**Perl语言编程**

**Chapter 1 绪论和Markdown**

**1.生物信息学常用编程语言诞生年份：**Perl（1987）、Python（1991）、Ruby（1995）、Java（1995）

**2.R语言IDE**：RStudio

**3.部分R语言模块：**

Readr：tabular data

Tidyr：tidy format

Magrittr：forward-pipe operator

Dplyr： fast data manipulation

Lubridate：dates and times

Stringr：regular expressions and character strings

Ggplot2：the grammar of graphics

Rmarkdown：markdown document

Knitr：reproducible research

**4.**一个Markdown段落前后要有**一个以上的空行**，如果想**强迫换行**，可以在**行尾插入至少两个空格。**

**5.Markdown中两种标题的语法：**

**①类Setext形式**是用底线的形式，利用=（最高阶标题）和-（第二 阶标题）。任何数量的=和-都可以有效果。

**②类atx形式**则是在行首插入1到6个#，对应到标题1到6阶。

**6.**Markdown标记区块引用使用>的引用方式，可以在每行的最前面加上>，也可以只在整个段落的第一行最前面加上>。区块引用可以嵌套，只要根据层次加上不同数量的>即可。

**7.Markdown列表：**

①无序列表使用星号（\*）、加号（+）或是减号（-）作为列表标记。

②有序列表则使用数字接着一个英文句点。（在列表标记上使用的数字并不会影响输出的 HTML结果）

1. 列表项目可以包含多个段落，每个项目下的段落都必须缩进**4个空格**或是**1个制表符**。

**Chapter 2 Perl 语言入门**

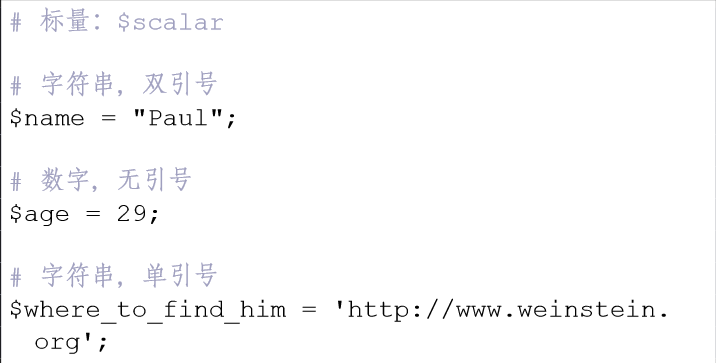
**1.**Perl语言创始人拉里·沃尔（Larry Wall），1987年12月18日。第三方代码库CPAN。

**2. perl -v**查看版本确认是否已安装，**perl -c**检查语法，**perltidy**格式化。

**3.Perl中三种变量类型及区别：**

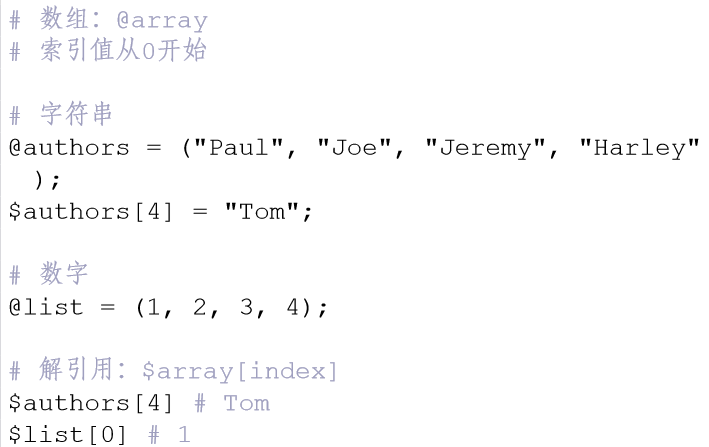
**①标量**：scalar；只包含一个元素的变量；以$开头 。

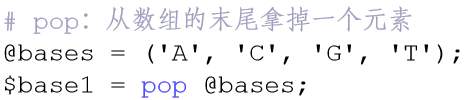
语法：

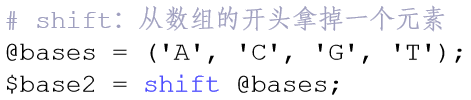


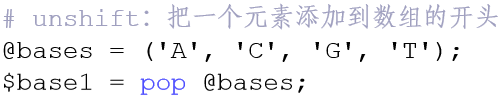
**②数组**：array；含有任意数量元素的变量，以其存储顺序作为索引； 以@开头 。

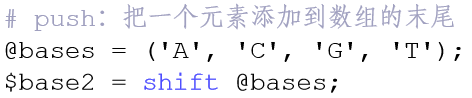
语法：

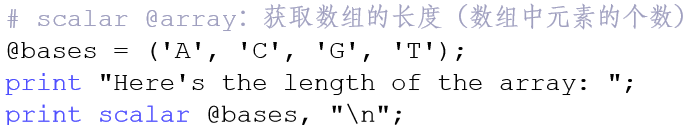


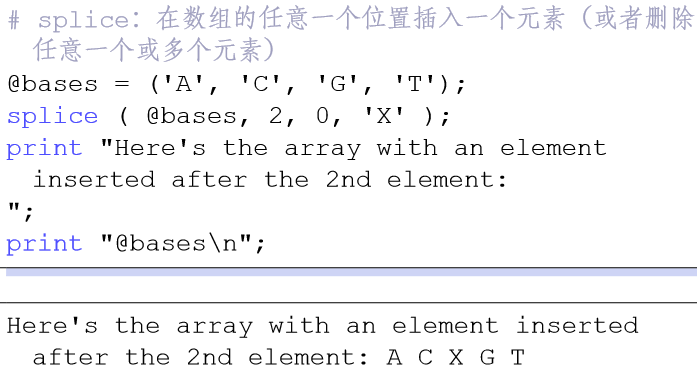












**③散列**：hash，associativearray（关联数组）；像字典一样，把不同的变量按照它们的逻辑关系组织起来，并以作为“键”的变量进行 索引；以%开头。

语法：



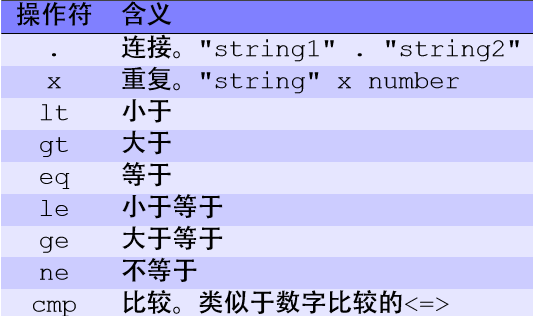
在任何需要逗号（,）的地方都可以用胖箭头（=>）代替，胖箭头左边的任何裸字（一连串的字母、数字和下划线，但不得以 数字开头）都会自动加上引号，因此胖箭头左边的裸字不需要加引号，列表末尾有一个额外的逗号，这种写法无伤大雅，而且便于维护 。

1. **操作符：**

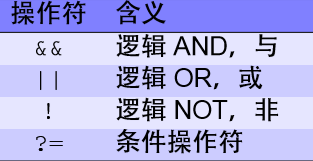
**①数字操作符：**



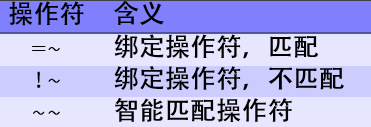
**②字符串操作符：**



**③逻辑操作符：**

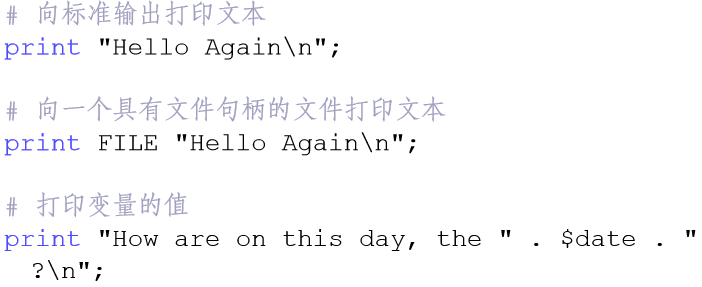


**④匹配操作符：**



1. **基本函数：**

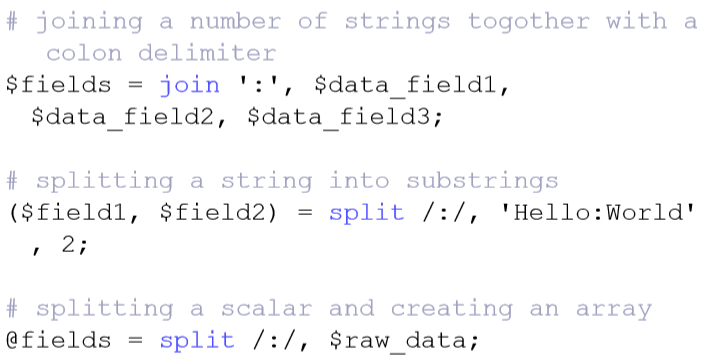
**①print：**



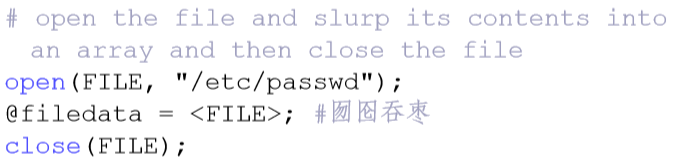
**②chomp:**



**③join, split:**

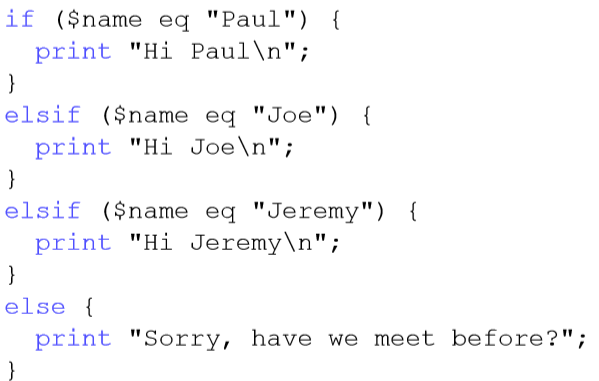


**④open,close:**

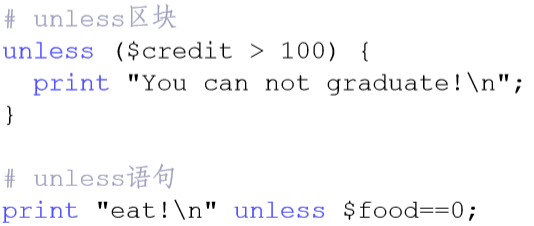


1. **判断语句：**

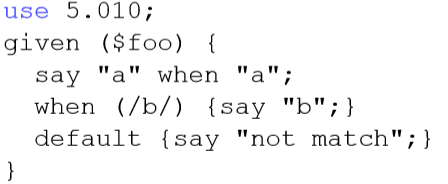
**①if-else语法:**



**②判断语句unless语法:**

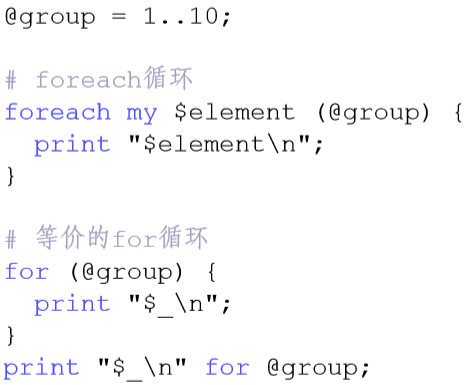


③判断语句given-when语法:

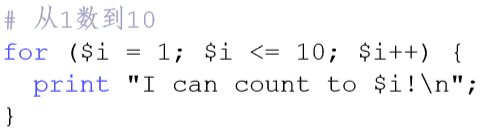


1. **循环语句 ：**

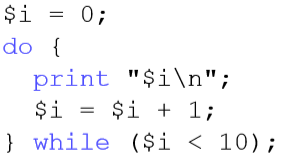
**①foreach语法：**



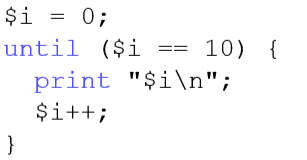
**②for语法：**



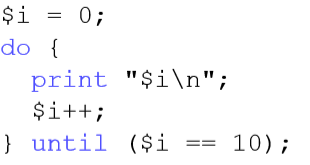
③do-while语法：



**④until语法：**

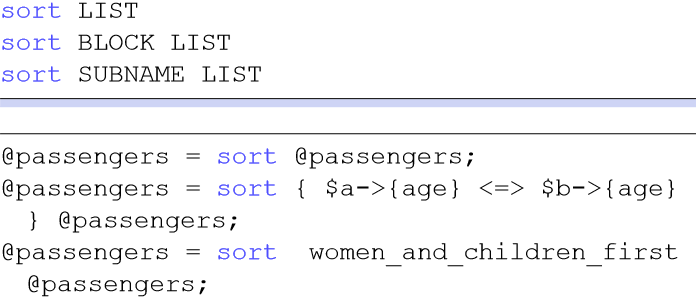


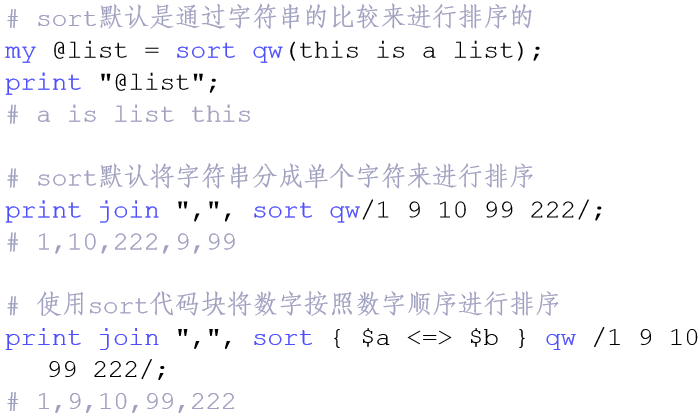
⑤do-until语法：

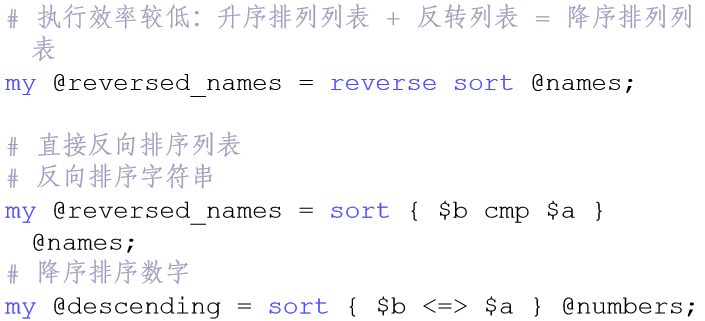


1. **列表操作：**

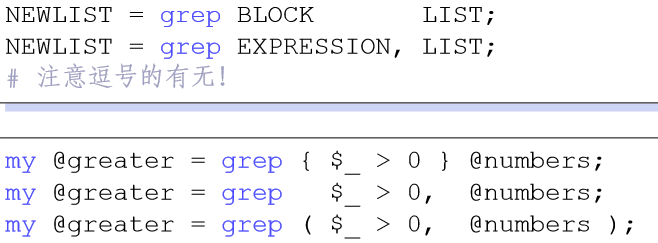
**①sort语法：**



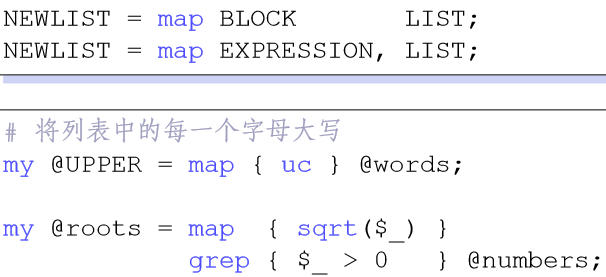




②grep语法：生成一个新的列表，该列表中的所有元素由一个给定列表中所有满足筛选标准的元素组成。



③map语法：在想要改变整个列表内容时使用 map。它逐个对给定列表中的元素进行 处理，然后新建一个列表，并将修改后的值返回到新的列表中。



1. **检修脚本**

不洁模式：#!/usr/bin/perl -T

打开警告：use warnings;

严格模式：use strcit;

开启诊断功能：use diagnostics;

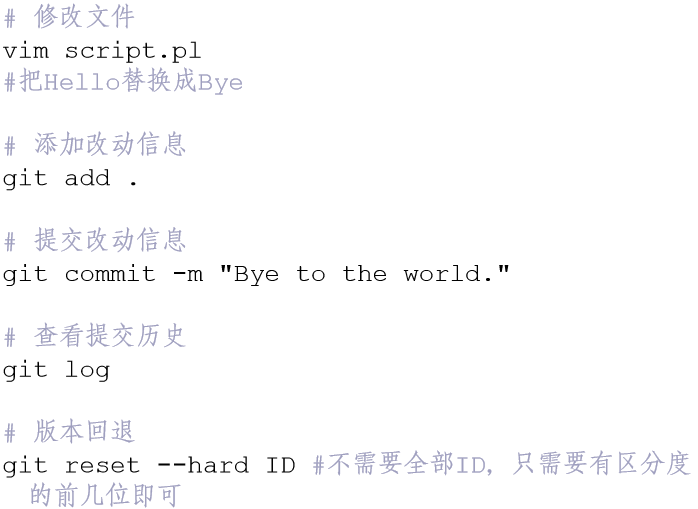
启用UTF-8编码：use utf8;

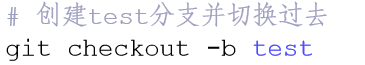
**Chapter 3 编程的艺术**

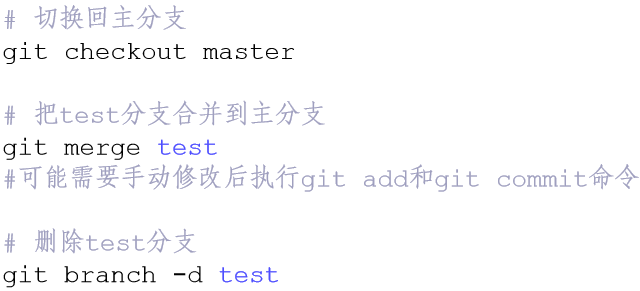
**1.Git**是一个**分散式版本控制软件**，最初由**林纳斯·托瓦兹**（Linus Torvalds）创作，于**2005**年以GPL释出，采用了**分布式版本库**的做法。

**2.Git使用语法：**

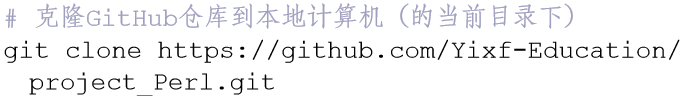












**3.编程过程总结：**

①分析任务属性：任务数量、处理频率、时间限制等。

②确定输入输出：数据格式、数据量、数据校验、输出形式等。

③选择合适算法：算法速度、算法难易、对应数据结构等。

④编写伪代码：整理思路、优化构思、选择编程范式、调整细节等。

⑤编写程序代码：编辑、调试、运行、完善等。

**Chapter 4 序列和字符串**

**1.**在Perl中，语句以分号“;”结尾。

**2.**Perl 中变量名的规范：只能由**大小写字母、数字和下划线“\_”**组成， 且**第一个字符不能是数字**。

**3.单双引号区别：**单引号（' '）不会进行变量内插。 双引号（" "）能够进行变量内插，可以使用转义字符。

**4.**在Perl中“=”不表示相等（数学），而是进行赋值，“==”表示数字相等。

**5.五种合并输出两条序列的方式：**

① print $DNA1, "\n";

print $DNA2, "\n\n";

② print "$DNA1$DNA2";

③$DNA3 = $DNA1 . $DNA2;

print "$DNA3\n\n";

④ print $DNA1, $DNA2, "\n";

⑤ $DNA3 = join ( "", ($DNA1, $DNA2) );

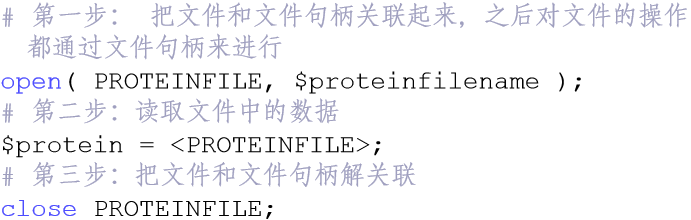
print "$DNA3\n\n";

**6.**替换语法：$RNA =~ s/T/U/g;

**7.**序列反转： $revcom = reverse $DNA;

**8.**替换字符集：$revcom =~ tr/ACGTacgt/TGCAtgca/;

**9.读取文件处理方法：**

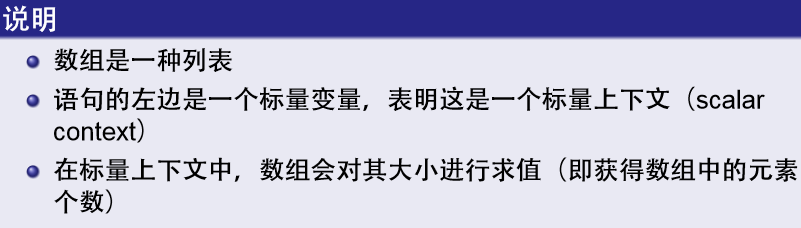


**对标量变量文件句柄一次只能读取输入文件的一行，下次自动读取文件的下一行。**

1. **上下文：**

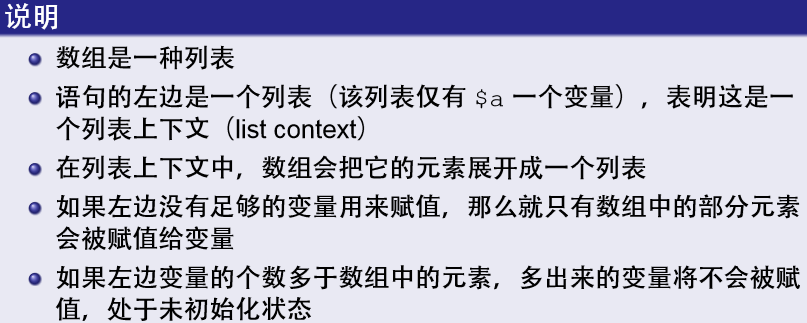
**①标量上下文：**

**$a = @bases;**



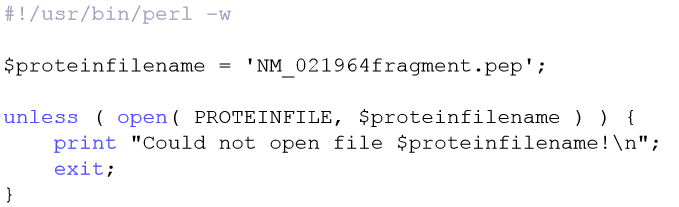
**②列表上下文：**

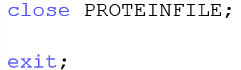
**($a) = @bases;**



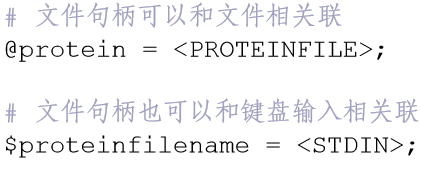
**Chapter 5 基序和循环**

**1.检查系统调用：**





**2.获取键盘输入：**



**3.chomp vs. chop：**chomp会去掉字符串末尾的换行符（有则去，没有则不进行任何处理） chop删除字符串末尾的最后一个字符（不管最后一个字符是什么，都会被去掉）。

**4.数组变标量：**



**join**把数组中的元素合并成一个字符串，元素之间用指定的字符串进行分隔 。

1. **字符串变数组：**



**split**会以指定的字符串（第一个参数）为分隔符来拆解字符串（第二个参数）如果第一个参数是空字符串，split会把字符串拆解成单个的字符。

**6.do-until 先执行一次代码块，之后再进行条件测试。（不是先测试后执行）**

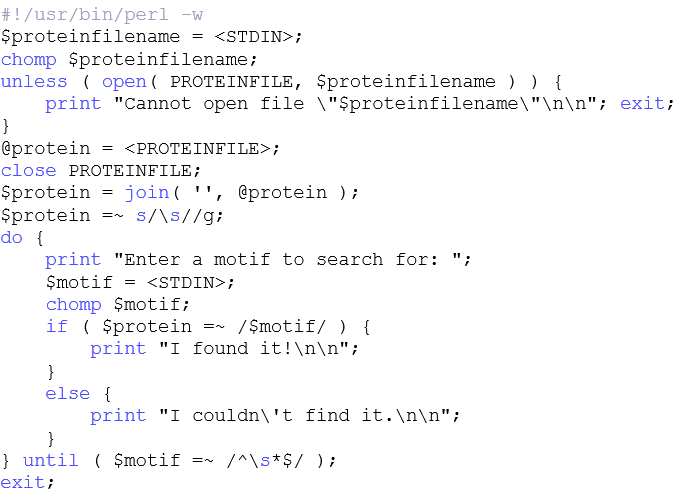
**7.正则表达式：**

元字符**\s**：匹配空格、制表符、换行符、换页符和回车符

字符集**[ \t\n\f\r]**：分别表示空格、制表符、换行符、换页符 和回车符

开头（^），结尾（$），有零个或者多个（\*）

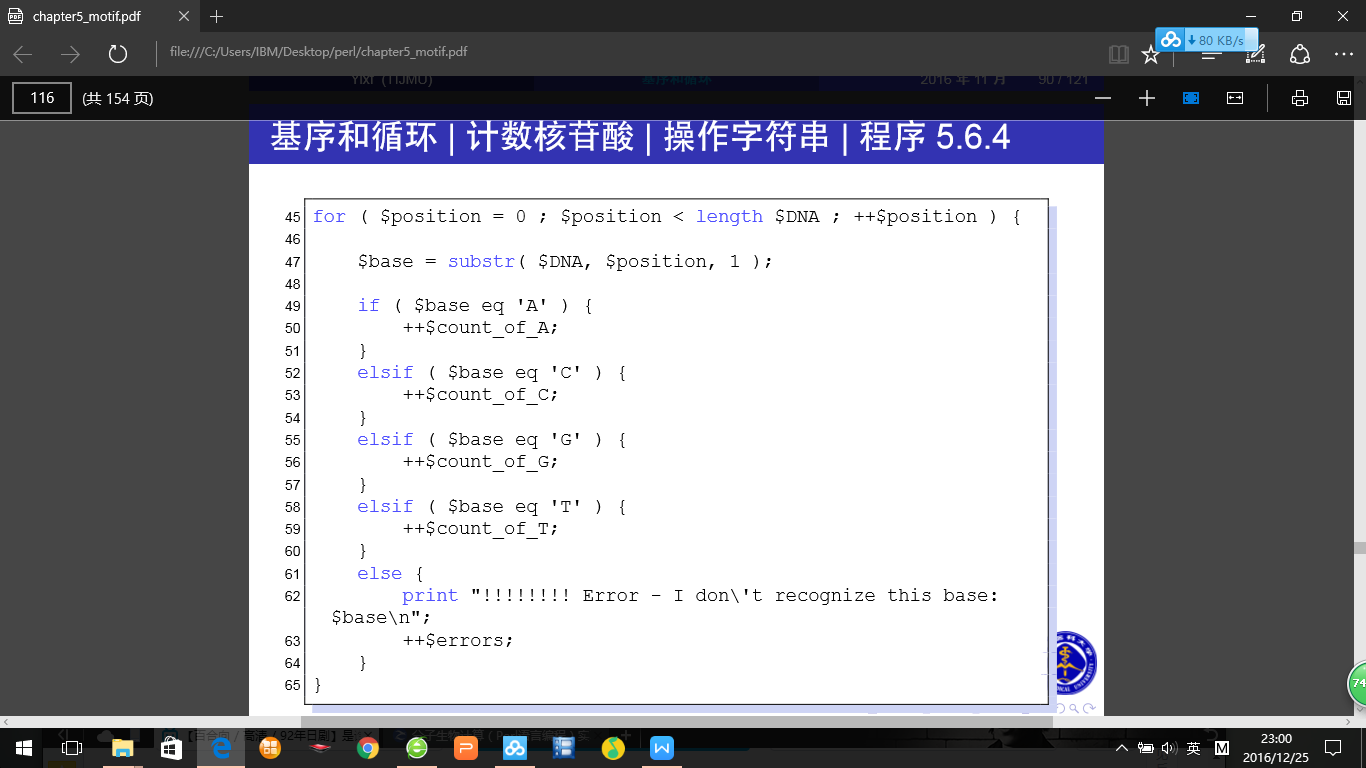
**8.模式匹配（查找基序）：**



**9.计数核苷酸策略：**

①把DNA拆解成单个碱基，存储到数组中，对数组中的元素进行迭代处理。

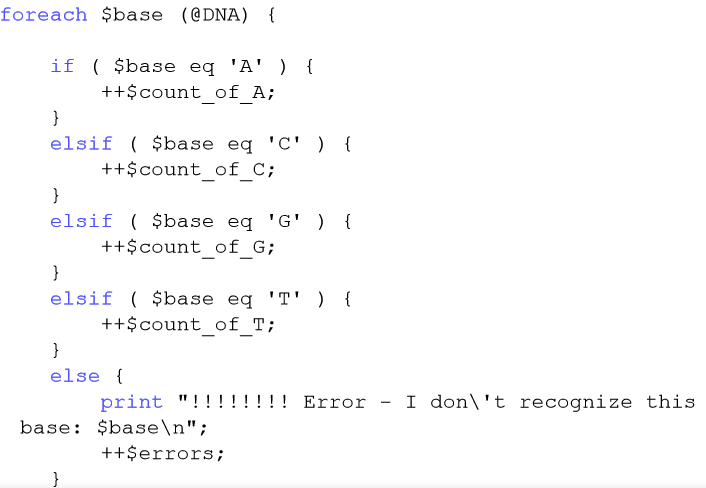




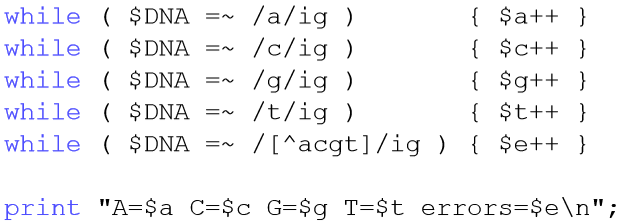
**substr：**获取位置索引为$position的那个碱基。第一个参数指定要操作的字符串；第二个参数指定要操作的位置索引（负值表示从字符串末尾开始）；第三个参数指定要操作的长度（负值表示字符串末尾剩余的字符数）；第四个参数指定要替换成的字符串。

②对DNA字符串中的位置进行迭代处理。





③



**while循环：**i:不区分大小写; g：全局修饰符，匹配字符串中的所有a,没有g的话，如果字符串中有a，会陷入死循环。

1. **加1的四种方法：**

++$count; （先加1，后返回值）

$count++; （先返回值，后加1）

$count = $count + 1;

$count += 1;

**11.**文件测试：

unless ( -e $dna\_filename )

**Chapter 6 子程序和Bugs**

1. **子程序调用：子程序的名字后面跟上用小括号包裹起来的参数列表(eg: addACGT($dna); )。**

**2.子程序分为三部分：**子程序定义的保留字sub、子程序的名字、包裹在大括号中的代码块

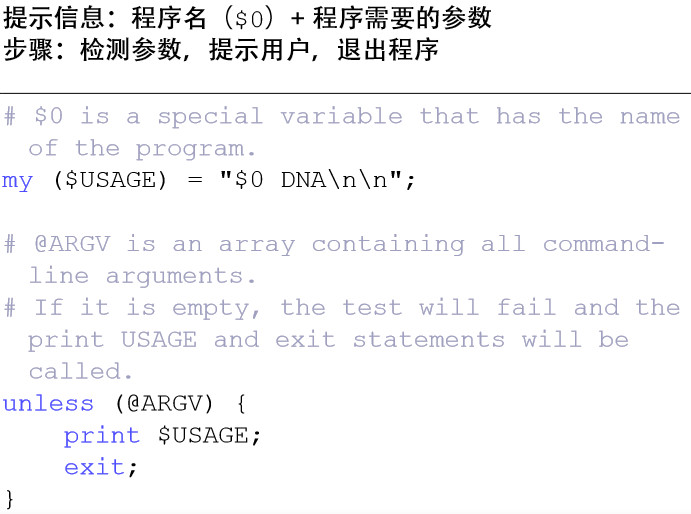
**3.**传递给子程序的参数：

参数：调用子程序时传递给它的值

使用特殊变量@\_向子程序传递参数值

**4.**返回值：使用return函数返回子程序的结果，可以返回：标量、标量列表、数组等。

**5.**特殊变量：**$0：程序名、@ARGV：所有的命令行参数**



1. **传递数据给子程序的两种方法：**

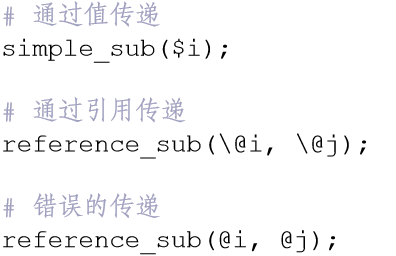
**①通过值（value）传递/调用：**调用子程序时，参数的值被复制并传递给子程序；子程序中这些值的变化不会影响到主程序中相应参数的值。

适用于：传递单个标量、标量列表、单个数组。

simple\_sub($i);

**②通过引用/参考/指针（reference）传递/调用：**在变量名前加一个反斜线（\）；在子程序中对参数变量值的操作会影响到主程序中参数的值；解引用时在变量名前有两个符号：（从左到右）表明变量类型的本来的符号和表明是引用的$符号（如：shift(@$j);）。

reference\_sub(\@i, \@j);



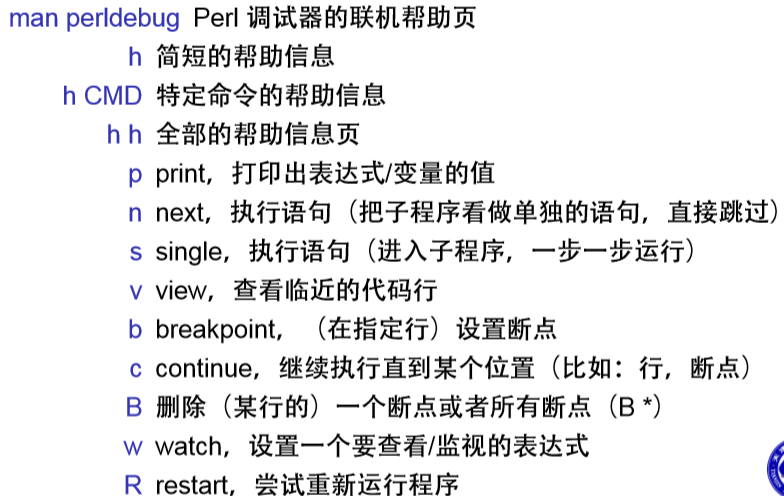
1. **模块：**

模块的后缀：.pm（比如：BeginPerlBioinfo.pm）

模块（.pm文件）的最后一行必须是：1;

使用模块：在靠近程序顶部的地方加上语句 use BiginPerlBioinfo;（此处不需要.pm后缀）

**8. Perl调试器常用命令：**



q：停止调试

调试器显示的是将要执行的那一行代码，而不是已经执行的代码行

使用v查看临近代码行时，当前行（即将被执行的行）会以==>进行标示

重复键入v可以持续显示更多代码，使用减号-会上翻一屏

**Chapter 7 突变和随机化**

1. **点突变的类型：**

转换、颠换

无义突变（终止）、错义突变（另一种氨基酸）、无表型突变（同一氨基酸）

1. **随机数生成器：**

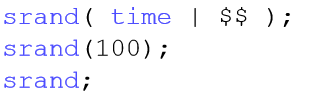
随机数生成器输出的数字是伪随机数，并不是真正随机的

一个随机数生成器作为一种算法，是可以被预测的

随机数生成器需要一个种子（seed）作为输入，种子改变，伪随机数随之改变

初始化使用的种子本身应该是随机选择的

**3.设置种子：**



srand;会自动设置种子

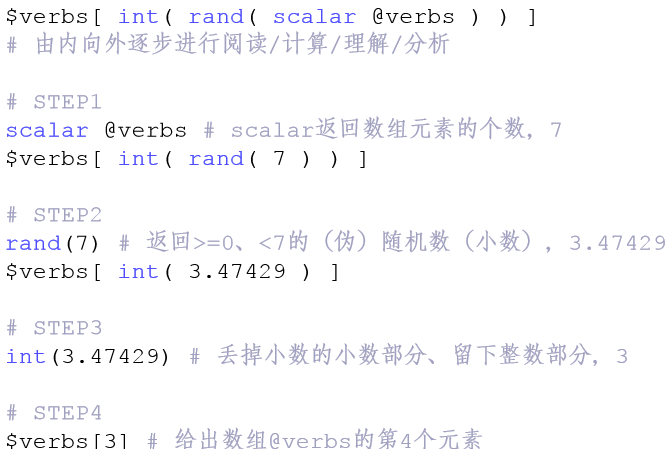
rand会自动调用srand;设置种子

time返回代表时间的数

$$返回运行的 Perl 程序的 PID（每次运行都会改变）

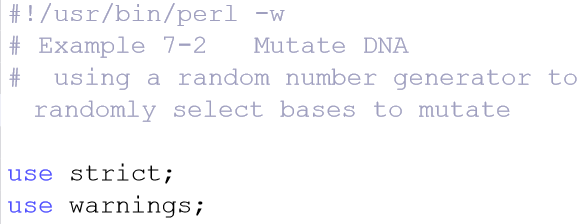
|表示位元的或运算（bitwiseOR），把两个数的位（bit）组合起来 逻辑或：0 or 0 = 0；0 or 1 = 1；1 or 0 = 1；1 or 1 = 1

**4.随机选取数组元素：**

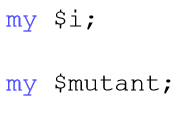




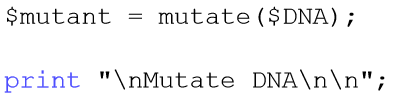
**5.模拟DNA突变：**

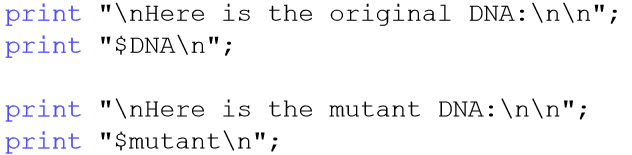


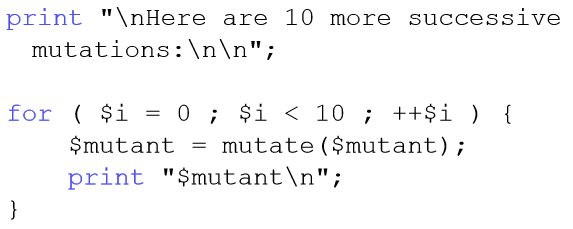


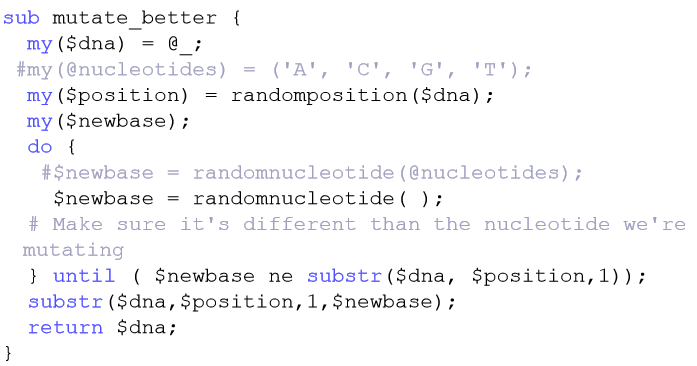


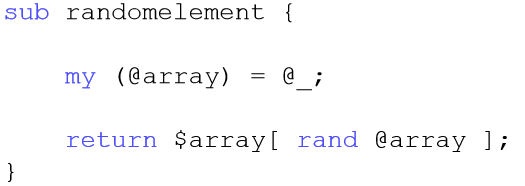


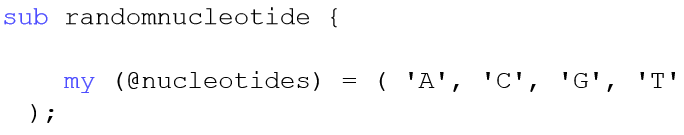










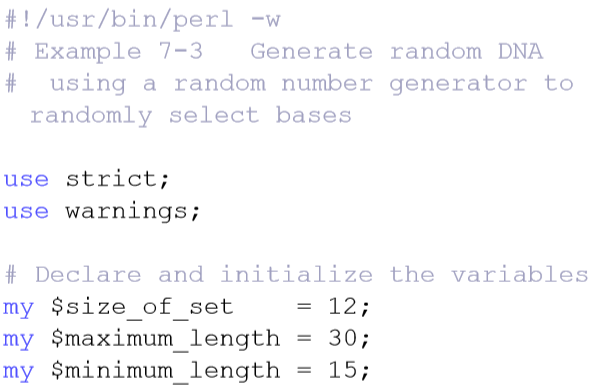






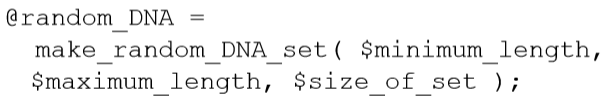


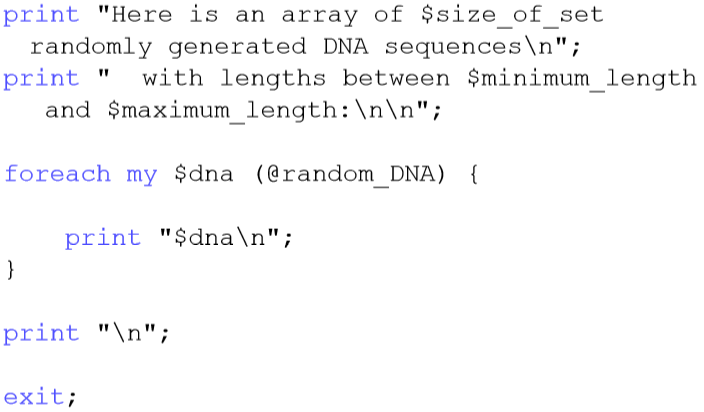
**6.生成随机DNA：**



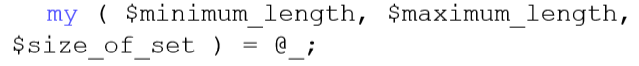














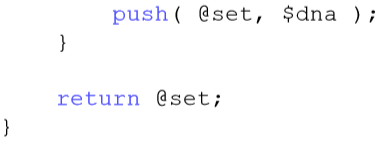


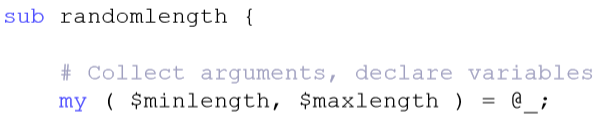


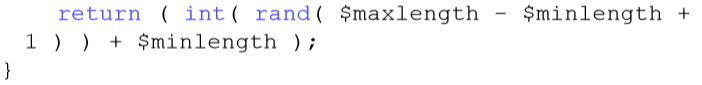


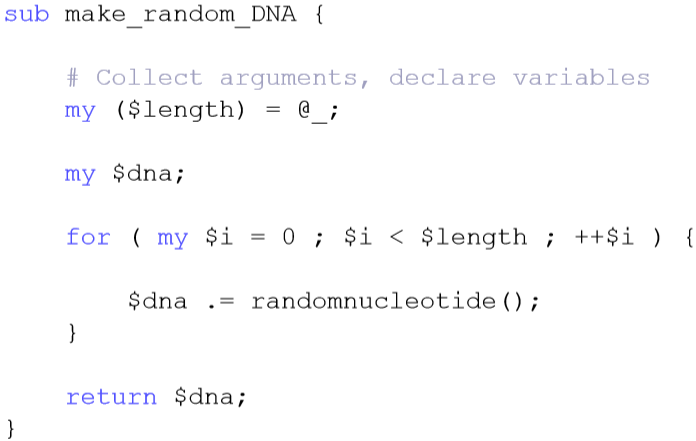




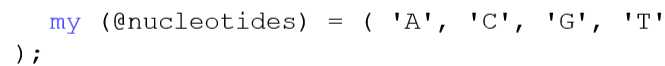




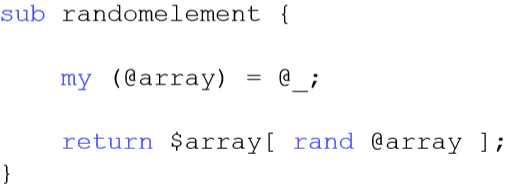




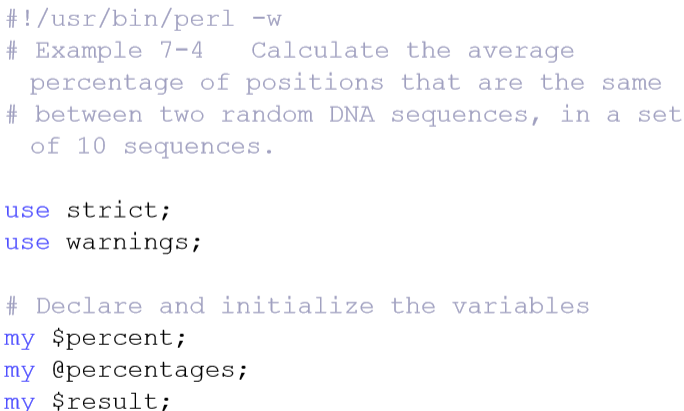






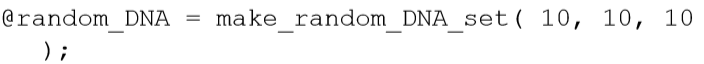


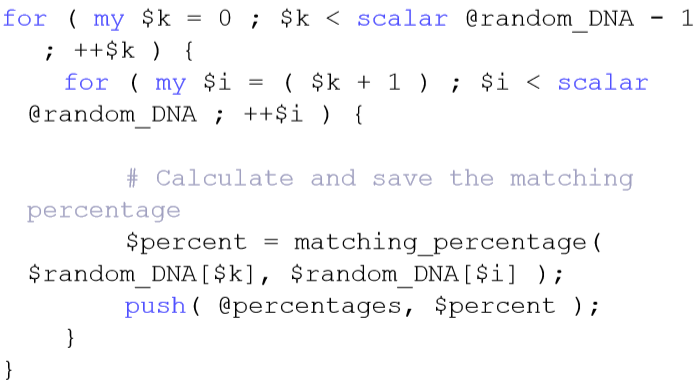
1. **计算两条DNA相似度：**

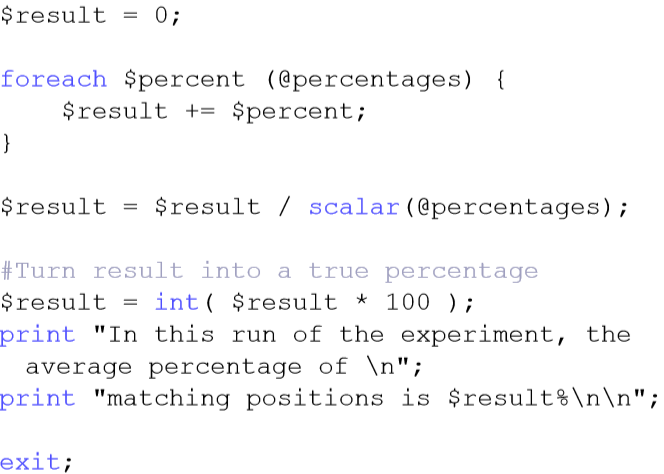


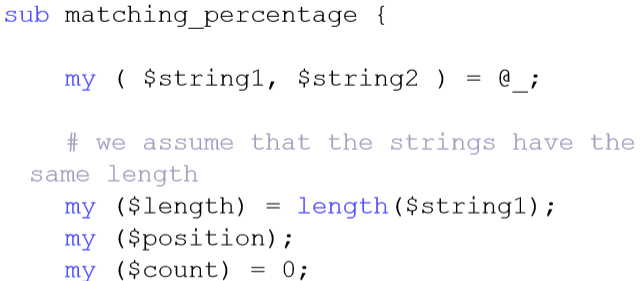


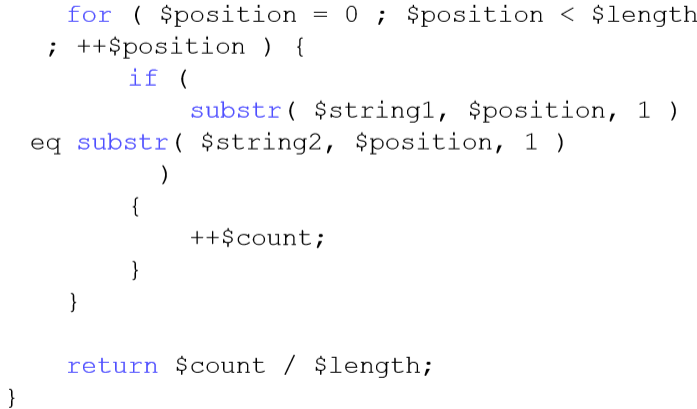




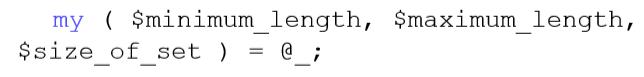












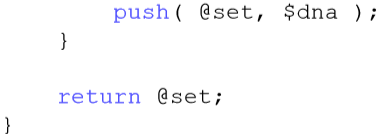






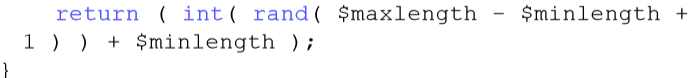


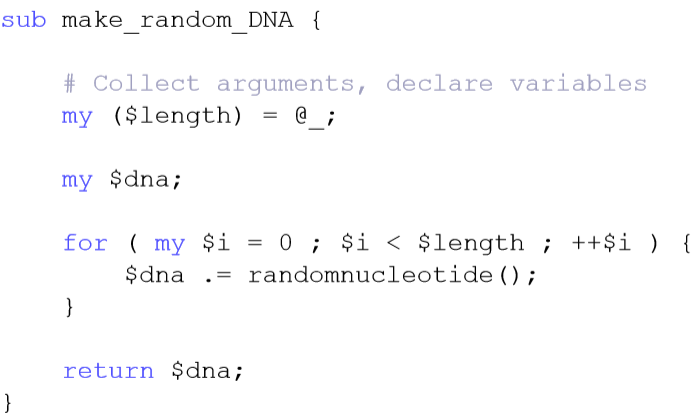


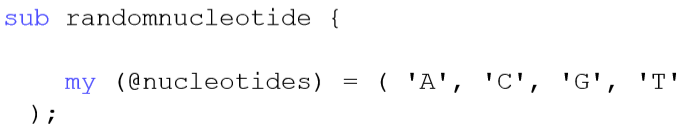




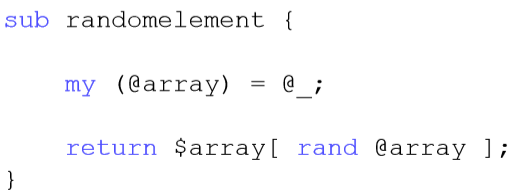








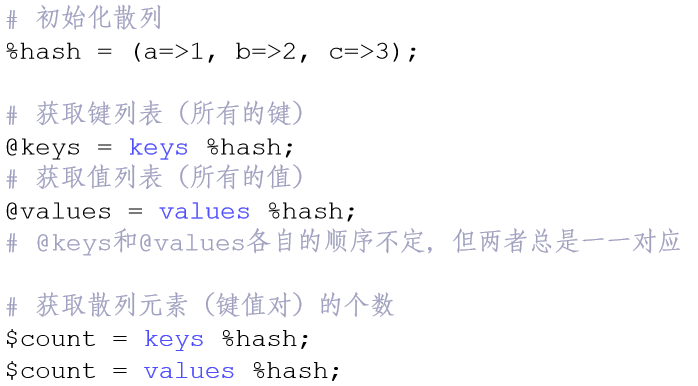


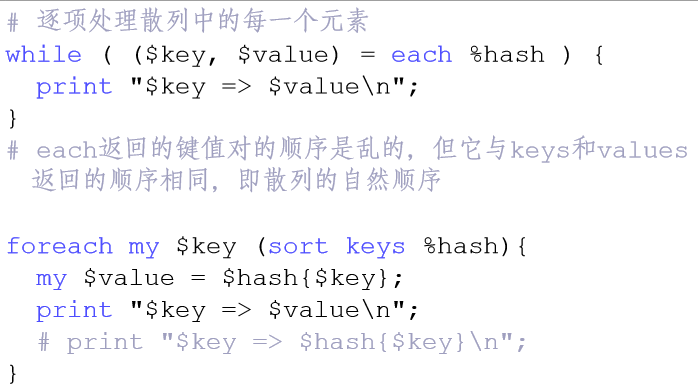


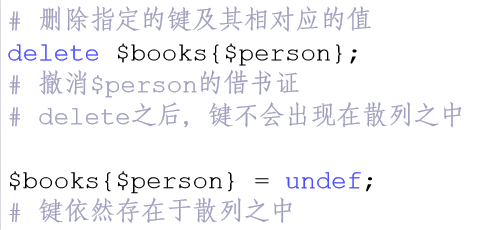
**Chapter 8 遗传密码**

1. **散列**

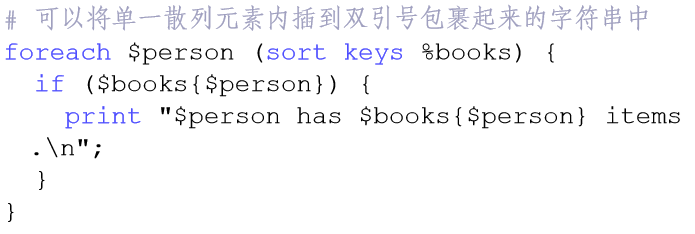
**给键赋值：$english\_dictonary{'recreant'} = "One who calls out in surrender.";**

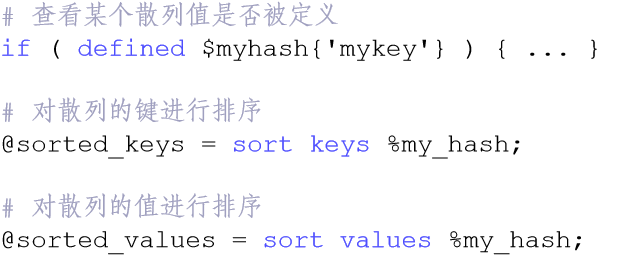






**键值反转： %inverse\_hash = reverse %hash;**



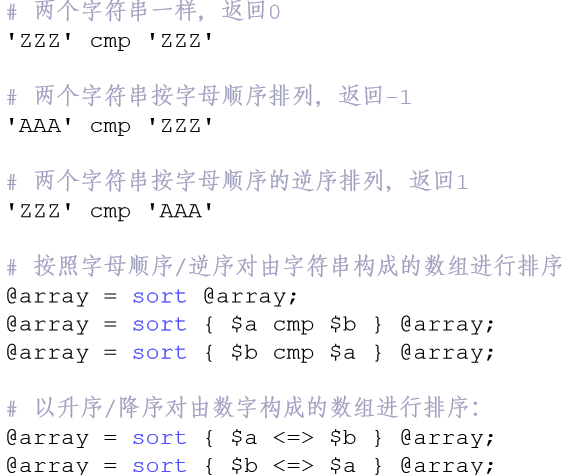


**defined & exists：**

defined：判断某个字符串是undef而不是空字符串，如果是undef，返回假；否则返回真。

exists：检查散列中是否存在某个键，和键对应的值无关如果键存在，返回真；否则返回假。

**2.比较与排序**



**3.文件句柄**

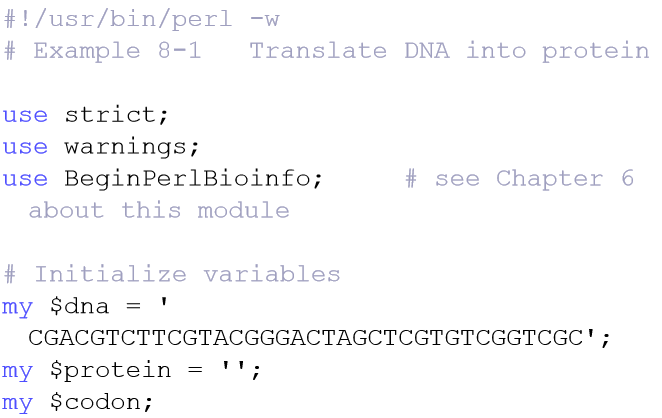
STDIN：标准输入（默认键盘）

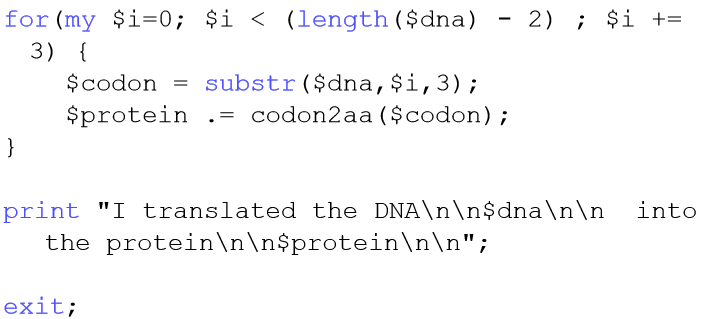
STDOUT：标准输出（默认屏幕），print默认使用STDOUT ，print可以使用一个文件句柄作为可选的参数

STDERR：标准错误输出（默认屏幕）

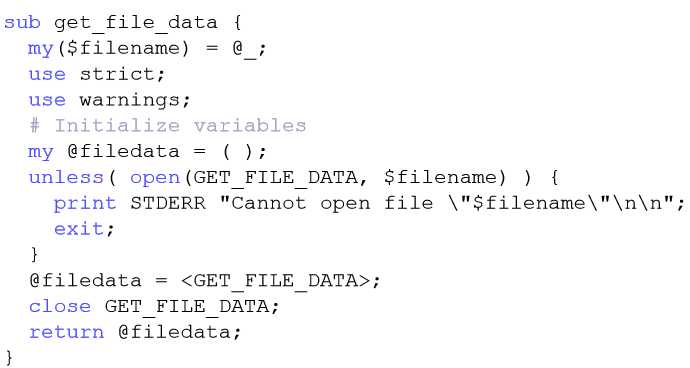
1. **翻译核苷酸：**

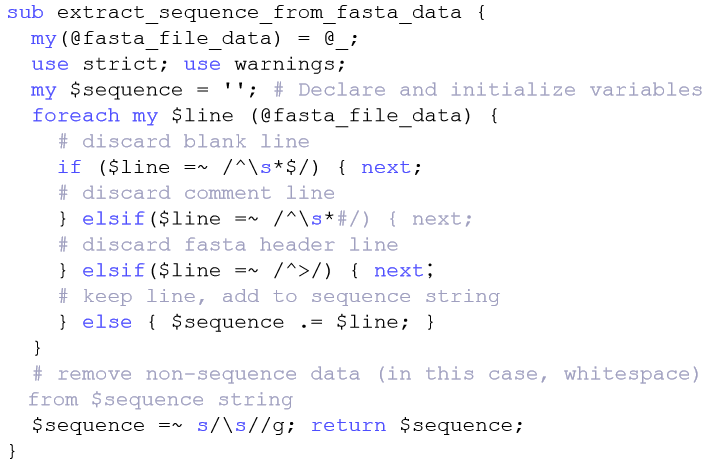
uc $codon：把输入的参数转换为大写

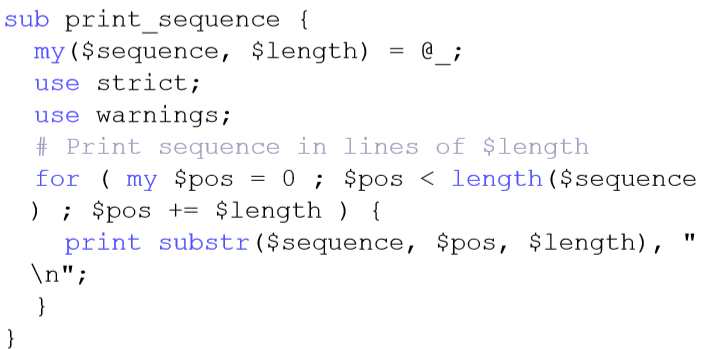


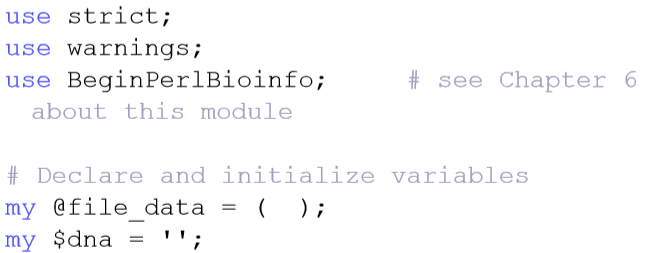


1. **读取FASTA文件：**

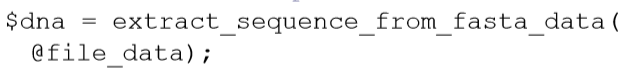


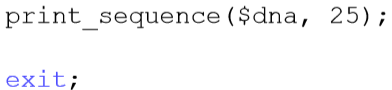




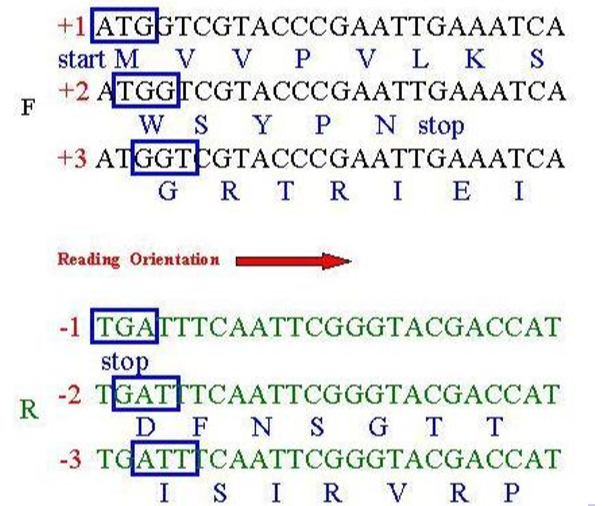




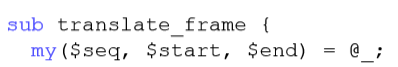


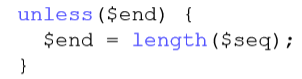


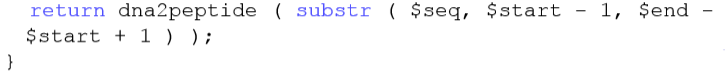
1. **阅读框架种类：**



1. **翻译指定区间子程序：**







**8.**阅读框翻译：

