In []:
```
pwd
cd shell
pwd
```

Should yield something like

```
/Users/username/2014-04-23-uib
```

In []:
```
ls
cd novice
pwd
```

Looking into a folder

In []:
```
ls -F shell
cd shell/filesystem
pwd
ls
cd users
```

Should give something like

```
/Users/username/2014-04-23-uib/novice/shell/filesystem/users
```

Exercise
Look around using ls only

In []:
```
cd vlad
pwd
cd ..
pwd
cd # go home
cd - # go back to folder before last cd
```

Exercise
`cd` in and out of folders in 'filesystem'

In []:
```
cd /Users/username/2014-04-23-uib/novice/shell/filesystem
/users/uib/data
```

In [12]:
```
ls
```

```
1952.txt  1962.txt  1972.txt  1982.txt  1992.txt  2002.tx
t
1957.txt  1967.txt  1977.txt  1987.txt  1997.txt  2007.tx
t
```

In []:
```
cat 2007.txt
less 2007.txt
```

In []:
```
wc -l 2007.txt
```

Is Norway a part?

In []:
```
grep Norway 2007.txt
```

Is Norway in all files?

In []:
```
grep Norway *.txt
```

Redirection: save output to a new file

In []:
```
grep Norway *.txt >Norway.txt
less Norway.txt
rm Norway.txt
```

Which continents, how many countries?

In []:
```
cut -f 4 2007.txt > continents.txt # <-- ???
```

Pipes!

In []:
```
cut -f 4 2007.txt |less # <-- !!!
cut -f 4 2007.txt |sort |less
cut -f 4 2007.txt |sort |uniq
cut -f 4 2007.txt |sort |uniq -c
```

Exercise:

- check for another file or two whether they show the same numbers

Sorting by population

In []:
```
sort -k 3 2007.txt |less

man sort

sort -n -k 3 2007.txt |less
sort -nr -k 3 2007.txt |less
```

Exercise:

- in 2007, which two countries have the highest life expectancy
- which two the lowest

In []:
```
sort -nr -k 5 2007.txt |less
sort -nr -k 5 2007.txt |head
sort -nr -k 5 2007.txt |head -2
sort -nr -k 5 2007.txt |tail -2
# Oh no
sort -nr -k 5 2007.txt |tail -3
```

Exercise:

- in 2007, which 1 country had the highest GPD
- which the lowest
- what about other years?

In []:
```
sort -nr -k 6 2007.txt |head -1
sort -nr -k 6 2007.txt |tail -2
sort -nr -k 6 2007.txt |tail -2 |head -1
```

'cut' command can also be used to display more than one column

In []:
```
sort -nr -k 6 2007.txt |head -1 |cut -f 1,6
```

Avoid all this typing and changing the year.
shell script!

In [ ]:
```
cd ..
mkdir scripts
cd scripts
touch highest_GDP.sh
```

Use 'history' to retrieve the command we used for the sorting
Replace '2007' with '$1'

In [ ]:
```
nano highest_GDP.sh

# type the following:
sort -nr -k 6 $1 |head -1 |cut -f 1,2,6
```

Now we run it

In [ ]:
```
cd ../data/
source ../scripts/highest_GDP.sh 2007.txt
source ../scripts/highest_GDP.sh 1952.txt
```

Exercise

- try this out on a bunch of years
- make another script that does the same for the life expectancy

In [ ]:
```
#../scripts/highest_lifeExp.sh
sort -nr -k 5 $1 |head -1 |cut -f 2,1,5
```

Now we want to automate
--> Loops!

In [ ]:
```
for f in *.txt
do echo $f
done
# OR
for f in *.txt; do echo $f; done
```

Putting it together

In [ ]:
```
for f in *.txt
do source ../scripts/highest_GDP.sh $f
done
```

Now we make a master script

In [ ]:
```
touch ../scripts/GDP_all.sh
nano ../scripts/GDP_all.sh

# enter

for f in *.txt
do source ../scripts/highest_GDP.sh $f
done
```

Add a header

In [ ]:
```
#GDP_all.sh
echo Country Year GDP
for f in *.txt
do source ../scripts/highest_GDP.sh $f
done
```

Finally, document your code (add comments)