Unix Basics Quick Review and Problem Set

Table of Contents

Quick Review

command	description
`ls`	list directory contents
`cd`	change directory
`mkdir`	make a directory
`rm`	remove, or delete files and directories. Use caution, it is easy to delete more that you want.
`head`	prints the top few lines to the terminal window
`tail`	prints the last few lines to the terminal window
`sort`	sorts the lines
`uniq`	prints the unique lines
`grep`	filnds the lines that contain a pattern
`wc`	counts the number of lines, characters and words
`mv`	move files

`cp`	copy files
`date`	returns the current date and time
`pwd`	return working directory name
`ssh`	remote login
`scp`	remote secure copy
`~`	shortcut for your home directory
`man <command>`	manual page for the command e.g. `man ls` to get the man page for `ls`

Useful UNIX command examples to try

The files you need later in this review are in a folder on your Desktop!

Let's go to a directory with a lot of files in it and list those files

```
cd /bin/
ls
```

What's the difference between these two commands?

Try them both!!

```
ls -l
ls -lt
```

Pipes

You can string more than one command together with a pipe `|`, such that the standard output of the first command is 'piped' into the standard input of the second command.

Try it!!

```
ls -lt | head
```

Semicolons

You can string more than one command together by putting a semi-colon `;` after the each command. Here, the commands will be run sequentially, but any output does not get passed from one command to the next.

Try it!!

```
date ; sleep 2 ; date
```

> If you want to know more about `sleep` type `man sleep`

Redirect STDOUT

You can redirect the output of a command into a file

```
cd ~
grep Chr7 cuffdiff.txt > fav_chr_cuffdiff.txt
```

Append STDOUT to the end of a file that already exists

You can append the output of a command to a file

```
grep Chr9 cuffdiff.txt >> fav_chr_cuffdiff.txt
```

Redirect STDERR

You can redirect STDERR to a file.

Let's review what STDERR actually is.

```
cat blablabla.txt
```

> file blablabla.txt does not exist so we get `cat: blablabla.txt: No such file or directory` printed to the terminal. This message is labeled by the operating system as an error message or STDERR.

STDERR is a labeled type of output we can redirect

```
cat blablabla.txt 2> errors.txt
```

> We can redirect the error messages, A.K.A. STDERR, to a new file called anything we want

What happens when you try to redirect STDOUT?

```
cat blablabla.txt > errors.txt
```

> `cat: blablabla.txt: No such file or directory` still gets printed to the screen because we only redirect STDOUT to our file. There is no STDOUT in this case and our file will be empty. How would you verify this?

Redirect STDOUT and STDERR

You can redirect both STDOUT and STDERR to **two separate** files in one command.

```
# just print it to the terminal first
cat fav_chr_cuffdiff.txt blablabla.file

# redirect to two files, STDOUT to out.txt, STDERR to err.txt
cat fav_chr_cuffdiff.txt blablabla.file > out.txt 2> err.txt
```

> Examine the contents of `out.txt` and `err.txt`

You can also redirect both STDOUT and STDERR to **the same** file.

```
cat fav_chr_cuffdiff.txt blablabla.file > all_out_err.txt 2>&1
```

Problem Set

1. Log into your machine.
2. What is the full path to your home directory?
3. Go up one directory?
    - How many files does it contain?
    - How many directories?
4. Make a directory called `problemsets`.
5. Navigate into this new directory called problemsets. Verify that you are in the correct directory by using `pwd`.
6. Use `wget` to copy https://raw.githubusercontent.com/srobb1/pfb2017/master/files/sequences.nt.fa from the web into your problemsets directory. If `wget` is not available on your system, use `curl -O` as an alternative.
7. Without using a text editor calculate or report these qualities for the file `sequences.nt.fa`. This file can be found here https://raw.githubusercontent.com/srobb1/pfb2017/master/files/sequences.nt.fa
    - How many lines does this file contain?
    - How many characters? (Hint: check out the options of wc)
    - What is the first line of this file? (Hint: read the man page of head)
    - What are the last 3 lines? (Hint: read the man page of tail)
    - How many sequences are in the file? (Hint: use grep) (Note: The start of a sequence is indicated by a `>` character.)
8. Rename `sequences.nt.fa` to `cancer_genes.fasta`. (Hint: read the man page for mv)
9. Copy this remote file: https://raw.githubusercontent.com/srobb1/pfb2017/master/files/cuffdiff.txt to your problemset directory.
10. Do the following to `cuffdiff.txt`. The descriptions of each column in the file are in the table below.
    - Look at the first few lines of the file
    - Sort the file by log fold change 'log2(fold_change)', from highest to lowest, and save in a new file in your directory called sorted.cuffdiff.out
    - Sort the file (log fold change highest to lowest) then print out only the first 100 lines. Save in a file called `top100.sorted.cuffdiff.out`.
    - Sort the file by log fold change, print out the top 100, print only first column. This will be a list of the genes with the largest change in expression. Make sure your list is sorted by gene name and is unique. Save this curated list in a file called `differentially.expressed.genes.txt`.

Cuffdiff file format

Column number	Column name	Example	Description
1	Tested id	XLOC_000001	A unique identifier describing the transcipt, gene, primary transcript, or CDS being tested

2	Tested id	XLOC_000001	A unique identifier describing the transcipt, gene, primary transcript, or CDS being tested
3	gene	Lypla1	The gene_name(s) or gene_id(s) being tested
4	locus	chr1:4797771-4835363	Genomic coordinates for easy browsing to the genes or transcripts being tested.
5	sample 1	Liver	Label (or number if no labels provided) of the first sample being tested
6	sample 2	Brain	Label (or number if no labels provided) of the second sample being tested
7	Test status	NOTEST	Can be one of OK (test successful), NOTEST (not enough alignments for testing), LOWDATA (too complex or shallowly sequenced), HIDATA (too many fragments in locus), or FAIL, when an ill-conditioned covariance matrix or other numerical exception prevents testing.
8	FPKMx	8.01089	FPKM of the gene in sample x
9	FPKMy	8.551545	FPKM of the gene in sample y
10	log2(FPKMy/FPKMx)	0.06531	The (base 2) log of the fold change y/x
11	test stat	0.860902	The value of the test statistic used to compute significance of the observed change in FPKM
12	p value	0.389292	The uncorrected p-value of the test statistic
13	q value	0.985216	The FDR-adjusted p-value of the test statistic
14	significant	no	Can be either "yes" or "no", depending on whether p is greater then the FDR after Benjamini-Hochberg correction for multiple-testing