数组

数组基础

含义：

数组就是一系列数据的集合体，他们按设定的顺序排列为一个“链的形状”。

注意：php 中的数组单元的顺序，跟下标无关！

数组定义（赋值）：

```
$arr1 = array(3,  11,  5,  18,  2  );//这是最常见的数组，下标为“默认下标”，就是从 0 开始的整数；
$arr2 = array(“a”=>3,  “bb”=>11,  “cc123”=>5,  ‘d1’=>18,  ‘xyz’=>2  );关联数组，下标为字符串，常见
$arr3 = array(1=>3,  10=>11,  3=>5,  0=>18,  88=>2  );下标可以人为给定；
$arr4 = array(1=>3,  ‘a1’=>11,  3=>5,  ‘mn’=>18,  88=>2  );下标可以数字和字符串混合使用；
$arr5 = array(5=>3,  11,  3=>5,  ‘mn’=>18,  2  ); //有指定下标，也有“自动下标”，
                                                    //此时下标为：5，6，3，”mn”, 7
                                                    //可见，自动下标为“前面最大数字下标+1”
$arr6 = array(5=>3,  7.7=>11,  3=>5,  ‘mn’=>18,  2  ); //此时下标为：5，7，3，”mn”, 8
$arr7 = array(5=>3,  true=>11,  false=>5,  ‘mn’=>18,  2  ); //此时下标为：5，1，0，”mn”, 6
$arr8 = array(1=>3,  3=>33,  true=>11,  ); //此时下标为：1，3,其对应值为：11, 33
//下标如果有重复，后面的值覆盖前面的值；
$arr9 = array(1=>3,  -3=>33,  11,  );      //此时下标为：1，  -3，  2，注意：最后一个逗号“可以有”。
其他一些形式：
$arr10[] = 3;
$arr10[] = 11;
$arr10[] = 5;  //该数组下标为 0,1,2，常规情况
$arr11[‘a’] = 3;
$arr11[‘bb’] = 11;
$arr11[‘cc123’] = 5;      //该数组下标为’a’,’bb’,’cc123’，常规情况
$arr12[1] = 3;
$arr12[] = 11; //此时下标为 2
$arr13[‘cc123’] = 5;      //该数组下标为 1,2,’cc123’
```

特别注意：php 中，数组单元的顺序，是由其“放入”顺序决定，而不是下标。

数组取值：

```
$v1 = $arr1[0];
$i = 3;
$v2 = $arr1[$i];     //取得数组下标为 3 的单元的值；
```

总体上，可以将取得一个数组的单元的值，看组取得一个变量的值完全一样！！！

数组的分类

按键值关系来分：

索引数组：通常认为，如果一个数组的下标是严格按照从 0 开始的连续的整数作为下标，则称其为索引数组——就是类似 js 数组的下标。例如：

$arr1 = array(3, 11, 5, 18, 2);//这是最常见的数组，下标为“默认下标”，就是从 0 开始的整数；

关联数组：通常认为，如果一个数组的下标都是一个“字符串”并一定程度上表名了该单元的“含义”，则称为关联数组，例如：

```
$conf = array(
    ‘host’=>”localhost”  ,
    ‘port’=>3306  ,
    ‘username’=>’root’  ,
    ‘password’ => ‘123’  ,
);
```

混合数组：既有数字下标，也有字符下标的情况：

$arr4 = array(1=>3, ‘a1’=>11, 3=>5, ‘mn’=>18, 88=>2);下标可以数字和字符串混合使用；

按数组的维数（复杂程度）分：

一维数组：

```
$a = array(1, 11, 111);
$b = array(2, 22, 222);
$c = array(3, 33, 333);
```

二维数组：

```
$dd = array(
    array(1, 11, 111),
    array(2, 22, 222),
    array(3, 33, 333)
);
```

多维数组：无非就是继续里面再用数组代替。

数组的基本使用

求一个一维数组的平均值

求一个二维数组的平均值

求一个一维数组的最大值

求交换一个一维数组的最大值和最小值的位置

有关数组的交换，再说两句：

```
$a = array( 3,  11,  5,  7,  20,  18);          //下标是 0,1,2,3,4,5
```

需求 1：交换数组第 0 项和第 3 项：

```
$v1 = $a[0];
$v2 = $a[3];
$t = $v1;
$v1 = $v2;
$v2 = $t;       //这种做法根本不行，因为 v1，v2 只是 2 个变量，跟数组没有关系了！
```

正确的做法是：

```
$t = $a[0];
$a[0] = $a[3];
$a[3] = $t;
```

需求 2：交换数组首项和末项：

```
$pos1 = 0;                  //首项的下标
$pos2 = count($a) - 1;   //最后一项的下标
$t = $a[$pos1];
$a[$pos1] = $a[$pos2];
$a[$pos2] = $t;
```

需求 3：交换数组最大项和最小项：

```
$pos_max = ....;     //经过一番计算得到最大项的下标；
$pos_min = ....;     //经过一番计算得到最小项的下标
$t = $a[$pos_max];
$a[$pos_max] = $a[$pos_min];
$a[$pos_min] = $t;
```

数组的遍历

foreach 基本语法

```
foreach( $数组变量名  as  【$key =>】 $value ){
```

```
    //循环体；这里可以去“使用”$key 和 value；
    //$key 和 $value 就是该遍历语句一次次取得的数组的每一个单元（项）的下标和对应值。
    //而且，它总是从数组的开头往后按顺序取数据。
}
```

数组的指针操作及遍历原理：

foreache 遍历流程原理图：

```
foreach( $数组变量名  as  $key => $value ){
    //循环体；这里可以去“使用”$key 和 value；
    //$key 和 $value 就是该遍历语句一次次取得的数组的每一个单元（项）的下标和对应值。
    //而且，它总是从数组的开头往后按顺序取数据。
}
```

使用 for 和 next 遍历数组

注意：对 php 数组，往往不能单纯使用 for 循环进行遍历。

或者说：php 中，使用 for 循环只能循环“下标为连续的纯整数数组”；

each()函数的使用

each()函数的作用：先取得一个数组的“当前单元”的下标和值（并放入一个数组），然后将指针移到下一个单元。

使用形式：

```
$a  =  each($数组名); //此时$a 就是一个数组了
```