

实验四 JavaScript 基础编程实验

一、教学课时

2 学时。

二、实验目的和要求

1. 掌握 JavaScript 的基本语法。
2. 掌握分支、循环以及循环嵌套。
3. 使用 document 对象的 write() 方法输出文本和 HTML 标签。

三、实验内容

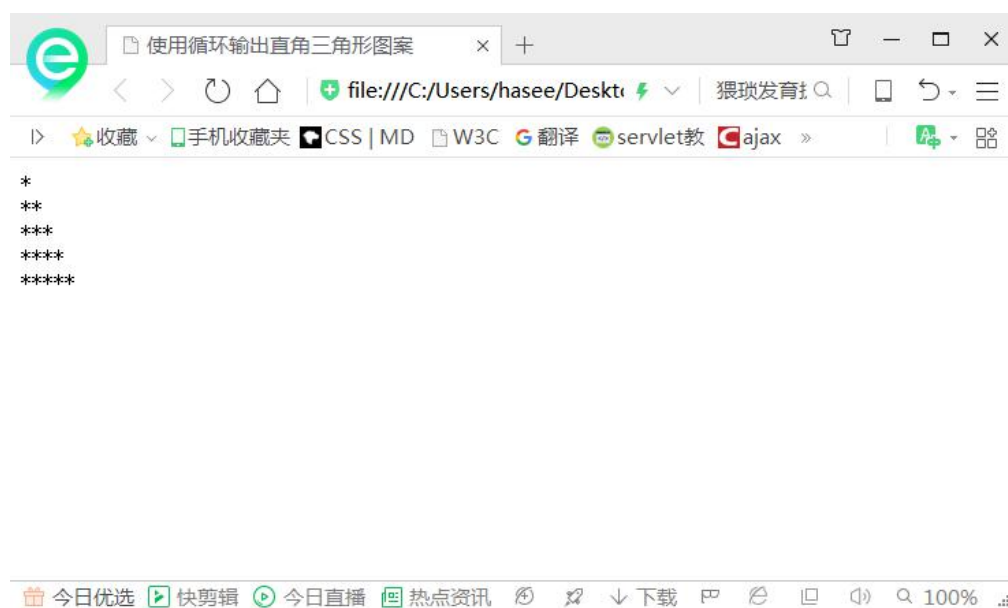
第一阶段：程序验证

1. 指导 1 使用循环输出直角三角形图案

知识点：

- for 循环
- 循环的嵌套
- 使用 document 对象的 write() 方法来输出文本和 HTML 标签
- 运行包含 JavaScript 的网页

(1) 问题：使用 Javascript 循环语句在页面中输出三角形图案，三角形由星号组成。如图上机 1.1 所示。



图上机 1.1 三角形图案

(2) 解决方案：创建网页文档，编写网页的基本结构代码。在网页正文 `<body></body>` 标签内添加 `<script></script>` 标签，并为它设置 `type` 属性，值为 `text/javascript`，然后在其中编写脚本代码。

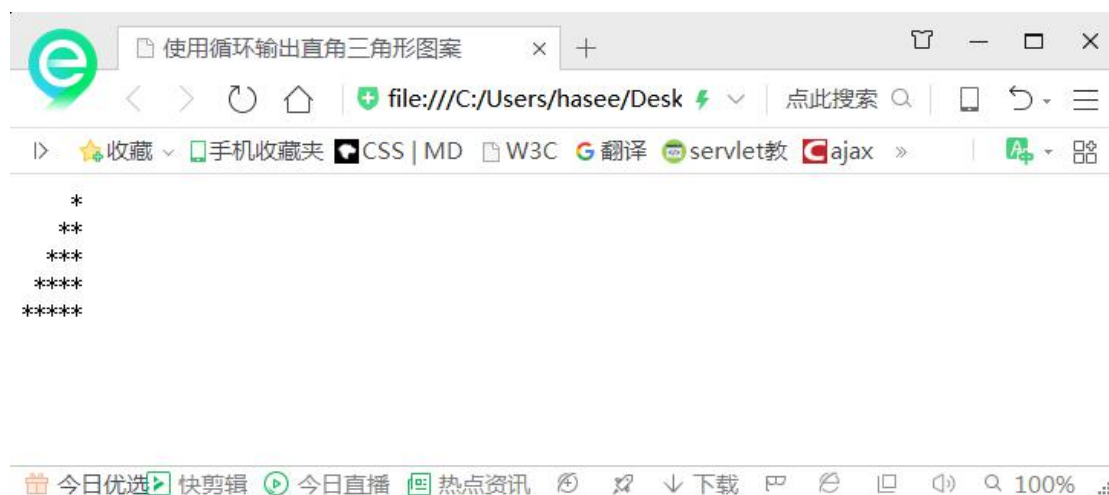
```
for(int i = 1; i <= 5; i++)
{
    for(int j = 1; j <= i; j++)
    {
        document.write("*");
    }
}
```

2. 指导 2 使用循环输出斜边在左侧的指教三角形图案

知识点：

- for 循环
- 循环的嵌套
- 使用 `document` 对象的 `write()` 方法输出文本和 HTML 标签。

(1) 问题：对上例进行修改，使三角形的直角边位于右侧。如图上机 1.3 所示。



图上机 1.3 三角形的直角边位于右侧

(2) 解决方案：创建网页文档，编写基本的结构代码。在网页正文<body></body>标签内，编写如下代码。

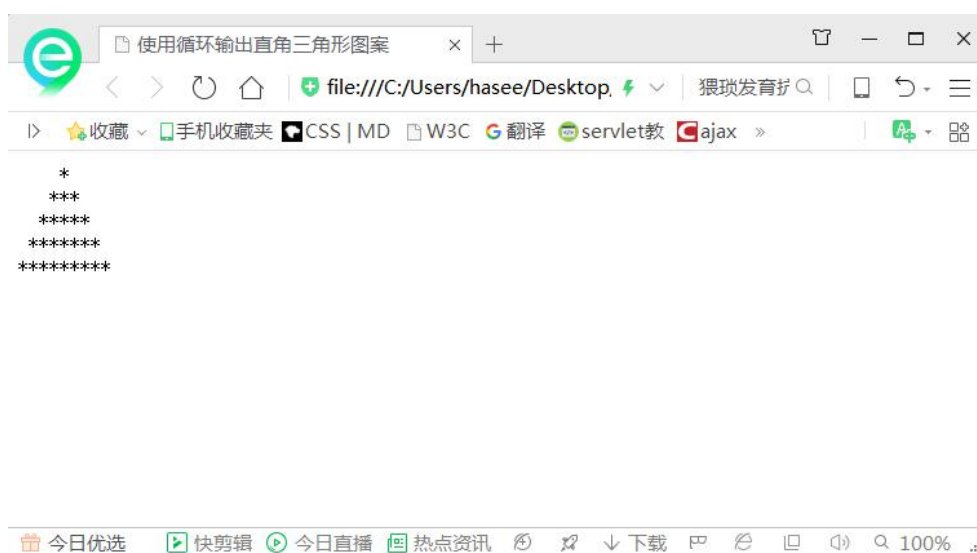
```
<script type="text/javascript">
for(var i = 1; i <=5; i++)
{
    for(var k = 1; k <= 5 - i; k++)
    {
        document.write("&nbsp;");
    }
    for( var j = 1; j<=i; j++)
    {
        document.write("*");
    }
}
</script>
```

(3) 在浏览器中运行，查看脚本的运行结果。

第二阶段：程序编写

1. 练习 1 使用循环输出等腰三角形图案

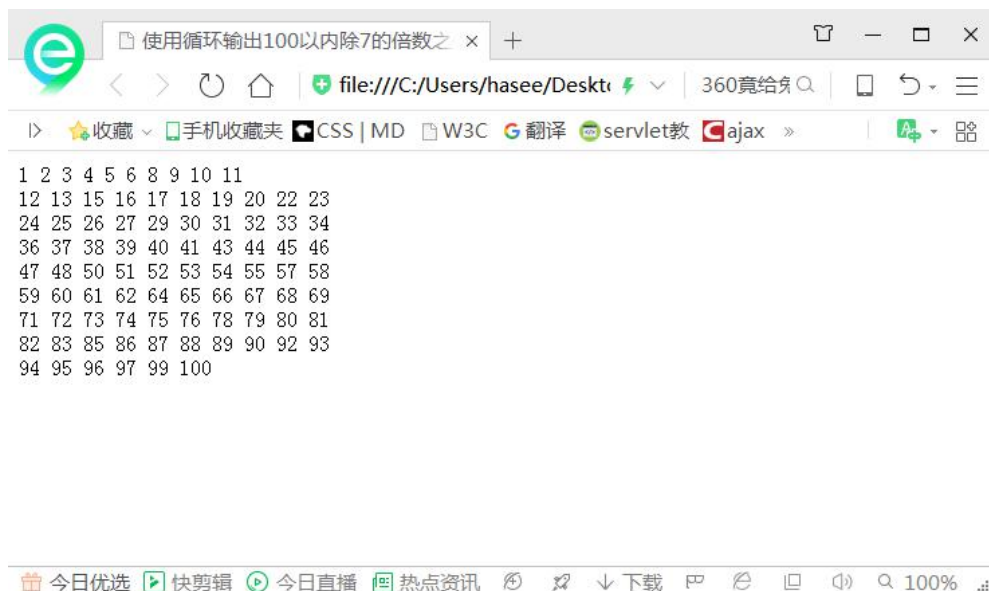
(1) 问题：实现左右对称的等腰三角形。如图上机 1.4 所示。



图上机 1.4 等腰三角形

2. 练习 2 使用循环输出 100 以内除 7 的倍数之外的整数。

- (1) 问题：在页面上输出 100 以内的正整数，但不包括 7 的倍数，每行输出 10 个。如图上机图 1.5 所示。



图上机 1.5 使用循环输出 100 以内的整数

3. 练习 3 使用锯齿数组

- (1) 问题：在 JavaScript 中模拟实现二维数组的功能，用于存储各个学期的所有课程名称，请根据运行结果在 JS 中自行构建数组并输出运行结果如图上机 1.6（当需要某一元素时必须从数组中调用）。
- (2) 提示：JavaScript 不支持二维数组以及多维数组，但是可以使用锯齿数组模拟二维数组的功能。锯齿数组又称为交错数组，简单理解就是数组的数组，即每一个元素都是另一个数组。
- (3) 例子：var myArray = [[“Jack”, “Mike”], [“Mary”, “Michael”]]; 这里创建了一个数组 myArray，myArray 中的两个元素都是数组。



4. 练习 4 在网页中显示日期时间

- (1) 问题：在页面中显示当前的日期时间。要求包含年月日星期时分秒，如图上机 1.7 所示。
- (2) 提示：调用 Date 对象的无参构造函数可以创建表示当前日期时间的 Date 对象（参见 Date 对象参考手册），可以获得 Date 对象包含的年份、月份、日期、星期、时、分、秒等信息。



第三阶段：思考题

- 1. 思考 JavaScript 的数组在定义与使用上与 Java 的数组定义与使用有什么不同？
- 2. 思考 JavaScript 的 Date() 对象与 Java 中的 Date 类有哪些异同。
- 3. 思考通过本次实验对 JavaScript 有哪些进一步的了解。

附录：

JavaScript Date 对象 参考手册（摘自 W3C）

Date 对象

Date 对象用于处理日期和时间。

创建 Date 对象的语法：

```
var myDate=new Date()
```

注释：Date 对象会自动把当前日期和时间保存为其初始值。

Date 对象属性

属性	描述
constructor	返回对创建此对象的 Date 函数的引用。
prototype	使您有能力向对象添加属性和方法。

Date 对象方法

方法	描述
Date()	返回当日的日期和时间。
getDate()	从 Date 对象返回一个月中的某一天（1 ~ 31）。
getDay()	从 Date 对象返回一周中的某一天（0 ~ 6）。
getMonth()	从 Date 对象返回月份（0 ~ 11）。
getFullYear()	从 Date 对象以四位数字返回年份。
getYear()	请使用 <code>getFullYear()</code> 方法代替。
getHours()	返回 Date 对象的小时（0 ~ 23）。

<code>getMinutes()</code>	返回 Date 对象的分钟 (0 ~ 59)。
<code>getSeconds()</code>	返回 Date 对象的秒数 (0 ~ 59)。
<code>getMilliseconds()</code>	返回 Date 对象的毫秒 (0 ~ 999)。
<code>getTime()</code>	返回 1970 年 1 月 1 日至今的毫秒数。
<code>getTimezoneOffset()</code>	返回本地时间与格林威治标准时间 (GMT) 的分钟差。
<code>getUTCDate()</code>	根据世界时从 Date 对象返回月中的一天 (1 ~ 31)。
<code>getUTCDay()</code>	根据世界时从 Date 对象返回周中的一天 (0 ~ 6)。
<code>getUTCMonth()</code>	根据世界时从 Date 对象返回月份 (0 ~ 11)。
<code>getUTCFullYear()</code>	根据世界时从 Date 对象返回四位数的年份。
<code>getUTCHours()</code>	根据世界时返回 Date 对象的小时 (0 ~ 23)。
<code>getUTCMinutes()</code>	根据世界时返回 Date 对象的分钟 (0 ~ 59)。
<code>getUTCSeconds()</code>	根据世界时返回 Date 对象的秒钟 (0 ~ 59)。
<code>getUTCMilliseconds()</code>	根据世界时返回 Date 对象的毫秒 (0 ~ 999)。
<code>parse()</code>	返回 1970 年 1 月 1 日午夜到指定日期 (字符串) 的毫秒数。
<code>setDate()</code>	设置 Date 对象中月的某一天 (1 ~ 31)。
<code>setMonth()</code>	设置 Date 对象中月份 (0 ~ 11)。
<code>setFullYear()</code>	设置 Date 对象中的年份 (四位数字)。
<code>setYear()</code>	请使用 <code>setFullYear()</code> 方法代替。
<code>setHours()</code>	设置 Date 对象中的小时 (0 ~ 23)。
<code>setMinutes()</code>	设置 Date 对象中的分钟 (0 ~ 59)。
<code>setSeconds()</code>	设置 Date 对象中的秒钟 (0 ~ 59)。
<code>setMilliseconds()</code>	设置 Date 对象中的毫秒 (0 ~ 999)。
<code>setTime()</code>	以毫秒设置 Date 对象。
<code>setUTCDate()</code>	根据世界时设置 Date 对象中月份的一天 (1 ~ 31)。
<code>setUTCMonth()</code>	根据世界时设置 Date 对象中的月份 (0 ~ 11)。
<code>setUTCFullYear()</code>	根据世界时设置 Date 对象中的年份 (四位数字)。
<code>setUTCHours()</code>	根据世界时设置 Date 对象中的小时 (0 ~ 23)。
<code>setUTCMinutes()</code>	根据世界时设置 Date 对象中的分钟 (0 ~ 59)。
<code>setUTCSeconds()</code>	根据世界时设置 Date 对象中的秒钟 (0 ~ 59)。

<code>setUTCMilliseconds()</code>	根据世界时设置 Date 对象中的毫秒 (0 ~ 999)。
<code>toSource()</code>	返回该对象的源代码。
<code>toString()</code>	把 Date 对象转换为字符串。
<code>getTimeString()</code>	把 Date 对象的时间部分转换为字符串。
<code>toDateString()</code>	把 Date 对象的日期部分转换为字符串。
<code>toGMTString()</code>	请使用 <code>toUTCString()</code> 方法代替。
<code>toUTCString()</code>	根据世界时, 把 Date 对象转换为字符串。
<code>toLocaleString()</code>	根据本地时间格式, 把 Date 对象转换为字符串。
<code>toLocaleTimeString()</code>	根据本地时间格式, 把 Date 对象的时间部分转换为字符串。
<code>toLocaleDateString()</code>	根据本地时间格式, 把 Date 对象的日期部分转换为字符串。
<code>UTC()</code>	根据世界时返回 1970 年 1 月 1 日 到指定日期的毫秒数。
<code>valueOf()</code>	返回 Date 对象的原始值。