IT350 Web and Internet Programming

SlideSet #14: Perl Misceleanous

(see online references)

Outline

- Regular expressions
- Customized sort
- Files

Regular Expressions

- Example: /^ a (a|b|c)* b $/x
- Precedence:

Some Regular Expression Quantifiers and Metacharactes

Quantifier/Symbol	Matches
{n}	
{m,n}	
{n,}	
+	
*	
?	
^	
$	
\b	
\w	
\d	
\s	
\S	

lect_regex.pl

Regular Expressions and Matching Operator

```
… usual prelude here

my $search = "Now is is the time";
print p("Test string is '$search'");

if ($search =~ /Now/){
   print p('Search 1 success'); }

if ($search =~ /^Now/){
   print p('Search 2 success'); }

if ($search =~ /Now$/){
   print p('Search 3 success'); }

if ($search =~ /\b ( \w+ ow ) \b/x){
   print p("Search 4 success: $1"); }

if ($search =~ /\b ( \w+ ) \s ( \1 ) \b/x){
   print p("Search 5 success: $1 $2"); }

print end_html();
```

Search and Replace

- $string =~ s/regex/replacement/g ;

- Example (replace aa with bb):
- $string = "This string has aa here and aa here"
- $string =~ s/aa/bb/g;

Exercise #1

- Write the expression to replace one or more newline characters in a string with "&&".
- Make it work for both Unix (\n) and Windows (\r\n)

lect_sort.pl

Sort

```
… #usual prelude here
my @theList = (4, 1, 2, 6, 93, 2, 65, 100);
print p("Initial list @theList\n");
my @theSortedList = sort @theList;
print p("Sorted list @theSortedList\n");
my @theNumbersSortedList = sort {$a <=> $b} @theList;
print p("Sorted list as numbers @theNumbersSortedList\n");
my @theReversedSortedList = sort {$b <=> $a} @theList;
print p("Sorted list on reverse @theReversedSortedList\n");

# $a, $b params: return < 0 if a<b, 0 if a=b, >0 if a>b
sub compareReversed($$){
   my ($a, $b) = @_;
   return $b-$a;
}
my @theReversedSortedList2 = sort compareReversed @theList;
print p("Sorted list on reverse @theReversedSortedList2\n");
print end_html();
```

Exercise #2

my @students = ('student x in IT350', 'student x in IC312', 'student y in IT350', 'student z in IC312', 'student y in IC312');

my @sortedStudents = ??????????????????

print p join('
',@sortedStudents);

- Complete the code to print the array with content sorted by CourseID

File Updates

```
… #standard header stuff here
1.  $filename = "myFile.txt";
2.  open (FILE, $filename) or print("Could not open file
   $filename for read");
3.  my @fileLines = <FILE>; #reads entire file into array
4.  close (FILE);
5.  open (OUTFILE, "> $filename") or print("Could not open
   file $filename for write");
6.  #read each line and find the one we are looking for
7.  my $aLine;
8.  foreach $aLine (@fileLines){
9.      chomp ($aLine);
10.     if ($aLine =~ /something/){
11.             #either modify the line and write it to file
12.             #or just skip the line (to delete it from file)
13.     }
14.     else{ print OUTFILE $aLine."\n";}
15. }
16. close (OUTFILE);
```

Exercise #3

- Modify precedent code to update the file enrollment.txt so student x moves from IT350 to SI340

Original enrollment.txt

student x in IC312
student z in IC312
student y in IC312
student x in IT350
student y in IT350

Updated enrollment.txt

student x in IC312
student z in IC312
student y in IC312
student x in SI340
student y in IT350

Multiple Files

Multiple Perl Files:

```
require "useful_functions.pl";
```

- **Be sure not to use same names (e.g., function names) in different files!**
- **The file to include (ex. useful_functions.pl) needs 1; on the last line**
- **When invoking the external function, use & before the function name**
  - **Ex: &testFunction();**

Example of using multiple files

array_functions.pl

```
#print array as a ul
sub printArrayAsList{
	#get array
	my @arr = @_;
	print '<ul><li>';
	print join ('</li><li>', @arr);
	print '</li></ul>';
}
1;
```

array_functions_test.pl

```
#!/usr/bin/perl
use strict;
use CGI qw( :standard );
use CGI::Carp qw(warningsToBrowser
    fatalsToBrowser);
require 'array_functions.pl';

print header(); print start_html();

my @array = qw (banana, apple, pear);
print h2('Print array as list');

&printArrayAsList(@array);

print end_html();
```