More grep and sed with regular expressions

Combining grep with find

The find command returns all files in a folder, recursively. Options include -E for extended regular expressions

```
find /home/lewis -name "*.bam"
```

```
find ../../ -name gene1.txt
```

```
find . -name "*.txt" -exec grep  'Bat' {} +
```

#will take the filenames from find and grep them with a pattern. Here, I am assuming superheroes file ends with .txt and is in your directory or subdirectory.

sed

We often use regular expressions to match text then replace it with different text.
The text editor sed (Section 5.8 of the text) can replace one text string with another text string. This substitution happens on a line by line basis. sed stands for Stream Editor and it effectively allows us to do a search and replace on our data.

sed '<expression>' [path]
We are going to focus only on the s command which means substitute. The syntax is:
grep 's/pattern/replacement/'

The "/" are delimiters. A trailing 'g' replaces all instances of the pattern on a line. If we omit it then it will only replace the first instance of the pattern on each line. A trailing 'i' makes the search case insensitive.

```
echo 'hello' | sed 's/l/g/'
echo 'hello' | sed 's/l/g/g'
```

We can use cat to feed sed lines from a file. Let's say we want to replace chrom with chr.

Make a file temp.txt

```
chrom 1     xx yy
chrom 2     zz aa
```

```
cat temp.txt | sed 's/chrom/chr/'
```

Or we can use sed directly. It does not open the entire file into memory, so it is fast.

```
sed 's/chrom/chr/' temp.txt
```

What does the following do?

```
sed 's/chrom//' temp.txt
```

We can use -E, like in grep, to enable Extended regular expressions

```
echo 'hello' | sed 's/[a-z]/a/'  #-E can be used too
echo 'hello' | sed 's/[a-z]/a/g' #-E can be used too
echo 'hello' | sed 's/[a-z]+/a/'   #what does this do? Hint- add
a + to 'hello' and see what happens
echo 'hello' | sed -E 's/[a-z]+/a/'   #what does this do?
```

sed is often useful for replacing spaces with tabs and similar tasks

```
echo 'hello how are you' | sed -E 's/ +/\t/g'
```

(Or for my mac that doesn't recognize \t, control v tab at the same time to make a tab)

```
echo 'hello how are you' | sed -E 's/ +/	/g' | cat -vet
```

#gives "invisible" punctuation to see if it worked.

We can capture chunks of text and use them in the replacement. Here, we capture one chunk:

```
echo 'George, happily' | sed -E 's/^([a-zA-Z ]+)/\1 is your
name/'
```

Here, we capture three chunks.

```
echo "chr1:23432432-23433000" | sed -E 's/^(chr[^:]+):([0-9]+)-
([0-9]+)/\1	\2	3/'
```

(Again, my mac's sed does not recognize \t for tab in the replacement string, so I put in "literal tabs" in the terminal using ctrl-v-tab.)

```
chr1	23432432	23433000
```

Aside: if you don't mind spaces, a more elegant way of doing this is:

```
echo 'chr1:200-400'|sed 's/[:-]/ /g'
echo 'chr1:200-400'|tr ':-' ' '
```

A more complicated example of replacing text: Transcripts have the following entry in a GFF file. Can you output the names of their genes? For example, the format is:
ID=transcript:AT1G01010

```
1	ensembl	gene 3631 5899 .	+	.
	ID=gene:AT1G01010;Name=NAC001;biotype=protein_coding;descri
ption=NAC domain-containing protein 1 [Source:UniProtKB/Swiss-
Prot%3BAcc:Q0WV96];gene_id=AT1G01010;logic_name=tair;version=1
```

Let's try:

```
grep 'ID=transcript:' Arabidopsis_thaliana.TAIR10.28.gff3 | cut
-f9 | sed -E 's/\w+=\w+:(AT[0-9]G[0-9]+).+/\1/' | head
```

By default, sed prints out everything. To print only matching lines, we suppress this feature using -n option, then we give the p flag at the end of the sed expression.

```
grep 'ID=transcript:' Arabidopsis_thaliana.TAIR10.28.gff3 | cut
-f9 | sed -nE 's/\w+=\w+:(AT[0-9]G[0-9]+).+/\1/p' | sort -u
```