

## 一、定义

## 二、调用

### 1、用&调用

### 2、先定义后调用

### 3、前向引用

### 4、用do调用

## 三、返回值

## 四、局部变量

## 五、子程序参数传递

### 1、形式

### 2、传送数组

## 六、递归子程序

## 七、用别名传递数组参数

## 八、预定义的子程序

## 一、定义

子程序即执行一个特殊任务的一段分离的代码，它可以使减少重复代码且使程序易读。PERL中，子程序可以出现在程序的任何地方。定义方法为：

```
sub subroutine{  
    statements;  
}
```

## 二、调用

调用方法如下：

### 1、用&调用

```
&subname;  
...  
sub subname{  
    ...  
}
```

### 2、先定义后调用，可以省略&符号

```
sub subname{  
    ...  
}  
...  
subname;
```

### 3、前向引用，先定义子程序名，后面再定义子程序体

```
sub subname;  
...  
subname;  
...  
sub subname{  
    ...  
}
```

### 4、用do调用

do my\_sub(1, 2, 3);等价于&my\_sub(1, 2, 3);

## 三、返回值

缺省的，子程序中最后一个语句的值将用作返回值。语句`return (retval);`也可以推出子程序并返回值`retval`，`retval`可以为列表。

#### 四、局部变量

子程序中局部变量的定义有两种方法：`my`和`local`。其区别是：`my`定义的变量只在该子程序中存在；而`local`定义的变量不存在于主程序中，但存在于该子程序和该子程序调用的子程序中(在PERL4中没有`my`)。定义时可以给其赋值，如：

```
my($scalar) = 43;
local(@array) = (1, 2, 3);
```

#### 五、子程序参数传递

##### 1、形式

```
&sub1(&number1, $number2, $number3);
...
sub sub1{
    my($number1, $number2, $number3) = @_;
    ...
}
```

##### 2、传送数组

```
&addlist (@mylist);
&addlist ("14", "6", "11");
&addlist ($value1, @sublist, $value2);
...
sub addlist {
    my (@list) = @_;
    ...
}
```

参数为数组时，子程序只将它赋给一个数组变量。如

```
sub twolists {
    my (@list1, @list2) = @_;
}
```

中`@list2`必然为空。但简单变量和数组变量可以同时传递：

```
&twoargs(47, @mylist); # 47赋给$scalar, @mylist赋给@list
&twoargs(@mylist); # @mylist的第一个元素赋给$scalar, 其余的元素赋给@list
...
sub twoargs {
    my ($scalar, @list) = @_;
    ...
}
```

#### 六、递归子程序

PERL中，子程序可以互相调用，其调用方法与上述相同，当调用该子程序本身时，即成了递归子程序。递归子程序有两个条件：1、除了不被子程序改变的变量外，所有的变量必须的局部的；2、该子程序要含有停止调用本身的代码。

#### 七、用别名传递数组参数

1、用前面讲到的调用方法`&my_sub(@array)`将把数组`@array`的数据拷贝到子程序中的变量`@_`中，当数组很大时，将会花费较多的资源和时间，而用别名传递将不做这些工作，而对该数组直接操作。形式如：

```
@myarray = (1, 2, 3, 4, 5);
&my_sub(*myarray);
sub my_sub {
    my (*subarray) = @_;
}
```

2、此方法类似于C语言中的传递数组的起始地址指针，但并不一样，在定义数组的别名之后，如果有同名的简单变量，则对该变量也是起作用的。如：

```
$foo = 26;
@foo = ("here's", "a", "list");
&testsub (*foo);
```

```
...
sub testsub {
    local (*printarray) = @_;
    ...
    $printarray = 61;
}
```

当子程序执行完，主程序中的\$foo的值已经成了61，而不再是26了。

3、用别名的方法可以传递多个数组，如：

```
@array1 = (1, 2, 3);
@array2 = (4, 5, 6);
&two_array_sub (*array1, *array2);
sub two_array_sub {
    my (*subarray1, *subarray2) = @_;
}
```

在该子程序中，subarray1是array1的别名，subarray2是array2的别名。

八、预定义的子程序

PERL5预定义了三个子程序，分别在特定的时间执行，它们是：BEGIN子程序在程序启动时被调用；END子程序在程序结束时被调用；AUTOLOAD子程序在找不到某个子程序时被调用。你可以自己定义它们，以在特定时间执行所需要的动作。如：

```
BEGIN {
    print("Hi! Welcome to Perl!\n");
}
AUTOLOAD{
    print("subroutine $AUTOLOAD not found\n"); # 变量$AUTOLOAD即未找到的子程序名
    print("arguments passed: @_ \n");
}
```

若同一个预定义子程序定义了多个，则BEGIN顺序执行，END逆序执行。

[上一章](#) [下一章](#) [目录](#)