2-4基础语法-流程控制与运算符

1. 课堂案例

课堂讲解

一、上节回顾

常量

变量

单双引号的区别

数据的类型

二、学习目标

算术运算符

字符串运算符

赋值运算符

比较运算符

逻辑运算符

其他运算符

优先等级

错误类型

判断语句

循环语句

三、教学过程描述

1、流程控制

1-1、if

单向分支结构:

```
if(条件表达式){
php语句;
php语句;
...
 }
```

条件表达式为真，执行{}中的所有PHP语句，作用范围(if表达式之后的{}中的所有内容)

双向分支结构:

```
if(条件表达式){
 一个表达式;
多个表达式;
}else{
  另外一个表达式;
  多个表达式;
}
```

多向分支结构:

```
if(条件表达式){

 }elseif(条件表达式){

  }elseif(条件表达式){

  }else{

}
```

1-2、switch

```
switch(变量){
```

case 值: PHP语句; break;

//break语句可选，不加break则满足条件后继续之后后面的语句不跳出分支结构

case 值: PHP语句; break;

...

default:

PHP语句;

}

1-3、while

while(条件表达式){

循环体;

}

1-4、do...while

do{

循环体;

}while(条件表达式);

先执行一遍循环体，在进行判断，根据判断结果决定是否继续执行循环

1-5、for

for(初始化变量1;给定变量范围;增量){

执行循环的语句;

}

【执行过程:】先是初始化->判断变量范围-> 满足执行循环->执行增量->判断变量范围->满足执行循环->执行增量…

1-6、foreach

foreach()数组遍历专用

foreach ($array as $key => $value)

{

要执行代码;

}

2、运算符

优先等级

无	clone new	clone 和 new
左	[	array()
右	**	算术运算符
右	++ -- ~ (int) (float) (string) (array) (object) (bool) @	类型和递增 / 递减
无	instanceof	类型
右	!	逻辑运算符
左	* / %	算术运算符
左	+ - .	算术运算符和字符串运算符
左	<< >>	位运算符
无	< <= > >=	比较运算符
无	== != === !== <> <=>	比较运算符
左	&	位运算符和引用
左	^	位运算符
左	|	位运算符
左	&&	逻辑运算符
左	||	逻辑运算符
左	??	比较运算符
左	? :	ternary
right	= += -= *= **= /= .= %= &= |= ^= <<= >>=	赋值运算符
左	and	逻辑运算符
左	xor	逻辑运算符
左	or	逻辑运算符

2-1、算术运算符

算术加(+)、

算术减(-)、

算术乘(*)、

算术除(/)、

算术取余(%)

PHP7+ 版本新增整除运算符 intdiv(),使用实例：

var_dump(intdiv(10, 3));//输出3

```php
$a = 8;
$b = 3;
echo $a + $b.'<br>';
echo $a - $b.'<br>';
echo $a * $b.'<br>';
echo $a / $b.'<br>';
echo $a % $b.'<br>';
```

2-2、赋值运算符

基本的赋值运算符是 "="。

它意味着左操作数被设置为

右侧表达式的值

运算符	等同于	描述
x = y	x = y	左操作数被设置为右侧表达
x += y	x = x + y	加
x -= y	x = x - y	减
x *= y	x = x * y	乘
x /= y	x = x / y	除
x %= y	x = x % y	模（除法的余数）
a .= b	a = a . b	连接两个字符串

```php
$x=10;
echo $x; // 输出10
$y=20;
$y += 100;
echo $y; // 输出120
$z=50;
$z -= 25;
echo $z; // 输出25
$i=5;
$i *= 6;
echo $i; // 输出30
$j=10;
$j /= 5;
echo $j; // 输出2
$k=15;
$k %= 4;
echo $k; // 输出3
```

2-3、位运算符

例子	名称	结果
`$a & $b`	And（按位与）	将把 *$a* 和 *$b* 中都为 1 的位设为 1。
`$a | $b`	Or（按位或）	将把 *$a* 和 *$b* 中任何一个为 1 的位设为 1。
`$a ^ $b`	Xor（按位异或）	将把 *$a* 和 *$b* 中一个为 1 另一个为 0 的位设为 1。
`~ $a`	Not（按位取反）	将 *$a* 中为 0 的位设为 1，反之亦然。
`$a << $b`	Shift left（左移）	将 *$a* 中的位向左移动 *$b* 次（每一次移动都表示“乘以 2”）。
`$a >> $b`	Shift right（右移）	将 *$a* 中的位向右移动 *$b* 次（每一次移动都表示“除以 2”）。

```php
$m=8;
$n=12;
$p=-109;
$mn=$m&$n;
echo $mn."<br>";//8
$mn=$m|$n;
echo $mn."<br>";//12
$mn=$m^$n;
echo $mn."<br>";//4
$mn=~$m;
echo $mn."<br>";//-9
$mn=~$p;
echo $mn."<br>";//108
```

2-4、比较运算符

例子	名称	结果
$a == $b	等于	**TRUE**，如果类型转换后 *$a* 等于 *$b*。
$a === $b	全等	**TRUE**，如果 *$a* 等于 *$b*，并且它们的类型也相同。
$a != $b	不等	**TRUE**，如果类型转换后 *$a* 不等于 *$b*。
$a <> $b	不等	**TRUE**，如果类型转换后 *$a* 不等于 *$b*。
$a !== $b	不全等	**TRUE**，如果 *$a* 不等于 *$b*，或者它们的类型不同。
$a < $b	小与	**TRUE**，如果 *$a* 严格小于 *$b*。
$a > $b	大于	**TRUE**，如果 *$a* 严格大于 *$b*。
$a <= $b	小于等于	**TRUE**，如果 *$a* 小于或者等于 *$b*。
$a >= $b	大于等于	**TRUE**，如果 *$a* 大于或者等于 *$b*。
$a <=> $b	结合比较运算符	当*$a*小于、等于、大于than *$b*时分别返回一个小于、等于、大于0的integer 值。PHP7开始提供.
$a ?? $b ?? $c	NULL 合并操作符	从左往右第一个存在且不为 **NULL** 的操作数。如果都没有定义且不为 **NULL**，则返回 **NULL**。PHP7开始提供。

三元运算符

条件?真:假

2-5、错误控制运算符

PHP 支持一个错误控制运算符：@。当将其放置在一个 PHP 表达式之前，该表达式可能产生的任何错误信息都被忽略掉。

```
$x = @$a["name"];
```

2-6、执行运算符

PHP 支持一个执行运算符：反引号（``）。注意这不是单引号！

注意：反引号运算符在激活了安全模式或者关闭了 shell_exec() 时是无效的。

2-7、递增减运算符

运算符	名称	描述
++ x	预递增	x 加 1，然后返回 x
x ++	后递增	返回 x，然后 x 加 1
-- x	预递减	x 减 1，然后返回 x
x --	后递减	返回 x，然后 x 减 1

2-8、逻辑运算符

运算符	名称	描述	实例
x and y	与	如果 x 和 y 都为 true，则返回 true	x=6 y=3 (x < 10 and y > 1) 返回 true
x or y	或	如果 x 和 y 至少有一个为 true，则返回 true	x=6 y=3 (x==6 or y==5) 返回 true
x xor y	异或	如果 x 和 y 有且仅有一个为 true，则返回 true	x=6 y=3 (x==6 xor y==3) 返回 false
x && y	与	如果 x 和 y 都为 true，则返回 true	x=6 y=3 (x < 10 && y > 1) 返回 true
x || y	或	如果 x 和 y 至少有一个为 true，则返回 true	x=6 y=3 (x==5 || y==5) 返回 false
! x	非	如果 x 不为 true，则返回 true	x=6 y=3 !(x==y) 返回 true

2-9、符串运算符

有两个字符串(string)运算符。

第一个是连接运算符(".")，它返回其左右参数连接后的字符串。

第二个是连接赋值运算符(".=")，它将右边参数附加到左边的参数之后。

```
$a = "Hello ";
$b = $a . "World!";

$a = "Hello ";
$a .= "World!";
```

2-10、数组运算符

例子	名称	结果
$a + $b	联合	$a 和 $b 的联合。
$a == $b	相等	如果 $a 和 $b 具有相同的键 / 值对则为 TRUE。
$a === $b	全等	如果 $a 和 $b 具有相同的键 / 值对并且顺序和类型都相同则为 TRUE。
$a != $b	不等	如果 $a 不等于 $b 则为 TRUE。
$a <> $b	不等	如果 $a 不等于 $b 则为 TRUE。
$a !== $b	不全等	如果 $a 不全等于 $b 则为 TRUE。

2-11、类型运算符

instanceof 用于确定一个 PHP 变量是否属于某一类 class 的实例

四、小结

1、if条件的使用

2、foreach和数组的使用

3、运算符的优先级别

2. 课堂练习

练习if

foreach遍历数组

三元运算符

简单加减乘除

3. 课后练习

1、while求出1-100的和

2、for求出1-100的和

3、for求出1-100的和但是不要算进50

4. 资料扩展

1、停止循环

1-1、break

在这里 跳出循环结构 switch break 跳出分支结构

1-2、continue

继续。结束本次循环（跳出本次，不执行），继续下一次的循环

2、单行错误抑制符

@但是这个玩意 效率太低 所以不用

3、逻辑运算短路现象

第一种： if($a && $b = '大爱苍老师'), 首先判断$a是不是true?返回true, 并且 $b = '大爱苍老师'(这其实是赋值,不是判断),也返回true, 所以整个if返回是true, $b = '大爱苍老师'

第二种 if($a || $b = '大爱苍老师') 2个条是或的关系, 那么首先判断$a是不是true?返回true, 那么直接if返回的就是true, $b = '大爱苍老师'这个根本就没有执行, $b=0

第三种 if($a && $b) 首先判断$a是不是true?结果是true; 然后判断$b是不是true?结果是false, 那么if的结果就是false, if的语句不执行. $b=0

4、流程控制

4-1、require

这个函数通常放在 PHP 程序的最前面，PHP 程序在执行前，就会先读入 require 所指定引入的文件，使它变成 PHP 程序网页的一部份。常用的函数，亦可以这个方法将它引入网页中。

4-2、require_one

require_once() 语句在脚本执行期间包括并运行指定文件。此行为和 require() 语句类似，唯一区别是如果该文件中的代码已经被包括了，则不会再次包括。

4-3、include

这个函数一般是放在流程控制的处理部分中。PHP 程序网页在读到 include 的文件时，才将它读进来。这种方式，可以把程序执行时的流程简单化。

4-4、include_once

语句在脚本执行期间包括并运行指定文件。此行为和 include() 语句类似，唯一区别是如果该文件中的代码已经被包括了，则不会再次包括。如同此语句名字暗示的那样，只会包括一次。