 78, 34]
```

降序排序

```
let arr = [4, 2, 5, 1, 3]
// 1.升序排列写法
arr.sort(function (a, b) {
  return a - b
})
console.log(arr)  // [1, 2, 3, 4, 5]
// 降序排列写法
arr.sort(function (a, b) {
  return b - a
})
console.log(arr)  // [5, 4, 3, 2, 1]
```

选择排序算法

选择排序原理：（了解）

就是从第一个数开始，与后面所有的数相比较，找出最小（最大）的数，放在第一个位置

以此类推，每一轮确定一个相对于这一轮最小（最大）的数

算法可视化网站

- https://visualgo.net/zh/sorting

算法：

核心： 利用循环嵌套比较，根据索引号来交换变量

①：外层循环是一共进行几轮相比较，通过观察，一共进行 数组长度-1 次比较

- for (let i = 0; i < arr.length - 1; i++)
- 把 i 作为最小值起始索引 minIndex

②：里层循环是每一轮的比较来查找最小值

- 里层循环起始值是 i + 1 个元素开始查找
- for (let j = i + 1; j < arr.length; j++)
- 进行比较的时候，发现最小的数组元素，把当前元素索引号给 minIndex

③：如果 minIndex 和 i 位置不一致，则交换变量

黑马程序员

www.itheima.com

传智教育旗下高端IT教育品牌