# 常见算法

请写出常见的排序算法，并用PHP实现冒泡排序，将数组安装从小到大的方式进行排序。

**冒泡排序原理和实现**

**算法的概念**

**时间复杂度和空间复杂度的概念**

**常见排序算法**

**常见查找算法**

**基本概念**

一个问题可以有多种算法，每种算法都不同的效率

一个算法具有五个特征：有穷性，确切性，输入项，输出项，可行性。

**算法评定**

算法分析的目的在于选择合适算法和改进算法。

一个算法的评价主要从时间复杂度和空间复杂度来考虑。

时间复杂度

执行算法所需要的计算工作量。一般来说，计算机算法是问题规模n的函数f(n),算法的时间复杂度也因此记做T(n)=O(f(n))

问题的规模n越大，算法执行的时间的增长率与f(n)的增长率正相关，称作渐进时间复杂度(Asymptotic Time Complexity)

#### 时间复杂度计算方式

O(n^2),O(1),O(n)?

得出算法的计算次数公式

例：

1+2+3+...+n

$sum = 0;

for($i =1;$i <= $n;$i ++){

$sum += $i;

}

从这可以看出i一直加到n，所以总共加了n次;

然后它的时间复杂度就是O(n)。

用常数1来取代所有时间中的所有加法常数

**在修改后的运行次数函数中，只保留最高阶项**

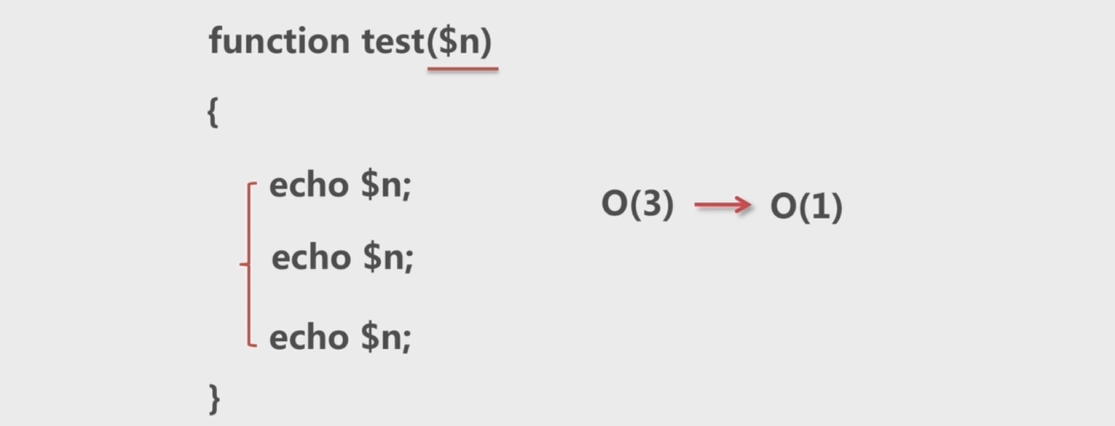
比如说我们最终呢i计算的次数是n^2+n+1我们次数算出来的表达式是这样，那我们的这个最终的表达式是什么！是O(n^2),其它的忽略掉它只拿最高阶;

**如果说最高阶存在且不是1，则去除与这个项相乘的常数。**

比如说我们最终得到的相是什么？是2n^2+3n+1那这个时候呢我们用到的这个时间复杂度就是n^2,也就是说我们的这个时间复杂度我们拿到的是一个什么！根据规模去拿到表达式的一个最阶相里面的一个最高的这样的一个值就可以，拿到n^2就可以不用去拿其它的。

**举例：**

**常数阶：O(1)**

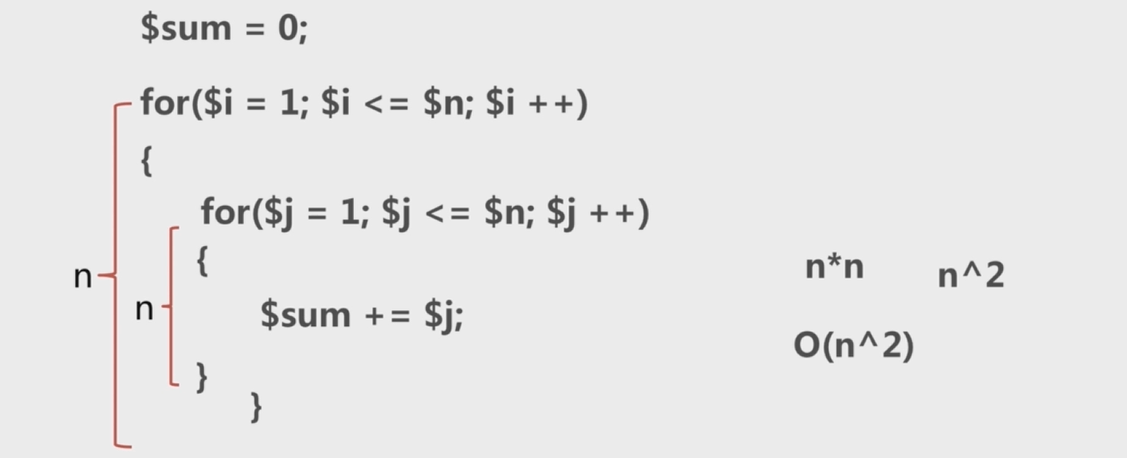


这个function 将来这个$n是多少最终只运行3次，所以这个function的时间复杂度是O(3)但是我们记作( o(1) ),把这个常数要替代成1

**线性阶:O(n)**

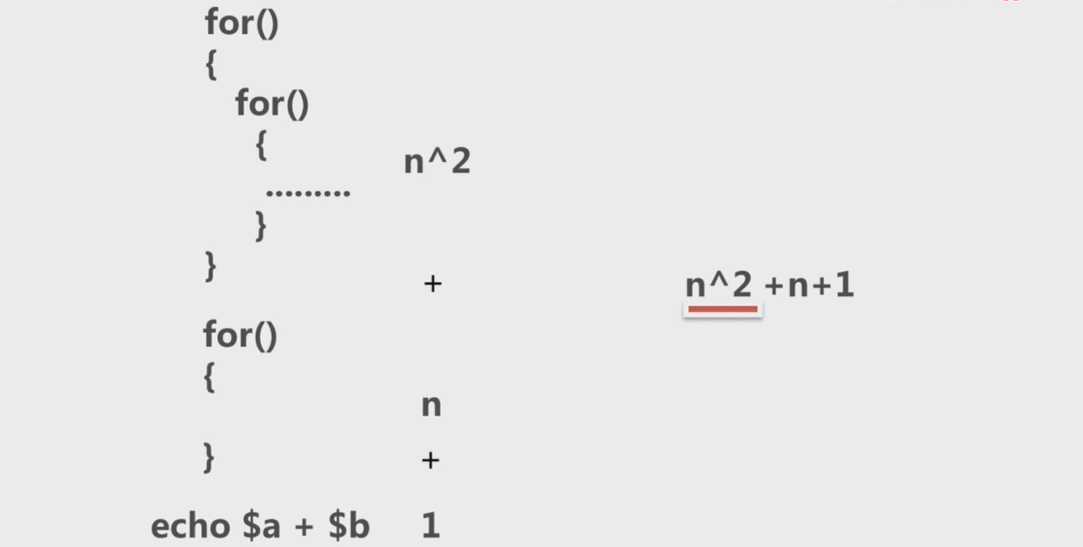
就是1加到n这样的概念

**平（立）方阶：(O(n^2)/O(n^3))**



整体来说我们可以看到，此时我们这个是不是循环套循环！里面这个循环运行n次外面这个循环也运行n次那我最终的表达式是不是应该是n乘以n啊！那也就是n^2那最终我们的这个时间复杂度就是O(n^2)，如果是三层循环的话那就是立方，所以这就是**平方阶和立方阶**; O(n^2)：读作n方

**特殊平方阶：O(n^2/2+n/2)->O(n^2)**



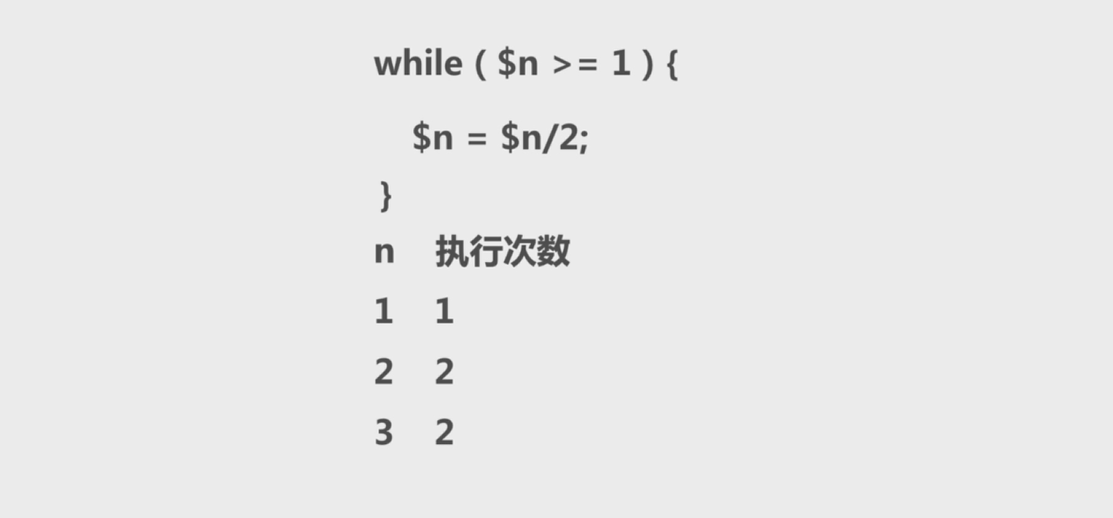
整体来说for双层for是 n^2 ,而下面单层for是n，echo $a + $b在这也计算了一下算做1。就是n^2+n+1。所以最终特殊的阶也就是n^2+n+1。

**对数阶:O(log2n)**

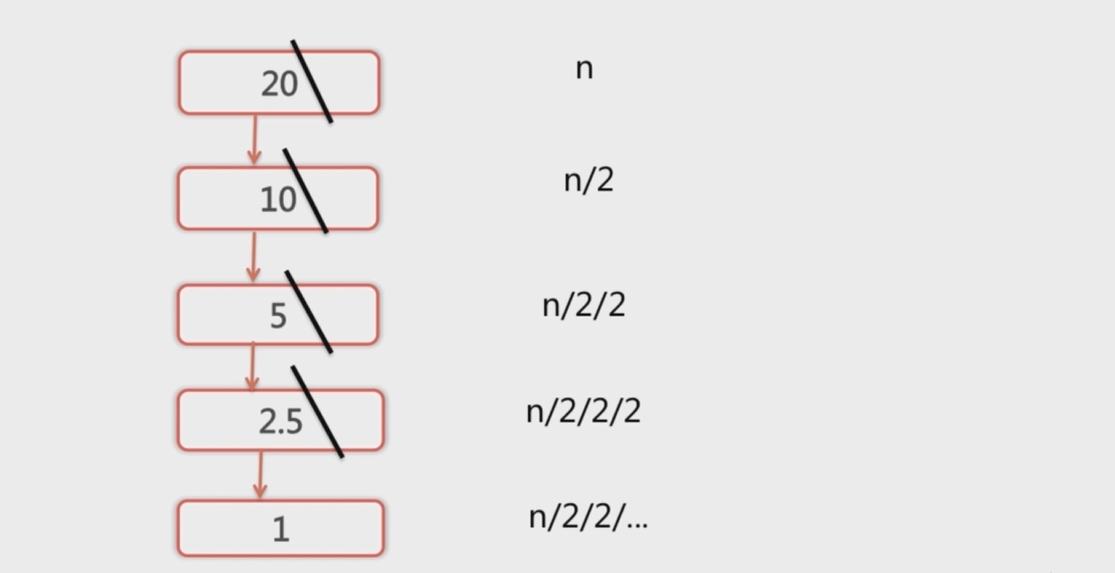
**常见时间复杂度：常数阶，线性阶，平方阶，立方阶，对数阶，log2n阶，指数阶。**

然后时间复杂度的效率我们通常情况下是常数阶是最高的高于谁啊！高于我们的log2n高于我我们的n高于我们的nlog2n高于n^2高于n^3高于2^n高于n!(n的阶乘)高于n^n (n的n次方)

O(1)>O(log2n)>O(n)>O(nlog2n)>O(n^2)>O(n^3)>O(2^n)>O(n!)>O(n^n)



比如说我们去写一个while循环while里面我去写一个$n>=1然后循环体里面我去写一个$n = $n/2，然后结束。那整体来说我们对于这样一个内容来说我们来看一下他的计算次数会有多少。首先n为1时是不是计算了一次n为以1，1大于等于1是不是进来然后在这个地方是二分之一，n变成二分之一，二分之一不大于等于一是不是跳出循环了？计算一次如果n等于二的话我们说二大于等于一进来然后这个里面我们二除以二是不是变成一然后在回来是不是又执行一次，它执行了两次如果n等于三的话大家注意三大于一没有问题三除以二等于多少！是不是1.5，1.5的话大于等于1吧！没有问题好又执行！两次但1.5除以2是不是不满足条件了！对不对.



好那这样的话我们可以看到通过这样的一个算的话我们好向看不太出来它的一个规律对不对好假设我们这个n是有20个数。他的值是20，然后呢20首先进来我们是不是要除以2，是不是劈一半，对除以2是不是变成10了！10然后他又比1大，又比1大的话那好OK进来我又劈一半，20劈一半变成10，10在劈一半是不是变成5。5又比1大怎么办，我又在劈一半变成2.5了吧！对不对。2.5我们取2就可以了然后在劈一半是不是变成1了对不对！就是这么一个概念所以整体来说呢我们可以看到他的一个这样的概念呢是什么呀！就是20是n，到10是除以2到5是n/2/2,到2.5是n/2/2/2(n除以2除以2除以2),一直到1 是n/2/2....

哪怎么样来做呢首先我们的次数多少是不是n除以2的比如我们的次数是m次最终让她等于1（ (n/2^m)=1）。所整体来说我们次数是不是这个m我。我们要拿这个m的吧！那这个m我们怎么算是不是就是2的m次方等于n（（2^m）=n）

那n就等于多少log2n（m=log2n）所以它的这个！我们的这个阶最终的阶就是什么O(n^2)有的时候我们在记这个 log2n的时候它可能把这个2给省略掉logn这么一个概念

所以在这来说呢大家注意一下我们的这个次数就是对数的这样一个概念。

**时间复杂度其它的概念**

最坏情况：最坏情况时的运行时间，一种保证，如果没有特别说明，说的时间复杂度即为最坏情况的时间复杂度。

平均情况：期望的运行时间

#### 空间复杂度

算法需要消耗的内存空间，记作S（n）=O(f(n))

包括程序代码所占用的空间，输入数据所占用的空间和辅助变量所占用的空间这三个方面

计算和表示方法与时间复杂度类似，一般用复杂度的渐近性来表示

###### 空间复杂度计算方式

**有时用空间换取时间**

**冒泡排序的元素交换，空间复杂度O(1)**(中间有一个临时变量存储要交换的值，这个临时的变量只会占用一个内存，然后不管这个数有多大我们始终这个变量的内容，永远它值只会有这么一个临时变量所以的的空间复杂度就是O(1))

**排序算法**

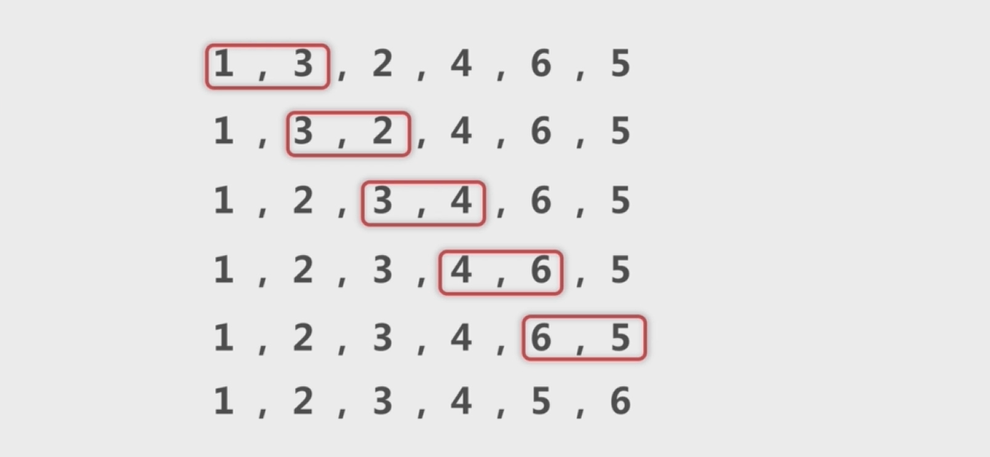
冒泡排序、直接插入排序、希尔排序、选择排序、快速排序、堆排序、归并排序

**冒泡排序**

**原理：**两两想领的数进行比较，如果反序就交换，否则不交换

**时间复杂度：**最坏(O(n^2)),平均(O(n^2))

**空间复杂度：**O(1)



1和3比较，3比1大部交换，3和2比较2比3小交换，3和4比较4比3大不交换,4和6比较6比4大不交换,6和5比较5比6小交换。

总体来说这是最优情况了，对于冒泡老师最坏的情况是有可能是一趟是排不出来的。这时我们就需要进行两层循环了。

$cnt = count($numbers);

for ($i = 0; $i < $cnt - 1; $i++) {

for ($j = 0; $j < $cnt - $i - 1; $j++) {

if ($numbers[$j] > $numbers[$j + 1]) {

$temp = $numbers[$j];

$numbers[$j] = $numbers[$j + 1];

$numbers[$j + 1] = $temp;

}

}

}

}

时间复杂度：O(n^2) 空间复杂度：O(1)

**直接插入排序**

**原理：**每次从无序表中取出第一个元素，把它插入到有序表的合适位置，使有序表仍然有序

**时间复杂度：**最坏(O(n^2)),平均(O(n^2))

**空间复杂度：**O(1)

**希尔排序**

**原理：**把待排序的数据根据增量分成几个子序列进入排序，直到增量为1，直接进入插入排序；增量的排序，一般是数组的长度的一半，再变为原来增量的一半，直到增量为1；

**时间复杂度：**最差(O(n^2)),平均（O(\*log2n)）

**空间复杂的：**O(1)

**选择排序**

原理：每次从待排序的数据元素中选出最小（或最大）的一个元素，存放在序列的起始位置，直到全部待排序的数据元素排完

**时间复杂度：**最坏(O(n^2)),平均(O(n^2))

**空间复杂度：**O(1)

**快速排序**

**原理：**通过一趟排序将要排序的数据分割成独立的两部分，其中一部分的所有数据都比另外一部分的所有数据都要小，然后在按照此方法对这两部分数据分别进行快速排序，整个排序过程可以递归完成。

时间复杂度：最差(O(n^2)),平均(O(nlog2n))

空间复杂度：最差(O(n)),平均(O(log2n))

**堆排序**

**原理：**把待排序的元素按照大小在二叉树位置上排列，排序好的元素要满足：父节点的元素要大于等于子节点；这个过程叫做堆化过程，如果根节点存放的是最大的数，则叫做大根堆，如果是最小，就叫小根堆，可以把根节点拿出来，然后在堆化，循环到最后一个节点。

**时间复杂度：**最差(O(nlog2n)),平均(O(nlog2n))

**空间复杂度：**O(1)

**归并排序**

**原理：**将两个（或两个以上）有序表合并成一个新的有序表，即把待排序序列分为若干个有序的子序列，再把有序的子序列合并为整体有序序列。

**时间复杂度：**最差(O(nlog2n)),平均(O(nlog2n))

**空间复杂度：**O(n)

**总结**

快速排序，归并排序的理想时间复杂度都是O(nlog2n),但是快速排序的时间复杂度并不稳定，最坏情况下复杂度为O(n^2),所以最理想的算法还是归并排序。

###### 查找算法

**二分查找**

**原理：**从数组的中间元素开始，如果中间元素正好是要找的元素，搜索结束，如果某一个特定元素大于或者小于中间元素，则在数组大于或者小于中间元素的那一半中查找，而且跟开始一样从中间开始比较，如果某一步骤数组为空，代表找不到。

**时间复杂度：**最差(O(log2n)),平均(O(log2n))

**空间复杂度：**迭代(O(1)),递归(O(log2n))

**顺序查找**

**原理：**按一定的顺序检查数组中每一个元素，直到找到所要寻找的特定值为止。

时间复杂度：最差(O(n)),平均(O(n))

空间复杂度：O(1)

**总结**

二分查找算法的时间复杂度最差是(O(log2n)),顺序查找的时间复杂度最差为O(n),所以二分查找法更快，但是递归情况下，二分查找法更消耗内容，时间复杂度为O(log2n)

**解题方法**

此类考点非常重要也较为复杂，需要考生充分理解各种排序算法和查找算法的原理以及实现方式，另外还需要理解时间复杂度和空间复杂度的计算方式和概念，此类考察点毋庸置疑是考察考生的逻辑思维能力，因此需要大家仔细研究各种算法的实现方式。

**请简述时间复杂度和空间复杂度的概念**

时间复杂度就是描述算法的时间的一个消耗的计算量，

空间复杂度就是内存的计算量

**对无序数组排序，最忧的时间复杂度是什么，用PHP或者JavaScript写出一个实际的例子（如了解，给出算法的名称），该算法的空间复杂度是什么？**

**一个有序数组中，查询特定item是否存在的最优算法是什么？时间复杂度是什么？**

# 常见数据结构

请简单描述一下数据结构的特征(stack,heap,list,doubly-linked-list,queue,array(vector))

**常见数据结构特征**

**Array：**

数组，最简单而且应用最广泛的数据结构之一

特性：使用连续的内存来存储，数组中的所有元素必须是相同的类型或类型的衍生（同质数据结构），元素可以通过下标直接访问

**LinkedList：**

链表，线性表的一种，最基本，最简单，也是最常用的数据结构

特性：元素之间的关系是一对一的关系（除了第一个和最后一个元素，其他元素都是首尾相接），顺序存储结构和链式存储结构两种存储方式。

**Stack：**

栈，和队列相似，一个带有数据存储特性的数据结构

特性：存储数据是先进后出的、栈只有一个出口，只能从栈定不增加和移除元素。

**Heap:**

堆：一般情况下，堆叫二叉堆，近似完全二叉树的数据结构

特性：子节点的键值或者索引总是小于它的父节点，每个节点的左右子树又是一个二叉堆，根节点最大的堆叫最大堆或者大根堆，最小的叫最小堆或者小根堆。

**List:**

线性表，由零个或多个数据元素组成的有序列

特性：线性表是一个序列，0个元素构成的线性表是空表，第一个元素无先驱，最后一个元素后继，其他元素都只有一个先驱和后继，有长度，长度是元素个数，长度有限。

**Doubly-linked-list**

双向链表

特性：每个元素都是一个对象，每个对象有一个关键字key和两个指针(next和prev)

**Queue**

队列

特性：先进先出(FIFO),并发中使用，可以安全将对象从一个任务传给另一个任务

**Set**

集合

特性：保存不重复元素

**Map**

字典

特性：关联数组，也被叫做字典或者键值对

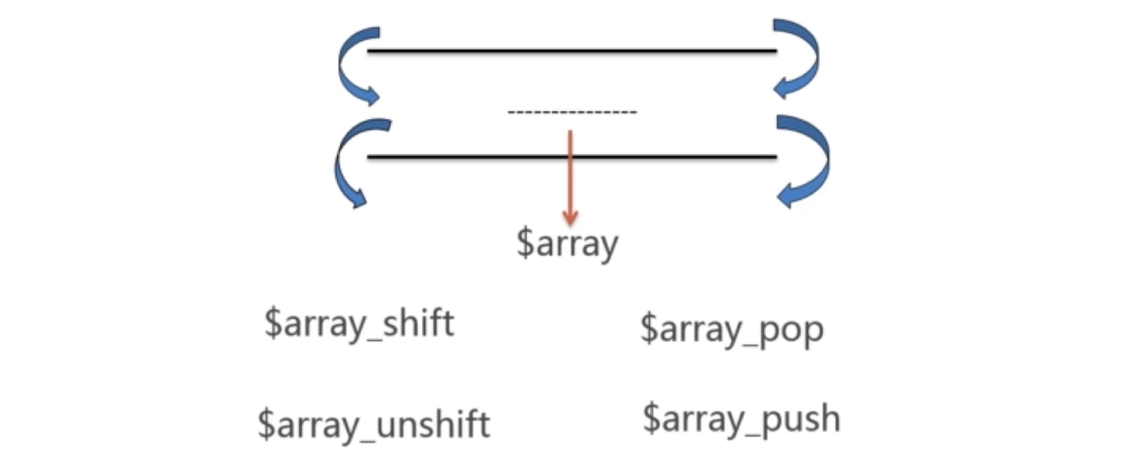
**Graph**

图

特性：通常使用邻接矩阵和邻接表表示，前者易实现但是对于稀疏矩阵会浪费较多空间，后者使用链表的方式存储信息但是对于图搜索时间复杂度较高

选理解每种数据结构的特征，有时会遇到让用PHP代码实现其中的数据结构，遇到此类题，先要理解该种数据结构的特性是什么，然后在实现，通常可以使用PHP的数组来模拟。

**用PHP实现一个双向队列**



**1,1,2,3,5,8,13,21,34.....求第30位的数是多少，请用伪代码描述其实方法。**

**解题思考**

找要处理的数组或者字符串的规律，也可能是一个数列的规律，如果要求使用伪代码实现，建议先说明规律，然后白话说明解题步骤即可。

**解题**

$arr = [1,1];

for($i=2;$i<30;$i++){

$arr[$i] = $arr[$i-1] + $arr[$i-2];

}

var\_dump($arr);

**例题**

请写一个函数，实现以下功能：字符串 "open\_door" 转换成"OpenDoor","make\_by\_id" 转换成 "MakeById"

function strHandle($str){

$resturn ='';

$arr = explode('\_',$str);

foreach($arr as $val){

$return . = ucfirst($val);//ucfirst头字母转大写

}

return $return;

}

echo strHandle('make\_by\_id');

# 模拟内置函数实现

**不使用PHP函数，用方法写一个反转字符串的函数**

此类考点会分为两类，一类是直接说明让考生实现什么样的操作，一类是让考生实现某函数的功能，因此不仅需要考生具备一定的数据处理能力，还要熟悉PHP内置函数的功能是什么，常见的处理主要是对字符串和数组的处理，因此建议大家多学习HP字符串和数组内置函数，尝试模拟其功能，有助于提升逻辑思维和熟悉函数功能。

function str\_rev($str){

for($i=0;true;$i++){

if(!isset($str[$i])){

break;

}

}

$return ='';

for($j=$i-1;$j>=0;$j--){

$return.=$str[$j];

}

return $return;

}

echo str\_rev('gfuritmgj');

**例题：**

写一个函数，要求不使用array\_merge完成多个数组的合并

//数组合并

function array\_mer(){

$return = [];

$arrays= func\_get\_args();//获取一个函数的所有参数

foreach($arrays as $arr){

if(is\_array($arr)){

foreach($arr as $val){

$return[]=$val;

}

}

}

return $return;

}

var\_dump(array\_mer([3,5,7]));