php

流程控制，函数，数组，字符串

一、运算符

运算符

一，算术运算符

算术运算符用于完成各种运算：

+	加法运算符	$a+$b;
-	加法运算符	$a-$b;
*	乘法运算符	$a*$b;
/	除法运算符	$a/$b;
%	取模运算符(求余数)	$a%$b;

二，赋值运算符

赋值运算符，将一个数据值赋给一个变量;
复合运算符，在赋值之前会完成某个运算;

$a = 5	赋值	
$a += 5	加法赋值	$a = $a + 5
$a -= 5	减法赋值	$a = $a – 5
$a *= 5	乘法赋值	$a = $a * 5
$a /= 5	除法赋值	$a = $a / 5
$a .= 5	拼接赋值	$a = $a.5

三，字符串运算

字符串运算符用于拼接字符串;

例:

```
$a = "hello";
$b = $a . “world";    //.表示拼接前后两个字符串
echo $b;

$b = "Hello";
$b .= “World!";    //.=  表示 $b = $b.”World”;
echo $b;
```

四，递增(++)和递减(—)运算符

递增和递减运算符将变量的当前值加1或减1，可以使代码更简洁;

++$i	先加	$i的值加1，然后再返回$i的值;
$i++	后加	先返回$i的值，然后再将$i的值加1;
--$i	先减	$i的值减1，然后再返回$i的值;
$i--	后减	先返回$i的值，然后再将$i的值减1;

五，逻辑运算符

利用逻辑运算符可以根据多个变量的值进行判断，这使得控制程序的流程成为可能，逻辑操作符常用于控制结构中，如if条件和while及for循环;

&&，and　　逻辑与

ll，　or　　逻辑或

!,　　逻辑非

xor,　　异或（有且仅有一个为true，则返回true）

六，比较运算符

比较运算符，返回一个布尔值 TRUE 或 FALSE;

>	大于
<	小于
>=	大于或等于
<=	小于或等于
!=	不等于
<>	不等于
==	等于
===	全等于 （两个比较的内容里，类型也要一样）
!==	全不等

七，三元运算符

语法：expression1 ？expression2 : expression3

例：
```
$a = 5;
$b = 2;
$res = $a > $b ? "yes":"no";
echo $res;
```

八，运算符的优先级

所谓运算符的优先级指的是哪一个运算符应该先计算。

具体运算符的优先级，参考php手册;

赋值运算，从右到左

例:

```
echo 1 + 2 * 3;                    //outputs 7
```

二、流程控制

判断语句

1. if 语句

```
if(expression ){
        //statement
}else if(expression){
        //statement
}else{
        //statement
}
```

if语句用括号中的表达式返回值（true 或 false)来控制是否执行指定的代码程序;
表达式为数字0、空、未定义的字符串，内置常量false都会返回false;

2. switch 语句

switch 语句可以看作是if-else组合的一种变体，如果需要比较有限值的变量，通常会使用switch语句;

语法格式：

```
switch (expression){
        case value:
        //statements
            break;
        default:
          //statements
}
```

在每个case块的末尾处都有break语句，如果没有break语句，就会执行所有后续的case块，直到遇到break语句为止;

循环语句

1.while

语法格式：

```
while(expression){
    //statements
}
```

例：

```
$count = 1;
while ($count < 5) {
    echo "$count 平方 = ".pow($count, 2)."<br />";
    $count++;
}
```

2. do…while
do…while循环是while的一种变体，它在代码块的结束处验证循环条件;

语法格式:

```
do {
    //statements
}while(expression);
```

例：

```
$count = 11;
do {
    echo "$count squared = ". pow($count, 2). "<br />";
    $count++;
}while($count < 10);
```

3. for循环

语法格式：

```
for(expression1; expression2; expression3){
        //statements
}
```

第一个表达式expression1在第一次循环时计算;
第二个表达式expression2在每次循环时进行计算，这个表达式确定循环是否继续执行;
第三个表达式exprression3在每次循环结束时计算;

例：

```
for($i = 1; $i <= 5; $i++){
        echo "$i squared = ".pow($count, 2)."<br/>";
    }
```

4. foreach 循环

foreach循环用来遍历数组，每次循环都将指针后移一位;

语法格式1：

```
foreach(array_expr as $value){
        //statements
}
```

语法格式2：

```
foreach(array_expr as $key=>$value){
        //statements
}
```

跳出循环

1. break

如果包含一个break语句，将立即结束 while、do…while、for、foreach、switch的执行。

2.continue

continue语句使当前循环执行结束，并从下一次循环开始执行;

三、函数

函数概念

函数是用来完成某种特定任务的可重用代码块;

函数可以使程序更具模块化,拥有良好的结构;

函数定义后在程序中可以重复调用;

函数分为内置函数和自定义函数

函数分类

1. 内置函数

PHP系统提供了大量功能强大的函数，帮助我们解决各种问题;

2.创建自定义函数

```
function function_name(parameters) {
        //function body
}
```

例：

```
function sayhello(){
        echo 'hello';
}
```

函数简介

函数用function关键字来声明;

函数名称是由字母或下划线开始,中间可以包含数字;

函数名不区分大小写,不过在调用函数的时候， 通常使用其在定义时相同的形式;

php不支持函数重载, 所以自定义函数不能和内置函数重名;

不能在一个文件中自定义同名的函数;

参数出现在括号中,如果有多个参数用逗号分隔;

函数调用

语法格式:

```
fun_name(parameters);
```

例：

```
$val = pow(4, 2);
echo pow(3, 3);
```

函数的区别

注意不同语言的区别：

在Java等强类型语言中方法的名字严格区分大小写；

C语言中的函数也是严格区分大小写；

但PHP到现在的版本,函数名称不区分大小写；

很多语言允许函数(方法)的重载,即函数有相同的名称但是函数参数不同；

PHP不支持函数的重载；

参数传递

值传递(传值)

函数内对参数值的改变不会影响函数外部的值;

引用传递(传址)

有些情况下，可能希望在函数体内对参数的修改在函数体外也能反映;

使用引用传递参数要在参数前加上&符号;

变量本身传入，传入后的变量与原变量建立联系;

函数体内变量的变化，会影响到原变量本身;

例：

```
$a = 25;
function modifyNum(&$num){
    $num = 100;
}
modifyNum(&$a);
```

默认参数值：可以为参数指定默认值，在没有提供其他值的情况下，则将默认值自动赋给该参数;

可选参数：可以指定某个参数为可选参数，这些参数需要放在参数列表的末尾，需且要指定其默认值为空;

如果指定了多个可选参数，可以选择性地传递某些参数;

返回值

通常情况下，只依靠函数做某些事情还不够; 需要将函数的执行结果返回给调用者，这时可以使用 return 语句返回结果;

return 语句变执行后，将使得函数立即结束运行，并且将控制权返回被调用的行;

例：
```
function mysquare($num){
    if($num == ‘’){
        return;
    }
    $res = $x * $x;
    return $res;
}
echo mysquare(4);
```

变量作用域和生命周期

由于引入了函数，程序中变量的能见度发生了变化，即变量的作用范围发生了改变;

变量分为:全局变量，局部变量，静态变量;

局部变量，函数体内定义的变量为局部变量，只在函数体内可见;

局部变量的作用域：从声明它的那条语句开始到函数结束;

例：

```
$str = 'hello php';
echo '1:'.$str.'<br />';
function change(){
        $str = 'hello everyone';
        echo '2:'.$str.'<br />';
}
change();
echo '3:'.$str;
```

include和require

1.include()

include()语句将在其被调用的位置处包含一个文件。

例：　include(“init.php");

2.include_once()

include_once()的作用与include()相同，不过它会首先验证是否已经包含了该文件，如果已经包含，则不再执行include_once();

3.require() 与include() 一样，只不过require()我们通常放在php程序的最前面；

4.require_once() 与include_once() 一样，但是也要放在php程序的最前面；

5.include和require的区别

require一个文件存在错误的话，那么程序就会中断执行了，并显示致命错误

include一个文件存在错误的话，那么程序不会中断，而是继续执行，并显示一个警告错误。

四、数组

数组的概念

一.数组的概念：

数组可以理解为有序的（键-值)对组成的数据值的集合;

如果我们把变量理解为单个值的容器，那么数组就是可以包含多个值的容器;

根据索引值的不同数组分为：索引数组和关联数组;

例：$day = array(“a”,”a”,”a”);　　//索引数组

$week = array("a"=> "星期一", "b"=>"星期二", "c"=> “星期三");　　//关联数组

数组的创建

1. 使用array()函数

$array = **array** ([mixed ...])

例：$arr = array();

$fruits = array("orange", "apple", "banana");

$languages = array("en"=> "english", "cn"=> "china"）；

与其它语言的数组实现方式不同，php不需要在创建数组时指定其大小;

因为php是一种松散类型的语言，所以甚至不需要在使用前先声明;

索引可以是整型数字或者是字符串;

索引数组：索引为整数,如果没有指定索引值则默认为零，依次递增;

关联数组：索引为字符串的数组;

2. 直接对数组变量赋值

$arr[*key*] = *value*;

例： $fruits[] = "orange";

$fruits[] = "apple";

$languages["en"] = "english";

$languages["cn"] = "china";

如果方括号中没有指定索引，则取当前最大整数索引值，新的键名将是 该值+ 1。如果当前还没有整数索引，则键名将为0。如果指定的键名已经有值了，该值将被覆盖。

3. 使用函数创建数组

range() 建立一个包含指定范围单元的数组

例： $num = range(1, 100);

print_r($num);

$letter = range('a','l');

print_r($letter);

数组的基本操作

1.unset($arr[0]) 删除数组元素

2.print_r($arr) 打印数组

3.count($arr) 取得数组大小

4.in_array(10, $arr) 检查数组中是否包含某个值

遍历数组

1. for 循环遍历数组

2，foreach 循环遍历数组

数组排序

1.sort() 、 rsort() 对数组进行升序和降序

例：
```
$fruits = array("lemon", "orange", "banana", "apple");
sort($fruits);
print_r($fruits);
```

2.ksort()、krsort() 对数组按索引进行升序或降序, 并保持索引关系

例：
```
$fruits = array("l"=>"lemon", "o"=>"orange", "b"=>"banana",
"a"=>"apple");
ksort($fruits);
print_r($fruits);
```

二维数组

数组元素的值也可以是数组;

例:
```
$result = array(
        array(
            'pname'=> 'nokia n73',
            'price'=>  1500,
        ),
        array(
            'pname'=> 'nokia 5800',
            'price'=>  2000,
        ),
);
```

遍历二维数组:

```
foreach($products as $product_k=>$product_v){
        foreach($product_v as $key=>$val){
                echo $key.'=>'.$val;
        }
}
```

五、字符串

输出字符串

1.echo

void **echo** (string arg1 [, string ...])

是一个语法，不是函数

echo 没有返回值;

echo 可以输出多个值，使用逗号分隔;

例：

```
$val = "world";
echo "hello", $val;
```

其他处理函数

1.strlen() 获取字符串长度

例：$passwd = "123456";

```
if(strlen($passwd) < 8){
        echo "密码不能少于8位";
}
```

2.strtolower() 将字符串转换为小写字母

例：$url = "HTTP://WWW.LANOU.COM/ ";

```
echo strtolower($url);
```

3. strtoupper() 将字符串转换为大写字母

例：$str = "中文 hello world";

```
echo strtoupper($str);
```

查找与替换

1.strpos()

int **strpos** (string,find[, start])

strpos()函数在 string 中以区分大小写的方式找到find 第一次出现的位置;如果没有找到则返回FALSE;

可选参数start 指定开始查找的位置;

例：echo strpos("Hello world!","wo");

2.stripos()

stripos()与strpos()功能相同，只是查找时不区别大小写;

3.str_replace()

string str_replace (search, replace, subject [, int &count])

str_replace()函数在subject中以区分大小写的方式搜索search ， 用replace替换找到的所有内容; 如果没有找到search,则subject保持不变;

如果定义了可选参数 count 则只替换subject中count个search

例： $str = "test@163.com";

$email = str_replace("@", "(at)", $str);

echo $email;

4.str_ireplace()

str_ireplace()与str_replace()功能相同， 只是不区分大小写;

截取字符串

1.substr()
string **substr** (string string, int start [, int length])

从start位置取出length长度的字符，字符串位置开始值为零;
如果没有指定length，那么默认一直到字符串末尾;

例：echo substr("Hello world", 6);
　　echo substr("hello world", 6, 5);

2.strstr()
string **strstr** (string str, string search, bool beforeSearch)
strstr() 作用从str中搜索search,并根据第三个参数返回search后面的数据还是前面的数据。

例： echo strstr("Hello world!","world");

3.stristr()
stristr()与strstr()功能相同，只是不区分大小写;

例： echo strstr("Hello world!","WORLD");

删除字符串

1.ltrim()
string ltrim (string str [, string charlist])
ltrim 函数删除字符串左侧空格或其他预定义字符;
如果未设置charlist参数，则删除以下字符：
"\0"　　NULL
"\t"　　制表符
"\n"　　换行
"\x0B"　垂直制表符
"\r"　　回车
" "　　空格

2.rtrim()
string rtrim (string str [, string charlist])
rtrim 函数删除字符串右侧空格或其他预定义字符;

3.trim()
trim 函数删除字符串两侧空格或其他预定义字符;

4.strrev() 反转字符串

例：　$str = "hello world";
　　echo strrev($str);

5. nl2br() 将字符串中换行 (\n) 转换成 HTML 换行标签 (
)

例：　$str = "hello
　　world";
　　echo nl2br($str);

6.strip_tags() 删除字符串中HTML XML PHP 标签
string strip_tags (string str [, string allowable_tags])
可选参数 allowable_tags 指定要保留的标签;

例：$str = "test 163";
　　echo strip_tags($str);

7. htmlspecialchars() 函数把一些预定义的字符转换为 HTML 实体

预定义的字符是：
& （和号） 成为 &
" （双引号） 成为 "
' （单引号） 成为 '
< （小于） 成为 <
> （大于） 成为 >

例：$str = " 这是一个段落 ";
echo htmlspecialchars($str);

数组字符串转换

1.explode() 返回由字符串组成的数组

例：
```
$str = "1,2,3,4,5,6";
$arr = explode(',', $str);
print_r($arr);
```

2.implode() 将数组元素连接成字符串

例：
```
$arr = array('a','b', 'c', 'd');
$str = implode('|', $arr );
echo $str;
```

谢　谢