Advanced Linux Usage

2021-11-23

Martin Dahlö
martin.dahlo@uppmax.uu.se

- **Same program, many files**

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
```

SciLifeLab UPPSALA

- ● **Same program, many files**

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
```

SciLifeLab UPPSALA

- Same program, many files

```
$ ls -l
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 16:42 sample_9.bam
$ my_prog sample_1.bam
$ my_prog sample_2.bam
$ my_prog sample_3.bam
$ my_prog sample_4.bam
$ my_prog sample_5.bam
$ my_prog sample_6.bam
$ my_prog sample_7.bam
$ my_prog sample_8.bam
$ my_prog sample_9.bam
$
```

- **Same program, many files**
  - 10 files? Ok
  - 1000 files? Not ok

- **Same program, many files**
  - 10 files? Ok
  - 1000 files? Not ok
- **Reproducibility**
  - Self and others

- **Same program, many files**
  - 10 files? Ok
  - 1000 files? Not ok
- Reproducibility
  - Self and others

A solution - write a script!

```
total 0
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo 0 Sep  1 17:18 sample_9.bam
$ nano analysis.sh
```

```
GNU nano 2.0.9              File: analysis.sh                        Modified
```

```
^G Get Help    ^O WriteOut    ^R Read File    ^Y Prev Page    ^K Cut Text      ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is     ^V Next Page    ^U UnCut Text    ^T To Spell
```

```
  GNU nano 2.0.9                    File: analysis.sh                              Modified

my_prog sample_1.bam
```

```
^G Get Help     ^O WriteOut     ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit         ^J Justify      ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

```
  GNU nano 2.0.9              File: analysis.sh                        Modified

my_prog sample_1.bam
my_prog sample_2.bam




^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

```
  GNU nano 2.0.9              File: analysis.sh                        Modified

my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam




^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

```
$ l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep  7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_9.bam
$
```

Basic script

```
$ l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep  7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_9.bam
$ bash analysis.sh
```

SciLifeLab
UPPSALA

```
$ l
total 4,0K
-rw-rw-r-- 1 dahlo dahlo 267 Sep  7 09:34 analysis.sh
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_1.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_2.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_3.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_4.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_5.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_6.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_7.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_8.bam
-rw-rw-r-- 1 dahlo dahlo   0 Sep  1 17:18 sample_9.bam
$ bash analysis.sh
```

Still not OK for 1000 or more files!

```
  GNU nano 2.0.9              File: analysis.sh                    Modified

my_prog sample_1.bam
my_prog sample_2.bam
my_prog sample_3.bam
my_prog sample_4.bam
my_prog sample_5.bam
my_prog sample_6.bam
my_prog sample_7.bam
my_prog sample_8.bam
my_prog sample_9.bam




^G Get Help    ^O WriteOut    ^R Read File    ^Y Prev Page    ^K Cut Text     ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is     ^V Next Page    ^U UnCut Text   ^T To Spell
```

```
GNU nano 2.5.3                    File: analysis.sh

my_prog -r references/human_genome.fa sample_1.bam
my_prog -r references/human_genome.fa sample_2.bam
my_prog -r references/human_genome.fa sample_3.bam
my_prog -r references/human_genome.fa sample_4.bam
my_prog -r references/human_genome.fa sample_5.bam
my_prog -r references/human_genome.fa sample_6.bam
my_prog -r references/human_genome.fa sample_7.bam
my_prog -r references/human_genome.fa sample_8.bam
my_prog -r references/human_genome.fa sample_9.bam




^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos      ^Y Prev Page
^X Exit        ^R Read File    ^\ Replace     ^U Uncut Text  ^T To Spell   ^  Go To Line   ^V Next Page
```

- ## Assigning
  ```
  my_variable=5
  my_variable="nice text"
  ```

- ## Assigning
  ```
  my_variable=5
  my_variable="nice text"
  ```
- ## Using
  ```
  $my_variable
  ```

- ## Assigning
  ```
  my_variable=5
  my_variable="nice text"
  ```
- ## Using
  ```
  $my_variable
  ```

```
$ my_variable="Dave"
```

- **Assigning**

```
my_variable=5
my_variable="nice text"
```

- **Using**

```
$my_variable
```

```
$ my_variable="Dave"
$ echo "Hello, $my_variable."
```

- Assigning

```
my_variable=5
my_variable="nice text"
```

- Using

```
$my_variable
```

```
$ my_variable="Dave"
$ echo "Hello, $my_variable."
Hello, Dave.
```

```
GNU nano 2.5.3                    File: analysis.sh

my_prog -r references/human_genome.fa sample_1.bam
my_prog -r references/human_genome.fa sample_2.bam
my_prog -r references/human_genome.fa sample_3.bam
my_prog -r references/human_genome.fa sample_4.bam
my_prog -r references/human_genome.fa sample_5.bam
my_prog -r references/human_genome.fa sample_6.bam
my_prog -r references/human_genome.fa sample_7.bam
my_prog -r references/human_genome.fa sample_8.bam
my_prog -r references/human_genome.fa sample_9.bam




^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     ^Y Prev Page
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^  Go To Line  ^V Next Page
```

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/human_genome.fa

my_prog -r $ref sample_1.bam
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_9.bam




                                       [ Read 12 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     ^Y Prev Page
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^  Go To Line  ^V Next Page
```

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

my_prog -r $ref sample_1.bam
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_9.bam




^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     ^Y Prev Page
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^  Go To Line  ^V Next Page
```

```
for var in 1 2 3;
do
    echo $var
done
```

```
$ bash loop_test.sh
1
2
3
$
```

```
for var in text works too;
do
    echo $var
done
```

```
$ bash loop_test.sh
text
works
too
$
```

```
for var in mix them 5;
do
    echo $var
done
```

```
$ bash loop_test.sh
mix
them
5
$
```

```
for var in *.txt;
do
    echo $var
done
```

```
$ bash loop_test.sh
all.txt
examples.txt
readme.txt
```

```
GNU nano 2.5.3                      File: analysis.sh

ref=references/goat_genome_version4.1.fa

my_prog -r $ref sample_1.bam
my_prog -r $ref sample_2.bam
my_prog -r $ref sample_3.bam
my_prog -r $ref sample_4.bam
my_prog -r $ref sample_5.bam
my_prog -r $ref sample_6.bam
my_prog -r $ref sample_7.bam
my_prog -r $ref sample_8.bam
my_prog -r $ref sample_9.bam




^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos     ^Y Prev Page
^X Exit        ^R Read File    ^\ Replace     ^U Uncut Text  ^T To Spell   ^  Go To Line  ^V Next Page
```

Basic script

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in *.bam ;
do
    my_prog -r $ref $file
done




^G Get Help    ^O Write Out    ^W Where Is     ^K Cut Text     ^J Justify      ^C Cur Pos      ^Y Prev Page
^X Exit        ^R Read File    ^\ Replace      ^U Uncut Text   ^T To Spell     ^  Go To Line   ^V Next Page
```

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in *.bam ;
do
    echo my_prog -r $ref $file
done




^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos      ^Y Prev Page
^X Exit        ^R Read File    ^\ Replace     ^U Uncut Text  ^T To Spell   ^  Go To Line   ^V Next Page
```

Debugging!

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in *.bam ;
do
    echo my_prog -r $ref $file
done




^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text
^X Exit        ^R Read File    ^\ Replace     ^U Uncut Te
```

```
$ bash analysis.sh
my_prog -r references/goat_genome_version4.1.fa sample_1.bam
my_prog -r references/goat_genome_version4.1.fa sample_2.bam
my_prog -r references/goat_genome_version4.1.fa sample_3.bam
my_prog -r references/goat_genome_version4.1.fa sample_4.bam
my_prog -r references/goat_genome_version4.1.fa sample_5.bam
my_prog -r references/goat_genome_version4.1.fa sample_6.bam
my_prog -r references/goat_genome_version4.1.fa sample_7.bam
my_prog -r references/goat_genome_version4.1.fa sample_8.bam
my_prog -r references/goat_genome_version4.1.fa sample_9.bam
```

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in *.bam ;
do
    my_prog -r $ref $file
done
```

^G Get Help   ^O Write Out   ^W Where Is   ^K Cut Text   ^J Justify   ^C Cur Pos   ^Y Prev Page
^X Exit   ^R Read File   ^\ Replace   ^U Uncut Text   ^T To Spell   ^ Go To Line   ^V Next Page

```
$ bash analysis.sh
```

```
$ bash analysis.sh data/
```

```
$ bash analysis.sh data/
```

$1

```
$ bash analysis.sh data/   second_argument
```

                        $1 $2

```
$ bash analysis.sh data/   second_argument third
                    $1          $2          $3
```

```
$ bash analysis.sh data/   second_argument third "fourth argument"
```

$1 $2 $3 $4

```
GNU nano 2.5.3                     File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in *.bam ;
do
    my_prog -r $ref $file
done




^G Get Help      ^O Write Out     ^W Where Is      ^K Cut Text      ^J Justify       ^C Cur Pos       ^Y Prev Page
^X Exit          ^R Read File     ^\ Replace       ^U Uncut Text    ^T To Spell      ^  Go To Line    ^V Next Page
```

```
GNU nano 2.5.3                    File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in $1/*.bam ;
do
    my_prog -r $ref $file
done
```

```
$ cat file.list
sample_1.bam
sample_3.bam
smaple_9.bam
```

```
GNU nano 2.5.3                        File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in $1/*.bam ;
do
    my_prog -r $ref $file
done
```

```
GNU nano 2.5.3                          File: analysis.sh

ref=references/goat_genome_version4.1.fa

for file in $( cat $1 ) ;
do
    my_prog -r $ref $file
done



^G Get Help    ^O Write Out    ^W Where Is    ^K Cut Text    ^J Justify    ^C Cur Pos      ^Y Prev Page
^X Exit        ^R Read File    ^\ Replace     ^U Uncut Text  ^T To Spell   ^  Go To Line   ^V Next Page
```

Arguments

```
ref=references/goat_genome_version4.1.fa

for file in $( cat $1 ) ;
do
    my_prog -r $ref $file
done
```

```
$ cat file.list
sample_1.bam
sample_3.bam
sample_9.bam
$ bash analysis.sh file.list
```

- Control statement

```
if condition; then
   action
fi
```

- **Control statement**

```
if true; then
  echo "This is true"
fi
```

result:
```
This is true
```

- Control statement

```
if false; then
    echo "This is true"
fi
```

result:

- **Control statement**

```
if [[ 5 < 9 ]]; then
  echo "This is true"
fi
```

result:
```
This is true
```

- **Control statement**

```
if [[ 5 > 9 ]]; then
  echo "This is true"
fi
```

result:

- **Control statement**

```
if [[ 5 == 9 ]]; then
   echo "This is true"
fi
```

result:

- **Control statement**

```
if [[ "Hello" == "Hello" ]]; then
  echo "This is true"
fi
```

result:
```
This is true
```

- **Control statement**

```
if [[ "Hello" == "Hi" ]]; then
  echo "This is true"
fi
```

result:

- Control statement

```
if [[ "Hello" == "Hel"* ]]; then
  echo "This is true"
fi
```

result:
```
This is true
```

- ## For all samples except dog

```
for file in $1/*.bam ;
do
    echo my_prog $file
done
```

- ## For all samples except dog

```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

- For all samples except dog

```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: $file is /path/to/dog_1.bam

- **For all samples except dog**

```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

`basename $file`

- For all samples except dog

```
for file in $1/*.bam ;
do
    if [[ ... != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` **is** `/path/to/dog_1.bam`

`basename $file`

`dog_1.bam`

- For all samples except dog

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then

        echo my_prog $file

    fi
done
```

Ex: `$file` is `/path/to/dog_1.bam`

`basename $file`

`dog_1.bam`

- **For all samples except dog**

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

Ex: `$file` **is** `/path/to/dog_1.bam`

`basename $file`

`dog_1.bam`

Different languages

- Programming is programming
  - Perl, Python, Bash, and more

Different languages

- Programming is programming
  - Perl, Python, **Bash**, and more

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

- Programming is programming
  - **Perl**, Python, Bash, and more

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

```
use strict;
use warnings;
use File::Basename;

foreach my $file (glob("$ARGV[0]/*.bam")) {

    if(basename($file) !~ "^dog.+"){

        system("my_prog", $file);
    }
}
```

- Programming is programming
  - Perl, **Python**, Bash, and more

```
for file in $1/*.bam ;
do
    if [[ $(basename $file) != "dog"* ]]; then

        my_prog $file

    fi
done
```

```
import glob
import sys
import subprocess
import os

for file in glob.glob( sys.argv[1] + "/*.bam" ):

    if not os.path.basename(file).startswith("dog"):

        subprocess.call( ["my_prog" , file] )
```

Different languages

- Programming is programming
  - ○ Perl, Python, Bash, and more

- Start with one, git gud, (learn another)
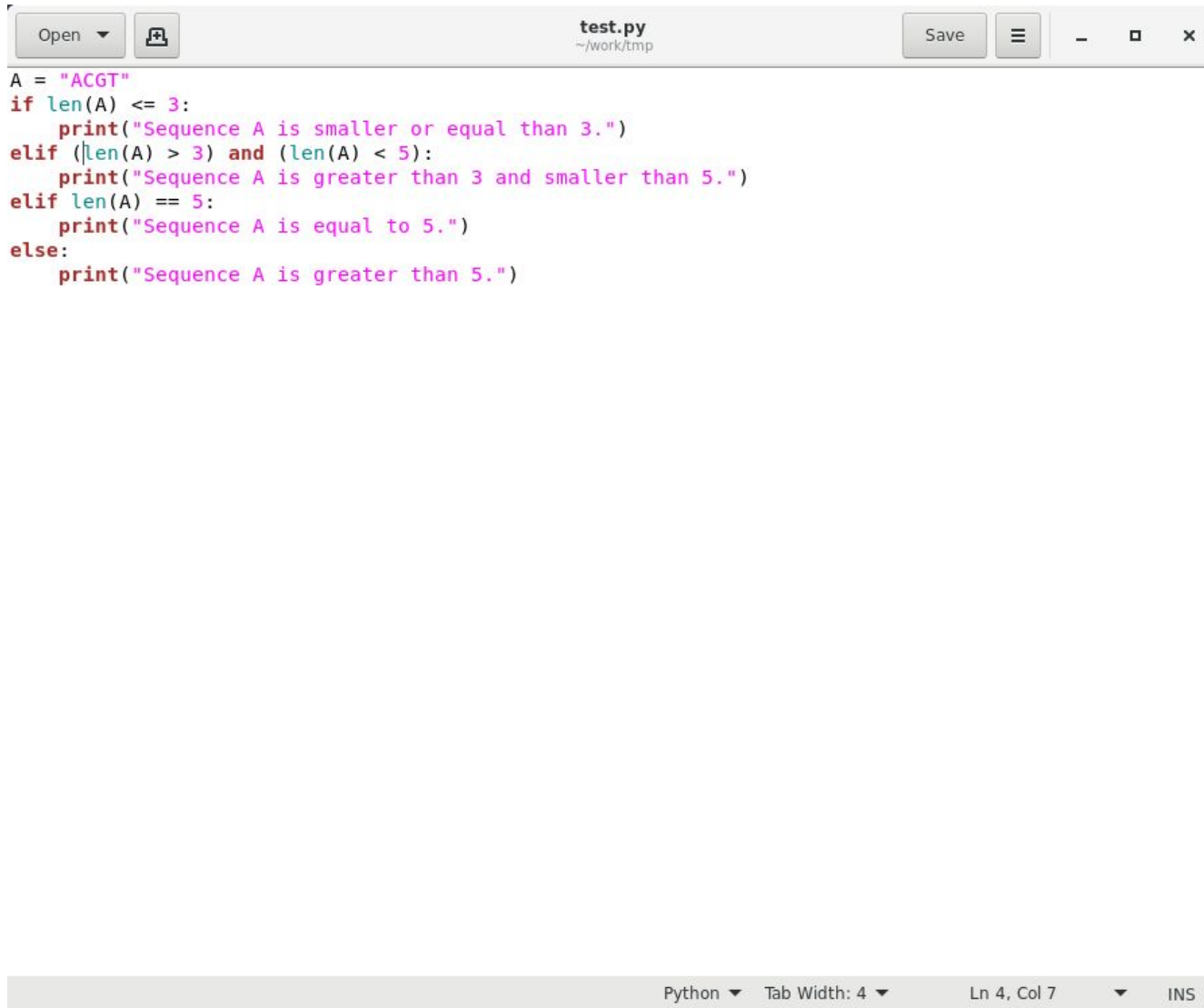
Different languages

- Programming is programming
  - Perl, Python, Bash, and more

- Start with one, git gud, (learn another)

PYTHON

- Graphical text editor more similar to what you might be used to
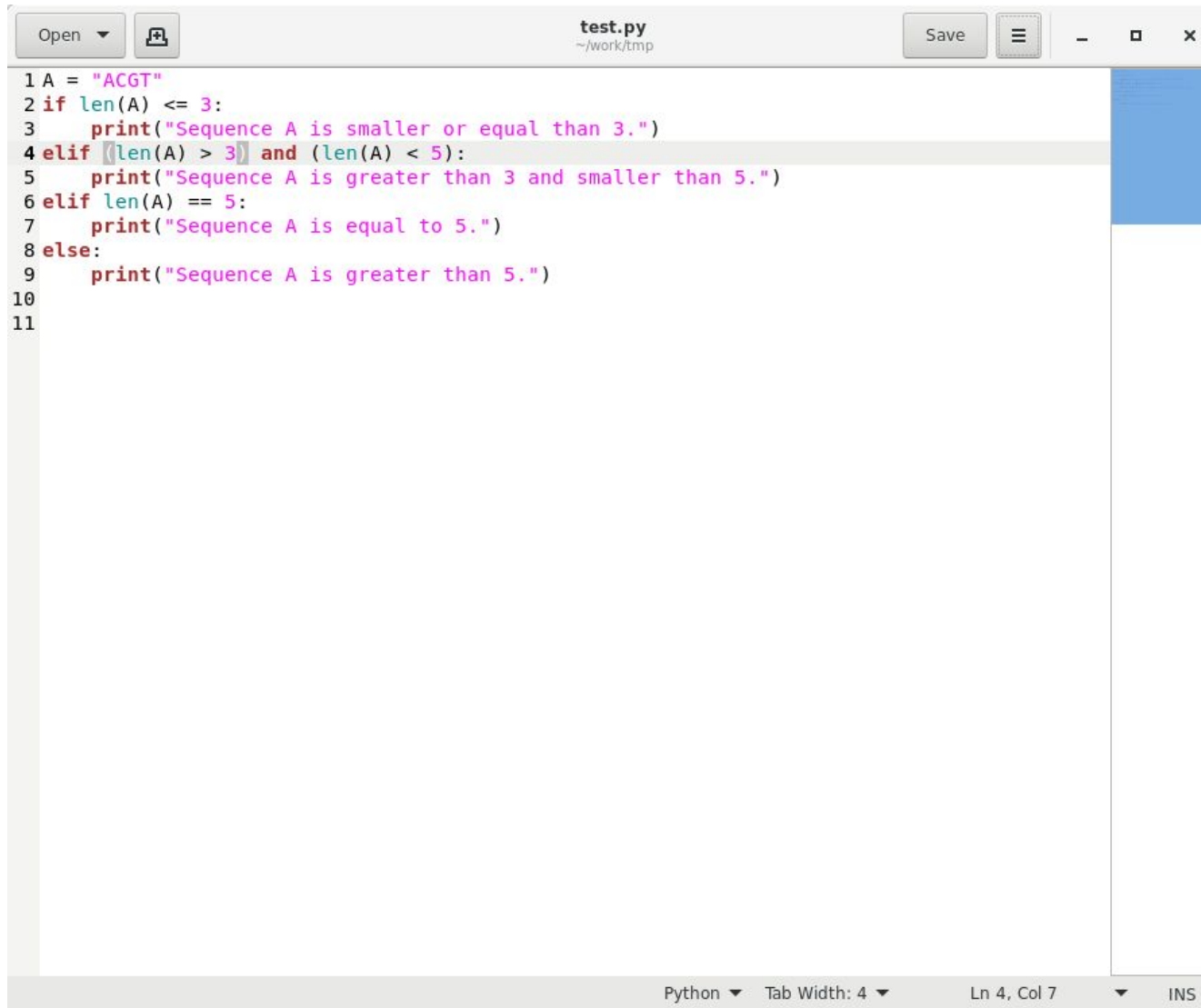
- Launch through command line:

```
$ gedit
```

Gedit

test.py
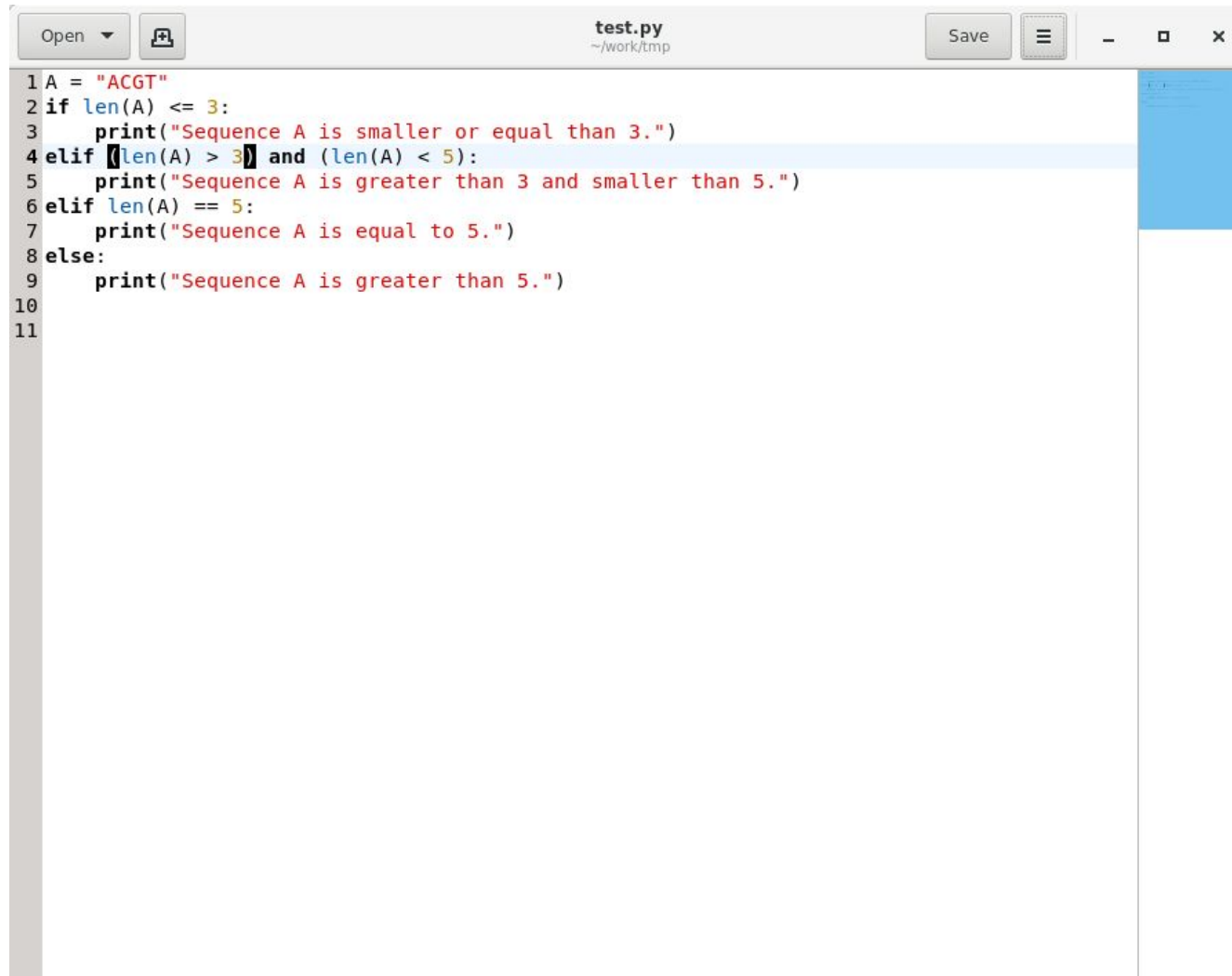~/work/tmp

```python
A = "ACGT"
if len(A) <= 3:
    print("Sequence A is smaller or equal than 3.")
elif (len(A) > 3) and (len(A) < 5):
    print("Sequence A is greater than 3 and smaller than 5.")
elif len(A) == 5:
    print("Sequence A is equal to 5.")
else:
    print("Sequence A is greater than 5.")
```

Python ▾   Tab Width: 4 ▾   Ln 4, Col 7   ▾   INS

Gedit


```python
A = "ACGT"
if len(A) <= 3:
    print("Sequence A is smaller or equal than 3.")
elif (len(A) > 3) and (len(A) < 5):
    print("Sequence A is greater than 3 and smaller than 5.")
elif len(A) == 5:
    print("Sequence A is equal to 5.")
else:
    print("Sequence A is greater than 5.")
```

Gedit


```python
1 A = "ACGT"
2 if len(A) <= 3:
3     print("Sequence A is smaller or equal than 3.")
4 elif (len(A) > 3) and (len(A) < 5):
5     print("Sequence A is greater than 3 and smaller than 5.")
6 elif len(A) == 5:
7     print("Sequence A is equal to 5.")
8 else:
9     print("Sequence A is greater than 5.")
10
11
```

Theme: Kate

Gedit


```python
A = "ACGT"
if len(A) <= 3:
    print("Sequence A is smaller or equal than 3.")
elif (len(A) > 3) and (len(A) < 5):
    print("Sequence A is greater than 3 and smaller than 5.")
elif len(A) == 5:
    print("Sequence A is equal to 5.")
else:
    print("Sequence A is greater than 5.")
```

Theme: Oblivion

Gedit


```python
A = "ACGT"
if len(A) <= 3:
    print("Sequence A is smaller or equal than 3.")
elif (len(A) > 3) and (len(A) < 5):
    print("Sequence A is greater than 3 and smaller than 5.")
elif len(A) == 5:
    print("Sequence A is equal to 5.")
else:
    print("Sequence A is greater than 5.")
```

Theme: Monokai

- Menu - Preferences - View
  - Display line numbers
  - Display overview map
  - Highlight current line
  - Highlight matching brackets
- Menu - Preferences - Editor
  - Tab width 4
  - Insert spaces instead of tabs
- Menu - Preferences - Fonts & Colors
  - Kate or Oblivion

Laboratory time once again!

https://uppsala.instructure.com/courses/47037/pages/linux-5-advanced-bash-lab