Introduction to Perl Programming

Recitation

Week 9

Regular Expressions

- Regular expressions are a way of describing a PATTERN:
  - "all the words that begin with the letter A"
  - "every 10-digit phone number"
- We create regular expression to match the different parts of the pattern we're looking for
  - Ordinary characters match themselves
  - Meta-characters are special symbols that match a group of characters
    - for example \d matches any digit
- Bioinformatics programs often have to look for patterns in strings:
  - Find a DNA sequences containing only C's and G's
  - Look for a sequence that begins with ATG and ends with TAG

Meta Characters

`.`	match any single character
`[atcg]`	match any single a, t, c, or g
`[A-Z]`	match any character in given range
`[^atcg]`	match any character NOT in the set
`\`*`CHAR`*	takes away meta meaning of character *CHAR* `[\.|\*]` matches "." or "|" or "*"
`^` or `\A`	true at start of string
`$` or `\z`	true at end of string
`\b` `\B`	true at word boundary true when not at word boundary
`\d` `\D`	match any digit match any non-digit
`\n` `\t`	match newline character match tab character
`\s` `\S`	match any white space character match any non-whitespace character
`\w` `\W`	match any "word" character (alphanumeric plus "_") match any non-word character

Ways to Control Patterns

PATTERN1|PATTERN2	**matches either PATTERN1 or PATTERN2**
PATTERN*	**matches zero or more instances of pattern.** **[A-Z]* = any number of capital letters (including 0)**
PATTERN+	**matches one or more instances of pattern.** **[A-Z]+ = one or more capital letters**
PATTERN{N}	**matches exactly N instances of pattern** **[ATCG]{3} = one codon**
PATTERN{MIN,MAX} **PATTERN{MIN,}**	**matches at least MIN but not more than MAX times** **A[C]{2,4}G matches ACCG, ACCCG, or ACCCCG** **matches at least MIN times**
***?** **+?** **{MIN,MAX}?**	**matches 0 or more time, minimally** **matches 1 or more time, minimally** **matches MIN to MAX times, minimally**

Examples

```
# match if string $str contains 0 or more white space characters
$str =~ /^\s*$/;

# string $str contains all capital letters (at least one)
$str =~ /^[A-Z]+$/;

# string $str contains a capital letter followed by 0 or more digits
$str =~ /[A-Z]\d*/;

# number $n contains some digits before and after a decimal point
$n =~ /^\d+\.\d+$/;

# string contains A and B separated by any two characters
$s =~ /A..B/;

# string does NOT contains ATG
$s !~ /ATG/;
```

Examples

```
# match if string $str contains any sequence of three consecutive A's
$str =~ /AAA/;
$str =~ /A{3}/;

# match if string $str consist of exactly three A's
$str =~ /^AAA$/;
$str =~ /^A{3}$/;

# match if $str contains a codon for Alanine (GCA, GCT, GCC, GCG)
$str =~ /GC./;

# match if $str contains a STOP codon (TAA, TAG, TGA)
$str =~ /TA[AG]|TGA/;
$str =~ /T(AA|AG|GA)/;
$str =~ /T(A[AG]|GA)/;
```

Examples

```
# string contains any word containing all capital letters
$str =~ /\b[A-Z]+\b/;

# A followed by any number of C or G's followed by T or A
$str =~ /A[CG]*(T|A)/;
$str =~ /A[CG]{0,}[TA]/;

# TT followed by one or more CA's followed by anything except G
$str =~ /TT(CA)+[^G]/;

# string begins with B and has between 5 and 10 letters
$str =~ /^B.{4,9}$/;

# string consists of a 10 digit phone number: ddd-ddd-dddd
$str =~ /^\d\d\d\-\d\d\d\-\d\d\d\d$/;
$str =~ /^\d{3}\-\d{3}\-\d{4}$/;

# string consists of an IPv4 address: ddd.ddd.ddd.ddd
$str =~ /^((25[0-5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])\.){3}(25[0-
5]|2[0-4][0-9]|1[0-9][0-9]|[1-9]?[0-9])$/
```

Capturing Matches

- When we match a string with a regular expression, we may want to find out what matched
- Do this by surrounding the part of interest with ( )
- Then access special variables $1, $2, etc to get matches:

```
$str = "Perl is a programming language used for bioinformatics.";
$str =~ /(.*) is.*(b.*)\./;
$first = $1;
$second = $2;
print "$first $second\n";   # prints "Perl bioinformatics"

# or, you can capture the results in a list assignment:
($first, $second) = $str =~ /(.*) is.*(b.*)\./;
print "$first $second\n";   # prints "Perl bioinformatics"
```

Capturing Matches

- When we match a string with a regular expression, we may want to find out what matched
- Do this by surrounding the part of interest with ( )
- Then access special variables $1, $2, etc to get matches:

```
$str = "Perl is a programming language used for bioinformatics.";
$str =~ /(P.*l)/;
$word = $1;
print $word;   # prints "Perl is a programming l"
```

Capturing Matches

- If no string is given to the match operators, $_ is assumed

```
@A = ('ATGGCT','CCCCGGTAT','GCAGTGG');
for (@A) {
   ($first, $second) = /(.+)GG(.+)/;
   print "$first $second\n" if ($first and $second);
}

OUTPUT:
AT CT
CCCC TAT
```

Q. Why no output for third string?

Regular Expression (What Did We Match?)

```
#!/usr/bin/perl
use strict;
use warnings;

my $string = "Several rapidly developing RNA interference (RNAi)
methodologies hold the promise to selectively inhibit gene expression in
mammals. RNAi is an innate cellular process activated when a
double-stranded RNA (dsRNA) molecule of greater than 19 duplex
nucleotides enters the cell, causing the degradation of not only the
invading dsRNA molecule, but also single-stranded (ssRNAs) RNAs of
identical sequences, including endogenous mRNAs.";

# find all words containing "RNA"
while ( $string =~ /(\w*RNA\w*)/g ) {
    print "$1\n";
}
exit;

Output:
RNA
RNAi
RNAi
RNA
dsRNA
dsRNA
ssRNAs
RNAs
mRNAs
```

Regular Expression (What Did We Match?)

```
#!/usr/bin/perl
use strict;
use warnings;

my $string = "Several rapidly developing RNA interference (RNAi)
methodologies hold the promise to selectively inhibit gene
expression in mammals. RNAi is an innate cellular process
activated when a double-stranded RNA (dsRNA) molecule of greater
than 19 duplex nucleotides enters the cell, causing the
degradation of not only the invading dsRNA molecule, but also
single-stranded (ssRNAs) RNAs of identical sequences, including
endogenous mRNAs.";

# find all words containing "RNA"
while ( $string =~ /(\w+RNA\w+)/g ) {
    print "$1\n";
}
exit;

Output:
ssRNAs
mRNAs
```

Regular Expression (What Did We Match?)

```
#!/usr/bin/perl
use strict;
use warnings;

my $string = "Several rapidly developing RNA interference (RNAi)
methodologies hold the promise to selectively inhibit gene
expression in mammals. RNAi is an innate cellular process
activated when a double-stranded RNA (dsRNA) molecule of greater
than 19 duplex nucleotides enters the cell, causing the
degradation of not only the invading dsRNA molecule, but also
single-stranded (ssRNAs) RNAs of identical sequences, including
endogenous mRNAs.";

# find anything containing "RNA"
while ( $string =~ /(\S+RNA\S+)/g ) {
    print "$1\n";
}
exit;

Output:
(RNAi)
(dsRNA)
(ssRNAs)
mRNAs.
```

Regular Expression (Where Did the Match Occur?)

```
#!/usr/bin/perl
use strict;
use warnings;

my $string = "Several rapidly developing RNA interference (RNAi)
methodologies hold the promise to selectively inhibit gene expression in
mammals. RNAi is an innate cellular process activated when a
double-stranded RNA (dsRNA) molecule of greater than 19 duplex
nucleotides enters the cell, causing the degradation of not only the
invading dsRNA molecule, but also single-stranded (ssRNAs) RNAs of
identical sequences, including endogenous mRNAs.";

# find all words containing "RNA"
while ( $string =~ /(\S+RNA\S+)/g ) {
    print "$1 ends at position ", pos($string)-1, "\n";
}
exit;

Output:
(RNAi) ends at position 49
(dsRNA) ends at position 211
(ssRNAs) ends at position 374
mRNAs. ends at position 431
```

Some Useful URLs

- http://docs.python.org/library/re.html
- http://www.regular-expressions.info/
- http://www.regular-expressions.info/tutorial.html
- http://www.bjnet.edu.cn/tech/book/perl/
  - Nice tutorial regexp discussed on Day 7
- http://www.troubleshooters.com/codecorn/littperl/perlreg.htm