第四章　语句和声明

概念型

声明：my 和 our

if/elsif/else

while/until

for/foreach

continue 块

next/last/redo/

LABEL(标签)

重要型

for(;;){...}

foreach VAR (LIST) {...}

循环控制，可以在循环上放上 LABEL。循环控制语法如下：

last LABEL

next LABEL

redo LABEL

LABEL 是可选的，如果省略，该操作符就选用最内层的封闭循环。last 操作符立即退出当事循环，即使有 continue 块也不会执行。next 操作符忽略当前循环的余下的语句，然后开始一次新的循环。redo 操作符在不重新计算循环条件的情况下，重新开始循环语句块，如果存在 continue 块也不会被执行。

while 和 until 语句可以有一个额外的块：continue 块，这个块在整个块每继续一次都执行一次，不管是退出第一个块还是用一个明确的 next，但实际中 continue 块用的不多。

裸块

case 控制结构，perl 本身没有实现，但是有多种替代方法。

goto，三种形式，goto LABEL，goto EXPR, goto &NAME

全局声明，子过程和格式声明是全局的。声明子过程后可以不用括号而调用，可以用 require 语句从其他文件装载定义，但这个方法不是很好了，最好的方式是使用 use 声明，它可以在编译时就 require 各模块。

范围声明，编译时作用。

范围变量声明，completely private variables(using my)

selective global variables(using our)

provide temporary values to global variables(using local)

if more than one variable is listed, the list must be placed in parentheses. For my and our, the

elements may only be simple scalar, array or hash variables. For local, the constraints are somewhat more relaxed, you may also localize entire typeglobs and individual elements or slices of arrays and hashes.

Using a local operator on a global variable gives it a temporary value each time local is executed, but it does not affect that variable's global visibility. When the program reaches the end of that dynamic scope, the temporary value is discarded and the original value resotred.

编译指示：

```
use warings;
use strict;
use integer;
use bytes;
use constant pi => (4*atan2(1, 1));
...
```

琐碎型

心得型