

Linux 系统概论

天津医科大学
生物医学工程与技术学院

2014-2015 学年下学期（春）
2013 级生信班

第九章 Perl 语言简介

伊现富 (Yi Xianfu)

天津医科大学 (TIJMU)
生物医学工程与技术学院

2015 年 7 月



教学提纲

1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

• unless 语句

• given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题

1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



Perl

- Perl 是高级、通用、直译式、动态的程序语言
- **Practical Extraction and Report Language**, 实用摘录与报表语言
- Pathologically Eclectic Rubbish Lister, 病态折中式垃圾列表器
- 拉里·沃尔 (**Larry Wall**) , 1987 年 12 月 18 日

特性

- 具有动态语言的强大灵活的特性
- 借用了 C、sed、awk、shell 等语言的特性, 提供了许多冗余语法
- 使用了语言学的思维 (泛型变量、动态数组、Hash 表等)
- 程序员可以忽略内部数据存储、类型、内存越界等细节
- 内部集成了正则表达式的功能
- 巨大的第三方代码库**CPAN** (Comprehensive Perl Archive Network, Perl 综合典藏网)

Perl

- Perl 是高级、通用、直译式、动态的程序语言
- **Practical Extraction and Report Language**, 实用摘录与报表语言
- Pathologically Eclectic Rubbish Lister, 病态折中式垃圾列表器
- 拉里·沃尔 (**Larry Wall**) , 1987 年 12 月 18 日

特性

- 具有动态语言的强大灵活的特性
- 借用了 C、sed、awk、shell 等语言的特性, 提供了许多冗余语法
- 使用了语言学的思维 (泛型变量、动态数组、Hash 表等)
- 程序员可以忽略内部数据存储、类型、内存越界等细节
- 内部集成了正则表达式的功能
- 巨大的第三方代码库**CPAN** (Comprehensive Perl Archive Network, Perl 综合典藏网)

书写规范

- Perl：程序语言本身
- perl：实际编译并运行程序的解释器
- PERL：错误的写法，外行的标志

应用领域

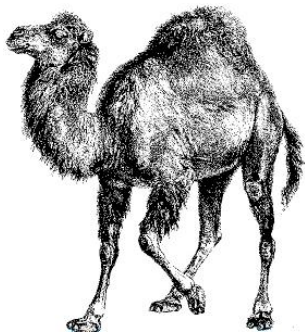
- 脚本语言中的瑞士军刀
- 网络编程、CGI（Common Gateway Interface，通用网关接口）
- 图形编程
- 系统管理
- 生物信息
- ...

书写规范

- Perl：程序语言本身
- perl：实际编译并运行程序的解释器
- PERL：错误的写法，外行的标志

应用领域

- 脚本语言中的瑞士军刀
- 网络编程、CGI（Common Gateway Interface，通用网关接口）
- 图形编程
- 系统管理
- 生物信息
- ...



- **TMTOWTDI** : There's More Than One Way To Do It. 不只一种方法来做一件事。发音为 “Tim Toady” 。
- **TIMTOWTDIBSCINABTE** : There's more than one way to do it, but sometimes consistency is not a bad thing either. 不只一种方法来做一件事，但有时保持一致也不错。发音为 “Tim Toady Bicarbonate” 。
- Easy things should be easy, and hard things should be possible. 简单的事情应该是简单的，复杂的事情应该变得可能。



优点

- 很容易：容易使用，但学习 Perl 并不简单
- 几乎不受限制：几乎没有什么事是 Perl 办不到的
- 速度通常很快

缺点

- 灵活、随意和“过度”的冗余语法
- write-only：代码有点难看，令人难以阅读
- 解释器耗费资源



优点

- 很容易：容易使用，但学习 Perl 并不简单
- 几乎不受限制：几乎没有什么事是 Perl 办不到的
- 速度通常很快

缺点

- 灵活、随意和“过度”的冗余语法
- write-only：代码有点难看，令人难以阅读
- 解释器耗费资源



```
1 #!/usr/bin/perl
2 # Classic "Hello World" as done with Perl
3
4 use strict;
5 use warnings;
6
7 print "Hello World!\n";
```

```
1 # Step 1: 编写脚本
2 vim hello.pl
3 # Step 2: 修改权限
4 chmod 755 hello.pl
5 # Step 3: 运行脚本
6 ./hello.pl
```



```
1 #!/usr/bin/perl
2 # Classic "Hello World" as done with Perl
3
4 use strict;
5 use warnings;
6
7 print "Hello World!\n";
```

```
1 # Step 1: 编写脚本
2 vim hello.pl
3 # Step 2: 修改权限
4 chmod 755 hello.pl
5 # Step 3: 运行脚本
6 ./hello.pl
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



变量

Perl 是一种无类型语言 (untyped)，换句话说，在语言层面上，Perl 和大多数编程语言不同，不把变量分成整数、字符、浮点数等等，而只有一种能接受各种类型数据的“无类型”变量。

Perl 中各种变量的运算也很自由，数和含有数的字符串是等效的，可以把数字字符串参与数学计算，也可以反之，让数字参与字符串的构成和操作。

类型

- 标量：scalar；只包含一个元素的变量；以\$ 开头
- 数组：array；含有任意数量元素的变量，以其存储顺序作为索引；以@ 开头
- 散列：hash, associative array（关联数组）；像字典一样，把不同的变量按照它们的逻辑关系组织起来，并以作为“键”的变量进行索引；以% 开头

变量

Perl 是一种无类型语言 (untyped)，换句话说，在语言层面上，Perl 和大多数编程语言不同，不把变量分成整数、字符、浮点数等等，而只有一种能接受各种类型数据的“无类型”变量。

Perl 中各种变量的运算也很自由，数和含有数的字符串是等效的，可以把数字字符串参与数学计算，也可以反之，让数字参与字符串的构成和操作。

类型

- 标量：scalar；只包含一个元素的变量；以\$ 开头
- 数组：array；含有任意数量元素的变量，以其存储顺序作为索引；以@ 开头
- 散列：hash, associative array（关联数组）；像字典一样，把不同的变量按照它们的逻辑关系组织起来，并以作为“键”的变量进行索引；以% 开头

1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



```
1 # 标量: $scalar
2
3 # 字符串, 双引号
4 $name = "Paul";
5
6 # 数字, 无引号
7 $age = 29;
8
9 # 字符串, 单引号
10 $where_to_find_him = 'http://www.weinstein.
    org';
```



1 引言

2 变量

- 标量
- **数组**
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

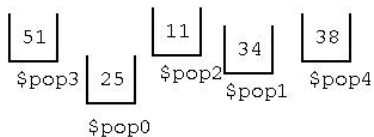
- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

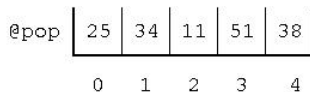
8 回顾与总结

- 总结
- 思考题

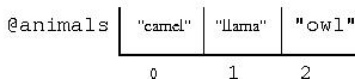




Individual scalar variables

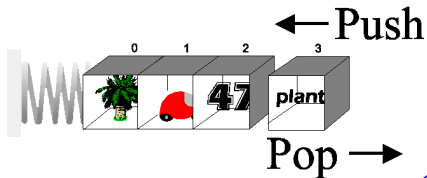
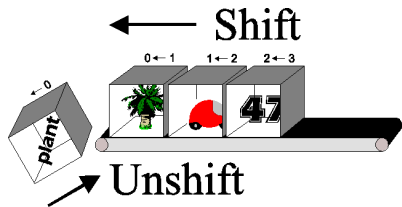
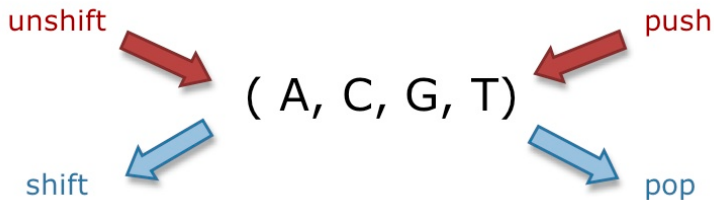


Array



```
1 # 数组: @array
2
3 # 字符串
4 @authors = ("Paul", "Joe", "Jeremy", "Harley"
5 );
6
7 # 数字
8 @list = (1, 2, 3, 4);
9
10 # 解引用: $array[index]
11 $authors[4] # Tom
12 $list[0] # 1
```





1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

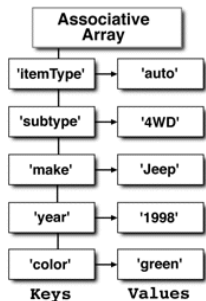
- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题





%Perez

Age	21
Name	"Jeff"
Eyes	"Br"
Temp	98.6

`$Perez{'Eyes'} = "Br";`

%Kleiner

Age	21
Name	"Dave"
Eyes	"Blue"
Temp	99.0

`$Kleiner{'Temp'} = 99.0;`



```
1 # 散列: %hash
2
3 # 创建散列
4 %person = (
5   name => 'Paul',
6   age  => '29',
7   url  => 'http://www.weinstein.org',
8 )
9
10 # 提取键值: $hash{key}
11 $person{"age"} # 29
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



变量 | 内置变量

Perl 提供了大量的预定义变量。下面列举了常用的一些预定义变量：

<code>\$_</code>	在执行输入和模式搜索操作时使用的默认空格变量
<code>\$.</code>	文件中最后处理的当前行号
<code>\$@</code>	由最近一个 <code>eval()</code> 运算符提供的 Perl 语法报错信息
<code>\$!</code>	获取当前错误信息值，常用于 <code>die</code> 命令
<code>\$0</code>	含有正在执行的程序名
<code>\$\$</code>	正在执行本脚本的 Perl 进程号
<code>\$PERL_VERSION / \$^V</code>	Perl 解释器的版本、子版本和修订版本信息
<code>@ARGV</code>	含有命令行参数
<code>ARGV</code>	一个特殊的文件句柄，用于遍历 <code>@ ARGV</code> 中出现的所有文件名
<code>@INC</code>	库文件的搜索路径
<code>@_</code>	在子例程中， <code>@_</code> 变量含有传给该子例程的变量内容
<code>%ENV</code>	关联数组型变量 <code>%ENV</code> 含有当前环境信息
<code>%SIG</code>	关联数组型变量 <code>%SIG</code> 含有指向信号内容的句柄



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



操作符	含义
+	加法
-	减法
*	乘法
/	除法
**	乘幂, 乘方
%	取模, 取余
<	小于
>	大于
==	等于
<=	小于等于
>=	大于等于
!=	不等于
<=>	比较。 $a<=>b$: a 等于 b 时返回 0, a 大于 b 时返回 1, a 小于 b 时返回 -1



操作符	含义
.	连接。"string1" . "string2"
x	重复。"string" x number
lt	小于
gt	大于
eq	等于
le	小于等于
ge	大于等于
ne	不等于
cmp	比较。类似于数字比较的<=>



操作符	含义
& &	逻辑 AND, 与
	逻辑 OR, 或
!	逻辑 NOT, 非
?=	条件操作符



操作符 | 文件测试操作符

操作符	含义
-r	可读
-w	可写
-x	可执行
-e	存在
-z	存在但没有内容
-s	存在且有内容
-f	普通文件
-d	目录
-l	符号链接
-T	看起来像文本文件
-B	看起来像二进制文件
-M	最后被修改后至今的天数
-A	最后被访问后至今的天数
-C	最后 inode 变更后至今的天数



操作符	含义
<code>=~</code>	绑定操作符，匹配
<code>!~</code>	绑定操作符，不匹配
<code>~~</code>	智能匹配操作符



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



基本函数 | `print, chomp`

```
1 # 向标准输出打印文本
2 print "Hello Again\n";
3
4 # 向一个具有文件句柄的文件打印文本
5 print FILE "Hello Again\n";
6
7 # 打印变量的值
8 print "How are on this day, the " . $date . "
   ?\n";
```

```
1 # 删除变量末尾的（多个）换行符，返回删除的换行符的个数
2 chomp $name;
3 chomp @authors;
```



基本函数 | `print, chomp`

```
1 # 向标准输出打印文本
2 print "Hello Again\n";
3
4 # 向一个具有文件句柄的文件打印文本
5 print FILE "Hello Again\n";
6
7 # 打印变量的值
8 print "How are on this day, the " . $date . "
   ?\n";
```

```
1 # 删除变量末尾的（多个）换行符，返回删除的换行符的个数
2 chomp $name;
3 chomp @authors;
```



基本函数 | join, split

```
1 # joining a number of strings together with a
   colon delimiter
2 $fields = join ':', $data_field1,
   $data_field2, $data_field3;
3
4 # splitting a string into substrings
5 ($field1, $field2) = split /\:/, 'Hello:World'
   , 2;
6
7 # splitting a scalar and creating an array
8 @fields = split /\:/, $raw_data;
```



基本函数 | open,close

```
1 # open the file and slurp its contents into
  an array and then close the file
2 open(FILE, "/etc/passwd");
3 @filedata = <FILE>;
4 close(FILE);
5
6 open my $IN, '<', $file_in or die "$0 :
  failed to open input file '$file_in' : $!\n
  ";
7 while(<$IN>) {
8     chomp;
9     actions;
10 }
11 close $IN or warn "$0 : failed to close input
  file '$file_in' : $!\n";
```



基本函数 | opendir, readdir, closedir

```
1 #!/usr/bin/perl -w
2
3 print "Read user's home directory\n";
4 opendir(HOMEDIR, ".");
5 @ls = readdir HOMEDIR;
6 closedir(HOMEDIR);
7
8 print "Create file dirlist.txt with a
    directory listing of user's home dir\n";
9 open(FILE, ">dirlist.txt");
10 foreach $item (@ls) {
11     print FILE $item . "\n";
12 }
13 close(FILE);
14 print "All done\n\n";
```



基本函数 | **my**,local

```
1 # Global variable $name is given a name
2 $name = "Paul";
3
4 # Enter our loop
5 foreach (@filedata) {
6     # declare a new variable for just the loop
7     my $current_file;
8
9     # create a local version of name to
10    temporarily assign values within the loop
11    to
12    local $name;
13    ...
14 }
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

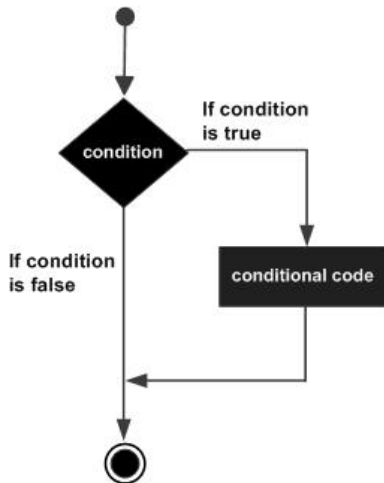
7 检修脚本

8 回顾与总结

- 总结
- 思考题



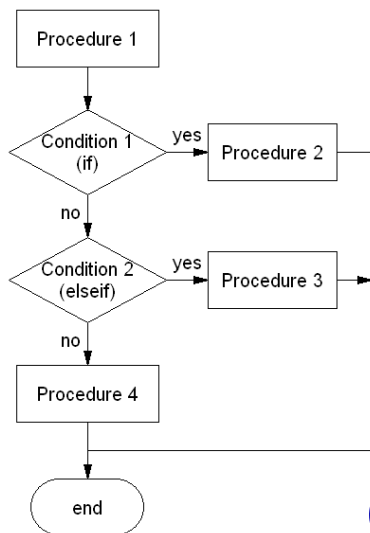
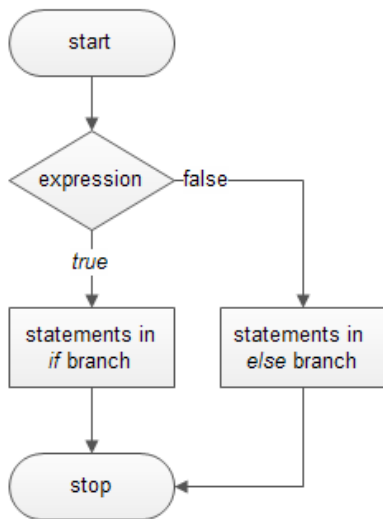
判断语句 | if | 逻辑流程



```
1 # if 区块
2 if ($hour > 22) {
3     print "should sleep...\n";
4 }
5
6 # if 语句
7 print "hello" if $guest >= 1;
```



判断语句 | if-else | 逻辑流程



判断语句 | if-else | 语法

```
1 if ($name eq "Paul") {  
2     print "Hi Paul\n";  
3 }  
4 elsif ($name eq "Joe") {  
5     print "Hi Joe\n";  
6 }  
7 elsif ($name eq "Jeremy") {  
8     print "Hi Jeremy\n";  
9 }  
10 else {  
11     print "Sorry, have we meet before?";  
12 }
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

● unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

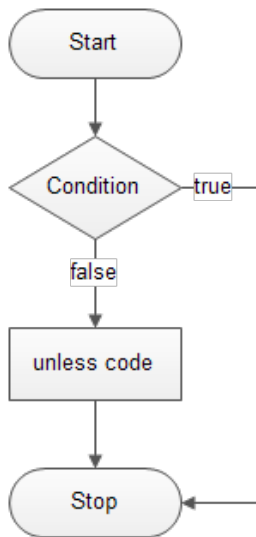
7 检修脚本

8 回顾与总结

- 总结
- 思考题



判断语句 | unless | 逻辑流程



```
1 # unless 区块
2 unless ($credit > 100) {
3     print "You can not graduate!\n";
4 }
5
6 # unless 语句
7 print "eat\n" unless $food==0;
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

● given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

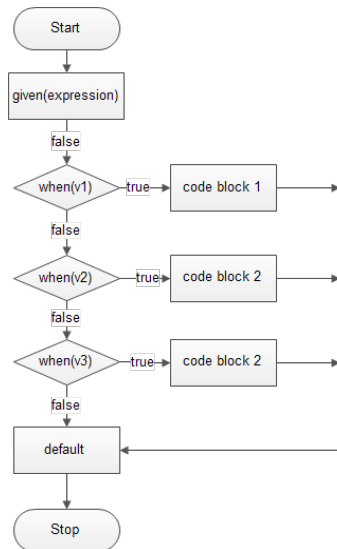
7 检修脚本

8 回顾与总结

- 总结
- 思考题



判断语句 | given-when | 逻辑流程



判断语句 | given-when | 语法

```
1 use 5.010;  
2 given ($foo) {  
3     say "a" when "a";  
4     when (/b/) {say "b";}   
5     default {say "not match";}   
6 }
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句

- for 语句

- while 语句

- until 语句

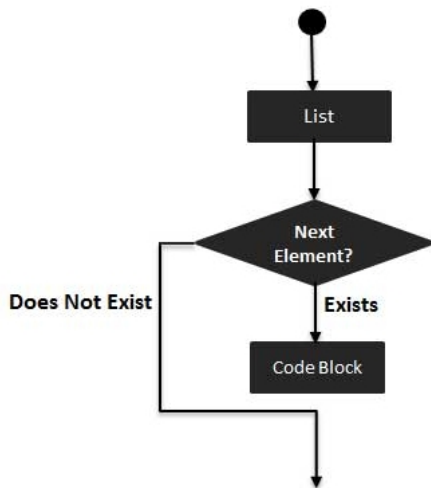
7 检修脚本

8 回顾与总结

- 总结

- 思考题





循环语句 | foreach | 语法

```
1 @group = 1..10;
2
3 # foreach循环
4 foreach my $element (@group) {
5     print "$element\n";
6 }
7
8 # 等价的for循环
9 for (@group) {
10     print "$_\n";
11 }
12 print "$_\n" for @group;
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句

- for 语句

- while 语句

- until 语句

7 检修脚本

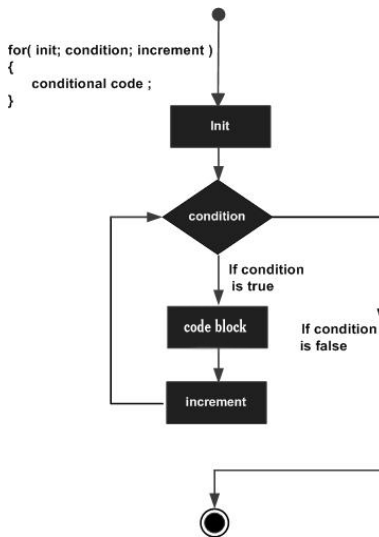
8 回顾与总结

- 总结

- 思考题



循环语句 | for | 逻辑流程



```
1 # 从1数到10
2 for ($i = 1; $i <= 10; $i++) {
3     print "I can count to $i!\n";
4 }
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句

- for 语句

- while 语句

- until 语句

7 检修脚本

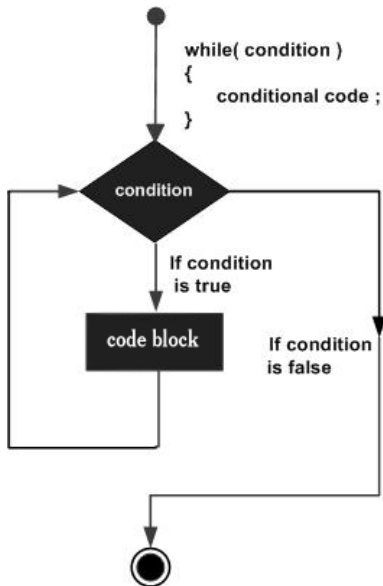
8 回顾与总结

- 总结

- 思考题



循环语句 | while | 逻辑流程

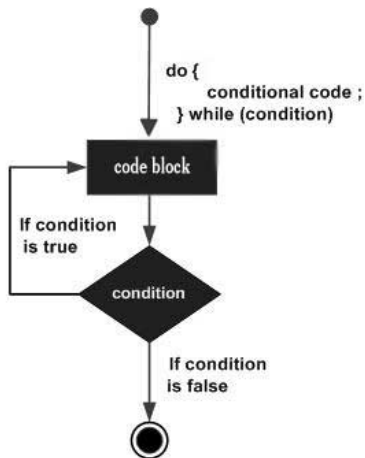


循环语句 | while | 语法

```
1 $i = 0;  
2 while ($i < 10) {  
3     print "$i\n";  
4     $i++;  
5 }
```



循环语句 | do-while | 逻辑流程



循环语句 | do-while | 语法

```
1 $i = 0;  
2 do {  
3     print "$i\n";  
4     $i = $i + 1;  
5 } while ($i < 10);
```



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

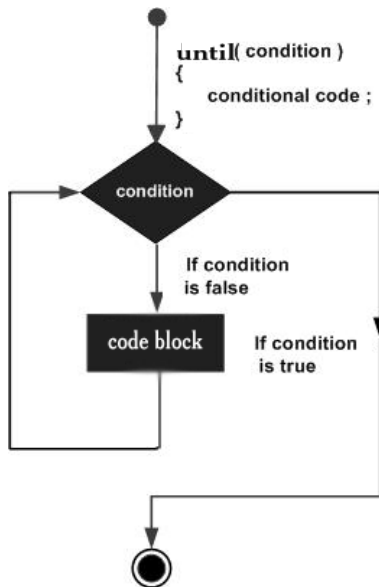
7 检修脚本

8 回顾与总结

- 总结
- 思考题



循环语句 | until | 逻辑流程

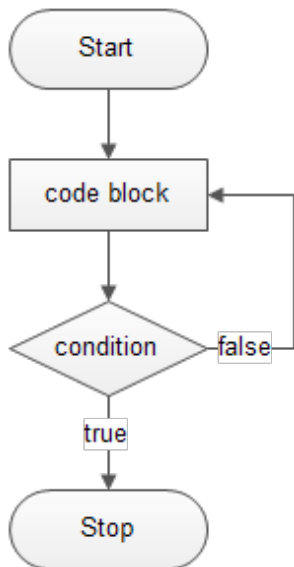


循环语句 | `until` | 语法

```
1 $i = 0;  
2 until ($i == 10) {  
3   print "$i\n";  
4   $i++;  
5 }
```



循环语句 | do-until | 逻辑流程



循环语句 | do-until | 语法

```
1 $i = 0;  
2 do {  
3     print "$i\n";  
4     $i++;  
5 } until ($i == 10);
```



1

引言

2

变量

- 标量
- 数组
- 散列
- 内置变量

3

操作符

4

基本函数

5

判断语句

- if 语句

● unless 语句

● given-when 语句

6

循环语句

● foreach 语句

● for 语句

● while 语句

● until 语句

7

检修脚本

8

回顾与总结

● 总结

● 思考题



```
1 # 不洁模式
2 #!/usr/bin/perl -T
3
4 # 打开警告
5 use warnings;
6
7 # 严格模式, 语法更加规范
8 use strict;
```



检修脚本

```
1 # 检查语法
2 perl -c script.pl
3
4 # 格式化脚本
5 perltidy script.pl
6
7 # 调试脚本
8 perl -d script.pl
```



命令	功能
s	仅执行脚本中的一行
n	按步执行以避免进入子程序
c	继续运行直至遇到断点
r	继续运行直至从当前子程序中执行返回命令
w	显示当前行前后的代码
b	设置断点
x	显示变量的值
W	设置监视表达式
T	显示栈回溯追踪
L	列出所有的断点
D	删除所有的断点
R	重新启动脚本以便再次测试



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



知识点

- Perl 语言简介：中心思想，优缺点，语法结构
- 变量：标量，数组，散列，内置变量
- 操作符：数字、字符串、逻辑、文件测试、匹配操作符
- 基本函数：print, chomp, join, split, open, close, my
- 判断语句：if, unless, given-when
- 循环语句：foreach, for, while, until
- 检修脚本：检查语法，格式化脚本，调试脚本

技能

- 掌握 Perl 语言的基本语法
- 使用 Perl 编写简单的应用脚本

1 引言

2 变量

- 标量
- 数组
- 散列
- 内置变量

3 操作符

4 基本函数

5 判断语句

- if 语句

- unless 语句

- given-when 语句

6 循环语句

- foreach 语句
- for 语句
- while 语句
- until 语句

7 检修脚本

8 回顾与总结

- 总结
- 思考题



- 1 Perl 语言的中心思想是什么？
- 2 Perl 中的变量主要有哪三大类？
- 3 列举 Perl 中的各种操作符。
- 4 chomp, join, split 的作用是什么？
- 5 在 Perl 中如何和文件进行交互？
- 6 Perl 中的判断语句有哪些，语法是怎样的？
- 7 Perl 中的循环语句有哪些，语法是怎样的？
- 8 如何调试 Perl 脚本？





TEX

LATEX

X_YTEX

Beamer

