

Perl 2

The long Perl handout has the full explanations -- these are just the short code snippets used in lecture

Variables

```
$x = 42;  
$y = $x + 13;  
$z = "hello";  
if (defined($z)) { ...
```

Strings

```
$str = $z . " twice " . $z;    ## concat  
$str = "x is $x\n";          ## interpolation  
("hello" eq $str)           ## string eq test  
lc(), uc()                   ## upper/lower case fns
```

Arrays

```
@a = (1, 2);  
$a[1]    ## == 2  
push(@a, 3);  ## add to RHS  
$x = shift(@a);  ## remove the 1
```

```
## iterate $elem over the array  
foreach $elem (@a) {...
```

Subroutines

```
sub() foo {
  my($x, $y) = @_;    ## parameters
  my($z);    ## local var
  return($x + $y);
}
```

```
...foo(6, 7) --> 13
```

@_

@_ is an array of the passed scalar values -- = parallel assigns the scalars to locals

my

my() declares something local

\$main::x

refer to outer "global" vars as \$main::x -- needed if the "use strict 'vars';" option is on.

File Processing

File Handle

Standards: STDIN, STDOUT, STDERR

Custom file handles are by convention all upper-case: FILE, FILE_IN, F

<FILE>

read one line from an input file handle, including the \n

```
print FILE $x, "hello", $y
```

write to an output file handle. The things to write are separated by commas, but the file-handle is not. File handle defaults to STDOUT

```
open(IN_FILE, "poem.txt");
```

Open for reading

```
open(OUT_FILE, ">poem2.txt");
```

Open for writing

```
## Standard file processing loop...
while (defined($line = <IN_FILE>)) {
  chomp($line);    ## remove EOLN
  print $OUT_FILE $line, "\n";
}
```

```
## || die -- error detection
open(F, $fname) ||
  die "Cannot open $fname";
```