天津医科大学实验课教案首页

课程名称：分子生物计算　　**实验名称：**实验 5　子程序和 Bugs

教师姓名：伊现富　　**职称：**讲师　　**教学日期：**2018 年 12 月 11 日 8:00-9:40

授课对象：生物医学工程与技术学院 2016 级生信班（本）　　**实验人数：**28

实验类型（验证型、综合型、设计型、创新型）：验证型　　**实验分组：**一人一机

学时数：2　　**教材版本：**Perl 语言在生物信息学中的应用——基础篇

实验目的与要求：

- 了解 Perl 语言中的模块和子程序库。
- 熟悉 Perl 调试器。
- 掌握 Perl 语言中的子程序。

实验内容及学时分配：

- (10') 子程序：回顾 Perl 语言中子程序的定义、使用等基本知识。
- (5') 程序调试：总结调试 Perl 程序的方法。
- (85') 实验操作：编写用于生物序列数据处理的子程序并进行调试。

主要仪器和实验材料：

- 主要仪器：一台安装有 Perl 语言（Linux 操作系统）的计算机。

实验重点、难点及解决策略：

- 重点难点：传递数据给子程序的方法；Perl 调试器的使用。
- 解决策略：通过演示进行学习，通过练习熟练掌握。

思考题：

- 比较传递数据给子程序的方法。
- 总结调试 Perl 程序的方法。

参考资料：

- Beginning Perl for Bioinformatics, James Tisdall, O'Reilly Media, 2001.
- Perl 语言入门（第六版），Randal L. Schwartz, brian d foy & Tom Phoenix 著，盛春 译，东南大学出版社，2012。
- Mastering Perl for Bioinformatics, James Tisdall, O'Reilly Media, 2003.
- 维基百科等网络资源。

一、 子程序（10分钟）

1. 定义：sub
2. 调用：&
3. 返回值：reture
4. 收集参数：@_
5. 作用域：my
6. 传递数据：值 vs. 引用

二、 程序调试（5分钟）

1. use warnings; 和 use strict;
2. 选择性注释
3. 添加 print 语句
4. Perl 调试器：p,n,s,v,c,b,B,w,R...

三、 实验操作（85分钟）

1. 编写子程序

```
#!/usr/bin/perl -w

$dna = 'CGACGTCTTCTCAGGCGA';
$longer_dna = addACGT($dna);
print "I added ACGT to $dna and got $longer_dna\n\n";

sub addACGT {
    my ($dna) = @_;
    $dna .= 'ACGT';
    return $dna;
}
```

2. 使用 my 限定作用域

```
#!/usr/bin/perl -w

$dna = 'AAAAA';
$result = A_to_T($dna);
print "I changed all the A's in $dna to T's and got $result\n\n";

sub A_to_T {
    my ($input) = @_;
    my $dna = $input;
    $dna =~ s/A/T/g;
    return $dna;
}
```

3. 传递数据给子程序

(1) 通过值传递

```
#!/usr/bin/perl -w
use strict;

my $i = 2;
simple_sub($i);
print "In main program, after the subroutine call, \$i equals $i\n

sub simple_sub {
    my($i) = @_;
    $i += 100;
    print "In subroutine simple_sub, \$i equals $i\n\n";
}
```

(2) 通过引用传递

```
#!/usr/bin/perl

use strict;
use warnings;

my @i = ('1', '2', '3');
my @j = ('a', 'b', 'c');
print "In main program before calling subroutine: i = " .  "@i\n";
print "In main program before calling subroutine: j = " .  "@j\n";
reference_sub(\@i, \@j);
print "In main program after calling subroutine: i = " .  "@i\n";
print "In main program after calling subroutine: j = " .  "@j\n";
exit;

sub reference_sub {
    my($i, $j) = @_;
    print "In subroutine : i = " . "@$i\n";
    print "In subroutine : j = " . "@$j\n";
    push(@$i, '4');
    shift(@$j);
}
```

4. 使用命令行参数

```
#!/usr/bin/perl -w

use strict;

my ($USAGE) = "$0 DNA\n\n";
unless (@ARGV) {
    print $USAGE;
    exit;
}
my ($dna) = $ARGV[0];
my ($num_of_Gs) = countG($dna);
print "\nThe DNA $dna has $num_of_Gs G\'s in it!\n\n";
exit;

sub countG {
    my ($dna) = @_;
    my ($count) = 0;
    $count = ( $dna =~ tr/Gg// );
    return $count;
}
```

5. 使用 Perl 调试器修复 Bugs（注意：程序中的 bug 不止一个）

```
#!/usr/bin/perl

my $dna = 'CGACGTCTTCTAAGGCGA';
my @dna;
my $receivingcommittment;
my $previousbase = '';
my $subsequence = '';

if (@ARGV) {
    my $subsequence = $ARGV[0];
}
else {
    $subsequence = 'TA';
}

my $base1 = substr( $subsequence, 0, 1 );
my $base2 = substr( $subsequence, 1, 1 );

@dna = split( '', $dna );

foreach (@dna) {
    if ($receivingcommittment) {
        print;
        next;
    }
    elsif ( $previousbase eq $base1 ) {
        if (/$base2/) {
            print $base1, $base2;
            $recievingcommitment = 1;
        }
    }
    $previousbase = $_;
}

print "\n";
exit;
```