Day 8: Standard Library

Suggested Reading:

http://perldoc.perl.org/index-functions.html

“Perl functions by category” + skim a few

Homework Review

Regular Expression Leftovers

Groups

(…) groups a match

```
/(in){2}/   dining, feminine
            in, ini, nine
```

matching groups are remembered in **$1**, **$2**, …

```
if ($phone =~ /(\d{3})-\d{3}-\d{4}/) {
  $area_code = $1;
}
```

can use captures in substitutions

```
$string =~ s/(\w+), (\w+)/$2, $1/;

'a big, great cat' => 'a great, big cat'
```

Alternatives

(…|…) matches one or the other

```
/d(og|im)/   dog, dim, dime
             dom, dig
```

( ) optional when | applies to whole pattern

```
/here|hear/   here, there, hear, heart, gatherer
              her, har, ...
```

gets messy…

```
if (/^((1[0-2])|([1-9])):([0-5]\d)$/) {
  print "hour: $1, minute: $4\n";
}
```

And Now…

(Some) Built-In Functions

Numeric Functions

int abs · sqrt exp log · sin cos atan2 · hex oct · srand rand

```
int(3.14159)    => 3
abs(-3)         => 3

sqrt(100)       => 10

hex('AF')       => 175
oct('257')      => 175

rand(10)        => 6.46860370253311
rand(10)        => 9.41780438684997
```

String Functions

chop chomp · lc lcfirst uc ucfirst · chr ord · tr///
length index rindex substr · sprintf · reverse

```
ord('A') => 65
chr(65)  => 'A'

index('start', 'tar') => 1
substr('stop', 1, 2)  => 'to'
substr('stop', -3)    => 'top'

sprintf('[%5.2f]', 3.14159) => [ 3.14]
```

Array & Hash Functions

push pop shift unshift · splice · split join · reverse
exists delete · keys values each

```
my @a = ( 1, 2, 3, 4, 5 );
my @b = splice(@a, 1, 3);
  # @a == ( 1, 5 )
  # @b == ( 2, 3, 4 )

splice(@a, 1, 0, 'a', 'b');
  # @a == ( 1, 'a', 'b', 5 )
```

3 Amazing Functions

sort()

Sort a list in ***string*** order

```
my @a = ( 1, 2, 3, 5, 7, 11, 13, 17 );
my @b = sort(@a);

=> ( 1, 11, 13, 17, 2, 3, 5, 7 )
```

Sort hash keys

```
foreach my $key (sort keys %hash) {
    print "$key => $hash{$key}\n";
}
```

Custom sort(): The Long Way

```
sub compare_numbers {
    if ($a < $b) { return -1; }
    if ($a > $b) { return 1; }
    return 0;
}

my @a = ( 10, 5, 22, 3, 1, 17 );
my @b = sort compare_numbers @a;

=> ( 1, 3, 5, 10, 17, 22 )
```

Custom sort(): The Perl Way

```
my @a = ( 10, 5, 22, 3, 1, 17 );
my @b = sort { $a <=> $b } @a;

=> ( 1, 3, 5, 10, 17, 22 )
```

grep()

```
grep BLOCK LIST
grep EXPR, LIST
```

```
my @list = ( 1, 2, 3, 10, 11, 12 );
my @filtered_1 = grep { $_ % 2 } @list;
my @filtered_2 = grep($_ % 2, @list);
```

```
open(INPUT, $filename) or die "hard $!\n";
my @comments = grep(/^\s*#/, <INPUT>);
close(INPUT);
```

```
my $hits = grep(exists $hash{$_}, @array);
```

map()

```
map BLOCK LIST
map EXPR, LIST
```

```
my @list = ( 1, 2, 3, 10, 11, 12 );
my @new_list = map($_ + 1, @list);

=> ( 2, 3, 4, 11, 12, 13 )
```

```
sub uniq {
    my %hash = map { $_ => 1 } @_;
    return keys %hash;
}
```

Back to Functions

Input/Output Functions

warn die · open getc <> (readline) read close
print printf · opendir readdir closedir

```
warn "Something bad happened!\n";

open(INPUT, $filename) or die "... $!\n";
while (my $character = getc(INPUT)) {
    # ...
}

printf('[%5.2f]', 3.14159);
```

File and Directory Functions

-X (-r, -w, …) stat lstat · glob · chdir mkdir rmdir
chmod chown · link symlink readlink · unlink rename

```
my ($dev, $ino, $mode, $nlink, $uid, $gid,
    $rdev, $size, $atime, $mtime, $ctime,
    $blksize, $blocks) = stat($filename);

chdir $new_directory;

if (-l $path) {
    my $real_path = readlink($path);
}
```

Modules

Modules

Perl ***Module*** = library of pre-written code

```
use Benchmark;
use POSIX qw/strftime floor ceil/;
```

CPAN : **C**omprehensive **P**erl **A**rchive **N**etwork

`www.cpan.org`

`search.cpan.org`

`perldoc.perl.org` — "Modules"

File::Basename

Pick apart file paths

```
use File::Basename;

my $path = '/usr/share/dict/words';
my $dir  = dirname($path);
my $name = basename($path);

$dir   =>  /usr/share/dict
$name  =>  words
```

File::Find

Recursive directory reader (like shell find)

```
use File::Find;
find(\&find_handler, @directories);

sub find_handler {
    return if /\.{1,2}/;  # $_ is filename
    if (-f $_) {
        print "In '$File::Find::dir', ";
        print "found '$File::Find::name'\n";
    }
}
```

POSIX 'strftime'

Convert dates to readable strings

```
use POSIX 'strftime';
strftime('%Y-%m-%d %H:%M:%S', localtime);

=> 2010-07-23 11:47:32
```

Last 2 Slides…

Other Scripting Languages

- All have standard libraries, functions, etc.
- Just a matter of learning what is available
- Find a good reference (book, website, …)
- Don't reinvent the wheel!

Homework

- Print interesting info about directories, recursively
- Use built-in functions and modules as appropriate
- Will involve a good data structure, too