* 1. //实现1+2+3+...+100的和；要求使用三种方式

//1.使用函数

function sum($sum){

return $sum =array\_sum(range(1,$sum));

}

//echo sum(100);

说明: 使用 array\_sum进行求和 range();生成1,100内的所有数

//2.首尾相加

function sum1($sum){

return $sum=($sum+1)\*($sum/2);

}

//echo sum1(100);

说明:搞死算法 首尾相加\*所有数字的一半

//3.循环累加

function sum2($n){

$sum=0;

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

$sum+=$i;

}

return $sum;

}

//echo sum2(100);

说明:给定一个最大值 和一个初始值 循环条件为小于等于最大值 每次循环让 初始值加$i

//n的阶乘

//非递归

function jc($n){

$sum=1;

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

$sum\*=$i;

}

return $sum;

}

//echo jc(6);

说明: 利用循环求 给定最大值和初始值 循环条件为小于等于最大值 每次循环让 初始值乘$i

//递归

function jc1($n){

if ($n==1) {

return 1;

}else{

$sum\*=jc1($n);

}

}

echo jc(6);

说明:给定值$N 如果$n=1就返回1;否则就在调用算法累乘

//判断一个字符串是否是回文字符串

function hui($str){

$string=strrev($str);

if ($str==$string) {

echo "是回文字符串";

}else{

echo "不是回文字符串";

}

return $string;

}

echo hui('abba');

//先使用函数strrev让字符串倒叙输出 形成新的字符串 让后判断两者是否相等 相等是回文不相等则不是回文

Person类

<?php

class person{

public $name;

public $age;

public function \_\_construct($name,$age){

$this->name=$name;

$this->age=$age;

}

}

?>

说明: 实例化一个person类 设置公共属性和公有的构造方法

<?php

interface walk{

public function run();

}

?>

说明:定义一个接口walk 并定义一个行走的方法;

<?php

include 'person.php';

include 'walk.php';

class student extends person implements walk{

public function run(){

}

}

?>

//定义一个student 类 应用person类和walk类

下面使用继承方法

<?php

include 'student.php';

$obj1=new student("张三1",101);

$obj2=new student("张三2",102);

$obj3=new student("张三3",103);

$obj4=new student("张三4",104);

$obj5=new student("张三5",105);

function index($n){

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

$c='obj'.$i;

global $$c;

}

$max=1;

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

$a='obj'.$i;

$b='obj'.$max;

if ($$a->age > $$b->age) {

$max=$i;

}

}

$c='obj'.$max;

return $$c->age;

}

echo index(5);

?>

说明:引入student类

实例化类并赋值

建立函数

先通过循环取得每个对象的名称并进行全局话

给定一值用来比对

如否后者大于前者就赋值给他

最后取到最大值所在的对象

根据对象输出值

//linux上传

Cd 到要创建的目录下

Mkdir WeekExam1 创建空目录

进入 WeekExam1

Rz 上传文件