

Summing up Unix, Python, etc.

UNIXBasicsA1.pdf - Adobe Acrobat Pro

File Edit View Window Help

Create [Icons] Customize [Icons]

1 / 15 79,2% Tools Sign Comment

jonkl@medisin.uio.no

Unix basics exercise – MBV-INF410

In order to start this exercise, you need to be logged in on a UNIX computer with a terminal window open on your computer. It is best if you are logged in on freebee.abel.uio.no. If you do not know how to open a terminal window, like the one below, on your desktop, please contact one of the instructors and get help.

In the terminal window you have a Unix shell open with a command-line (a shell prompt) where you can type your commands. The command-line interpreter will execute your commands. There are several alternative and widely used Unix shells, for example C shell (csh), Bourne shell (sh), tcsh (used to be default at UiO), and Bourne-again shell (bash) (now default at UiO). In this exercise we will use bash. Hopefully you already have a bash shell running. If you are uncertain, type "bash" at the command prompt and press enter. Now you have opened a bash shell inside the other shell, and you are ready to start.

1. Your first command will be *pwd* (short for *print working directory*). It will tell you exactly where in the file system hierarchy you are, i.e. the full pathname of the directory you are in. Type *pwd* and press enter.

The command-line interpreter have interpreted your command. It knows that *pwd* is a program and runs that program. The output from the program was sent to your

1

Very important to be able to use
basic Unix

Make your own Unix scripts is
not very central

Most likely you will instead use
Python for writing small
programs/scripts

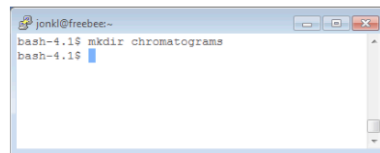
jonkl@medisin.uio.no

A brief practical Unix/Python exercise – MBV-INF410

We will in this exercise work with a practical task that, it turns out, can easily be solved by using basic Unix and Python.

Let us pretend that an engineer in your group has spent several weeks growing and Sanger sequencing several hundred clones. All the successful sequencing data has been “packed” in one single file called `allChroms.tar.gz`.

1. Log onto `freebee.abel.uio.no` and create a new directory in your home area. Let us call it “chromatograms”.



```
jonkl@freebee:~$  
bash-4.1$ mkdir chromatograms  
bash-4.1$
```

2. Download the file `allChroms.tar.gz` from the wiki page and put it in the new directory. How you do this will depend on your laptop. When you have done this, make sure you understand what you did. We will do similar operations more times during the course (*and very likely for the exam...*). *This is important!*
3. The file has a double ending, “.tar.gz”. This indicates that this is a compressed file that has been compressed, or packed to save space, by the `gzip` software application (hence the “.gz”). It is also a “tar file”, also known as a “tarball”, which usually means that many files have been packed into a single file. This is often done to make file transfer and/or file storage easier. Use `ls -l allChroms.tar.gz` to see the size of the

Main points:

- Demonstrate a practical task where making a script is absolutely necessary
- Show you that some tasks *cannot* easily be solved by using «databases and web servers» - including Galaxy/LifePortal

A brief practical Unix/Python exercise – MBV-INF410

We will in this exercise work with a practical task that, it turns out, can easily be solved by using basic Unix and Python.

Let us pretend that an engineer in your group has spent several weeks growing and Sanger sequencing several hundred clones. All the successful sequencing data has been “packed” in one single file called `allChroms.tar.gz`.

1. Log onto `freebee.abel.uio.no` and create a new directory in your home area. Let us call it “chromatograms”.

- After this exercise you *must* be able to:
 - In a Unix shell, create directories, move around between the directories and copy and move files. Use `ls` with options to see which files and directories you have in a directory.
 - Use `cat` and `more` to view the contents of a file, and use `nano` to write text into a file or create a new one.
 - Delete files
 - Transfer files from your laptop to `freebee.abel.uio.no` and back
 - Download a compressed tarball file from somewhere and store it on your laptop and on `freebee.abel.uio.no`
 - Uncompress and extract the files from the tarball
 - Make a tarball archive file with `tar` and compress with `gzip`
 - Make programs executable and files visible or hidden from other users with `chmod`
 - Know how to use `grep`, “pipe” (`|`), and how to redirect output with “`>`”
 - If you obtain a little Python script, you should be able to run it and, at least if it is not too complicated, be able to figure out what it is doing

This is
most
important!

Python

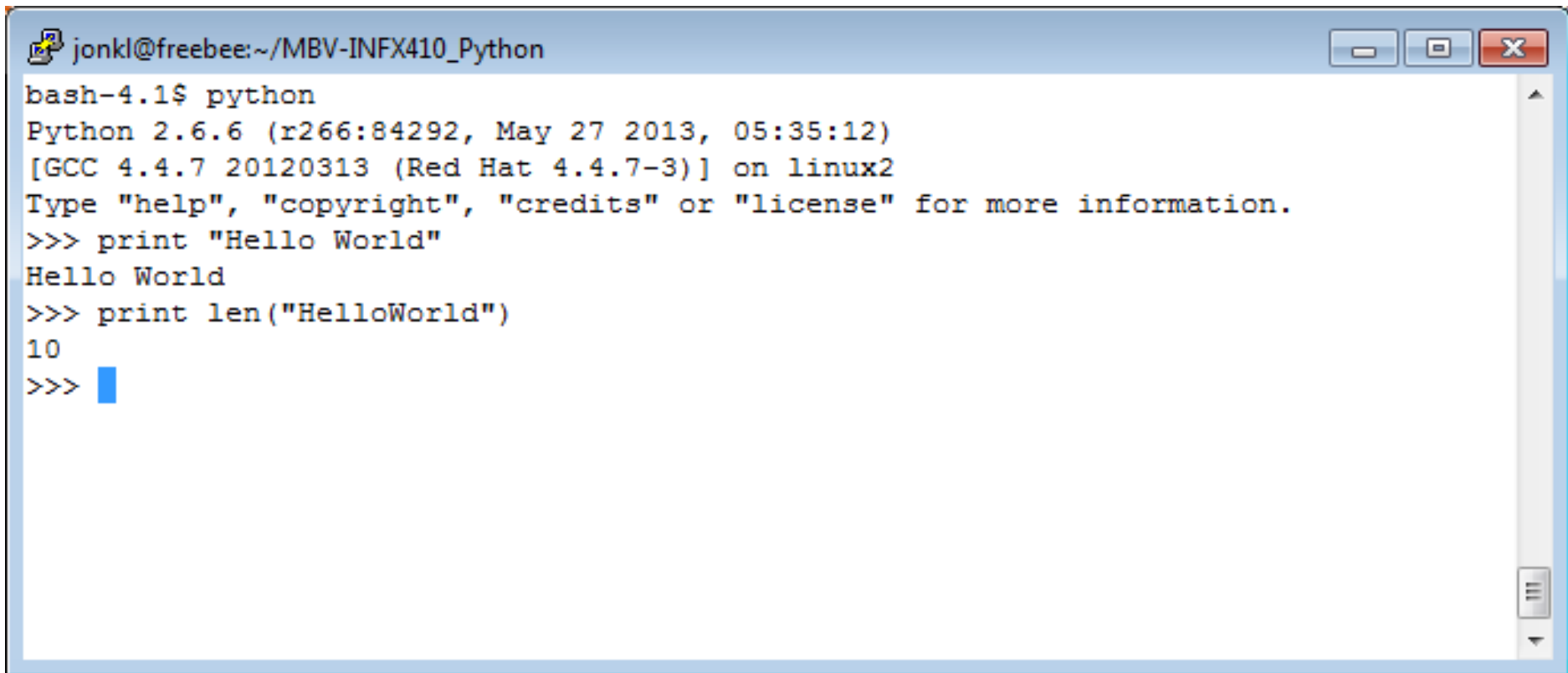
- Steep (or not?) learning curve
- Day 1
- Day 2
- Classes, objects and Biopython
- You are not expected to become experienced programmers
- The exercise on MSAs and “the Python script for moving files” illustrates what I expect you to be able to do with Python

More Unix stuff

- You ***must*** be able to move files between laptop and freebee.abel.uio.no!
- You ***must*** be able to uncompress and get all files out of a gzipped tarball, for example bigfile.tar.gz
 - And make this kind of archive file yourself

More Unix stuff

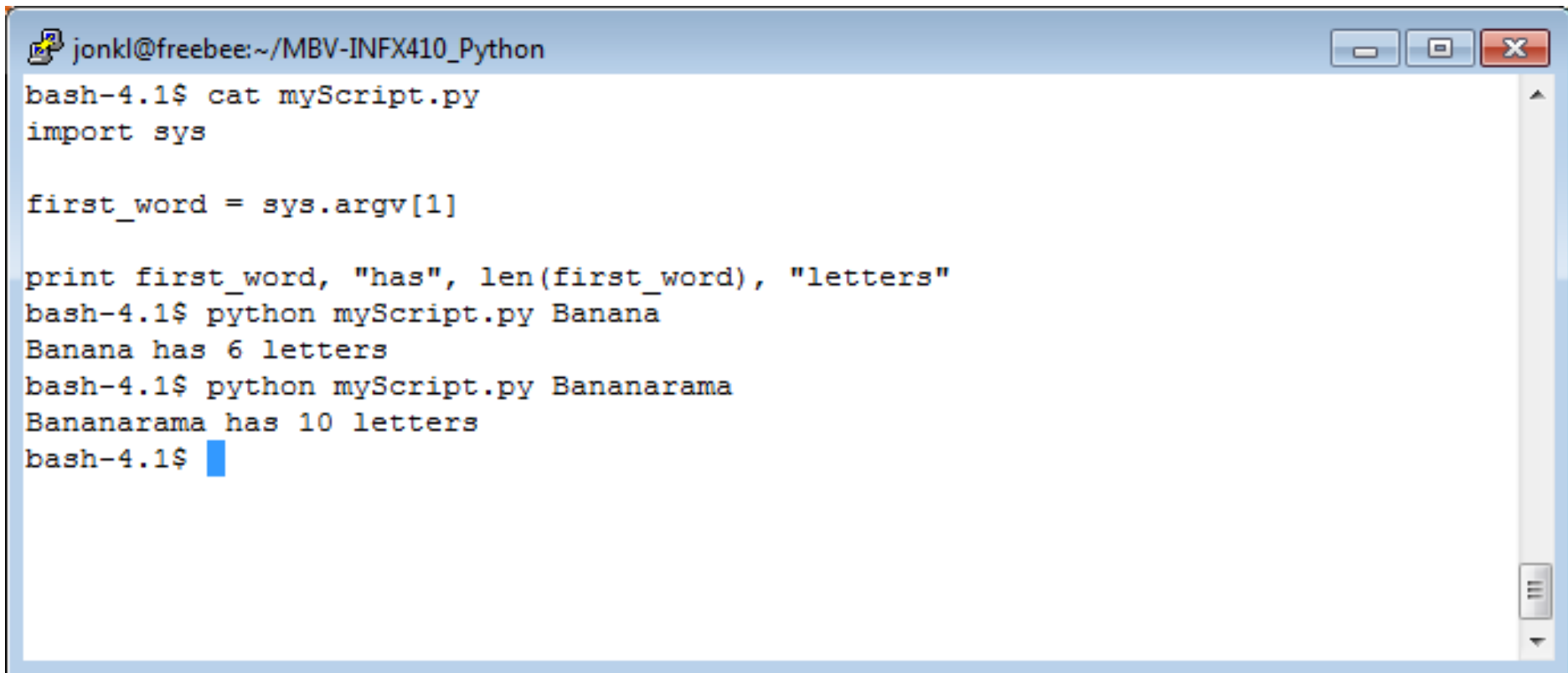
- Three ways to run a Python script
- 1: In an interactive shell

A screenshot of a terminal window titled 'jonkl@freebee:~/MBV-INF410_Python'. The terminal shows a bash prompt 'bash-4.1\$' followed by the command 'python'. The output displays the Python version 'Python 2.6.6 (r266:84292, May 27 2013, 05:35:12)' and the compiler '[GCC 4.4.7 20120313 (Red Hat 4.4.7-3)] on linux2'. It then prompts the user to type 'help', 'copyright', 'credits', or 'license' for more information. The user enters '>>> print "Hello World"', and the output is 'Hello World'. The user then enters '>>> print len("HelloWorld")', and the output is '10'. The prompt '>>>' is followed by a blue cursor.

```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ python
Python 2.6.6 (r266:84292, May 27 2013, 05:35:12)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-3)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print "Hello World"
Hello World
>>> print len("HelloWorld")
10
>>> 
```

More Unix stuff

- Three ways to run a Python script
- 2: As a script sent as input to Python



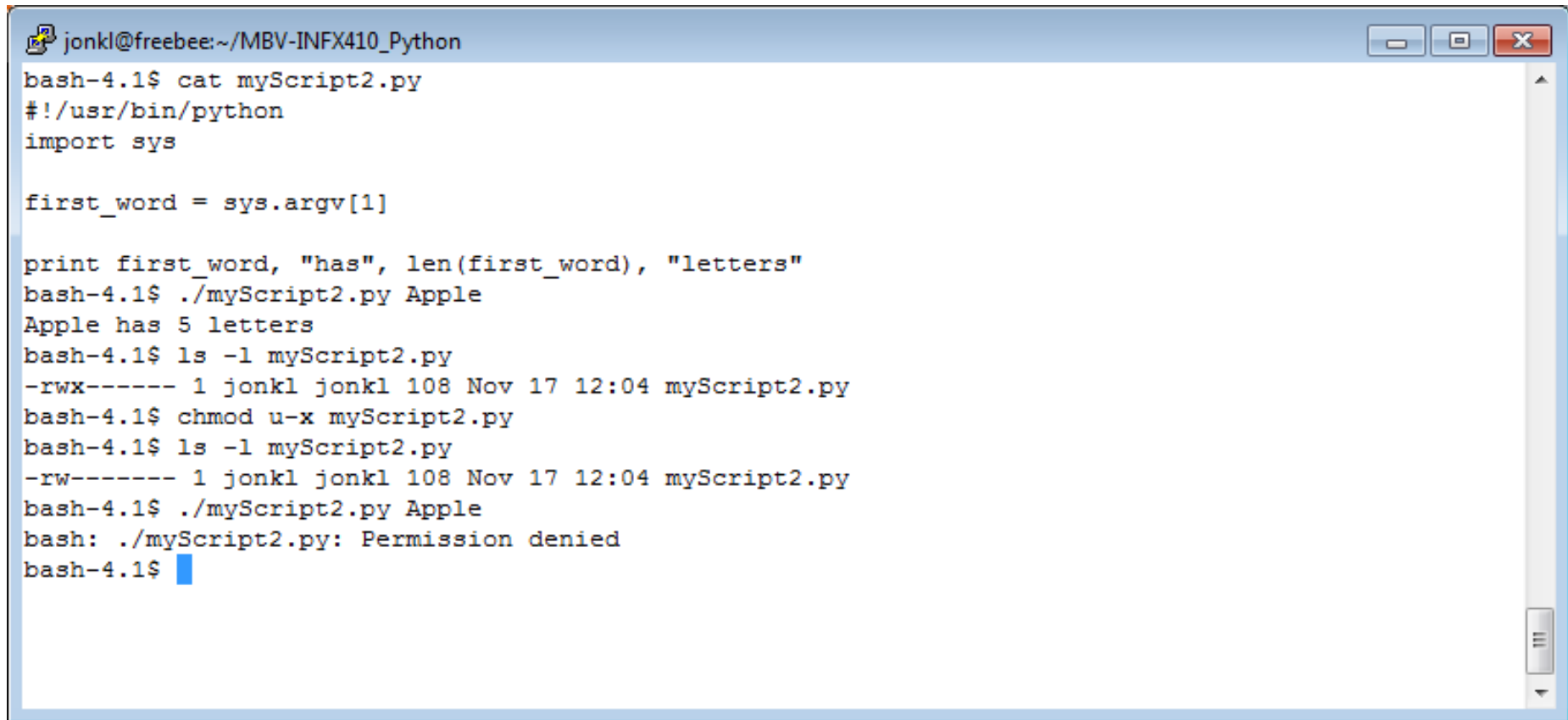
```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ cat myScript.py
import sys

first_word = sys.argv[1]

print first_word, "has", len(first_word), "letters"
bash-4.1$ python myScript.py Banana
Banana has 6 letters
bash-4.1$ python myScript.py Bananarama
Bananarama has 10 letters
bash-4.1$
```


More Unix stuff

- Three ways to run a Python script
- 3: As a script with `#!` at first line



```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ cat myScript2.py
#!/usr/bin/python
import sys

first_word = sys.argv[1]

print first_word, "has", len(first_word), "letters"
bash-4.1$ ./myScript2.py Apple
Apple has 5 letters
bash-4.1$ ls -l myScript2.py
-rwx----- 1 jonkl jonkl 108 Nov 17 12:04 myScript2.py
bash-4.1$ chmod u-x myScript2.py
bash-4.1$ ls -l myScript2.py
-rw----- 1 jonkl jonkl 108 Nov 17 12:04 myScript2.py
bash-4.1$ ./myScript2.py Apple
bash: ./myScript2.py: Permission denied
bash-4.1$
```

More Unix stuff

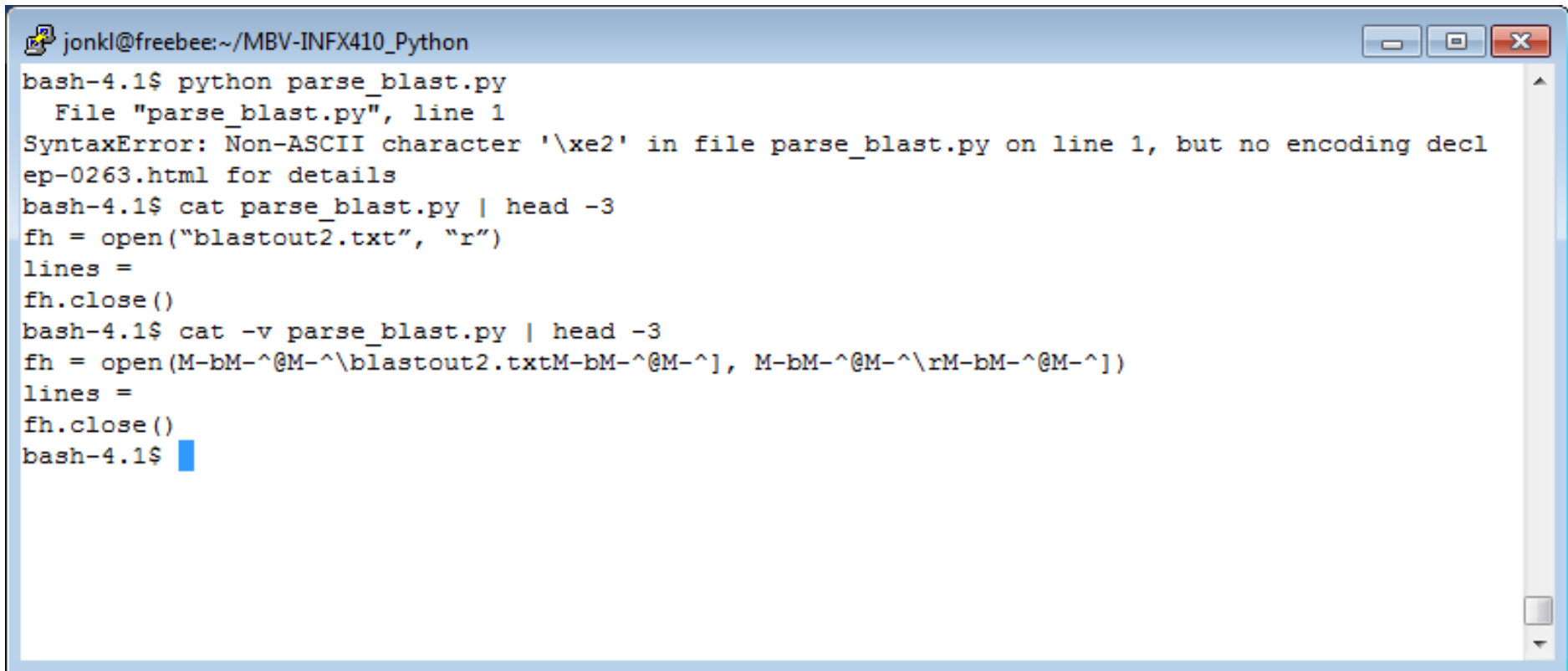
- Why can we do `ls`, `cat`, `rmdir` etc, but have to do `./myScript2`
 - Environment variables in the shell
 - `$PATH`, `$HOME` etc
 - `$HOME` is same as `~/`

```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ ./myScript2.py Hi
Hi has 2 letters
bash-4.1$ myScript2.py Hi
bash: myScript2.py: command not found
bash-4.1$ ls myScript2.py
myScript2.py
bash-4.1$ head -n 1 myScript2.py
#!/usr/bin/python
bash-4.1$ echo $PATH
/usr/lib64/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:/opt/bin
bash-4.1$ which ls
/bin/ls
bash-4.1$ echo $HOME
/usit/abel/u1/jonkl
bash-4.1$
```



Non-printing characters

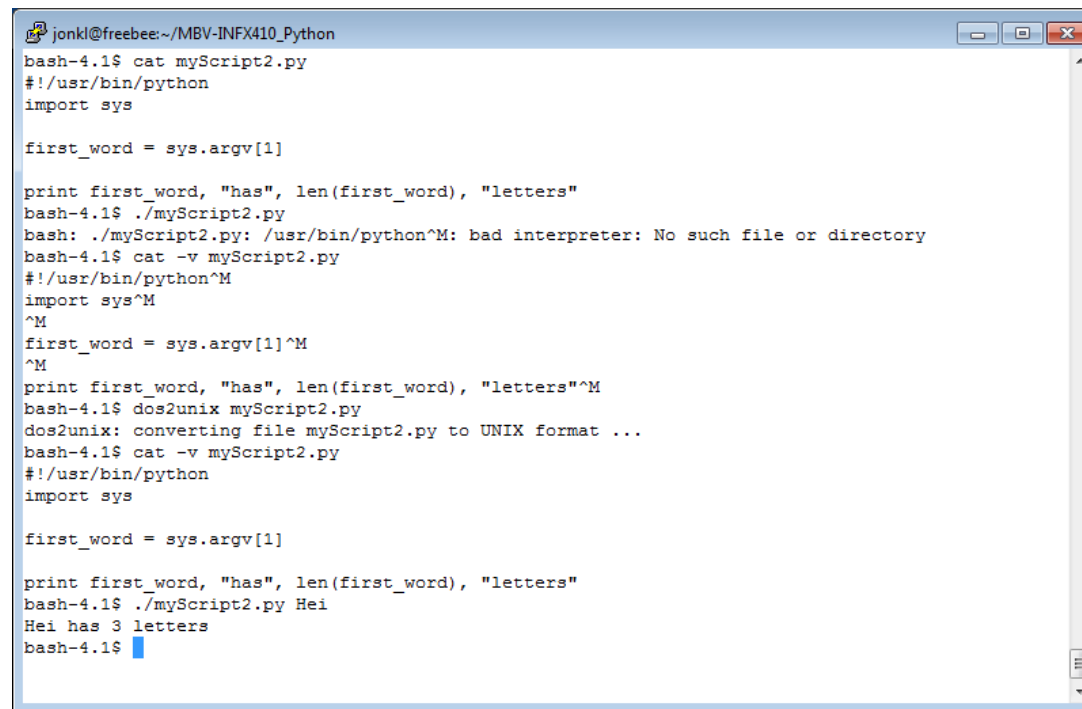
- Use *cat -v* to see “invisible” control codes in the text



```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ python parse_blast.py
  File "parse_blast.py", line 1
SyntaxError: Non-ASCII character '\xe2' in file parse_blast.py on line 1, but no encoding decl
ep-0263.html for details
bash-4.1$ cat parse_blast.py | head -3
fh = open("blastout2.txt", "r")
lines =
fh.close()
bash-4.1$ cat -v parse_blast.py | head -3
fh = open(M-bM-^@M-^\blastout2.txtM-bM-^@M-^], M-bM-^@M-^\rM-bM-^@M-^])
lines =
fh.close()
bash-4.1$
```

End-of-line markers (EOLs) are *not* the same in Unix and MS Windows!

- Use *cat -v* to see “invisible” control codes, for example Windows type EOLs
- Use *dos2unix* command to go from MS Windows to Unix
- Use *unix2dos* command to do the reverse



```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ cat myScript2.py
#!/usr/bin/python
import sys

first_word = sys.argv[1]

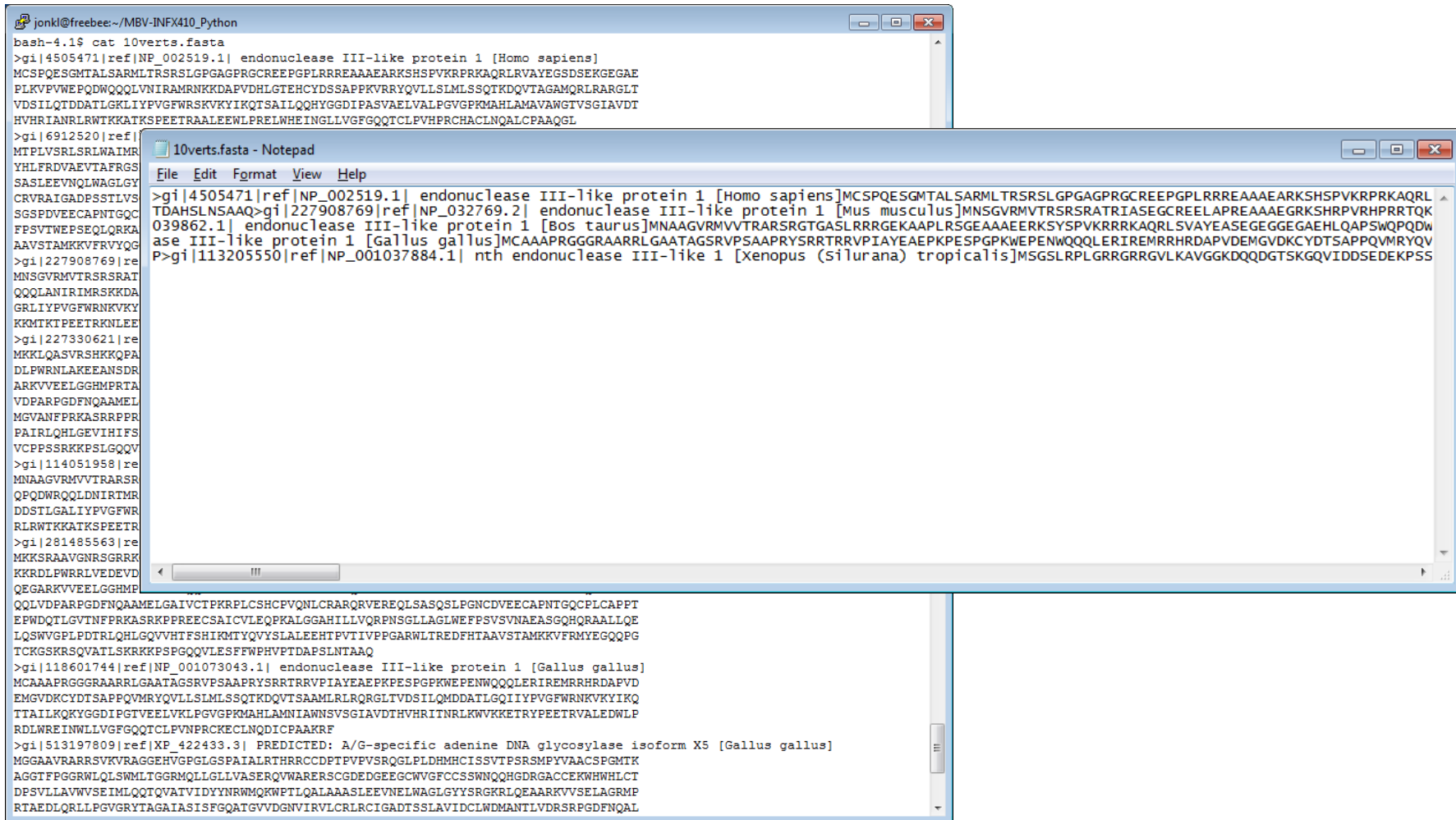
print first_word, "has", len(first_word), "letters"
bash-4.1$ ./myScript2.py
bash: ./myScript2.py: /usr/bin/python^M: bad interpreter: No such file or directory
bash-4.1$ cat -v myScript2.py
#!/usr/bin/python^M
import sys^M
^M
first_word = sys.argv[1]^M
^M
print first_word, "has", len(first_word), "letters"^M
bash-4.1$ dos2unix myScript2.py
dos2unix: converting file myScript2.py to UNIX format ...
bash-4.1$ cat -v myScript2.py
#!/usr/bin/python
import sys

first_word = sys.argv[1]

print first_word, "has", len(first_word), "letters"
bash-4.1$ ./myScript2.py Hei
Hei has 3 letters
bash-4.1$
```

End-of-line markers (EOLs) are *not* the same in Unix and MS Windows!

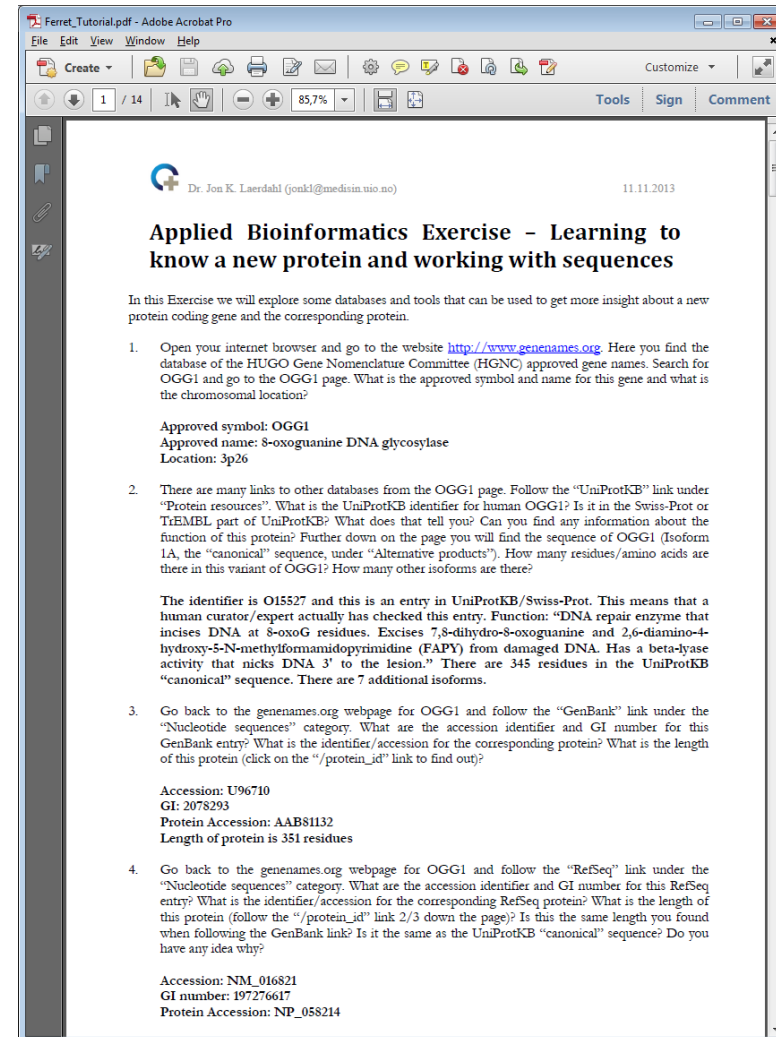
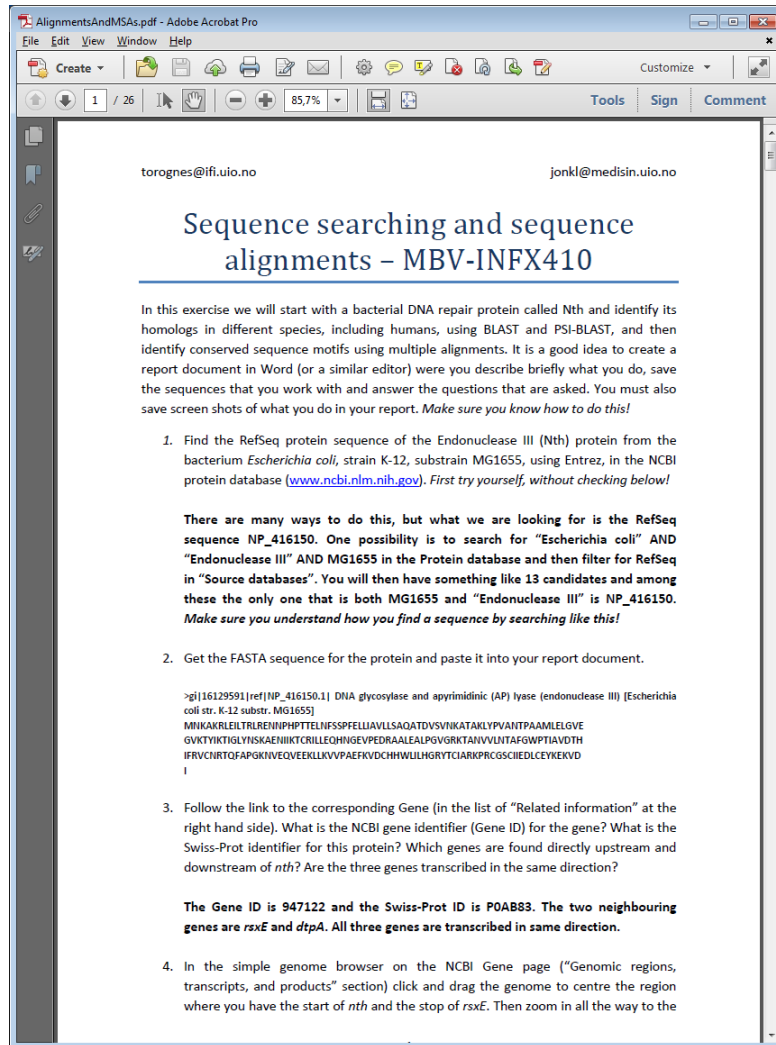
Jon K. Lærdahl,
Structural Bioinformatics



```
jonkl@freebee:~/MBV-INF410_Python
bash-4.1$ cat 10verts.fasta
>gi|4505471|ref|NP_002519.1| endonuclease III-like protein 1 [Homo sapiens]
MCSPESGMTALSARMLTRSLGPGAGPRGCREEPGLRRREAAAEARKSHSPVKRPRKAQRLRVAYEGSDSEKGEAE
PLKVPVWEPQDQQLVNIRAMRNKKDAPVDHLGTEHCYDSSAPKVRRYQVLLSLMLSSQTKDQVTAGAMQRLRARGLT
VDSILQTDATLGLKIYPVGFWRSKVKYIKQTSAILQQHYGGDIPASVAELVALPGVGPKMAHLAMAVAWGTVSGIAVDI
HVHRIANRLRWTKKATKSPEETRAALEEWLPRELWHEINGLLVGFQQTCLPVHPRCHACLNQALCPAAQGL

>gi|6912520|ref|
MTPLVSRLSRLWAIMR
YHLFRDVAEVTAFRGS
SASLEEVNQLWAGLGY
CRVRAIGADPSSTLVS
SGSPDVEECAPNTGQC
FSPVTWEPSEQLQRKA
AAVSTAMKKVFRVYQG
>gi|227908769|re
MNSGVRMVTSRSRAT
QQQLANIRIMRSKKDA
GRLIYPVGFWRNKVKY
KKMKITPEETRNLEE
>gi|227330621|re
MKKLQASVRSRSHKQPA
DLPWRNLAKAEANSR
ARKVVEELGGHMPRTA
VDPARPGDNQAMEL
MGVANFFPKASRRPFR
PAIRLQHLGEVIHIFS
VCPSSRRKKPSLQQQV
>gi|114051958|re
MNAAGVRMVVTRARSR
QFQDWRQQLDNIRTM
DDSTLGALIYPVGFWR
RLRWTKKATKSPEETR
>gi|281485563|re
MKKSRAAVGNRSRRK
KKRDLFWRLVEVD
QEGARKVVEELGGHMP
QQQLVDPARPGDNQAMELGAICTPKRPLCSHCVPQNLCRARQVREREQLSASQSLPGNCDVEECAPNTGQCPLCAPPT
EPWDQTLGVTINFPKASRKPPREECISAICVLEQPKALGGAHILLVQRPNSGLLAGLWFFSVSVNAEASGQHQAALLQE
LQSNVGLPDLRLQHLGQVVTFSHKIMTYQVYSLALEHTPVTIIVPGARWLTREDHTAAVSTAMKKVFRMYEGQQPG
TCKGSKRSQVATLSKRRKPSPGQQVLESFFWPHVPTDAPSLNTAAQ
>gi|118601744|ref|NP_001073043.1| endonuclease III-like protein 1 [Gallus gallus]
MCAAAPRGGRRAARRLGAATAGSRVPSAAPRYSRRTRRVPIAYEAEKPESPGPKWEPENWQQQLIREMRRHRDAPVD
EMGVDKCYDTSAFPQVMRYQVLLSLMLSSQTKDQVTSAMRLRLRQRLTVDSILQMDATLGQIYPVGFWRNKVKYIKQ
TTAILKQKYGGDIPGTVEELVKLPVGVGPKMAHLAMNIWNSVSGIAVDTHVHITNRLKWKVKETRYPEETRVVAEDWLP
RDLNREINWLLVGFQQTCLPVNPRCKECLNQDICPAARF
>gi|513197809|ref|XP_422433.3| PREDICTED: A/G-specific adenine DNA glycosylase isoform X5 [Gallus gallus]
MGGAARARRSVKVRAGGHHVGPGLGSPAIALRTHRRCCDPTVPVSRQGLPLDHMHCISSVTPSRSMFYAACSPGMITK
AGGTFPGGRWLQLSWMLTGGRMQLGLGLVASERQVWARERSCGDEDEEGCWVGFCCSSWNQHGDRGACKEKHHWLCT
DPSVLLAVVNSEIMLQQTQVATVIDYNNWMQKWFTLQALAAASLEEVNELWAGLGYYSRGKRLQEAARKVVSSELGRMP
RTAEDLQRLPGVGRYTAGAIASISFGQATGVVDGNVIRVLCRLRCIGADTSSLAVIDCLWDMANTLVDRSRPGDNQAL
```

These are *very* important!



Articles and written material

- Curriculum comprises
 - All lectures
 - All «handouts»
 - All exercises, demos and computers lab sessions
 - ***All articles listed on the wiki***
 - Obligatory assignment

https://wiki.uio.no/projects/clsi/index.php/MBV-INF410_2015

	Week 46	Week 47	Week 48	Week 49	Week 50	Week 51
Mon	Lecture/ exercises 1 st day of course – Nov 9	Lecture/ exercises	Study day/work on obligatory take-home assignment	Lecture/ exercise	Study day	
Tues	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Study day	
Wed	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Take-home exam start	Take-home exam finish
Thurs	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises		
Fri	Lecture/ exercises	Lecture/ exercises	Hand in oblig take-home assignment	Study day		
Sat/Sun						

You will only be able to take the exam if you follow all teaching!

If you cannot finish the exercises during
the exercise sessions, finish them at
home!

	Week 46	Week 47	Week 48	Week 49	Week 50	Week 51
Mon	Lecture/ exercises 1 st day of course – Nov 9	Lecture/ exercises	Study day/work on obligatory take-home assignment	Lecture/ exercise	Study day	
Tues	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Study day	
Wed	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Take-home exam start	Take-home exam finish
Thurs	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises		
Fri	Lecture/ exercises	Lecture/ exercises	Hand in oblig take-home assignment	Study day		
Sat/Sun						

You will only be allowed to take the exam if your obligatory assignment has been approved!

- Exercise for oblig will be handed out at the end of week 2
- Will be relatively easy and similar to exercises in weeks 1 and 2
- Must be returned before the first lecture in week 3 (4)
- PhD students (MBV-INF9410) must in addition write an assignment/essay
 - Describe how you would use some of the methods covered in the course in your own research
 - > 2500 words

	Week 46	Week 47	Week 48	Week 49	Week 50	Week 51
Mon	Lecture/ exercises 1 st day of course – Nov 9	Lecture/ exercises	Study day/work on obligatory take-home assignment	Lecture/ exercises	Study day	
Tues	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Study day	
Wed	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises	Take-home exam start	Take-home exam finish
Thurs	Lecture/ exercises	Lecture/ exercises	Study day/work on oblig take- home assignment	Lecture/ exercises		
Fri	Lecture/ exercises	Lecture/ exercises	Hand in oblig take-home assignment	Study day		
Sat/Sun						

UNIVERSITETET I OSLO

Det matematisk-naturvitenskapelige fakultet

Obligatory assignment: MBV-INF4410 and MBV-INF9410

Deadline: November 30th 2015, at 09:00

Permitted materials: All written material, including all internet resources

Only students that gets this assignment approved will be permitted to take the course home exam.

Your completed assignment must be returned, at the latest, at 9 am, Monday November 30. It should be sent by e-mail to the course coordinator Jon K. Lærdahl (e-mail address: jonkl@medisin.uio.no). Please put the course code and your name in the subject field (e.g. Oblig MBV-INF4410 Dolly Duck).

The oblig must be handed in as a single PDF document (Microsoft Word or an Open Office Document is also acceptable). Please also include your name and course code in the document and in the document title.

You are encouraged to use screenshots and other figures in order to improve your explanations.

THE WORK MUST REPRESENT YOUR OWN ANSWERS

Answers should contain only what is asked for. Some questions have multiple parts. Your answers may be given in English or in Norwegian. Technical questions about the oblig can be answered by Jon K. Lærdahl (e-mail address: jonkl@medisin.uio.no).

EXTRA Obligatory assignment: MBV-INF9410 (only)

Write an essay of at least 2500 words on one of these two topics

- How can some of the methods described in this course be used in your own research?
- A course relevant topic of your own choice. In this case you must get this approved by Jon K. Lærdahl *before* you write the essay