

# Unix Basics Quick Review and Problem Set

Table of Contents \* Quick Review \* Problem Set

## Quick Review

command	description
ls	list contents
cd	change directory
mkdir	make a directory
rm	use caution, it is easy to delete more than you would like
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	finds 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
~	represents your home directory
man [command]	manual page for the command ex. man ls

### How do these two commands compare?

Try it!!

```
1 ls -l
2 ls -lt
```

### Pipes

You can string more than one command together with a pipe (`|`), such that the output of the first command is received by the second command.

Try it!!

```
1 ls -lt | head
```

### Semicolons

You can string more than one command together with a semi-colon (`;`), such that the commands run sequentially, but that output does not get passed into the next command.

Try it!!

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

If you want to know more about `sleep` use `man sleep`

### Redirect STDOUT

You can redirect the output of a command into a file

Try it!!

```
1 grep Chr7 cuffdiff.txt > fav_chr_cuffdiff.txt
```

### Redirect STDOUT and Append

You can append the output of a command to a file

Try it!!

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

### Redirect STDERR

You can redirect STDERR to a file.

Let's review what STDERR actually is.

```
1 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

Try it!!

```
1 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`?

```
1 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

## Redirect STDOUT and STDERR

You can redirect both STDOUT and STDERR to **TWO SEPARATE** files in one command.

Try it!!

```
1 # just print it to the terminal first
2 cat fav_chr_cuffdiff.txt blablabla.file
3
4 # redirect to two files, STDOUT to out.txt, STDERR to err.txt
5 cat fav_chr_cuffdiff.txt blablabla.file > out.txt 2> err.txt
```

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

You can redirect both STDOUT and STDERR to **ONE SINGLE** file in one command.

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

## Problem Set

1. Log into your machine or account.
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 `sequences.nt.fa` from the web into your `problemsets` directory.
7. Without using a text editor calculate or report these qualities for the file `sequences.nt.fa`. This file can be found here
  - 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)
8. Rename `sequences.nt.fa` to `cancer_genes.fasta`. (Hint: read the man page for mv)
  9. Copy this remote file: `cuffdiff.txt` to your problemset directory.
  10. Do the following to `cuffdiff.txt` - the descriptions of each column in the file are 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, print only first column. Get a unique list of the genes, then print only the top 100. Save 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 transcript, gene, primary transcript, or CDS being tested
2	Tested id	XLOC_000001	A unique identifier describing the transcript, 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	$\log_2(\text{FPKMy}/\text{FPKMx})$	0.06531	The (base 2) log of the fold change y/x

Column number	Column name	Example	Description
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