CMSC423: Bioinformatic Algorithms, Databases and Tools
Lecture 4

Perl Modules

BioPerl

Biological databases

Writing modules

Also see: http://mathforum.org/~ken/perl_modules.html

```
package CMSC423::ParseFasta;
BEGIN {
  use strict;
  use Exporter;
  use vars qw(@EXPORT @EXPORT_OK @ISA %EXPORT_TAGS);

  @ISA = qw(Exporter);
  @EXPORT = qw();     # variables/functions to export
}

use vars @EXPORT;     # exported by default
use vars @EXPORT_OK;  # can be exported
```

Intermission ... FASTA files

```
>ATRNA01TF 3000 4000 3500 29 586
GTAANAAAGTGCTCTTGCGGAAGCCTTGAATGGTTCGCTGCTAAAGCTGCGAGCTGGCCA
TTGCAATGTTCTTAGAAAAACACGAACTTATCGGAGAGTGTCGTTACTGCGAGCTGTTGC
CGGTCGGTTTTCTCTGACAACTGCTGAAGCAGCTGCTTGATGTCGTCGAGGGTGGAGGTT
TAGTCGCCGGAACTCTGACCGTCGGTGTTGCTCATGGTGAATTGATCGTTGCTCTGAAGT
>ATCDA10TR 3000 4000 3500 22 487
TACCAGTCNCGAACCTATTAGAGGACTAGAGGGGGACCCTCCATTAAACTCATCCAAAGG
...
```

```
> ATRNA01TF 3000 4000 3500 29 586
GTAANAAAGTGCTCTTGCGGAAGCCTTGAATGGTTCGCTGCTAAAGCTGCGAGCTGGCCA
TTGCAATGTTCTTAGAAAAACACGAACTTATCGGAGAGTGTCGTTACTGCGAGCTGTTGC
CGGTCGGTTTTCTCTGACAACTGCTGAAGCAGCTGCTTGATGTCGTCGAGGGTGGAGGTT
TAGTCGCCGGAACTCTGACCGTCGGTGTTGCTCATGGTGAATTGATCGTTGCTCTGAAGT
> ATCDA10TR 3000 4000 3500 22 487
TACCAGTCNCGAACCTATTAGAGGACTAGAGGGGGACCCTCCATTAAACTCATCCAAAGG
...
```

Writing modules ... cont

```
sub new () # create a new instance of the "object"
{
  my $pkg = shift; # CMSC423::ParseFasta
  my $file = shift; # first parameter - file to read
  my $linesep = shift; # second parameter - end of line char

  my $self = {};   # point to current object
  bless $self;     # tell object self it belongs to package

  # record parameters into object
  $self->{file} = $file;
  $self->{linesep} = ''; # by default return one line
  $self->{linesep} = $linesep if defined $linesep;

  # create a buffer and read one line from file
  $self->{buffer} = <$file>;
  if (! defined $self->{buffer}) {
    print STDERR "File empty\n";
    return undef;
  }
```

Writing modules ... cont

```
  if ($self->{buffer} !~ /^>/){
    print STDERR "File doesn't start with a header\n";
    return undef;
  }

  chomp $self->{buffer};  # clean up end-of-line character

  return self;
} # new

sub GetRecord()
{
  my $self = shift;
  my $head; # header information
  my $data; # actual sequence

  if (! defined $self->{buffer} || $self->{buffer} !~ /^>/){
    return ();  # something's wrong... just bail
  }
```

Writing modules ... cont

```
  $head = $self->{buffer};# we make sure that we always leave
                         # header of the next object in buffer
  $head =~ s/^>//;       # get rid of initial >
  $self->{buffer} = <$self->{file}>; # read one more line
  chomp $self->{buffer};

  while (defined $self->{buffer} && $self->{buffer} !~ /^>/)
  { # while we read non-header lines
    $data .= $self->{buffer} . $self->{linesep}; # add to data
    $self->{buffer} = <$self->{file}>; # read another line
    chomp $self->{buffer} if (defined $self->{buffer});
  }
  return ($head, $data);
} # GetRecord

END {}

1;  # end of package
```

Using our module

```
#!/usr/bin/perl

open(IN, $ARGV[0]) || die ("cannot open $ARGV[0]: $!\n");

my $fastaReader = new AMOS::ParseFasta(\*IN);
# my $fastaReader = new AMOS::ParseFasta(\*IN, "\n");
# my $fastaReader = new AMOS::ParseFasta(\*IN, ",");

if (! defined $fastaReader){
  die ("Bad reader\n");
}

while (($head, $body) = $fr->GetRecord()){
  print "Header $head\n";
  print "Body $body\n";
}

exit(0);
```

WWW.BIOPERL.ORG

- A set of Perl modules ("objects") and functions for handling biological data

```
use Bio::Seq;
use Bio::SeqIO;

# create a sequence object of some DNA
my $seq = Bio::Seq->new(-id => 'testseq', -seq =>
'CATGTAGATAG');

# print out some details about it
print "seq is ", $seq->length, " bases long\n";
print "revcom seq is ", $seq->revcom->seq, "\n";

# write it to a file in Fasta format
my $out = Bio::SeqIO->new(-file => '>testseq.fsa', -format =>
'Fasta');
$out->write_seq($seq);
```

ParseFasta in Bio::Perl

```
use Bio::Seq;
use Bio::SeqIO;

$in  = Bio::SeqIO->new(-file => "inputfilename",
                       -format => 'Fasta');

while ( my $seq = $in->next_seq() ) {
  print "Header ", $seq->display_id(), "\n";
  print "Body ", $seq->seq(), "\n";
}
```

Biological databases

- General
  - GenBank - US
  - EMBL - Europe
- Specialized by data type
  - NCBI Trace Archive – raw sequencing data
  - SwissProt – curated protein information
  - KEGG – biological pathways
  - Gene Expression Omnibus – microarray data
- Specialized by organism
  - ZFIN – zebrafish
  - SGD – yeast
  - WormBase - worms

What data gets stored?

- DNA
  - string of letters
  - quality information, maybe chromatograms
  - location of genes (ranges along a chromosome)
- Proteins
  - string of letters
  - protein domains
  - 3D coordinates of each atom
- Pathways
  - graph of interactions between genes

For all – often store link to scientific articles related to data