

Introduction to Perl Programming

Beginning Perl, Chap 1

Getting started

1. unix shell prompt
2. learn an editor – emacs
3. run perl: “perl -v”, “perl -h”
4. logout: “exit”, “^D”

Unix commands

- `ls` – list files (`ls -l` long listing)
- `rm file` – remove (delete) a file
- `mv old_name new_name` – move (rename) a file
- `cp old new` – copy (rename) a file
- `mkdir` – make a directory (folder)
- `cd` – change to a directory (folder)
 - `cd bioch508` move down into bioch508
 - `cd ..` move up
 - `cd` move to your home directory
- `pwd` – print working (current) directory
- `less filename` – list contents of a file
- `emacs filename` – edit a file
- `^D` (`ctrl-D`) – logout

Tutorials:

www.ee.surrey.ac.uk/Teaching/Unix/

www.molecularevolution.org/eur/resources/computing.php

Unix case sensitivity

- Unix commands and file names are case sensitive.
- `Fasta35` is different from `fasta35` (the former does not exist)
- `Sequence.AA` is different from `sequence.aa`
- In general, always use lower case letters, numbers, and "." and "_" in Unix file names. Never use spaces inside a file name, and avoid special characters (\$, %, etc)

Unix – running programs

The Unix "shell" can run programs from many different locations. The locations that are searched when a command is typed are set by the \$PATH shell variable.

```
watson 21% echo $PATH
./seqprg/bin:/uva/bin:/usr/local/bin:/bin:/usr/bin
watson 22%
```

If you mis-type the name of a command, or if the command is not in your path, you will see the message:

```
watson 22% fasta36_t
fasta36_t: Command not found.
watson 23%
```

You can see where a command is coming from with the "which" command:

```
watson 48% which fasta35_t
/seqprg/bin/fasta35_t
watson 49%
```

The \$PATH variable is set during login initialization. Bioch508 students should have /seqprg/bin in their \$PATH

Unix – transferring files between computers

Because so much information (and so many sequence files) are available on the WWW, you will probably want to transfer files from your computer to watson.achs for analysis, and back from watson.achs to your computer for presentations.

The easiest way to transfer files back and forth between your computer and watson.achs is using the Home Directory service:

www.itc.virginia.edu/homedir/

You can use other commands – scp, sftp, secureFX (Windows) – to transfer files, but mounting your Home Directory (which is also your login directory on watson.achs) allows you to edit your unix files on your personal computer (but don't use MS Word – use emacs).

You can also transfer sequences from your computer to emacs using copy and paste.

Using emacs

```
sh>emacs
^x^c      exit
sh>emacs filename
type some stuff
^f,^b,^p,^n forward,back,prev, next
^x^s      save it
^x^c      exit
sh>
```

Intermediate emacs

```
sh>emacs random.pl
^s, ^r search forward, reverse
^a, ^e start, end of line
esc = M-
M-<, M-> start, end of buffer
M-% query-replace
^k kill-line (and put in kill buffer)
^k^k delete line and linefeed (EOL)
^y (yank – insert kill buffer)
^x 2, ^x 1, ^x o (multiple windows)
^u (repeat number)
^h (help, ^h-t tutorial, ^h-a apropos)
```

A Perl Template

```
#!/usr/bin/perl -w
use strict;      # required
use diagnostics; # optional

# your program begins here:
print "Goodbye, cruel world.\n";
```

Getting Help

```
Top-level help:
% perldoc perl
General help:
% perldoc perlfunc
Help with a specific function
% perldoc -f print
Search FAQ by keyword:
% perldoc -q sort
```

Running Perl

Running a script:

```
% perl myscript.pl
```

Perl "one-liners":

```
% perl -e 'print "Howdy\n";'
```

Spontaneous Perl:

```
% perl
```

```
Print "Here we are.\n";
```

```
<ctrl-D>
```

Executable scripts:

```
% chmod +x myscript.pl
```

```
% myscript.pl
```

Literals: strings and numbers

```
% perl -e 'print 2 + 2; print "\n";'
```

```
% perl -e 'print "abc"; print "def\n";'
```

```
# string "addition" (concatenation operator)
```

```
% perl -e 'print "one two" . " and three\n";'
```

```
# mixing numbers and strings:
```

```
% perl -e 'print (2 * 2) . "\n";'
```

```
% perl -e 'print "2 + 2 = " . (2 + 2) . "\n";'
```

```
# decimals and concatenations:
```

```
% perl -e 'print 2.3 + 2 . "\n";'
```

```
% perl -e 'print 2 . 3 + 2 . "\n";'
```

Exercises:

- Unix – login to `watson.achs.virginia.edu`
 - Create a subdirectory for `bioch508`
 - Go into that subdirectory
 - Transfer a protein sequence (FASTA format) file from your computer to that subdirectory
 - List the file on your screen
 - Make a copy of your sequence file so you have two files
- Emacs –
 - Open duplicate protein sequence file using emacs
 - Duplicate the middle two lines of the protein sequence
 - Save the file
 - Transfer the edited file back to your computer

Exercises (cont 1):

- Unix –
 - Run the `fasta35` program, using your original protein file as a query. Search the SwissProt database. Save your output to a third file.
 - Run the `fasta35` program from the command line, with the same query file and the same library – save the output to a second file
 - Run the `lalign35` program to compare your original protein sequence to the sequence with the duplicated region. Save the output to a third file.
 - Compare the results of your `lalign` analysis to the results at the FASTA WWW site (fasta.bioch.virginia.edu/fasta_www2)

Exercises (cont 2):

- Perl –
 - Write a program to print a random number
 - Print 10 random numbers between 1 and 20.
- Perl Challenges –
 - print the same random number 10 times.
 - write a Perl program to run the lalign35 alignment and save the output to a file
 - Modify the program to run the alignment using three different scoring matrices, saving the output to three different files