

Introduction to Perl Programming
– Working on the WWW - Perl Modules

Beginning Perl, Chaps 12, 13

Previous Exercises

1. Write an extraction program to report
 - (a) the most significant library hit
 - (b) all significant library hitsthat result from a FASTA search. Several example search results are available at watson.achs:/r0/seqlib/bioch508/perl/fasta.result*
2. Write the same program for blast output. Sample results are in the same directory

FASTA output

FASTA searches a protein or DNA sequence data bank
version 3.4t11 Mar 4, 2002
Please cite:
W.R. Pearson & D.J. Lipman PNAS (1988) 85:2444-2448

1

gtt1_drome.aa: 209 aa
>GTT1_DROME GLUTATHIONE S-TRANSFERASE 1-1 (EC 2.5.1.18) (CLASS-THETA). - DROS
vs NBRF PIR1 Annotated Protein Database library
searching /slib2/blast/pir1.lseg library

	opt	E()	
< 20	9	0:=	
22	0	0:	one = represents 25 library sequences
24	1	0:=	
26	0	0:	

FASTA (3.45 Mar 2002) function [optimized, BL50 matrix (15:-5)] ktup: 2
join: 36, opt: 24, open/ext: -10/-2, width: 16
Scan time: 2.350

2

The best scores are:

		opt	bits	E(14548)
XUFF11	glutathione transferase (EC 2.5.1.18) 1 -	(210)	1399	333 4.7e-92
XUZM32	glutathione transferase (EC 2.5.1.18) III	(223)	173	48 2.2e-06
XUZM31	glutathione transferase (EC 2.5.1.18) III	(221)	164	46 9.3e-06
XUZM1	glutathione transferase (EC 2.5.1.18) I -	(214)	144	41 0.00023

3

>>XUFF11 glutathione transferase (EC 2.5.1.18) 1 - frui (210 aa)
initn: 1399 initl: 1399 opt: 1399 Z-score: 1759.9 bits: 332.6 E(): 4.7e-92
Smith-Waterman score: 1399; 100.000% identity (100.000% ungapped) in 209 aa overlap (1-209:1-209)

	10	20	30	40	50	60
GTT1_D	MVDFYILPGSSPCRSVIMTAKAVGVELNKKLLNLQAGEHLKPEFLKINPQHTIPTLVDNG					

FASTA -m9

The best scores are:

		opt	bits	E(14548)
XUFF11	glutathione transferase (EC 2.5.1.18) 1 -	(210)	1399	333 4.7e-92
XUZM32	glutathione transferase (EC 2.5.1.18) III	(223)	173	48 2.2e-06
XUZM31	glutathione transferase (EC 2.5.1.18) III	(221)	164	46 9.3e-06
XUZM1	glutathione transferase (EC 2.5.1.18) I -	(214)	144	41 0.00023
RGECSS	stringent starvation protein - Escherichi	(213)	140	40 0.00043

\/\t/

%_id	%_gid	sw	alen	an0	ax0	pn0	px0	an1	ax1	pn1	px1	gapq	gapl	f
1.000	1.000	1399	209	1	209	1	209	1	209	1	210	0	0	0
0.264	0.296	183	212	4	199	1	209	6	210	1	223	16	7	0
0.307	0.342	164	127	4	127	1	209	6	122	1	221	3	10	0
0.257	0.291	144	179	32	203	1	209	34	198	1	214	7	14	0
0.263	0.274	140	118	43	160	1	209	49	161	1	213	0	5	0

FASTA no result

Scan time: 2.167
!! No sequences with E() < 0.001000
200 residues in 1 query sequences

Parsing FASTA

1. get the query file name/description:

```
while ($line=<FILE>) {
    if ($line =~ m/^\s*/) {
        $desc = $line;
        ($file,$qlen) = $prev =~ /^(.+)\s+(\d+)\s+aa$/;
        last;}
    else {$prev = $line;}
}
```

2. get the high score:

```
while ($line=<FILE>) {
    if ($line =~ m/^The best scores are: / ||
        $line =~ m/^\!\! No/) {last; }
    $hit = <FILE>;
    ($bits,$eval)= (split(/\s+/, $hit))[-2,-1];
```

3. get the alignments

Blast output:

BLASTP 2.1.2 [Nov-13-2000]

Reference: Altschul, Stephen F., Thomas L. Madden, Alejandro A. Schaffer,
. . .

Query= GTT1_DROME GLUTATHIONE S-TRANSFERASE 1-1 (EC 2.5.1.18)
(CLASS-THETA). - DROS
(209 letters)

Database: /slib2/blastdb/pir1
14,548 sequences; 5,554,453 total letters

. . .
Sequences producing significant alignments: Score E
(bits) Value

XUFF11 glutathione transferase (EC 2.5.1.18) 1 - fruit fly (Dro...	436	e-123
XUZM32 glutathione transferase (EC 2.5.1.18) III (version 2) - ...	51	4e-07

>XUFF11 glutathione transferase (EC 2.5.1.18) 1 - fruit fly
(Drosophila melanogaster)
Length = 209

Score = 436 bits (1108), Expect = e-123
Identities = 209/209 (100%), Positives = 209/209 (100%)

Query: 1 MVDFFYYLPGSSPCRSVIMTAKAVGVELNKKLLNLQAGEHLKPEFLKINPQHTIPTLV DNG 60

Parsing BLAST

1. get the query file name/description:

```
while ($line=<FILE>) {  
    if ($line =~ m/Query=/) {  
        ($qdesc) = $line =~ /Query=\s*(\.+)  
    }  
    $/; last;  
}
```

2. get the high score:

```
while ($line=<FILE>) {  
    if ($line =~ m/^Sequences producing/ ||  
        $line =~ m/\*\*\*\* No hits found \*\*\*\*/)  
    {last; }  
    $hit = <FILE>;  
    ($bits,$eval)= (split(/\s+/, $hit))[-2,-1];
```

3. get the alignments

Subroutines in Perl

```
#!/usr/bin/perl -w  
use strict;  
my $name = getname();  
print "my name is: $name\n";  
exit; # not necessary, only a visual cue  
sub getname {  
    print "Enter your name: ";  
    my $line = <STDIN>;  
    chomp($line);  
    return $line;  
}
```

Passing Parameters

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

my $fname = getname('first');
my $lname = getname('last');
print "Your name is: $fname $lname\n";

sub getname {
    my ($type) = @_;
    print "Please enter your $type name: ";
    my $line = <STDIN>; chomp($line);
    return $line;
}
```

Parameter arrays are 1-D!

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

my @a = qw(a b c d);
my @b = 1..10;

print "Parameter Count : ", countparam('foo', @a, @b), "\n";
# prints 15, not 3!

sub countparam {
    return scalar @_;
}
```

Variable "scope"

```
my $var1 = 0; # global, available everywhere
sub mysub { # the open brace opens a new scope
    my $var2 = 1; # $var2 only available within 'mysub'
    local $var1 = 2; # make our own private copy of $var1,
                    # leaving $var1 unchanged outside of
                    # 'mysub'; useful with global vars
                    # that you want to temporarily change
} # end of this scope

{ local $var1 = 1; dostuff(); }
print "$var1\n"; # prints 0;

sub dostuff { print "$var1\n"; } # prints 1, not 0 nor 2.
```

Packaging useful subroutines

```
# MyMath.pm
package MyMath;
use strict;

sub cuberoot {
    my $y = shift @_;
    return exp(log($y)/3);
}

sub nthroot {
    my ($y, $n) = @_;
    return exp(log($y)/$n);
}

1;
```

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

use lib "."; # where to find it
use MyMath;

$cube = MyMath::cuberoot(10);
$nth = MyMath::nthroot(10, 5);
```

Modules - prepackaged methods

```
#!/usr/bin/perl
use strict;
use LWP::Simple 'get'; # import the 'get' subroutine

my $url = "http://www.weather.com/weather/local/22908";
my $html = get($url);
my ($temp) = $html =~ m/(\d+)\&deg\;F/m;
print "Current temperature: $temp F\n";
```

Modules and Object Oriented Programming

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

use LWP::UserAgent;
use HTTP::Common;

# $ua and $req are objects built with "new" method:
my $ua = new LWP::UserAgent;
my $req = new HTTP::Common
    POST => 'www.ncbi.nlm.nih.gov/genome/guide/gquery.cgi',
    [ db => 1, term => 'gtml_human' ];

# "request" is a method of $ua;
# it returns a "response" object
my $res = $ua->request($req);

# "content" is a method of $res:
my $html = $res->content();
# do stuff with $html
```

Exercises

1. Write program that takes a list of PubMed ID's (that you would get from Entrez) and downloads the references from PubMed into a file. Download the text in "Medline" format, removing <html> tags. (For information on downloading information from Entrez and Pubmed, see:

http://eutils.ncbi.nlm.nih.gov/entrez/query/static/eutils_help.html

2. 2. (extra credit) Convert the Medline format to "bib" format:

```
%A W. R. Pearson
%A J. T. Reese
%D 2002
%T Empirical estimates of gap penalties
%J Bioinformatics
%V 123
%P 45-51
%X 123456   <- PMID
```