

Day 2: Basic Syntax

Suggested reading:

Learning Perl (4th Ed.)

Chapter 1: Introduction

Chapter 2: Scalar Data

Turn In Homework

Housekeeping

- If you haven't enrolled:
 - please consider enrolling or auditing
 - you may attend regardless
 - I cannot provide help (homework, office hours, ...)
 - I can add you to the mailing list (email me)
- CSL accounts
 - old accounts may still be active
 - otherwise, see login screen on instructional machine
 - problems? stop by CS 2350 (the CSL)
or email **lab@cs.wisc.edu**

Office Hours

Wednesdays, 10–11 a.m.

Thursdays, 2–3 p.m.

Computer Sciences 4265

Still best to email first

**Write code.
At least a little.
Every day.
Have fun!**

Basic Perl Syntax

(in ~~12~~ 13 slides)

Hello World

```
#!/usr/bin/perl

use strict;
use warnings;

# Everyone's first Perl program
print "Hello, world!\n";
```

Numbers & Math

- literals: **42**, **3.141**, **-6.5e9**, **0377**, **0xff**
- operators: **+** **-** ***** **/** ****** **%** **()**

```
4 + 7           => 11  
17.8 - 3.5     => 14.3  
16 * 0x10      => 256  
2 ** 8         => 256  
10 / 3         => 3.333333...  
10 % 3         => 1  
(2 + 3) * 4    => 20
```


Strings

- literals: `'...'` and `"..."`
- escapes: `\n`, `\t`, `\x7f`, `\\`, `\"`
- operator: `.`

`'foo\tbar'` \Rightarrow `foo\tbar` [*literally*]

`"foo\tbar"` \Rightarrow `foo bar`

`'foo' . "\n"` \Rightarrow `foo\n` [*with newline*]

`'Don\'t let "quotes" confuse you!'`

`"Don't let \"quotes\" confuse you!"`



`Don't let "quotes" confuse you!`

Time to Write Code!

Simple °F \Rightarrow °C Conversion

Simple Variables

- prefix with `$`
- on first use, declare with `my`
- assignment: `=` [`+=` `-=` *etc.*]
- increment/decrement: `++` `--`

```
my $name = 'Tim';  
my $counter = 0;  
my $odd_value_1 = $counter + 7;  
$counter += 2;  
$name .= ' Cartwright';  
$counter++;
```

print() Revisited

- writes to standard output by default
- variable interpolation in "..."
- with or without ()

```
my $name = 'Tim';  
print $name;  
print "Hello, $name!\n";  
print($name . " is teaching\n");
```

Basic Input

- terminal input is a bit tricky...
- for today's homework, just use this:

```
chomp(my $user_input = <STDIN>);
```

More Coding!

Take input and use variables

Conditionals

- **if** (*bool*) {...} **elsif** (*bool*) {...} **else** {...}
- **unless**
- { and } are required (unlike C)

```
if (boolean expression) {  
    print "Condition is true!\n";  
} else {  
    print "Condition is false.\n";  
}
```

```
unless (boolean expression) {  
    print "Condition is false.\n";  
}
```

Comparisons

numeric	==	!=	<	>	<=	>=
string	eq	ne	lt	gt	le	ge

```

2 == 2           => true
2 != 2           => false
1 + 1 < 3        => true
2 <= 2.0         => true

'yes' ne 'no'    => true
'2' eq '2.0'     => false
'Tim' gt 'Nick'  => true [ha!]
'Tim' gt 'nick'  => false

```


Boolean-Like Values

- **undef**: "not defined"; like **null/nil**
- **false**: **0**, **' '**, empty array/hash, **undef**
- **true**: everything else

```
my $foo;
defined($foo)    => "" [i.e., empty string]
defined(undef)  => ""
defined(' ')    => 1
defined(0)      => 1

2 == 2          => 1
2 != 2          => ""
```

Conversions

- Perl converts values to fit the environment
 - operators require specific types
 - boolean interpretation

```
"6" * '5'      => 30
```

```
6 . 5          => '65'
```

```
'2' == '2.0'   => true
```

```
'2' eq '2.0'   => false
```

```
'2' eq 2.0     => true
```

Hey, Let's Code!

Add conditional conversions

Basic Loops

- **while** (*bool*) {...} **until** (*bool*) {...}
- **for** (*init*; *condition*; *change*) {...}
- **next**, **last**

```
my $counter = 10;
while ($counter > 0) {
    print "$counter...\n";
    $counter--;
}
```

```
my $sum = 0;
for (my $i = 1; $i <= 10; $i++) {
    $sum += $i;
}
```

Statement Modifiers

- Modify statement using condition or loop
 - **if, unless**
 - **while, until, foreach**
- () around condition optional
- Tim says: "Use **only** when clear and natural!"

```
$parameter = $MAX if $parameter > $MAX;  
die("Invalid data.") if $no_input_found;  
print($i-- . "\n") while $i > 0;
```

Operators & Precedence

→	()
—	++ --
←	**
←	! + - (<i>unary</i>)
→	* / % x
→	+ - . (<i>binary</i>)
→	<< >>
—	< <= > >= lt le gt ge
—	== != eq ne

→	&
→	^
→	&&
→	
←	? :
←	= += -= ...=
←	not
→	and
→	or xor

```
my $x = 6 * 3 - 2 & 0xF ? ' ' : 'b';
$x ||= 'c';
```

You Made It!

Other Scripting Languages

- The cellphone metaphor...
- Check for different or additional:
 - **Literals** (true/false, null/nil/None, 1_234, ""'")
 - **Operators** (===, =~, overloading)
 - **Conditionals** (elsif *vs.* else if)
 - **Loops** (syntax, object-oriented iterators)
 - **Block syntax** ({...} *vs.* do...end *vs.* indentation)
 - **Object syntax** (foo.bar, foo.bar())

Homework

- Simple number-guessing game
 - *You* pick the number & the *computer* guesses
 - Seek a straightforward solution
- **BE SURE TO LABEL YOUR PRINTOUT!!!**

```
#!/usr/bin/perl
```

```
# Homework for CS 368-1
```

```
# Assigned on Day 02, 2011-07-12
```

```
# Written by Your Name Here
```

```
use strict;  
use warnings;
```