

第3章 语言结构

+ 在前面的章节中，我们已经学习了PHP的基本语法。为本章的学习做好了基础，在本章里，我们将要学习的是PHP语言结构。语言结构是PHP开发的核心部分，它是整个程序的整体框架。框架搭建出了问题，以后再做什么也是白费力气，因此我们一定要理解好本章的内容。我们将会先从最简单的语句开始学习，逐步深入到后面的流程控制，让读者不至于学习得很费力。

3.1 语句

- + 在前面的章节中我们已经学习了表达式。语句通常就是由表达式组成的，PHP程序就是由多条语句组成的。可以说没有语句，PHP程序就无法建立起来，下面我们就进入语句的学习。

3.1.1 什么是语句

- + 语句就是一条完整的句子。在PHP中，语句通常是由一个表达式加分号 (;) 结尾构成。最简单的语句就是空语句（只有一个分号）。下面我们就举例出几条语句，让读者来认识一下语句。
- + 以下都是正确的语句。
- + `$a=10;`
- + `$b="你好!";`
- + `$c=$v+5;`
- + `echo "hello PHP";`
- + `;` // 虽然只有一个分号，但是它也是一条语句

3.1.2 语句块

+ 在PHP中，我们为了程序结构更加清晰和阅读更加容易，通常把完成某个功能的多个语句用花括号括起来，它就构成了一个语句块。语句块是作为一个整体，在程序执行时，当遇到该语句块时，就执行该语句块中的语句。下面来看一个示例。

语句块
开始

{

.....
.....
.....

这里是多条语句

语句块
结束

}

3.1.2 语句块

不使用语句块

```
<?php
    $a=78;
    $b=85;
    $c=92;
    $d=$a+$b+$c;
    $e=$d/3;
    echo $e;
?>
```

使用语句块

```
<?php
    $a=78;
    $b=85;
    $c=92;
    {
        $d=$a+$b+$c;
        $e=$d/3;
    }
    echo $e;
?>
```

3.1.3 语句的执行流程——顺序执行

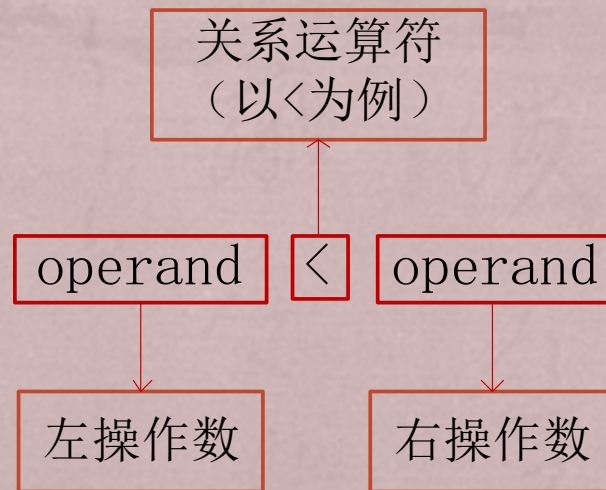
- + 虽然示例中的程序由很多个语句构成，并且其中还包括语句块。但是输出结果，仍是由上而下顺序执行。我们来看代码。

3.2 条件的构成

+ 在上一节的最后，我们提出了很多时候顺序执行程序有些时候是不能解决问题的。我们就需要学习另一种语句结构了，那就是分支结构。我们在学习语句分支结构之前，首先要学习条件的构成。因为语言结构实现分支，通常是通过条件判断来实现的。在PHP中常用来做条件判断的语句是关系运算表达式和逻辑运算表达式。只有掌握了这些运算，才可以完成分支语句。这里我们就来学习它们。

3.2.1 关系运算

- + 关系运算符是在PHP中比较常用的二元操作符，又被称作条件运算符或者比较运算符。它的作用是用对运算符两边的操作数进行比较，从而构成一个条件。关系运算表达式的值为布尔值，也就是关系成立即为真（TRUE），关系不成立即为假（FALSE）。



3.2.1 关系运算

+ 关系运算符较简单，大部分等同于我们数
学中学到的比较运算。这里不做太多讲解，
我们用表列出这些运算符及其作用。

运算符	名称	说明	示例
>	大于	左操作数大于右操作数返回真，否则返回假	\$a>\$b
<	小于	左操作数小于右操作数返回真，否则返回假	\$a<\$b
>=	大于等于	左操作数大于等于右操作数返回真，否则返回假	\$a>=\$b
<=	小于等于	左操作数小于等于右操作数返回真，否则返回假	\$a<=\$b
==	等于	左操作数等于右操作数返回真，否则返回假	\$a==\$b
===	全等于	左操作数等于右操作数（包括类型）返回真，否则返回假	\$a=== \$b
<>或!=	不等于	左右操作数不相等返回真，否则返回假	\$a!= \$b
!==	非全等	左右操作数数值或者类型不相等返回真，否则返回假	\$a!== \$b

3.2.1 关系运算

+ 下面我们来看一段输出比较运算结果的代码。

```
<?php
```

```
$a=3;
```

```
$b=5;
```

```
echo '$a<$b '.($a<$b).'
```

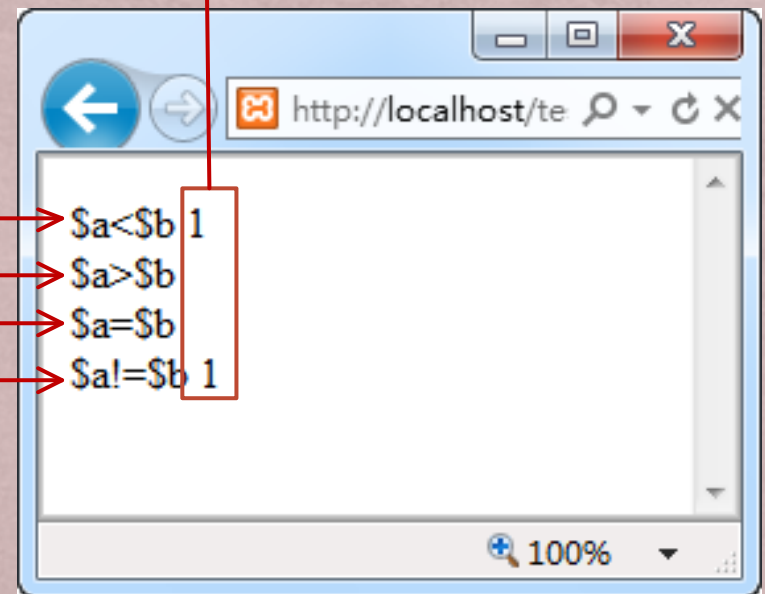
```
echo '$a>$b '.($a>$b).'
```

```
echo '$a=$b '.($a==$b).'
```

```
echo '$a!=$b '.($a!=$b).'
```

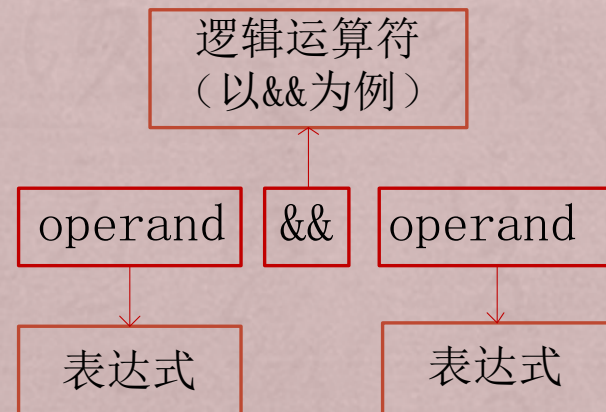
```
?>
```

这里就是比较结果
1为TRUE，空为FALSE



3.2.2 逻辑运算

+ 逻辑运算是用来判断一条或者多条表达式是成立还是不成立的，逻辑运算的核心是逻辑运算符。它可以帮助我们构建多个条件的组合。在PHP中，逻辑运算符只能操作布尔型的表达式，而且返回的值也是布尔值。常被用作结构语言的条件判断。



3.2.2 逻辑运算

运算符	名称	说明	示例
and或&&或&	逻辑与	当两个操作数均为TRUE则返回TRUE，否则返回FALSE	\$a&&\$b
or或 或	逻辑或	当两个操作数中有一个为TRUE则返回TRUE，否则返回FALSE	\$a \$b
not或!	逻辑非	操作数为FALSE则返回TRUE，为TRUE则返回FALSE	!\$a
xor或^	逻辑异或	当两个操作数一个为TRUE另一个为FALSE则返回TRUE，否则返回FALSE	\$a xor \$b

&&	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

^	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	FALSE

/	TRUE	FALSE
结果	FALSE	TRUE

3.2.2 逻辑运算

以下代码输出逻辑运算的结果。

以下框中就是逻辑运算果
1为TRUE，空为FALSE

```
<?php
```

```
$a=TRUE;  
$b=FALSE;
```

```
echo '$a&&$b'.('.$a&&$b').<br/>'
```

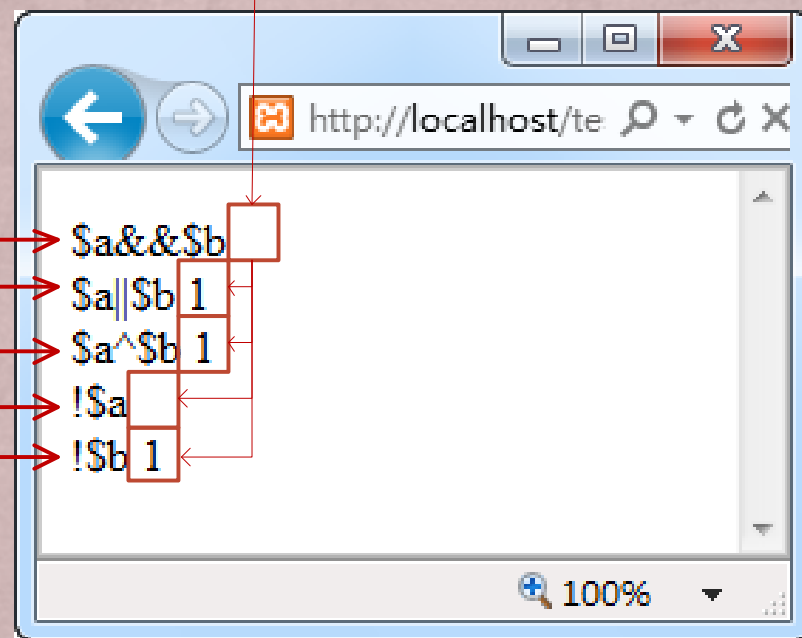
```
echo '$a||$b'.('.$a||$b').<br/>'
```

```
echo '$a^$b'.('.$a^$b').<br/>';
```

```
echo '!'$a'.('.$a).<br/>'
```

```
echo '!'$b'.('.$b).<br/>'
```

```
?>
```



3.2.2 逻辑运算

在逻辑判断语句中我们还需要注意的是逻辑与和逻辑或的短路问题，在平常编程中我们常用的是 (`&&`) 和 (`||`)，他们和 (`&`) (`|`) 的区别就是有短路原则。也就是说，逻辑与，只要第一个操作数的值为 `FALSE`，它就不会再去验证或者执行第二个表达式了，因为表达式结果已经确定为 `FALSE` 了。

如为TRUE

exp1

||

exp2

则不执行

如为FALSE

exp1

&&

exp2

则不执行

3.2.2 逻辑运算

以下代码演示了使用逻辑与（&&）、逻辑或（||）的短路原则。

```
<?php
    $a=1;
    $b=2;

    $a||($b++);

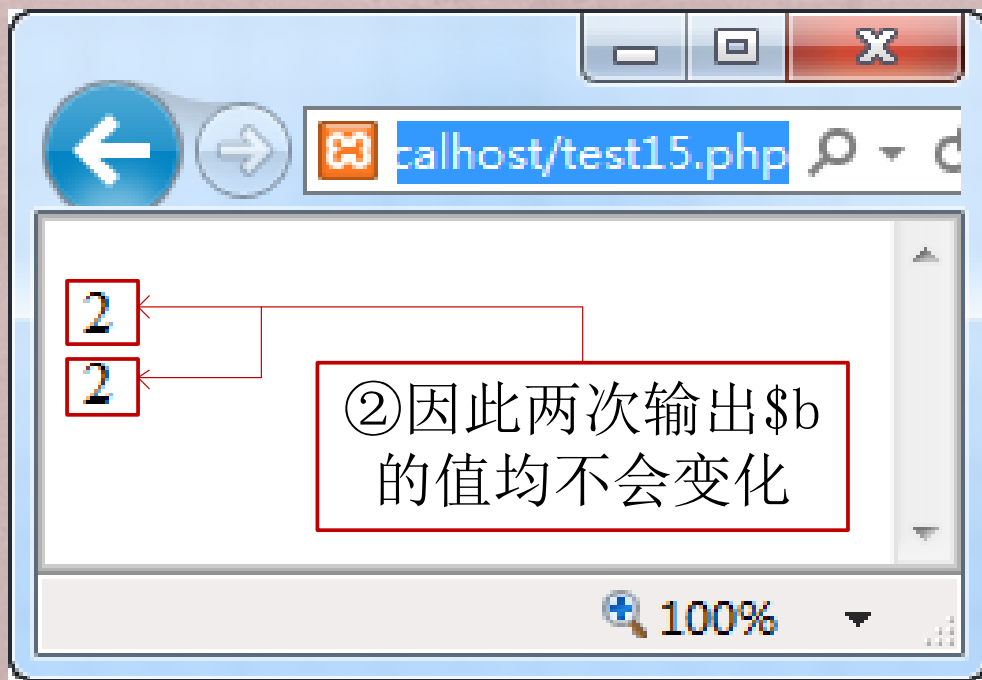
    echo $b;

    $a=FALSE;

    $a&&($b++);

    echo '<br />'.$b;
?>
```

①这两处均不会被执行



3.2.2 逻辑运算

以下代码演示了使用逻辑与 (&)、逻辑或 (|) 消除逻辑与 (&&) 和逻辑或 (||) 的短路原则。

```
<?php
$a=1;
$b=2;

$a | ($b++);

echo $b;

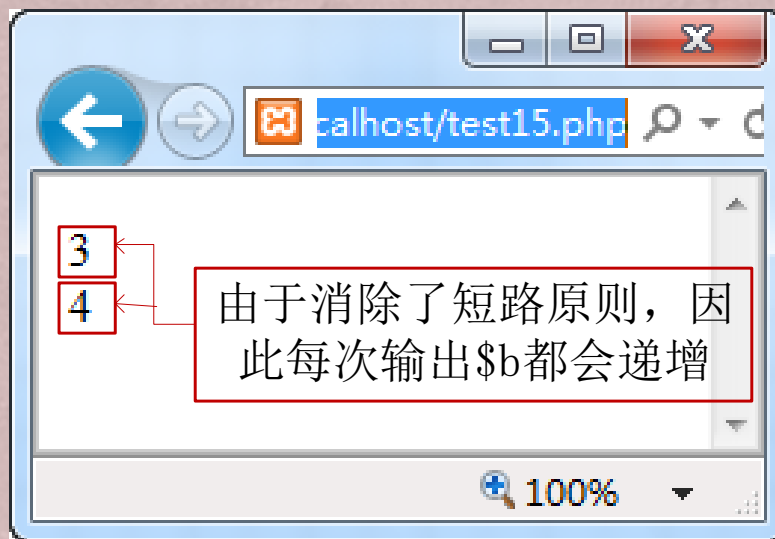
$a=FALSE;

$a & ($b++);

echo '<br />'.$b;
?>
```

这里我们用单
竖线 (|) 或

这里我们用单
and (&) 与



3.3 分支结构

+ 在前面我们所学习讲解和自己写的案例中，我们可以看出所写的代码基本没有什么转折，都是从上到下顺序就执行下来了，而这样的结构有的功能往往是不能实现或者是很难实现的，于是分支结构语句就随之产生了。分支结构语句主要包含四种。

+ if语句

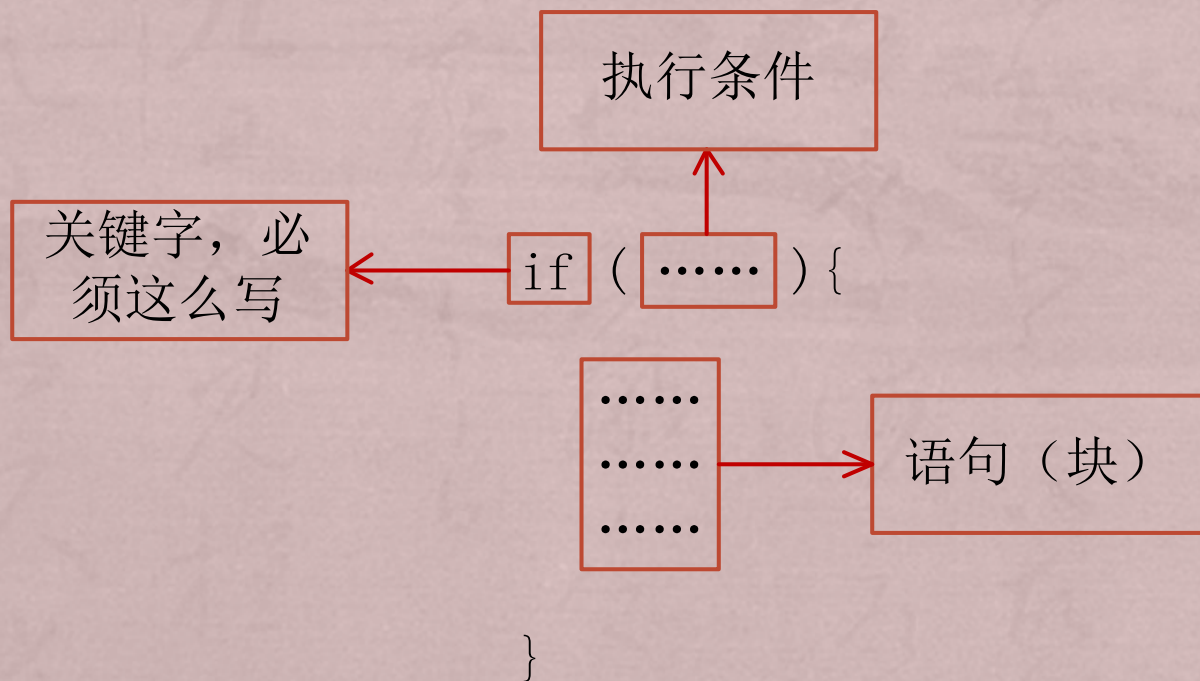
+ if...else语句

+ if...elseif...if语句

+ switch语句

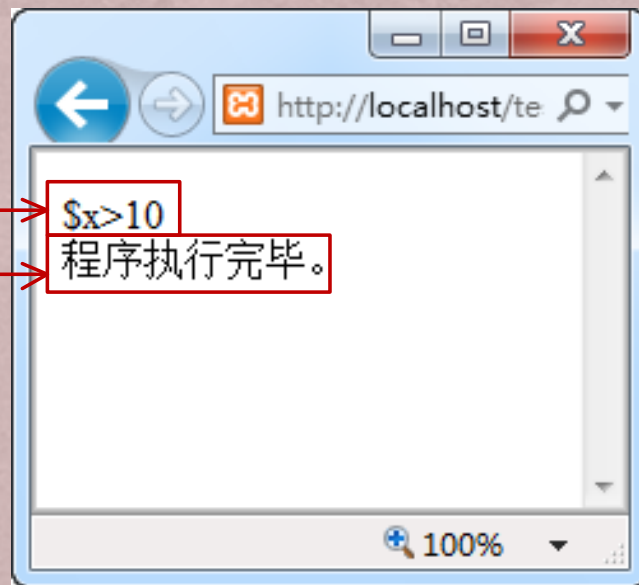
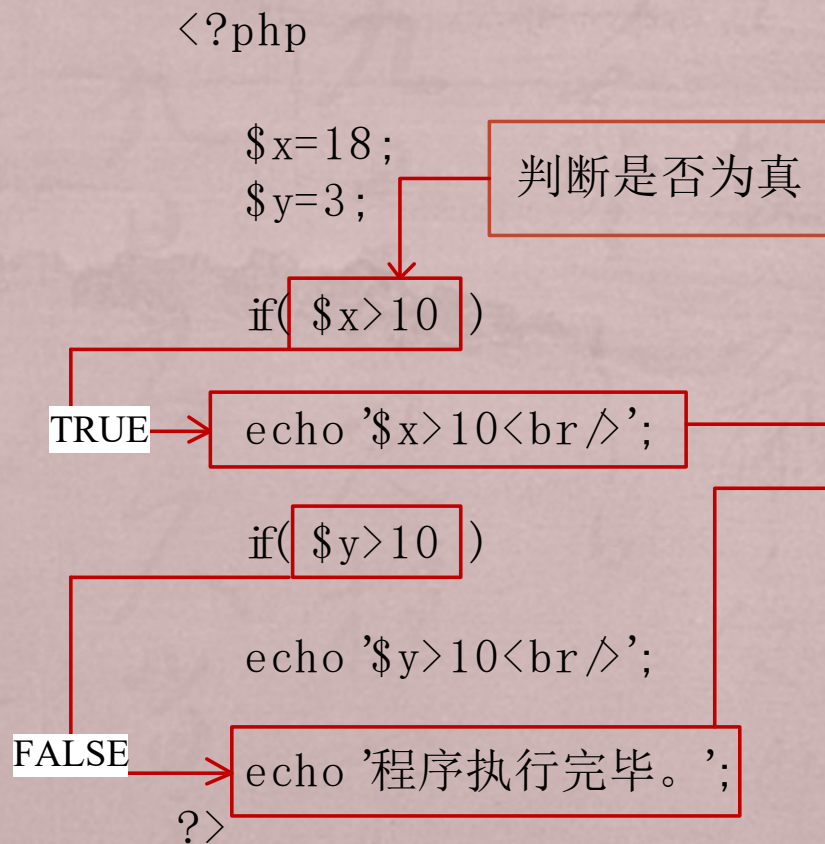
3.3.1 IF语句

- + if翻译成我们的汉语就是如果的意思，我们可以这样理解：如果某个条件成立，就做某件事情。在if语句中就是如果条件语句成立，就执行条件判断语句后面的一句语句。



3.3.1 IF语句

- + 以下代码定义两个变量，然后使用if语句依次判断是否大于10，并输出判断结果。



3.3.1 IF语句

+ 以下代码定义两个变量，通过if语句判断是否为‘小明’和‘小陈’，并输出判断结果。

```
<?php
```

```
$a='xiaom ing';  
$b='xiaochen';
```

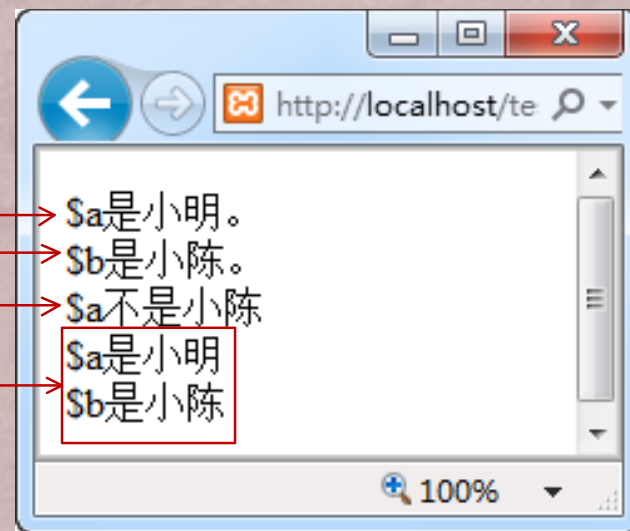
```
if($a=='xiaom ing')echo '$a是小明。';
```

```
if($b=='xiaochen'){  
    echo '<br/>$b是小陈。';  
}
```

```
if($a=='xiaochen')echo '$a是小陈';echo '<br/>$a不是小陈';
```

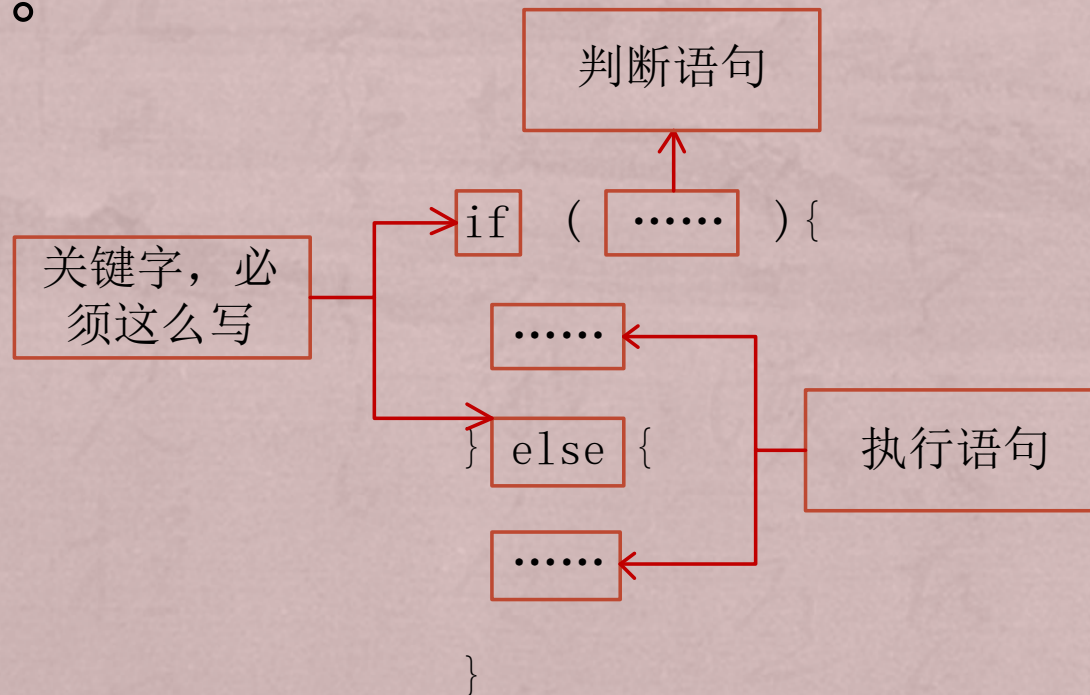
```
if($a=='xiaom ing'&&$b=='xiaochen'){  
    echo '<br/>$a是小明';  
    echo '<br/>$b是小陈';  
}
```

```
?>
```



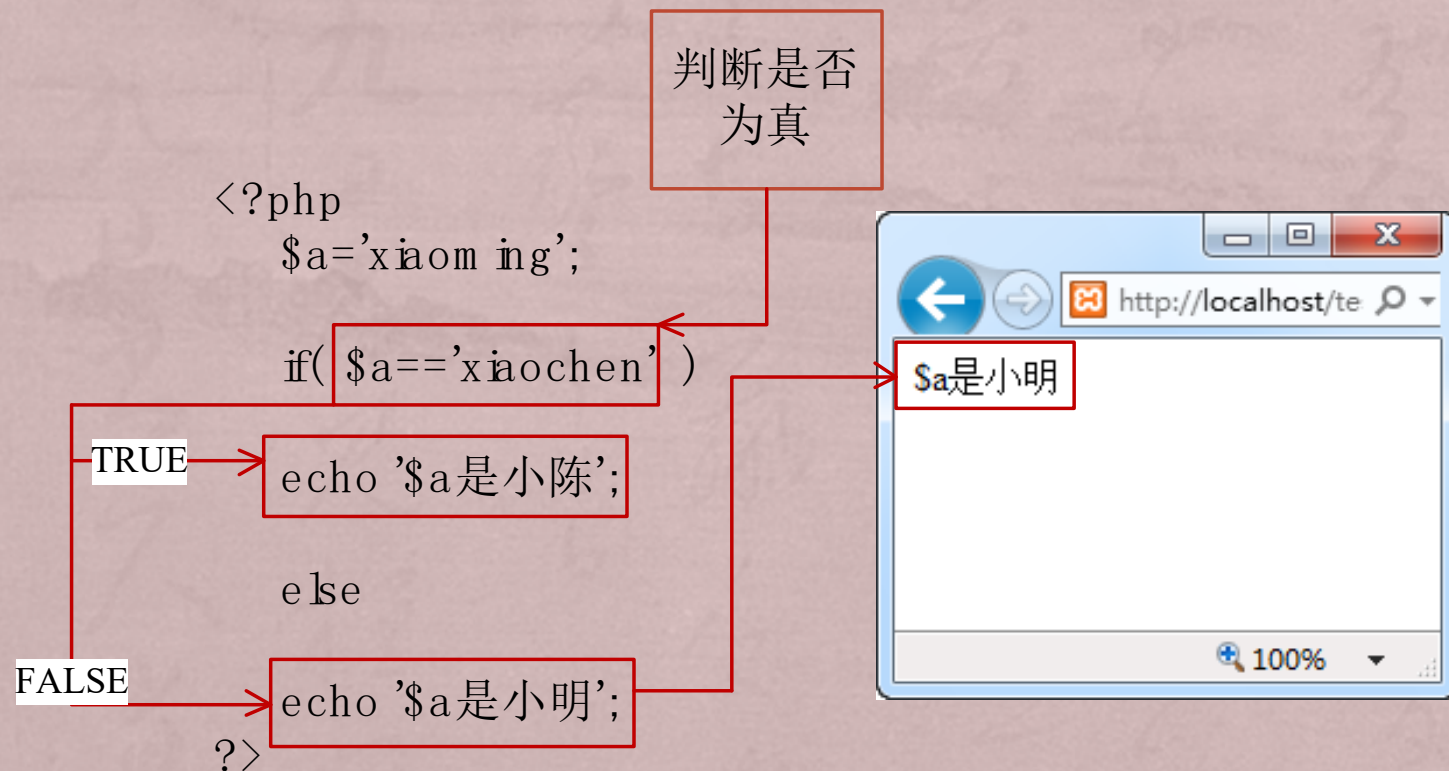
3.3.2 IF...ELSE语句

- + if...else语句和if语句很类似，只是多了一个条件判断。else中文含义就是否则的意思，从字面意义我们就可以这样理解：if后面的判断条件不成立，那么就执行else后的语句。



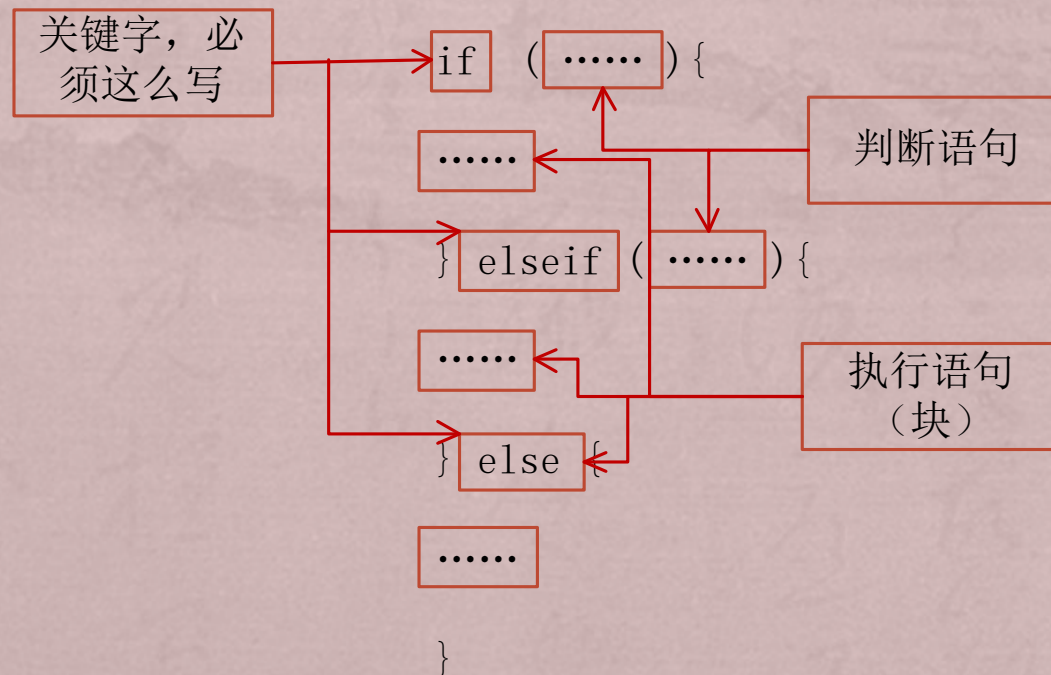
3.3.2 IF...ELSE语句

- + 以下代码演示使用if...else语句判断定义的一个变量的值是 'xiaochen' 还是 'xiaoming' 的执行步骤。



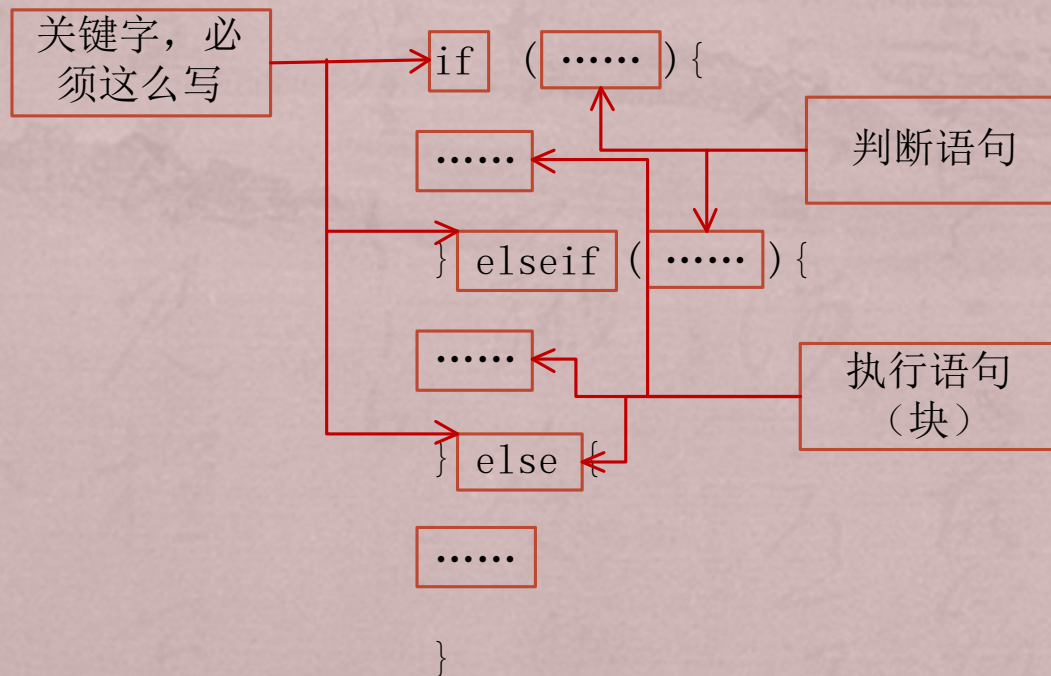
3.3.3 IF...ELSEIF...ELSE语句

- + if...elseif...else语句也是if语句的一种衍生，它的作用是根据不同的条件执行不同的语句，类似于多个if...else嵌套。下面我们看它语法的结构。



3.3.3 IF...ELSEIF...ELSE语句

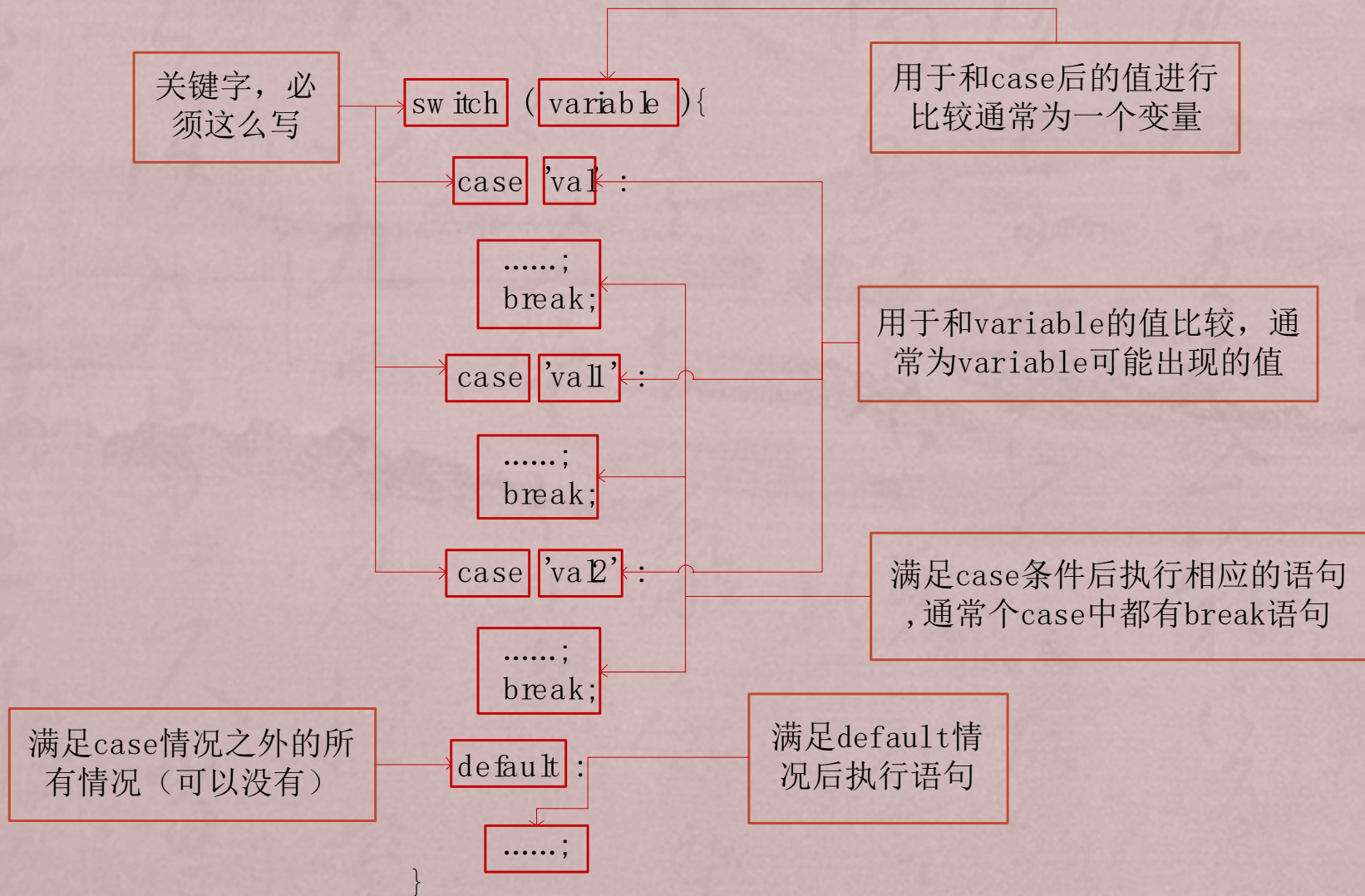
- + 以下代码是通过使用if...elseif...else语句来判断一个变量在什么范围内，并输出对应的结果的案例。



3.3.4 SWITCH语句

- + switch语句和前面讲到的if...elseif...else语句相类似，也是根据不同的条件执行不同的语句。和if...elseif...else语句不同点在于switch常用于对不同的值判断作出响应，图中表示switch语句的语法结构。

3.2.4 SWITCH语句



3.3.4 SWITCH语句

- (1) 使用switch语句判断电脑的三种状态‘运行’、‘重启’、‘关闭’，并输出相应提示。
- (2) 使用switch语句判断电脑的三种状态‘运行’、‘重启’、‘关闭’，并输出相应提示。（无break）
- (3) 使用switch结构不写break实现计算本周剩余天数。
- (4) 展示case语句为空的情况。

3.3.5 分支结构的嵌套

+ 前面我们所讲的分支结构都是包含一种分支结构语句的程序，其实在分支语句中还可以使用分支语句，这就叫做嵌套。这种结构运用在某个分支选择后又出现新的分支的情况，下面我们就用适婚年龄的例子来讲解多分支结构。

3.4 循环结构

+ 在讲本节之前我们先假设有这么一个需求，就是我们需输出同样的话五次，这个对读者来说，就最简单不过了，我们可以这样写：

+ <?php

+ echo '你好， 中国';

+ echo '你好， 中国';

+ echo '你好， 中国';

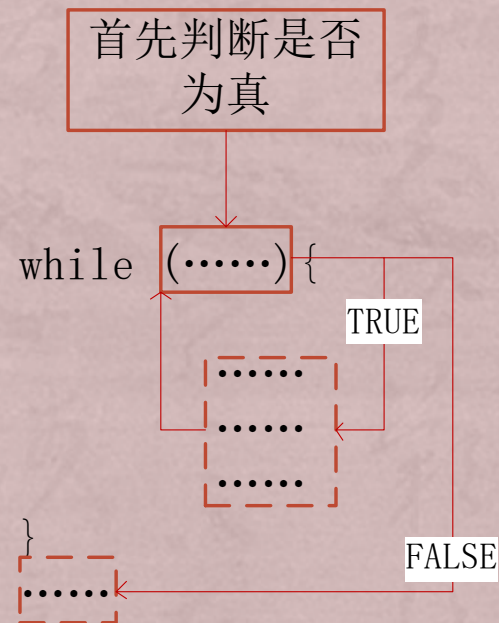
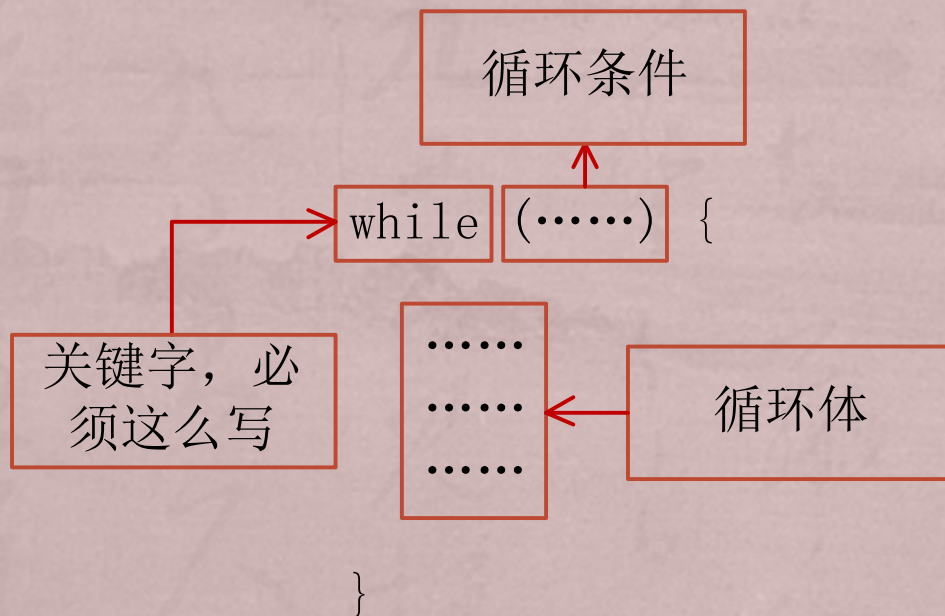
+ echo '你好， 中国';

+ echo '你好， 中国';

+ ?>

3.4.1 WHILE语句

+ while循环是PHP中最简单的循环类型。

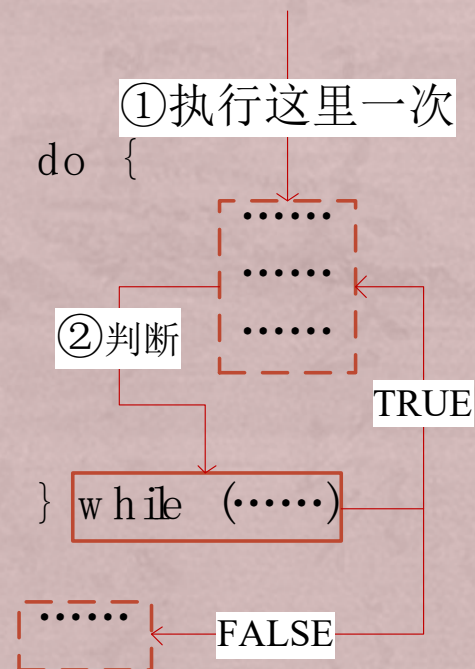
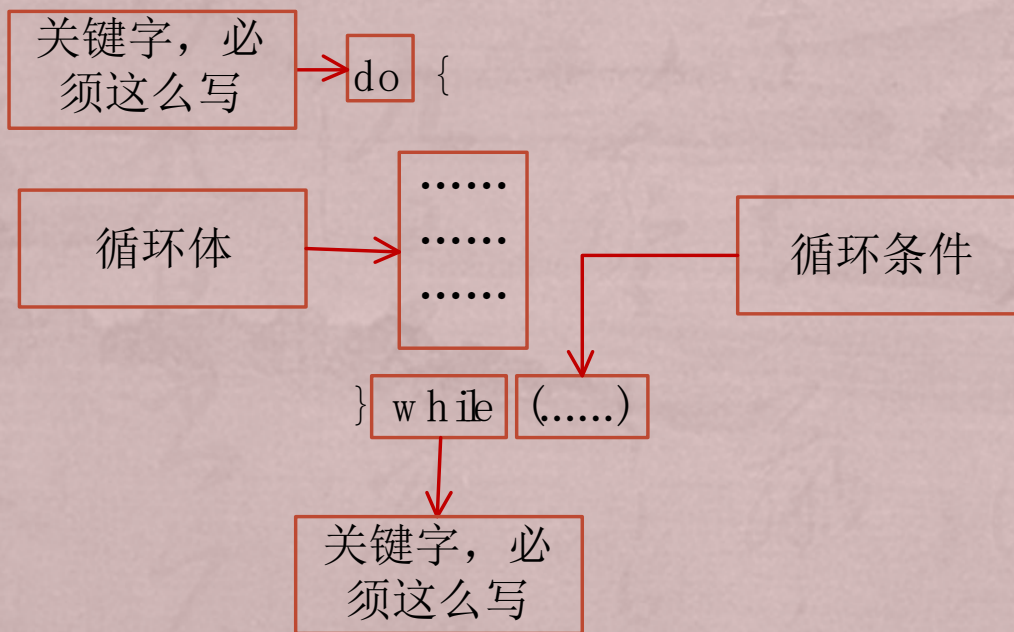


3.4.1 WHILE语句

- + 使用while循环实现输出输出五条“你好，中国”。

3.4.2 DO...WHILE语句

+ do...while循环和while循环非常相似。

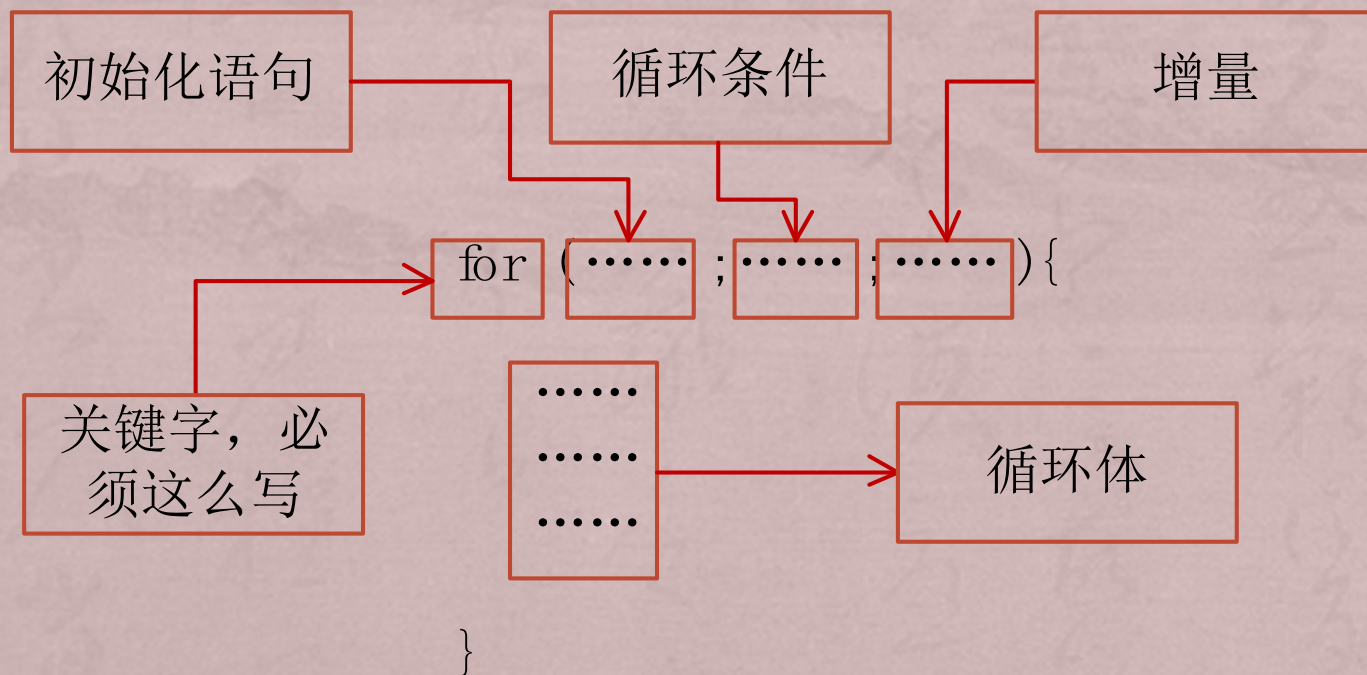


3.4.2 DO...WHILE语句

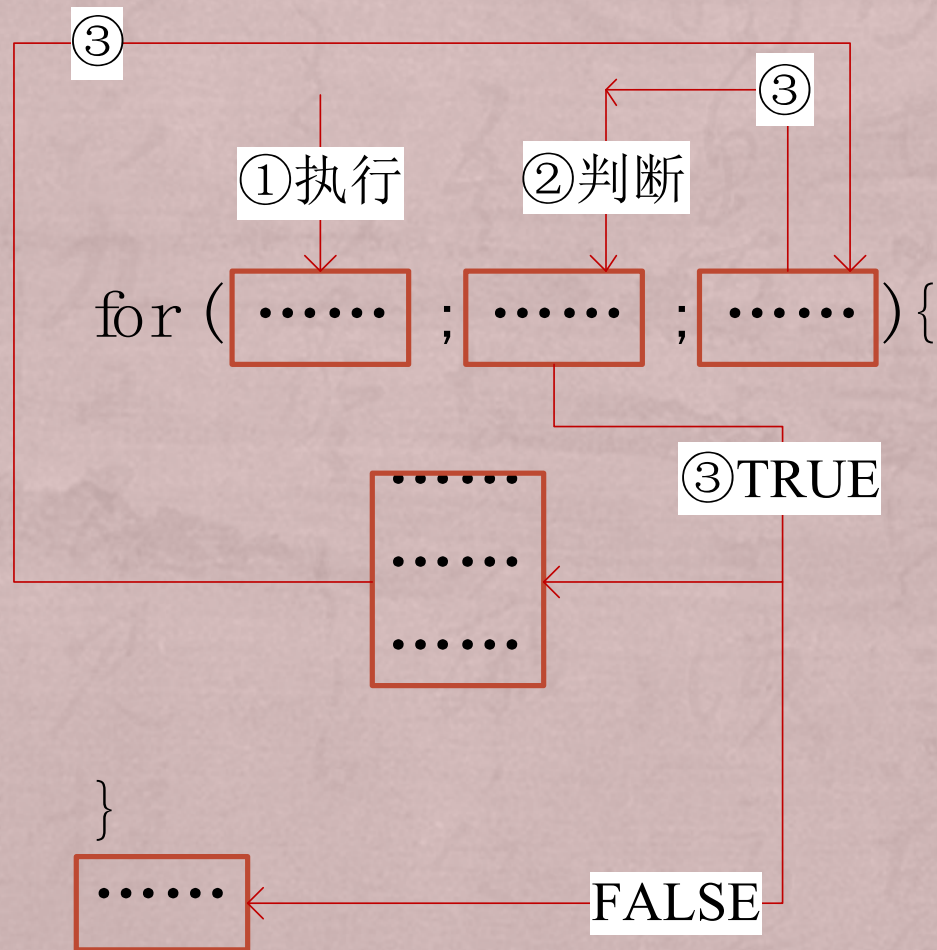
- + (1) 使用do...while循环实现输出五句“你好，中国”。
- + (2) do...while循环的特殊使用方式。(判断条件为0)

3.4.3 FOR循环语句

- + for循环语句常被说成是循环语句里面最复杂的循环语句，其实完全没有什么复杂的，for循环语句是简洁强大的循环语句。



3.4.3 FOR循环语句



3.4.3 FOR循环语句

for循环语句的写法也就比较多样，如下所示。

```
for($a=0;$a<10;$a++){.....}
```

```
for(;$a<10;$a++){.....} //初始化语句为空
```

```
for(;;$a++) //初始化和循环条件语句为空
```

```
for(;;){.....} //所有语句都为空
```

```
for($x=1,$y=3,$z=5;$x<8,$y>3,$z<=10;$x++,$y++,$z++){.....}
```

//各语句中有多个表达式，表达式间用逗号

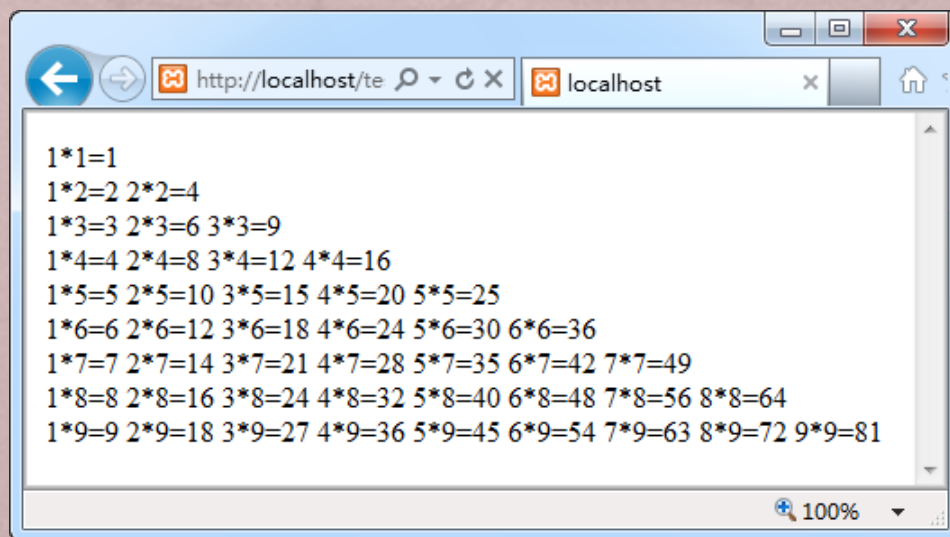
```
for($c='apple',$d=5;$d<10;$d++){.....} //初始化语句中有多个表达式
```

(1) 使用for循环实现输出五句“你好，中国”。

(2) 演示for循环语句的特性。

3.4.4 循环结构的嵌套

+ 前面我们在学习分支结构的时候，学习了分支结构的嵌套，那么作为同是语言结构的循环结构，也是可以嵌套的，我们常用九九乘法表的案例来讲解本节的内容。九九乘法表相信大多数读者都比较熟悉。



A screenshot of a web browser window showing a 9x9 multiplication table. The browser's address bar displays 'http://localhost/te' and the page title is 'localhost'. The multiplication table is rendered as a single block of text with each row on a new line. The bottom right corner of the browser window shows a zoom level of 100%.

```
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
```

3.4.4 循环结构的嵌套

- + (1) 使用for循环实现输出九九乘法表。
- + (2) 使用while循环实现输出九九乘法表。

3.4.5 跳转语句

- + 在PHP中有一些跳出结构的语句，就比如我们将switch语句的时候用到的break就是一种跳出结构的语句，跳转语句也是也是一种控制结构的语句。
- + 常用的跳转语句有continue、break、return。

1.CONTINUE语句

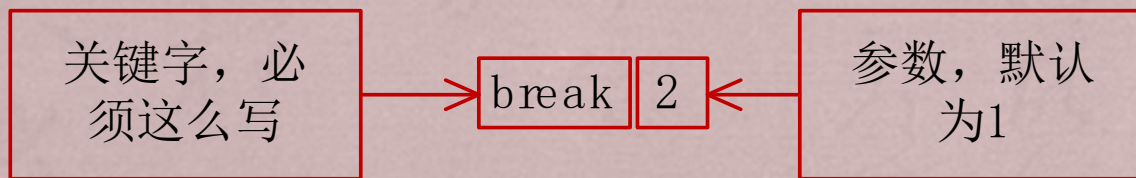
- + continue语句用来在循环结构中跳过本次循环中剩余的代码，并在循环条件为真时开始执行下一次循环。我们用循环输出1到15间的奇数来认识一下continue的作用。

1.CONTINUE语句

- + 输出1到15间的奇数。看代码。
- + continue语句后面可以接受一个整型参数，用来控制一次跳出几层循环结构。看代码。

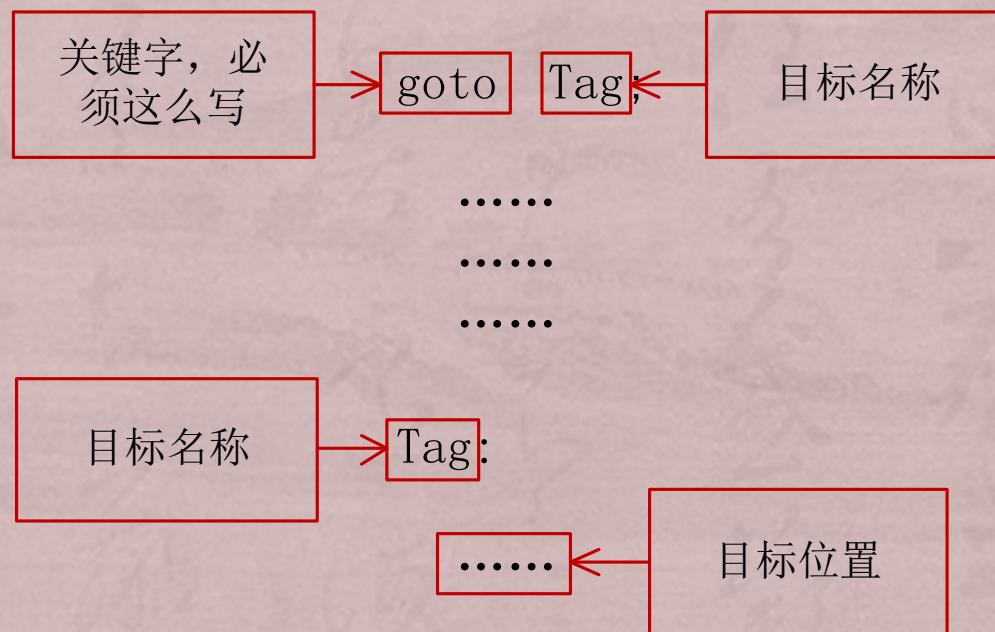
2.BREAK语句

- + break语句常用来结束当前for、foreach、while、do-while或者switch结构的执行。也就是说，程序结构执行中遇到break语句就会跳出这个结构，执行结构后面的语句。我们已经在学习switch结构的时候多次使用到了break语句。看代码。



3.GOTO语句

+ goto 操作符可以用来跳转到程序中的某一指定位置。该目标位置可以用目标名称加上冒号来标记。PHP中的goto是有一定限制的，它无法跳入到任何循环或者switch结构中。常见的用法是用来跳出循环或者switch，可以代替多层的break。看代码。



4.RETURN语句

- + 在PHP中，除了以上我们讲的三种跳转语句外，还有一种跳转语句叫做return语句，这个跳转语句由于我们现在所学的知识还不足以理解它的用法，因此我们将会讲解函数的章节里面讲解。

3.4 小结

- + 本章我们学习了PHP的语言结构，通常来说，掌握一门语言最主要的就是在语言结构上，结构就是一个大的框架，它会指导程序该按什么样的顺序执行，就比如程序出什么问题了，我们就需要用哪个方法来解决，是要出现分支的，而不是不管有什么问题，程序都一口气往下执行。那就什么也做不成了。只有这个结构做好了，我们才可以添砖加瓦来一步步地完善它。