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 7 生成随机 DNA 并进行分析

教师姓名：伊现富　　**职称：**讲师　　**教学日期：**2016 年 12 月 22 日 10:00-12:00

授课对象：生物医学工程与技术学院 2013 级生信班（本）　　**实验人数：**30

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：

- 了解伪代码和子程序的设计与编写。
- 熟悉自上而下和自下而上的程序设计理念。
- 掌握 Perl 语言中的嵌套循环。

实验内容及学时分配：

- (100）实验操作：编写 Perl 程序生成随机 DNA 序列并计算它们的相似性百分比。

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：嵌套循环。
- 解决策略：通过演示进行学习，通过练习熟练掌握。

思考题：

- 比较自上而下和自下而上的两种程序设计理念。
- 解释嵌套循环的逻辑步骤。

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、 实验操作（100分钟）

1. 生成一系列长短不一的随机DNA序列片段

```
#!/usr/bin/perl
use strict; use warnings;
my $maximum_length = 30; my $minimum_length = 15;
my $size_of_set = 12; my @random_DNA = ();
srand( time | $$ );
@random_DNA = make_random_DNA_set( $minimum_length, $maximum_length, $
print "Here is an array of $size_of_set randomly generated DNA sequenc
\n";
print "  with lengths between $minimum_length and $maximum_length:
\n\n";
foreach my $dna (@random_DNA) {
    print "$dna\n";
}
print "\n";

sub make_random_DNA_set {
    my ( $minimum_length, $maximum_length, $size_of_set ) = @_;
    my $length;
    my $dna;
    my @set;
    for ( my $i = 0 ; $i < $size_of_set ; ++$i ) {
        $length = randomlength( $minimum_length, $maximum_length );
        $dna = make_random_DNA($length);
        push( @set, $dna );
    }
    return @set;
}
sub randomlength {
    my ( $minlength, $maxlength ) = @_;
    return ( int( rand( $maxlength - $minlength + 1 ) ) + $minlength )
}
sub make_random_DNA {
    my ($length) = @_;
    my $dna;
    for ( my $i = 0 ; $i < $length ; ++$i ) {
        $dna .= randomnucleotide();
    }
    return $dna;
}
sub randomnucleotide {
    my (@nucleotides) = ( 'A', 'C', 'G', 'T' );
    return randomelement(@nucleotides);
}
sub randomelement {
    my (@array) = @_;
    return $array[ rand @array ];
}
```

2. 计算随机 DNA 的相似性百分比

```
#!/usr/bin/perl
use strict; use warnings;
my $percent; my @percentages; my $result; my @random_DNA = ();
srand( time | $$ );
@random_DNA = make_random_DNA_set( 10, 10, 10 );
for ( my $k = 0 ; $k < scalar @random_DNA - 1 ; ++$k ) {
    for ( my $i = ( $k + 1 ) ; $i < scalar @random_DNA ; ++$i ) {
        $percent = matching_percentage( $random_DNA[$k], $random_DNA[$
        push( @percentages, $percent );
} }
$result = 0;
foreach $percent (@percentages) { $result += $percent; }
$result = $result / scalar(@percentages); $result = int( $result * 100
print "In this run of the experiment, the average percentage of \n";
print "matching positions is $result%\n\n";
sub matching_percentage {
    my ( $string1, $string2 ) = @_;
    my ($length) = length($string1); my ($position); my ($count) = 0;
    for ( $position = 0 ; $position < $length ; ++$position ) {
        if ( substr( $string1, $position, 1 ) eq substr( $string2, $po
        { ++$count; }
    }
    return $count / $length;
}
sub make_random_DNA_set {
    my ( $minimum_length, $maximum_length, $size_of_set ) = @_;
    my $length; my $dna; my @set;
    for ( my $i = 0 ; $i < $size_of_set ; ++$i ) {
        $length = randomlength( $minimum_length, $maximum_length );
        $dna = make_random_DNA($length); push( @set, $dna );
    }
    return @set;
}
sub randomlength {
    my ( $minlength, $maxlength ) = @_;
    return ( int( rand( $maxlength - $minlength + 1 ) ) + $minlength )
}
sub make_random_DNA {
    my ($length) = @_; my $dna;
    for ( my $i = 0 ; $i < $length ; ++$i ) {
        $dna .= randomnucleotide(); }
    return $dna;
}
sub randomnucleotide {
    my (@nucleotides) = ( 'A', 'C', 'G', 'T' );
    return randomelement(@nucleotides);
}
sub randomelement { my (@array) = @_; return $array[ rand @array ]; }
```