JavaScript语句

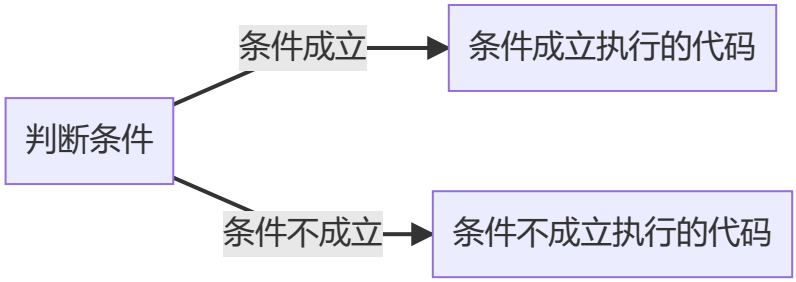
语句是构成编程系统当中的最小单位，它是用于完成某些特定的逻辑功能，也是为了完成代码的流程控制的，所以语句也叫流程控制

语句大体上来说可以分为以下几类

1. 分支语句
2. 循环语句
3. 选择语句

所有的语句都应该是由一个或多个关键字来完成的

if...else



`if...else`是我们的语句当中的分支语句，它用于做条件判断，来控制代码的执行流程，它的语法格式如下

```
if(条件1){
    //符合条件1的时候
}
else if(条件2){
    //后面还有好多条件....
}
else{
    //不满足条件的时候
}
```

如下所示

```
 var a = 10;
 var b = 20;
 // 如果a>b则输出"你好",否则就输出“你不好”
 // console.log()
 if (a > b) {
     console.log("你好");
 }
 else{
     console.log("你不好");
 }
```

上面的代码是一个非常简单的流程控制，我们只有一个条件。但是这里仍然有几个注意事项给大家说一下

```
var a = 10;
var b = 20;
if(a>b)
    console.log("你好");
else
    console.log("你不好");
```

> 当代码体只有一行的时候，我们可以省略花括号【不建议这么支操作，很容易形成岐义，也很容易出错】

请看下面的要求及代码

```
var score = 88;
//如果分数在90或以后，就输入优秀
//如果分数在80或以后，就输入良好
//如果分数在70或以后，就输入中等
//如果分数在60或以后，就输入及格
//否则就输出不及格
```

第一种情况下的代码

```
 if(score>=90){
     console.log("优秀");
 }
 if(score>=80){
     console.log("良好");
 }
 if(score>=70){
     console.log("中等");
 }
 if(score>=69){
     console.log("及格");
 }
 else{
     console.log("不及格");
 }
```

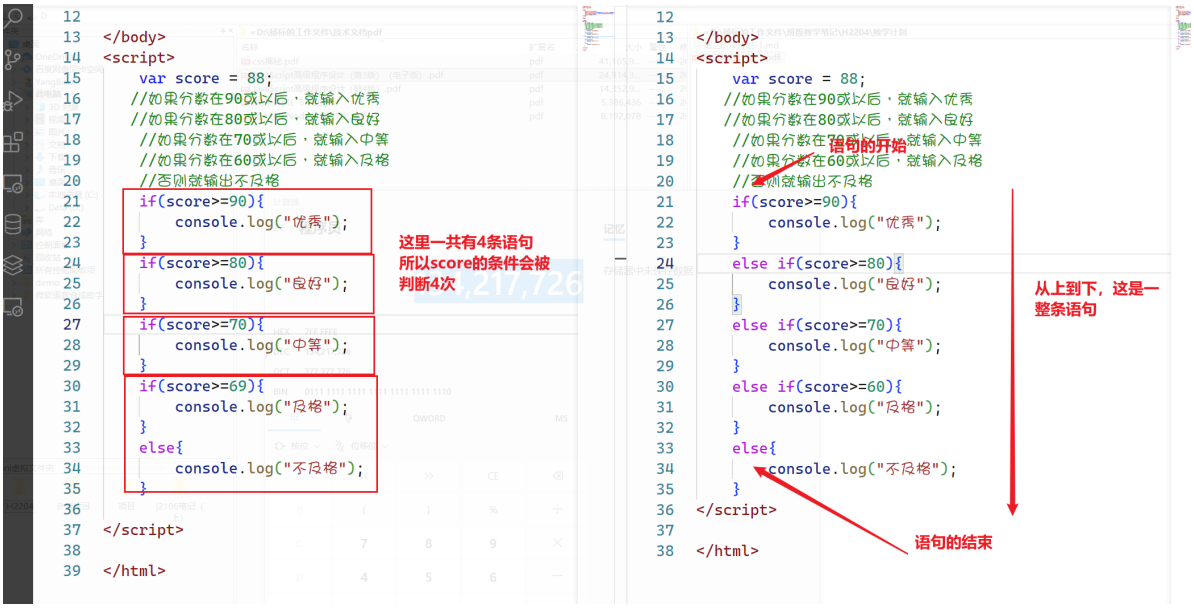
> 上面的代码写法是有问题的，它有所有的if都是一个**并行条件**，它会把所有的 `if` 条件都测一遍，只要是符合要求了，就执行了。
>
> 最终打印的结果就是“良了，中等，及格”

第二种情况下的代码

```
 if(score>=90){
     console.log("优秀");
 }
 else if(score>=80){
     console.log("良好");
 }
 else if(score>=70){
     console.log("中等");
```

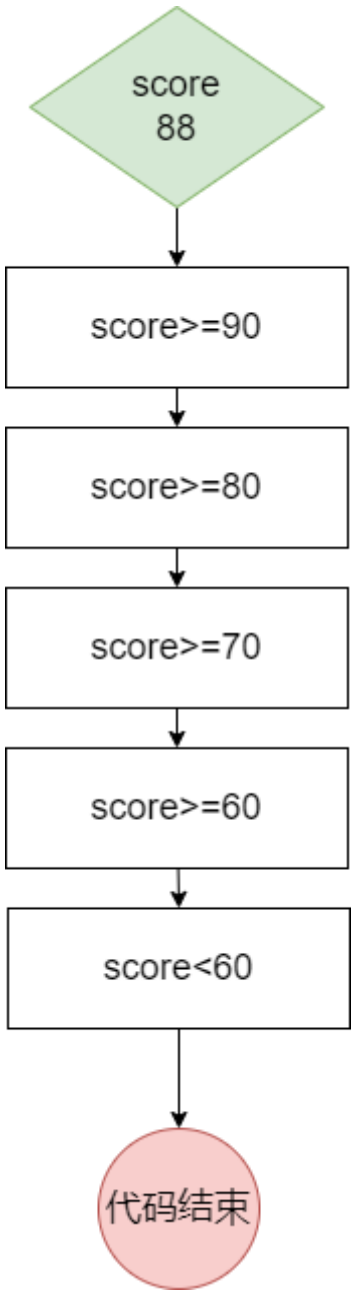
```
9   }
10  else if(score>=60){
11      console.log("及格");
12  }
13  else{
14      console.log("不及格");
15  }
```

> 上面的代码最终打印输出“良好”，它所有的条件都是串行的条件，它要符合了一个要求，就会中断语句，提前结果

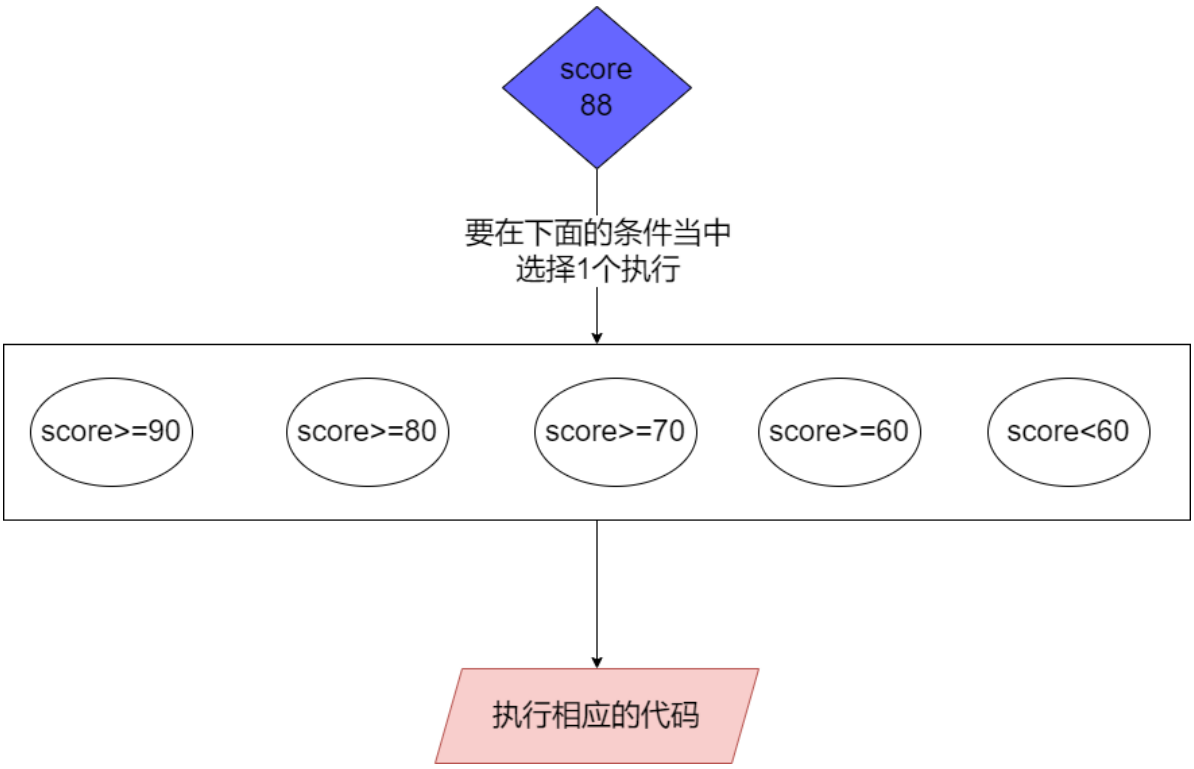


现在我们分别用2张图来说明这个情况

多个if的情况



if...else的情况



在 `if` 与 `else ...if` 里面，一共定要弄清楚它们的区别点是什么

if是对条件进行判断的的时候成立的时候执行的语句，那么如果 `if` 的条件的值不是一个 `boolean` 类型怎么办，如下所示

```
var a = 123;
var b = 0;

if(a){
    console.log("你好");
}
else{
    console.log("世界");
}

if(b){
    console.log("hello");
}
else {
    console.log("world");
}
```

其中的 condition（条件）可以是任意表达式；而且对这个表达式求值的结果不一定是布尔值。ECMAScript 会自动调用 `Boolean()` 转换函数将这个表达式的结果转换为一个布尔值。如果对 condition 求值的结果是 `true`，则执行 statement1（语句 1），如果对 condition 求值的结果是 `false`，则执行 statement2（语句 2）。而且这两个语句既可以是一行代码，也可以是一个代码块（以一对花括号括起来的多行代码）。

> 一定要弄清楚之前给大家讲过了JS里面6个明确的false条件

for循环语句

for 语句也是一种 ， 它会循环的执行代码体里面的代码，它的语法格式如下

```
for(初始值;循环前测试条件;自变量){
    //代码体
}
```

现在我们先从生活当中的最基本的例子来讲起，请看以下场景

场景一：标哥有钱了，办了一个砖厂，颜一鸣同学在标哥这里在搬砖，今天标哥交给颜一鸣的任务就是要搬10块砖。因为颜一鸣的个子比较小，所以一次只能搬动1块砖。那么请用代码来说明一下颜一鸣搬砖的过程

针对上面的地场景，我们要弄清楚几个问题

1. 颜一鸣从第1块开始搬砖
2. 颜一鸣要搬10块
3. 颜一鸣每次搬1块

首先，我们就使用搬砖的思维方式来考虑这个问题

```
for (var count = 1; count <= 10; count++) {
     console.log("颜一鸣从在搬" + count + "块砖");
}
```

场景二：颜一鸣经过多年的搬砖生活以后，肌肉变得发达了，现在可以一次搬2块砖了，标哥很欣慰，看到了颜一鸣的成长，今天又给颜一鸣派了一个新的任务，让颜一鸣还是砖10块砖，颜一鸣听后非常高兴说道：“小意思啦，我现在是大力士了，可以一次搬2块，来，看我的表演”

现在请根据上面的场景，来使用编程思维解决

1. 颜一鸣从第1块开始搬砖
2. 颜一鸣要搬10块
3. 颜一鸣每次搬2块

```
for (var count=1; count<=10; count+=2) {
    //代码体
    console.log("颜一鸣从在搬" + count + "块砖");
}
```

场景三：功夫不负有心人，颜一鸣经过了这么长时间在标哥这里的打拼，小有资本，它偷偷的搬了4块砖藏起来了。今天看到标哥，格外的神气一些，标哥不舒服了。说道：“来，小伙子，给你个光荣而又艰巨的任务，今天要搬20块砖，随便你怎么搬”。颜一鸣听后，无所谓的道：“没事，可以的，我可以一次搬2块砖”，同时颜一鸣心底还乐道：“嘿嘿，我之前还有4块砖藏着，可以直接使用，这样我就可以从第5块直接开始了”

现在请根据上面的场景来使用编程的思维解决

1. 颜一鸣从第5块开始
2. 颜一鸣要搬20块
3. 颜一鸣每次搬2块

```
for(var count = 5;count<=20;count+=2){
    console.log("颜一鸣从在搬"+count+"块砖");
}
```

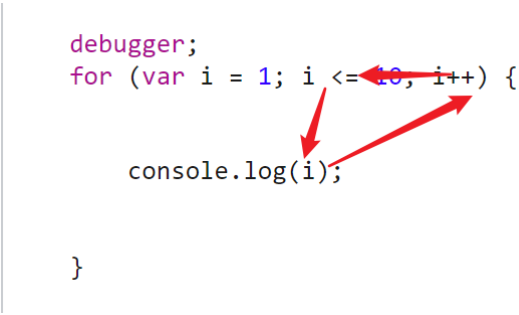
> 在上面的3个场景里面，我们可以看到，for循环里面的三个条件我们是可以根据实际的场景去做出相应的改变的，这个三个条件代表的是什么意思，同学样一定要知道

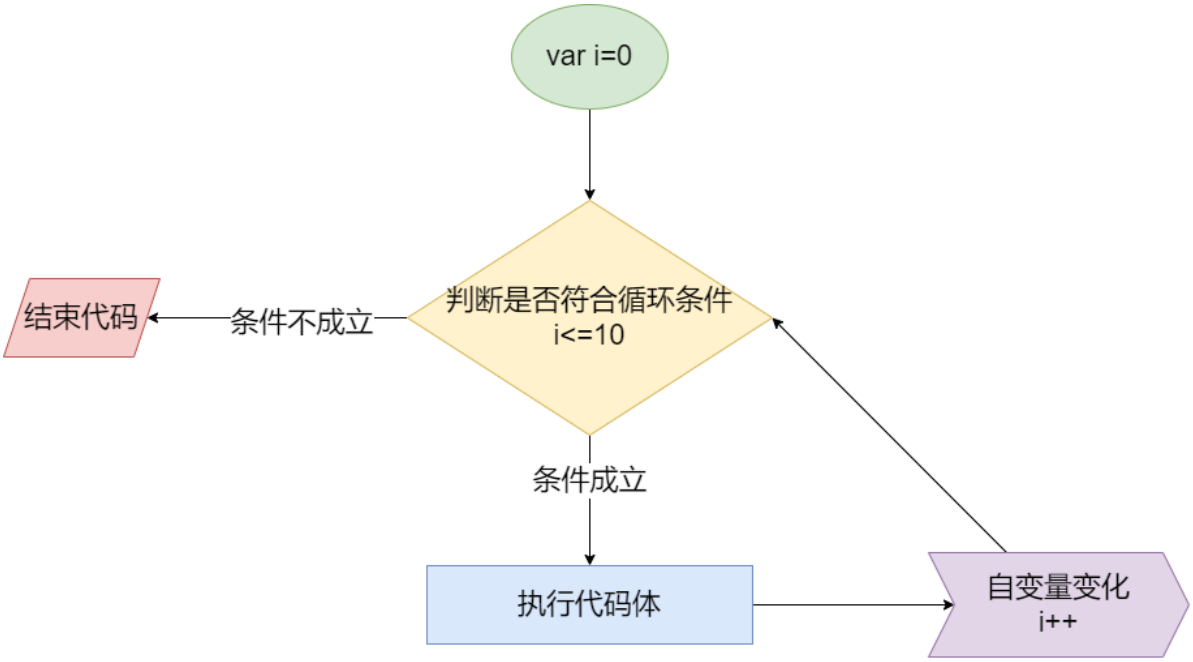
循环的本质

for循环的本质其实就是指for循环代码的执行过程是什么样式，我们现在通过断点调试的方式来进行

```
debugger;
for (var i = 1; i <= 10; i++) {
    console.log(i);
}
```

当我们在代码里面添加了`debugger`这个关键字以后，代码在运行的时候，就会在这个地方暂停下来





通过上面的图，我们可以知道，循环的本质就是在不停的去执行三个地方的代码

1. 判断循环条件是否成立
2. 执行代码体
3. 自变量要变化

循环的本质，我们已经知道了，现在我们弄清楚一个点，这三个条件是否都是必须的

第一种情况，如果没有初始变量，怎么办

```
var count = 1;       //循环的初始变量是可以写在外边的
for (; count <= 10; count++) {
    console.log(count);
}
```

第二种情况，省略循环的判断条件

```
for(var count=1;;count++){
    console.log(count);
}
//省略了循环的结束判断条件，那么这个循环就会一直进行，它就是一个死循环
```

第三种情况，省略自变量

```
for(var count=1;count<=10;){    //省略了自变量，它也会变成一个死循环
    console.log(count);
}
```

综上所述，for里面的三个东西，是都可以省略了

```
for(;;){
    console.log("hello world");
}
```

上面的语法是没有错的，但是它是一个死循环

小练习

1. 请打印1~100之间的偶数

```
/**
     * 开始条件：2
     * 结束条件：100
     * 自变量：2
*/
for (var i = 2; i <= 100; i = i + 2) {
    console.log(i);
}
```

还有一种写法

```
 /**
      * 开始条件：1
      * 结束条件：100
      * 自变量：1
 */
 for (var i = 1; i <= 100; i++) {
     // 下面打印这句话，不是必须的，它有条件执行
     // 条件成立我就执行，条件不成立，我就不执行
     if (i % 2 === 0) {
         console.log(i);
     }

 }
```

2. 计算1+2+3+....+10的总和

```
/**
     * 初始值：1
     * 结束值：10
     * 自变量：1
*/
var sum = 0;
```

```
for (var i = 1; i <= 10; i++) {
    sum = sum + i;
}
console.log(sum);
// 第一次循环 sum  = 0 + 1;
// 第二次循环 sum  = 0 + 1 + 2;
// 第三次循环 sum = 0 + 1 + 2 + 3;
// 第十次循环 sum = 0 + 1+ ......+10
```

---

while循环语句

while循环语句与for循环的语句的本质是一样的，都是一样**前测试循环语句**，它的语法格式如下

```
while(循环条件){
    //代码体
}
```

while是当循环条件成立的时候会一直执行的代码，这个和我们之前所学习的for循环是一样的。我们之前在讲for循环的时候讲到过循环的三个条件

1. 初始值
2. 循环条件
3. 自变量

```
for(var i=1;i<=10;i++){
    console.log(i);
}
```

上面的`for`循环可以写成下面的样式

```
//之前我们都讲过，循环里面的三个部分，是可以拆分来写的
var i=1;
for(;i<=10;){
    console.log(i);
    i++;
}
```

现在我们就将上面的`for`转写成`while`循环

```
var i=1;
while(i<=10){
    console.log(i);
    i++;
}
```

---

do...while循环语句

do...while循环语句它是一个**后测试循环语句**，先执行代码体，再判断条件是否成立

语法格式

```
//初始值
do{
    //代码体
    //自变量
}while(循环条件)
```

> 通俗来讲就是“先上车，后买票”

```
 // 想打印1~10之间的数
// for(var i=1;i<=10;i++){
//     console.log(i);
// }

// var i = 1;
// while (i <= 10) {
//     console.log(i);
//     i++;
// }

// do...while的写法
var i = 1;
do {
    console.log(i);
    i++;
} while (i <= 10);
```

通过上面的代码对比，我们发现，三种循环最终得到的结果是一样的，那么为什么还需要 `do...while` 呢

我们将上面的代码的初始条件更改一出，我们就发现区别

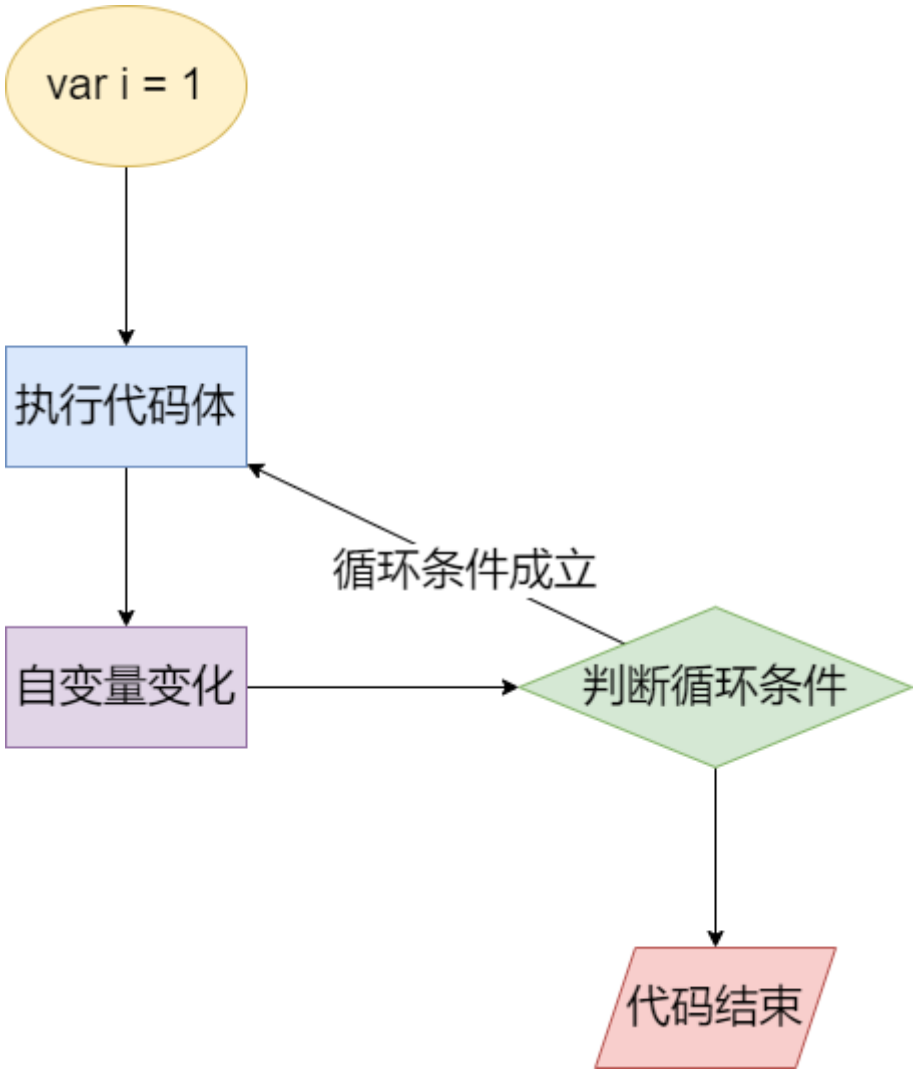
```
// 想打印1~10之间的数
// for (var i = 11; i <= 10; i++) {
//     console.log(i);
// }

// var i = 11;
// while (i <= 10) {
//     console.log(i);
//     i++;
// }

// do...while的写法
var i = 11;
do {
    console.log(i);
    i++;
} while (i <= 10);
```

这个时候我们可以看到，因为 `for` 与 `while` 都是前测试循环语句，所以在最初的时候循环条件就不成立，它就不会执行，而 `do...while` 是后测试循环语句，它会先执行一遍代码然后再判断条件是否成立，**所以 `do...while` 至少会执行一次**【因为即使后期条件不成立，我也执行了一次】

```
var a = 10;
do {
    console.log("我要牵女孩子的手");
    a++;
} while (a < 10);
```



break与continue

在上面的章节讲循环的时候，我们提到了循环的次数是由3个条件共同决定的，但是在某些特定的条件下面，它会提前结束循环

正常情况下，我们的循环代码当条件不成立以后会自动退出循环，但是在JS里面，仍然有2个关键字会让循环中途退出，这2个关键字就是 `break` 与 `continue`

- `break` 有中断循环的意思，提前结束循环，后面的循环不再进行了【半途而废】
- `continue` 的意思就是跳过本次循环，继续执行下一次循环【浪子回头】

> 在多重for循环的嵌套里面， `break` 与 `continue` 执行的是就近原则，它会的对当前这个循环起作用

场景一：假设颜一鸣给标哥搬10块砖，但是颜一鸣在搬到第5块砖的时候就接到他爸爸的电话，说：“儿啊，你爸我中了彩票，5000万，你要做富二代了”，颜一鸣听了这个话以后，非常兴奋，直接把手上的砖一丢，说：“去TMD，我也是富二代了”

面对上面的场景，我们怎么办呢?

```
/*
    初始值：1
    结束值：10
    自变量：1
*/
for(var count=1;count<=10;count++){
    //现在颜一鸣要他爸电话
    if(count==5){
        break;
        //因为是break,所以提前结束了循环，后面的次数也不再进行了
    }
    console.log("颜一鸣正在搬"+count+"块砖");
}
```

场景二：颜一鸣拿着他爸爸给了5000，怀揣着梦想去了一个心仪已久的城市---东莞，结果被骗了，身心疲惫的回到了标哥的身边继续搬砖，突然之间它觉得，在这个炎炎夏日里，还是只有标哥里的心是最温暖的，只有标哥对自己好！下定决心的颜一鸣决定要在标哥这里好好奋斗，做一个最强的搬砖人。标哥给颜一鸣派发了又一次的任务，**搬10块砖，从第1块开始，因为天气太热，所以就1次只搬一次**，这个时候正当颜一鸣开心的搬砖的时候，在耳边双听到了标哥的声音：”一鸣 ，你爸电话“

颜一鸣听到标哥的声音以后，身体一震，**把手上的砖一丢**，带着期望的眼神走向了办公室去接听电话，颤抖的声音说道：”爸呀，你又买彩票了？“

只见电话那头许久未曾出声，接着，一个声音慢悠悠的说道："儿啊，你爸我失业了，给爸打点钱"

标哥在旁边看了许久，也不知道电话那头具体说了些什么，只见颜一鸣听完电话以后，神低落的走出了办公室 ，嘴里还喃喃的说道：“其它那些砖也不那么烫手了，我要继续搬砖，争取今天搬到100块，拿到今天的饭钱，这样我就可以请我的爸吃饭了”

针对上面的场景 ，我们又应该怎么样使用编程的思维去解决呢?

```
for (var count = 1; count <= 10; count++) {
    // 当搬到第5块砖的时候 ，颜一鸣接电话去了
    if(count==5){
        //接电话
        continue;
    }
    console.log("颜一鸣正在搬" + count + "块砖");
}
```

嵌套的循环

循环也是可以嵌套的，如多个for循环是可以嵌套在一起的

```
for(var a=1;a<=10;a++){
    for(var b = 1;b<=10;b++){
        //代码体
    }
}
```

在上面的代码里面，我们可以看到，循环是可以进行嵌套的，嵌套的循环也可以有多层的嵌套关系，只要循环的语法是正确的，并且不要形成死循环就好

现在请根据下面的图片打印九九乘法表，这里会使用到2层的嵌套循环

1x1=1								
1x2=2	2x2=4							
1x3=3	2x3=6	3x3=9						
1x4=4	2x4=8	3x4=12	4x4=16					
1x5=5	2x5=10	3x5=15	4x5=20	5x5=25				
1x6=6	2x6=12	3x6=18	4x6=24	5x6=30	6x6=36			
1x7=7	2x7=14	3x7=21	4x7=28	5x7=35	6x7=42	7x7=49		
1x8=8	2x8=16	3x8=24	4x8=32	5x8=40	6x8=48	7x8=56	8x8=64	
1x9=9	2x9=18	3x9=27	4x9=36	5x9=45	6x9=54	7x9=63	8x9=72	9x9=81

```
for (var row = 1; row <= 9; row++) {
    var str = "";
    for (var col = 1; col <= row; col++) {
        str += col + "*" + row + "=" + col * row + "\t";
    }
    console.log(str);
}
```

label语句

label语句是一个非常简单语句，在这一个简单的标记里面，它主要是用于配置嵌套的循环来使用的，用于表明 `break` 或 `continue` 跳到哪一层去

```
for (var row = 1; row <= 9; row++) {
    var str = "";
    for (col = 1; col <= row; col++) {
        str += row + "*" + col + "=" + row * col + "\t";
        if (col == 5) {
            break;
        }
    }
    console.log(str);
}
```

在上面的代码里面，我们在第二个循环里面使用了 `break` ，这个时候得到的结果如下所示

```
1*1=1
2*1=2   2*2=4
3*1=3   3*2=6   3*3=9
4*1=4   4*2=8   4*3=12  4*4=16
5*1=5   5*2=10  5*3=15  5*4=20  5*5=25
6*1=6   6*2=12  6*3=18  6*4=24  6*5=30
7*1=7   7*2=14  7*3=21  7*4=28  7*5=35
8*1=8   8*2=16  8*3=24  8*4=32  8*5=40
9*1=9   9*2=18  9*3=27  9*4=36  9*5=45
>
```

从上面的图里面，我们可以看出，`for`里面的`break`中断是内部的循环，这是因为`break`与`continue`执行的是就近原则，有没有什么办法让`break/continue`去指定的循环那里跳出或中断呢，这个时候我们就需要使用`label`

```
aaa:for (var row = 1; row <= 9; row++) {
    var str = "";
    bbb:for (col = 1; col <= row; col++) {
        str += row + "*" + col + "=" + row * col + "\t";
        if (col == 5) {
            break aaa;   //在这里通过label来跳出了指定的for循环
        }
    }
    console.log(str);
}
```

switch语句

switch语句也叫选择语句，它会选择符合条件的语句去执行，它与我们的`if`语句关系是最为亲密

`switch`语句应该与`case`语句结合在一起使用，它的语法格式如下

```
switch(条件的值){
    case 条件1:
        //条件1的代码体
        break;
    case 条件2:
        //条件2的代码体
        break;
    //这里可能还有很多个case
    default:
        //所有的条件都不符合要求的时候
        break;
}
```

上面的场景与我们之前所学习的if语句是非常相像的，它是根据一个条件来选择某一个部分的代码去执行，所以在某些情况下，`if`和`switch`两种语句是可以互相转换的

案例：今天的天气很好，同学们都想出去玩，但是桃子说要看下行的天气情况再决定，她说如果下雨，我们就在教室自习，如果下雪，我们就出去打雪仗，如果起风了，我们就去放风筝，如果是晴天，我们就去郊游，否则我们就在教室里面上课！

现在我们先使用`if`语句来完成

```
var weather = "晴天";
if (weather == "下雨") {
    console.log("自习");
}
else if (weather == "下雪") {
    console.log("打雪仗");
}
else if (weather == "起风") {
    console.log("放风筝");
}
else if (weather == "晴天") {
    console.log("郊游");
}
```

```
14    else {
15        console.log("上课");
16    }
```

> 上面的代码里面，我们使用了很多个 `if` 去完成，if太多对我们的阅读代码并不友好，所以我们可以换一种方式来完成这种业务逻辑

接下来，我们使用 `switch...case` 来完成

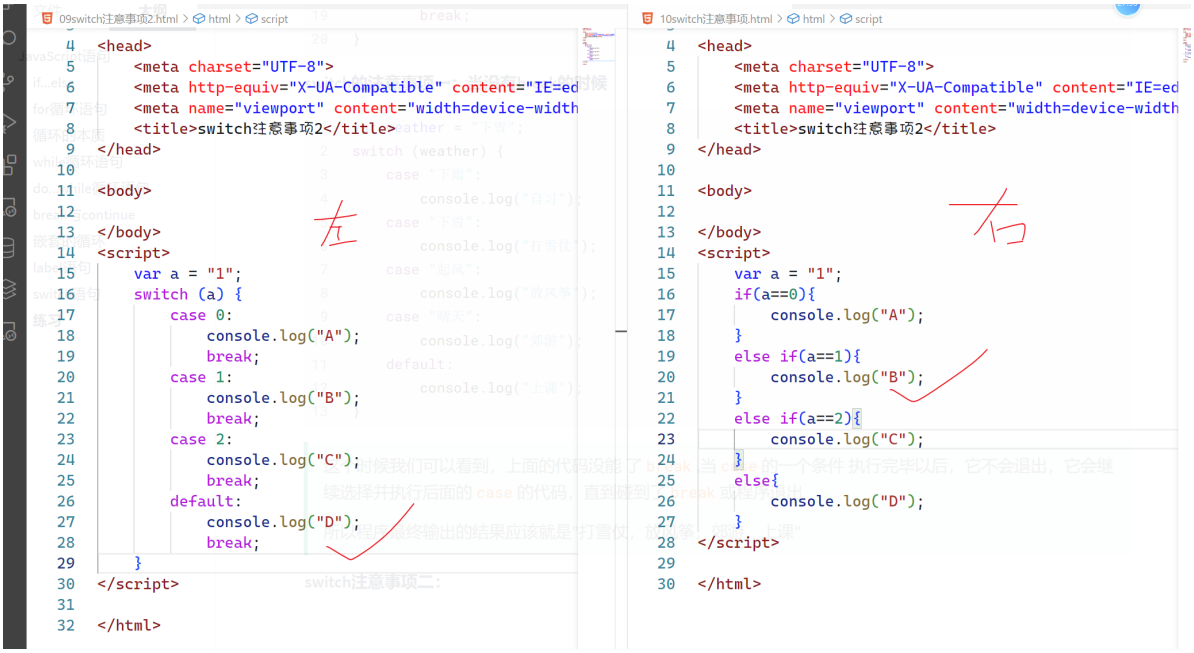
```
1   var weather = "下雪";
2   //swtich会根据weather的值来选择某一个
3   //符合要求的条件去执行
4   switch (weather) {
5       case "下雨":
6           console.log("自习");
7           break;
8       case "下雪":
9           console.log("打雪仗");
10          break;
11      case "起风":
12          console.log("放风筝");
13          break;
14      case "晴天":
15          console.log("郊游");
16          break;
17      default:
18          console.log("上课");
19          break;
20  }
```

switch的注意事项一：当没有break的时候

```
1   var weather = "下雪";
2   switch (weather) {
3       case "下雨":
4           console.log("自习");
5       case "下雪":
6           console.log("打雪仗");
7       case "起风":
8           console.log("放风筝");
9       case "晴天":
10          console.log("郊游");
11      default:
12          console.log("上课");
13  }
```

> 这个时候我们可以看到，上面的代码没能 了 `break` ,当 `case` 的一个条件 执行完毕以后，它不会退出，它会继续选择并执行后面的 `case` 的代码，直到碰到了 `break` 或程序退出
>
> 所以程序最终输出的结果应该就是"打雪仗，放风筝，郊游，上课"

switch注意事项二： `switch...case` 在执行判断的时候使用的是 `====` 全等操作



在上面的代码里面，我们可以很清楚的就看互它们的区别，左边打印的结果是 `D` ，右边的打印结果是 `B`

原因： `switch...case` 在做条件选择的时候，它使用的是强判断全等操作 `===` ，而右边的 `if` 语句里面我们使用的是 `==` 的相等操作，在相等操作里面 `"1"==1` 这是成立的，而全等里面则不相等

switch 语句在比较值时使用的是全等操作符，因此不会发生类型转换（例如，字符串"10"不等于数值 10）。

swtich...case注意事项三：当有多个 `case` 条件符合要求的时候，优先选择第一个执行

```
var a = 1;
switch (a) {
    case 1:
        console.log("A");
        break;
    case 2:
        console.log("B");
        break;
    case 1:
        console.log("D");
        break;
    default:
        console.log("F");
        break;
}
```

案例一：现有一个学生成绩的变量 score ，要根据学生成绩来定等级。等级划分如下： 100~90为优秀，80~89为良好，70~79为中等，60~69为及格，否则就不及格，请使用 switch 语句去完成代码

```
var score = 86;
switch (true) {
    case score >= 90:
        console.log("优秀");
        break;
    case score >= 80:
        console.log("良好");
        break;
```

```
    case score >= 70:
        console.log("中等");
        break;
    case score >= 60:
        console.log("及格");
        break;
    default:
        console.log("不及格");
        break;
}
```

下面还有一种思路

```
var score = 100;
var n = ~~(score / 10);
switch (n) {
    case 10:
    case 9:
        console.log("优秀");
        break;
    case 8:
        console.log("良好");
        break;
    case 7:
        console.log("中等");
        break;
    case 6:
        console.log("及格");
        break;
    default:
        console.log("不及格");
        break;
}
```

练习

1. 一张纸的厚度是0.0001米，将纸对折，对折多少次厚度超过珠峰高度8848米。

```
/**
     * 实始值:厚度0.0001米
     * 终止值:厚度8848米
     * 自变量：在原来的上面x2
*/
var count = 0;   //次数
for (var height = 0.0001; height <= 8848; height *= 2) {
    count++;      //每次对折以后，次数都要加1
}
console.log(count);
```

上面的代码其实我更愿意以下面的方式来完成

```
var count = 0;
var height = 0.0001;
for (; height <= 8848;) {
    count++;
    height *= 2;
}
console.log(count);
```

在上面的场景 里面，我们可以把 `for` 换成 `while` ，这样更好理解一些

```
var height = 0.0001;
var count = 1;

while(height<=8848){
    height *= 2;
    count++;
}
console.log(count);
```

2. 有一篮球从5米高处自由落下，每次弹起的高度是上一次的1/3，当篮球弹起的高度小于0.1米以后就不再弹起了，请问，这个篮球会弹起多少次?

```
// 结束条件：小于0.1
// 初始值：5m
var height = 5;
var count = 0;
// 一直弹，直到小于0.1
while (height >= 0.1) {
    height = height / 3;
    count++;
}
console.log(count);
```

3. 打印出所有的水仙花数(提示：水仙花数的范围在111~999之间) 水仙花是指一个三位数，它的每个位上的数字的3次幂之和等于它本身（例如：$1^3 + 5^3 + 3^3 = 153$）

第一种解法

```
for (var num = 111; num <= 999; num++) {
    // 百位
    var a = ~~(num / 100);
    // 十位
    var b = ~~(num % 100 / 10);
    // 个位
    var c = num % 10;
    if (num === a * a * a + b * b * b + c * c * c) {
        console.log(num);
    }
}
```

第二种解法

```
// 假设百位是a,十位是b,个位是c
for (var a = 1; a <= 9; a++) {
    for (var b = 0; b <= 9; b++) {
        for (var c = 0; c <= 9; c++) {
```

```
            // a:1,b:5,c:3
            /*
            if (a * a * a + b * b * b + c * c * c === a * 100 + b * 10 + c)
{
                console.log(a * 100 + b * 10 + c);
            }
            */
            if (a * a * a + b * b * b + c * c * c === +("" + a + b + c)) {
                console.log(a * 100 + b * 10 + c);
            }
        }
    }
}
```

4. 有一个台阶，如果一次跨2个，则最后剩下1阶，如果一次跨3个，则最后剩下2阶，如果一次跨5个，则最后剩下4阶，如果一次跨6个，则最后剩下5阶，如果一次跨7个，则刚刚好跨完，请问这个台阶最少为多少阶?

```
 // 我要一个一个的去找这个数，直到找到符合要求的为止
var num = 1;
while (true) {
    if (num % 2 === 1 && num % 3 === 2 && num % 5 === 4 && num % 6 === 5 &&
num % 7 === 0) {
        console.log(num);
        break;
    }
    num++;
}
```

5. 羽毛球拍15元，球3元，水2元。200元每种至少一个，有多少可能?

```
// 球拍是a，球是b，水是c
// 全部买球拍 200/15            1
// 全部买球 200/3               1
// 全部买水 200/2               1
for (var a = 1; a <= 200 / 15; a++) {
    for (var b = 1; b <= 200 / 3; b++) {
        for (var c = 1; c <= 200 / 2; c++) {
            if (a * 15 + b * 3 + c * 2 === 200) {
                console.log(a, b, c);
            }
        }
    }
}
```

6. 百马百担问题，有100匹马，驮100担货，大马驮3担，中马驮2担，2匹小马驮1担，求大、中、小各多少匹?

```
 /*
        设大马是a,中马是b,小马是c,现在列出如下的方程式
        a+b+c=100;
        a*3+b*2+c*0.5=100;

        a取值范围    0<=a<=100/3;
        b的取值范围   0<=b<=100/2;
        c的取值范围   0<=c<=100
```

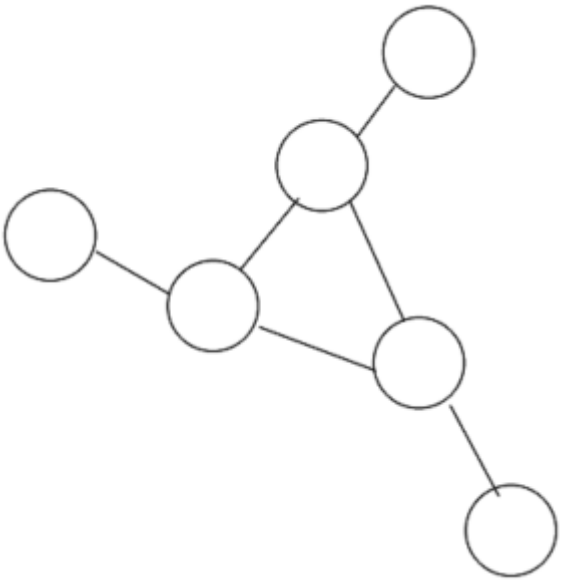
```
*/
for (var a = 0; a <= 100 / 3; a++) {  //大马
    for (var b = 0; b <= 100 / 2; b++) {    //中马
        for (var c = 0; c <= 100; c++) {        //小马
            if (a + b + c === 100 && a * 3 + b * 2 + c * 0.5 === 100) {
                console.log("大马：" + a + ",中马：" + b + ",小马：" + c);
            }
        }
    }
}
```

7. 括号里面只能放加或减，如果要使等式成立，括号里面应该放什么运算符

```
12 () 34 () 56 () 78 () 9 = 59
 /*
        在上面的代码里面，我们可以分析
        12+34可以理解为 12 + 34*1
        12-34可以理解为 12 + 34*-1
        所以每个括号里面，我们可以认为是-1~1这两个数

        现在我们把每个括号里面的东西用a,b,c,d去表示，则a,b,c,d应该都是-1 到 1
*/
outer:for (var a = -1; a <= 1; a += 2) {
    for (var b = -1; b <= 1; b += 2) {
        for (var c = -1; c <= 1; c += 2) {
            for (var d = -1; d <= 1; d += 2) {
                if (12 + 34 * a + 56 * b + 78 * c + 9 * d === 59) {
                    console.log(a,b,c,d);
                    break outer;
                }
            }
        }
    }
}
```

8. 请完成下面的题目



把1~6的数填入圆中，每条线上的数相加为10
使用编程思维解决

```
/**
    把小圆用abcdef6个变量去代替就可以了
```

```
*/
for (var a = 1; a <= 6; a++) {
    for (var b = 1; b <= 6; b++) {
        for (var c = 1; c <= 6; c++) {
            for (var d = 1; d <= 6; d++) {
                for (var e = 1; e <= 6; e++) {
                    for (var f = 1; f <= 6; f++) {
                        if (a + b + c === 10 && b + e + f === 10 && d + c + e == 10 && a + b + c + d + e + f === 21 && a * b * c * d * e * f === 720) {
                            console.log(a, b, c, d, e);
                        }
                    }
                }
            }
        }
    }
}
```

9. 打印出100以内的素数。

> 素数：除了1与自身以外不能被其它的数整除的数叫素数

```
// 除了1和自身以外，不能被其它的数整数
// 2~100之间，素数是从2开始的
//2<=x<=100

for (var x = 2; x <= 100; x++) {
    //假设型思维
    //我先假设这个数是素数，然后再反推
    var flag = true;
    //开始反推验证
    for (var n = 2; n < x; n++) {
        if (x % n === 0) {
            flag = false;
            //上面的假设已经被推翻了，就不再对这个数进行后面的验证了
            break;
        }
    }
    //在这里看一下假设的结果
    if(flag){
        console.log(x);
    }
}
```

10. 题目：日本某地发生了一起谋杀案，警察通过排查确定杀人凶手为4个嫌疑犯的一个，以下为四个嫌疑犯的供词

A说：不是我

B说：是C

C说：是D

D说：C在胡说

已知三个人说了真话，一个人说了假话，请编程找出凶手

```
/*
        1.四个人里面只有1个凶手
        2.四个人里面有3个说了真话

        假设条件:凶手我们用1表示，不是凶手就用0表示
        a + b + c + d == 1;
        所有人应该都有可能是凶手，也有可能不是凶手
        a,b,c,d四个人都有可能是0，1两个数
    */
outer:for (var a = 0; a <= 1; a++) {
    for (var b = 0; b <= 1; b++) {
        for (var c = 0; c <= 1; c++) {
            for (var d = 0; d <= 1; d++) {
                var aSay = a == 0;
                var bSay = c == 1;
                var cSay = d == 1;
                var dSay = !cSay;
                if (aSay + bSay + cSay + dSay == 3 && a + b + c + d == 1) {
                    console.log(a, b, c, d);
                    break outer;
                }
            }
        }
    }
}
```

11. 5位运动员参加了10米台跳水比赛，有人让他们预测比赛结果

    A选手说：B第二，我第三。

    B选手说：我第二，E第四。

    C选手说：我第一，D第二。

    D选手说：C最后，我第三。

    E选手说：我第四，A第一。

    比赛结束以后，每位选手都说对了一半，请编程确定比赛的名次

```
for (var a = 1; a <= 5; a++) {
        for (var b = 1; b <= 5; b++) {
            for (var c = 1; c <= 5; c++) {
                for (var d = 1; d <= 5; d++) {
                    for (var e = 1; e <= 5; e++) {
                        // 第一步： abcde不相等
                        if (a + b + c + d + e == 15 && a * b * c * d * e == 120) {
                            // 第二步： 按每位运动员的表述来判断
                            var aSay = (b == 2) ^ (a == 3);
                            var bSay = (b == 2) ^ (e == 4);
                            var cSay = (c == 1) ^ (d == 2);
                            var dSay = (c == 5) ^ (e == 3);
                            var eSay = (e == 4) ^ (a == 1);
```

```
                                // 第三步： 每位运动员都说对了一半
                                if (aSay && bSay && cSay && dSay && eSay) {
                                    console.log(a, b, c, d, e);
                                }
                            }
                        }
                    }
                }
            }
```

12. 有一对幼兔，幼兔1个月后长成小兔，小兔1个月后长成成兔并生下一对幼兔，问8个月后有多少对兔子，幼 兔、小兔、成兔对数分别是多少?

```
var yt = 1;
var xt = 0;
var ct = 0;
for (var month = 1; month <= 8; month++) {
    ct = xt + ct;
    xt = yt;
    yt = ct;
    console.log("第" + month + "月有成图" + ct + "只" + xt + "只" + yt + "只");
}
```

13. 猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不过瘾，又多吃了一个，第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第10 天早上想再吃时，见只剩下一个桃子了。求第一天共摘了多少？ （提示：采用逆向思维的方向， 从后向前推算）

```
var peach = 1;
for (var day = 9; day >= 1; day--) {
    peach = (peach + 1) * 2
}
console.log(peach);
```

终级题

1. 猜测产品质量评奖 5 家工厂的产品在一次评比中分获 1， 2， 3， 4， 5，在公布结果前，已知 E 厂产品肯定不是第二、三名，五厂代表猜测评比结果，

   A 厂的代表说： E 厂一定能获得第一名。

   B 厂的代表说：我厂的产品可能获第二名。

   C 厂的代表说： A 厂的产品质量最次。

   D 厂的代表说： C 厂的产品不是最好的。

   E 厂的代表说： D 厂的产品会获得第一名。

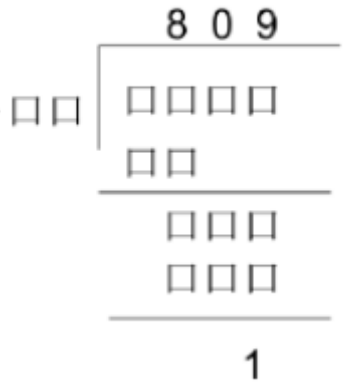
   公布结果后，证明只有产品获第一名和第二名的两个厂的代表猜对了。求 5 个厂产品各获第几名。

```

```

```
 /*
        1.E厂产品肯定不是第二、三名
        2.第一名和第二名的两个厂的代表猜对了
        3.1+2+3+4+5=15;1*2*3*4*5=120
        1<=a<=5;
        1<=b<=5;
        1<=c<=5;
        1<=d<=5;
        1<=e<=5;
*/
for (var a = 1; a <= 5; a++) {
    for (var b = 1; b <= 5; b++) {
        for (var c = 1; c <= 5; c++) {
            for (var d = 1; d <= 5; d++) {
                for (var e = 1; e <= 5; e++) {
                    if (a + b + c + d + e === 15 && a * b * c * d * e === 120) {
                        if (e != 2 && e !== 3) {
                            // 判断说的话
                            var aSay = a <= 2 && e == 1;
                            var bSay = b <= 2 && b == 2;
                            var cSay = c <= 2 && a == 5;
                            var dSay = d <= 2 && c != 1;
                            var eSay = e <= 2 && d == 1;
                            if (aSay + bSay + cSay + dSay + eSay == 2) {
                                console.log(a, b, c, d, e);
                            }
                        }
                    }
                }
            }
        }
    }
}
```

2. 填写数字 设有算式如图所示,求出口中的数字,并打印出完整的算式



```
for (var x = 10; x <= 99; x++) {
    for (var y = 1000; y <= 9999; y++) {
        if (y % x === 1 && ~~(y / x) === 809) {
            console.log(y,x);
        }
    }
}
```

3. 海滩上有一堆桃子，五只猴子来分，第一只猴子把这堆桃子平均分为五份，多了一个，这只猴子把多的一个扔入海中，拿走了一份；第二只猴子把剩下的桃子又分成了五份，多了一个，它同样把多的一个扔入海中，拿走一份，第三、四、五只猴子都是这么做的，问海滩上原来最少有多少个桃子?

4. 题目：有一分数序列：2/1，3/2，5/3，8/5，13/8，21/13...求出这个数列的前10项之和。

```
var fz = 2;
var fm = 1;
var sum = 0;
for (var i = 1; i <= 10; i++) {
    sum += fz / fm;
    var temp = fz;      //temp就是原来的fz
    // 新的数的分子就应该是 原来的分母 + 原来的分子，
    fz = fm + fz;
    // 新的分母就应该是原来的分子，
    fm = temp;
}
console.log(sum);
```

5. 分书问题

   有A、B、C、D、E五本书，要分给张、王、李、赵、钱五位同学，每人只能选一本，事先让每人把自己喜爱的书填于下表，编程找出让每个人都满意的方案。

	A	B	C	D	E
张			√	√	
王	√	√			√
李		√	√		
赵	√	√		√	
钱		√			√