Regular Expressions
Perl programming Examples

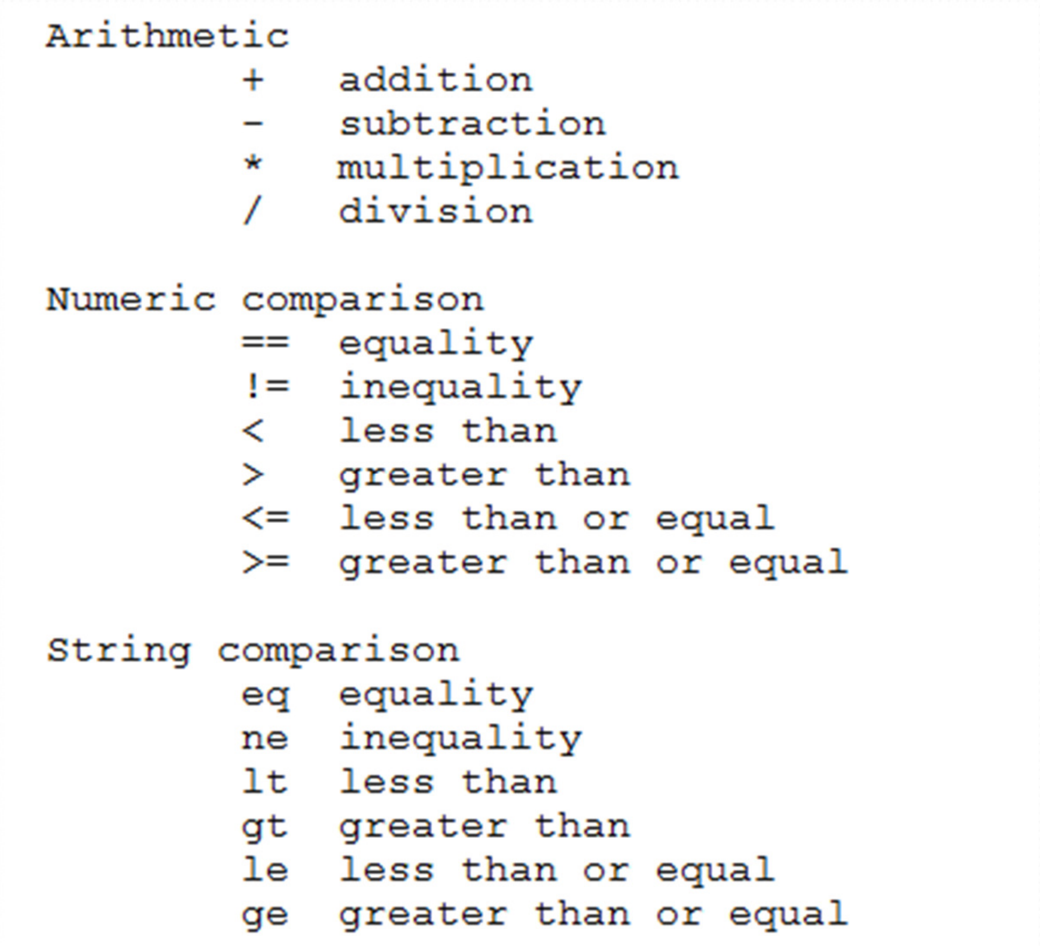
15-123

Systems Skills in C and Unix

Plan

- Perl programming basics
  - Operators
  - loops, arrays, conditionals
  - file processing
  - subroutines, references
  - Systems programming
  - Command line arguments
  - Perl intro
- Unix utilities
  - sed basics
- Regular expression grammar
  - Alternation, Grouping,Quantification
- Perl and RegEx
  - Examples

Operators

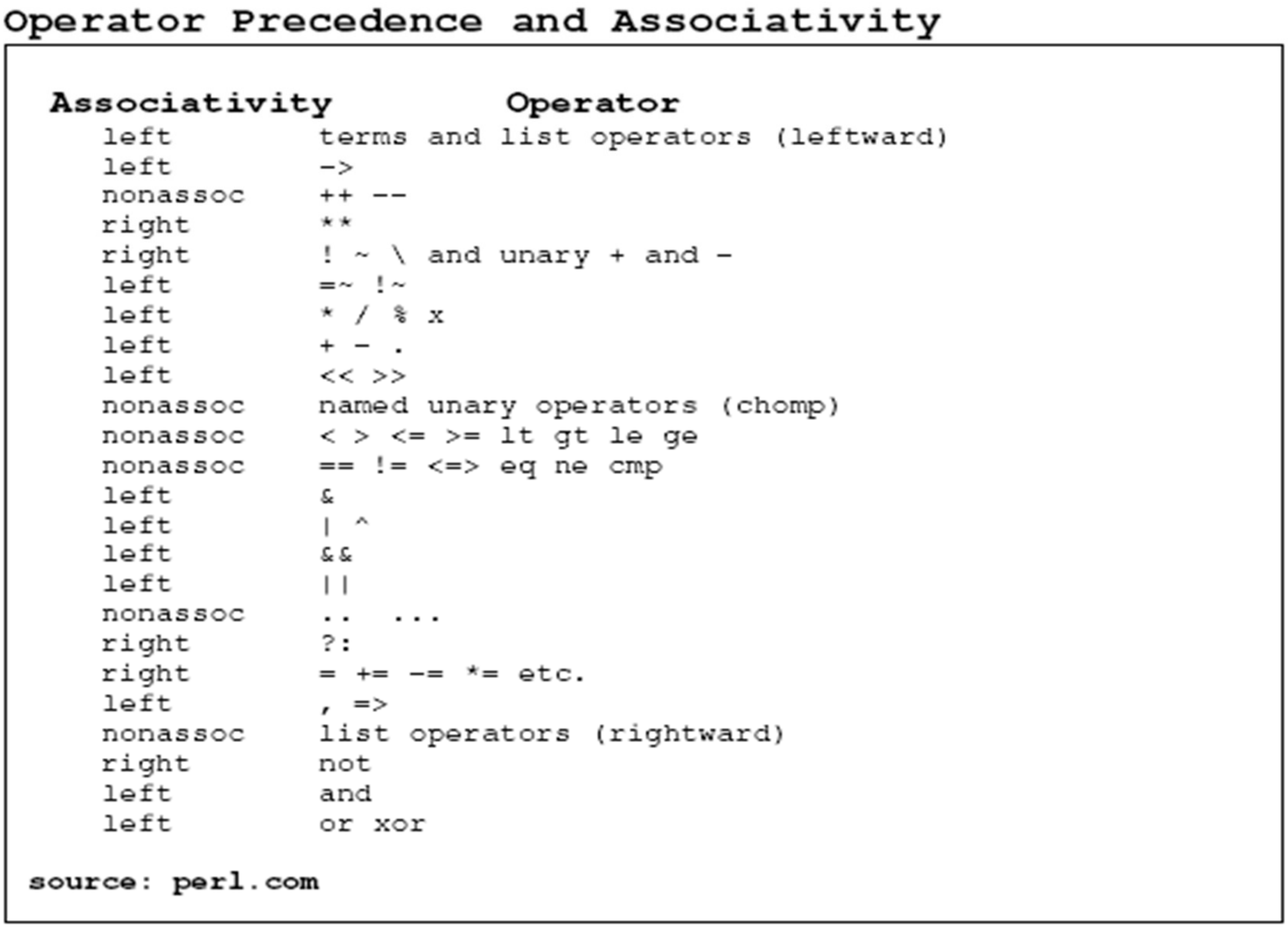


```
Arithmetic
        +    addition
        -    subtraction
        *    multiplication
        /    division

Numeric comparison
        ==   equality
        !=   inequality
        <    less than
        >    greater than
        <=   less than or equal
        >=   greater than or equal

String comparison
        eq   equality
        ne   inequality
        lt   less than
        gt   greater than
        le   less than or equal
        ge   greater than or equal
```

Source: unix man



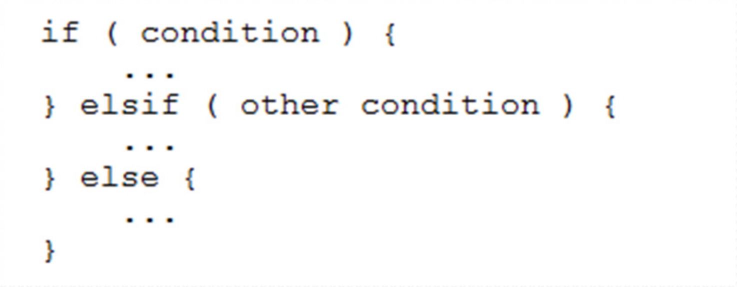
Operator Precedence and Associativity

Associativity	Operator
left	terms and list operators (leftward)
left	->
nonassoc	++ --
right	**
right	! ~ \ and unary + and -
left	=~ !~
left	* / % x
left	+ - .
left	<< >>
nonassoc	named unary operators (chomp)
nonassoc	< > <= >= lt gt le ge
nonassoc	== != <=> eq ne cmp
left	&
left	| ^
left	&&
left	||
nonassoc	.. ...
right	?:
right	= += -= *= etc.
left	, =>
nonassoc	list operators (rightward)
right	not
left	and
left	or xor

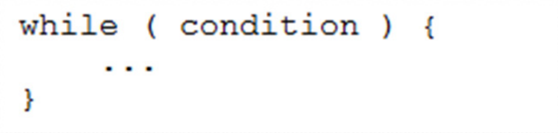
source: perl.com

More at: http://www.perl.com/doc/manual/html/pod/perlop.html

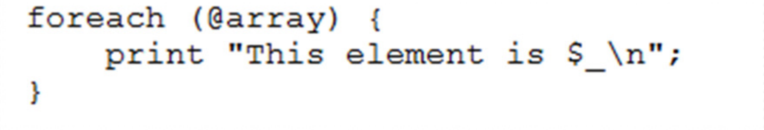
Loops, arrays, conditionals



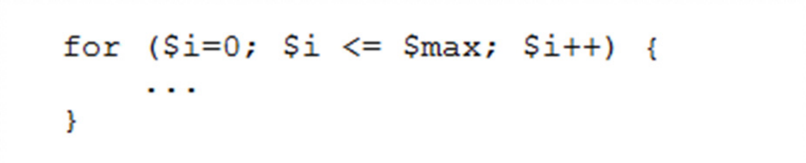
```
if ( condition ) {
    ...
} elsif ( other condition ) {
    ...
} else {
    ...
}
```



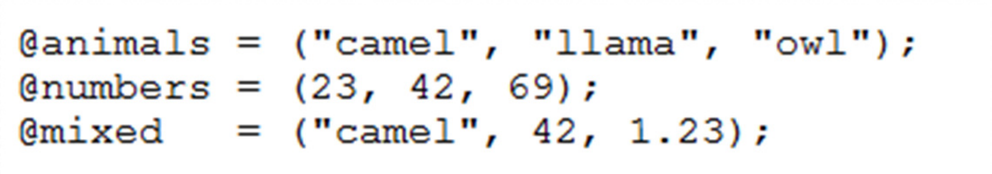
```
while ( condition ) {
    ...
}
```



```
foreach (@array) {
    print "This element is $_\n";
}
```



```
for ($i=0; $i <= $max; $i++) {
    ...
}
```



```
@animals = ("camel", "llama", "owl");
@numbers = (23, 42, 69);
@mixed   = ("camel", 42, 1.23);
```

Source: unix man

Perl IO

```
$size = 10;
open(INFILE, “file.txt”);
$#arr = $size-1; # initialize the size of the array to 10
$i = 0;
foreach $line (<INFILE>) {
   $arr[$i++] = $line;
   if ($i >= $size) {
       $#arr = 2*$#arr + 1; # double the size
       $size = $#arr + 1;
   }
}
```

Perl IO

- open(OUT, “>out.txt”);

- print OUT “hello there\n”;

- Better file open
  - open (OUT, “>out.txt”) || die “sorry out.txt could not be opened\n”

Subroutines

```
sub sum {
return $a + $b;
}
```

So we can call this as:

```
$a = 12; $b = 10;
$sum = sum();
print “the sum is $sum\n”;
```

Passing Arguments

Passing Arguments

A perl subroutine can be called with a list in parenthesis.

Example:

```
sub add {
  $tmp = 0; # this defines a global variable
  foreach $_ (@_) {
      $tmp += $_;
   }
  return $tmp;
}
```

Local variables

Local variables

Perl subroutines can define local private variables.

Example

```
sub product {
  my ($x); # defines the local variable x
  foreach $_ (@_) { $x *= $_;}
  return $x;
}
```

You can have a list of local variables by simply expanding the list as:

```
my ($x, $y, @arr);
```

Command line arguments

Command Line Arguments in Perl

- A Perl program can take command line arguments. One or

more command line

- arguments can be passed when calling a perl program.
  - **perl program.pl infile.txt outfile.txt**
- The number of command line arguments is given by **$#ARGV + 1 and** command line arguments are named

$ARGV[0], **$ARGV[1]**, etc

LWP
Library for www in Perl

- LWP contains a collection of Perl modules
  - ***use LWP::Simple;***
  - ***$_ = get($url);***
  - ***print $_;***
- *Good reference at*
  - *http://www.perl.com/pub/a/2002/08/20/perlandlwp.html*

Getopt

- The Getopt::Long module implements an extended getopt function called GetOptions().
- Command line arguments that are given as
  - **-n 20   or –num 20**
  - **-n 20 -t test**
- Can be extracted using getOptions
- Example
  - ***use Getopt::Long;***
  - ***$images_to_get = 20;***
  - ***$directory = ".";***
  - ***GetOptions("n=i" => \$images_to_get, "t=s" => \$directory);***
- *Note: any remaining arguments can be extracted by using $ARGV[0] etc..*

References: http://perldoc.perl.org/Getopt/Long.html

Hashes or associative arrays

```
%hash = ( );
# initializes a hash to empty set. We can add elements later
$hash{‘guna’} = “aa”;
$hash{‘neil’} = “ab”;
$hash{‘george’}=”ac”;
```

each function

- Each function allows us to extract both value and key from a hash table

<u>example</u>

```
%table = {“guna”, 10, “me”, 20};
while (($key, $value) = each(%table) ) {
      print "$key => $value\n";
}
```

References

Creating a Reference to a scalar

```
$num = 10;
$numref = \$num;
```

Creating a Reference to an array

```
@array = (guna, me, neil);
$arrayref = \@array;
```

Creating a Reference to a hashtable

```
%hash = {guna, aa, me, bb, him, cc};
$hashref = \%hash;
```

Dereferences

Dereferencing a Reference to a scalar

```
$num = 10;
$numref = \$num;
print $$numref; # prints the value 10
```

Dereferencing a Reference to an array using -> operator

```
@array = (guna, me, neil);
$arrayref = \@array;
print $arrayref->[0]; # prints ‘guna’
```

Dereferencing a Reference to an array using -> operator

```
@array= [guna, me, [blue, red]];
$arrayref = \@array;
print $arrayref->[2][1]; # prints ‘red’
```

Systems programming

```
opendir(DIR,".");
foreach $file (readdir(DIR)) {
   print "$file \n";
}
close(DIR);
```

Examples

```
print "which directory to change to : ";
chomp($dir = <STDIN>);
  if (chdir $dir){
      print "we are now in $dir \n";
  } else {
       print "we could not change to $dir \n";
  }
```

Modifying Permissions

```
foreach $file (“guna.c”, “temp.o”) {
  unless chmod (O666, $file) {
      warn “could not chmod the file $file \n”;
  }
}
```

Renaming a file

- **Rename($file1, $file2);**
- Exercise: Write a perl script that will take a folder as command line argument and rename all .txt files to .htm files

Copying a file

- **use File:Copy;**
- **copy($file1, $file2);**
- Exercise: Write a perl script that will create a duplicate folder given a folder. Name the new folder, dup_folder

Running a perl script from another

```
#!/usr/local/bin/perl
system 'perl mkdir.pl file.txt';
```

Encoding pages

- Example

```
PAGE:  for ($page=1;  ; $page++){
      $url = “http://www.cs.cmu.edu/”.$page;

      last PAGE if ($count == $maxcount );
}
```