```
foreach(iterable_expr as $value){
  statement
}
```

或

```
foreach(iterable_expr as $key=>$value)
{statement}
```

其中 iterable_expr 为数组或对象；

可用：写法；可以用&$value来通过引用赋值来直接改变数组数据，在引用foreach语句结束后

用 unset(value)来解除，否则数组最后单元的数据将保持更新为最新一个数组更新的数据。

eg. $arr1 = array('a'=>1, 'b'=>2, 'c'=>3)

$arr2 = array('x'=>4, 'y'=>5, 'z'=>6);

```
foreach ($arr1 as &$value){
    value*=2 ; }

unset(value);

foreach($arr2 as $value) {
    value*=2 ;}
```

此时

$arr1 = array('a'=2, 'b'=4, 'c'=6)

$arr2 = array('x'=8, 'y'=10, 'z'=12)

若不用offset value)，则'c'不是6

而是最后更新的数组数据'z'=12

的值12.

对于$arr = array(

(1, 2)

(3, 4)

));

...以此类推，可以用

这样的数组：

$arr as list($a, $b)

将二维单元的数据分别赋予$a和$b；list维度低于单元维度时，分别将前几维赋予list，list高于单元维度时报错.

eg. $arr = [
    [1, 2, 3],
    [2, 3, 4],
    ];

... as list($a)

若 $arr

则 $a 分别被赋值 1, 2.

若 $arr as list($a, $b, $c)

则 ($a, $b, $c) 分别被赋值

(1, 2, 3) 和 (2, 3, 4)

若 $arr as list($a, $b, $c, $d)

会报错.