Table of Contents

Introduction to Unix, Part 3 - grep and regular expressions

Welcome to the Programming for Evolutionary Biology workshop!!

Giovanni M. Dall'Olio and Alvaro Perdomo-Sabogal, 03/03/2019. All materials available here: https://github.com/dalloliogm/evop2019/archive/master.zip (https://github.com/dalloliogm/evop2019/archive/master.zip)

In this third part we will see the **grep** command to search the contents of files, and some basic regular expressions.

Press space or down key to continue.

In [6]:

```
# Configuration - this will not appear in the slideshow
%alias grep='grep --color'
%cd
%cd workspace/Peb2019/exercises/
```

```
/Users/gmd78366
/Users/gmd78366/workspace/Peb2019/exercises
```

Searching patterns into file: grep

The instructions for the next exercises are stored in the **2_searching_patterns.txt** file.

However, if you look at this file with head or less, you will see that its contents have no meaning!

In [3]:

```
head 2_searching_patterns.txt
```

```
N6jorDR84qO1Wm9oQSwq NFP5fuIUt    dicCiojstp  RVFEDyvlCiQgJludpQz3Dmh
mfsFAdF6K87GCcFxIuPJU3KV bOqbM    nl1Kliq0kejysuyzQjjHD4uLwCiz9uDfGJ8
k8KXugdUgQWis6TJj5Bs3r0O zhucJ    Jdrh5eLocvbvx3c1nOECQLPX3Zxoed RdWF
a55O3BZvOWND7NgRpTGYNvagOPftki    QwYSIr6uFambWWjUyHWSc3ayRD6dovvuBAr
0LJ4YFDvRbpOz DBcKgo45oRKFFN83    T6fRFSpMzZUbXz lwpT6LC98uiNSdFXRrdV
YpGM8k5BhVkL FSAlpKaF1aOjKOX9K    Z5ov kGnjbKosQQeRmHt0KlrobawghapA8f
51WCoVFad 5NUN8cd9MKOkSmqHkEAy    bTKIT3Xl9zZkwXpflOG0Ka0CiS77Q3Nq9Bn
wIZTwypDgNcFN3JxFrJuOHJxsdIKPD    hnTBOWnngoVfOrNeR6af7mZKCFLHLov7pPF
 lVzzIoSD03ll3peasKZyaLEsRW1er    GAI9wh7LzYicG647nz1KA9Rv3TuHujuJaDZ
ocuB5qUMrxMbopSc6IymtjPCVmhhFd    Mdh8d7kTkAqMP0Fqutt6B33Etn93XAcXDnV
```

This is because the instructions are hidden in the file, just to make things more interesting!

Press space or the down key to see how to continue.

The first instruction for the first exercise is in the lines containing the word "start" somewhere.

To find them, type:

```
$: grep start 2_searching_patterns.txt
```

In [4]:

```
grep start 2_searching_patterns.txt
```

```
cyiywNQhsTaFvrX3F0 start6473DC
DRyKuiS dTekI3ctrstartv7VXoOCC     ______________________________
S4xFQNhv0ipsyg75yCoxA3hestartJ    / Congrats!                    \
t start e6YBF5VeAJd1niRCAm54fv    |  You've used grep correctly, |
UQhstart S6CADhyFEGtamLmgUw5mN    \  and found a cow.            /
na3xDyV start nEn FcDkDoHlJ5U8     ------------------------------
vB64co0qVmDSfMy startHBRBNmMou            \   ^__^
Ucm64KuL6ci0HB startg3B91ubyVR             \  (oo)\_______
dmATFfwkAnvmm  starthJsUB7Zhpa                (__)\       )\/\
x startdyxGG 530mUC7MDaokCAKkP                    ||----w |
TnuS 6WyvWQffstart bhC8FgngIfA                    ||     ||
Lpjd8fy4HwOj3start KFmLWhcpH1D
qrcD DUVmU67qINstart eimdNwsk7
x startYmyFAMljNGPoBZxWvdugLOE
Egmqh5pXm eestart 5JkazumfP9      The command grep allows to search
T0Zq6 startfnvP8cIXnfqbB4q5la0    for a pattern in a text file.
0apOeBLv7YvlPA start gkQLhw5LJ
CxfzRe541ystartJj68PrNKawesz08    It will print all the matching
bd7175m w start 5YFVqNF1A8ZJS6    lines to the screen.
Hj6BoO startsRWuvPNGBg4ZSxdy5K
FrowUgP0Rww startD56jcTvQQWEMt    =================
f6exIg44GXvkBKEj5O0sPt startuT    Next Exercise
pUasyifskHstart AA4QhdNI5CYOqp    =================
WpEr7I1I7ajn3Vb startyV77kLktG
PULXHsnrjVtqGQJP0T start x  mT    In the next exercise we will see
e94IRW6tna8G9gzXnMFeyeWiMstart    how to access grep's documentation.
Yj7ItSnjwa xpIIJ6Pnstarty4c9pv
amG5UYNApOEml0 startgehiEmraPN    Grep the following word to continue:
c5zgU3wsGstartLXkuSEU5X6htL4vh      _            _
nYoEzZOnC18fdKoy start SwGylYg     | |          | |
vEj5 start qFNICtaBpVNgDRpfaJk     | |__    ___ | | _ __
ik7aYCKUtVVn4s3mQS8NEKx6 start     | '_ \  / _ \| || '_ \
LscAgVefstartXT3xEhEESN8qzhyU0     | | | ||  __/| || |_) |
r4XFjstartez4gqwdMPsmjysywXiW5     |_| |_| \___||_|| .__/
KQS1yymt0dfm start 6IRkDDywzwg                     | |
Y5XQOdrDQ1uU 8uexqOkpQy start                      |_|
cUhpfXSO startWCjg6ImRVLQoLcwx
1PKSKJRIAYZ65HBustart6yeNy4ti6
```

Accessing grep documentation

To access the documentation of a command, we can use the **man** command.

Let's type the following:

```
$: man grep
```

This will allow to navigate the documentation for grep, in the same modality as with the **less** command. Use the arrows to scroll, and q to exit.

For the next exercise, you will have to identify two options in the man page, and use them to do a case-insensitive search for "ignorecase", and count the number of lines.

In [5]:

```
grep help 2_searching_patterns.txt
```

```
3rC1S7O mhelp uDa8dTWUinNSVSDX
aeda4XBrj7YzD help xnpmxY3J1I0    The documentation for grep can
kFRgBPrGHX9GLsjfWYQ help A3QIf    be accessed through man:
GJD8rk1Aahelp rFvCwxsfVGy8 YJy
DJoDl10eVzXkHLo7helpDTfeA3GYvX        $: man grep
I help FGSlQgjeD7MN6Z13g1x0vNs
3RT8c1IHNBSaYmEGoNhelp PCfxdsz    Scroll down to see all the
O Z DWnhzuyIoPw9BQKzhelpaJKm6c    parameters for grep and their descri
ption.
hRomqLnWx4geJlErVCz0X ZVvhelp
aPP69R8T5oHzJa0C W help WUMyv4    Use / to search for text.
ot1fC vOE1 help BbP6Fp1jjli63x    Press the q key to exit.
haluPmjhelp kO1Hqxi5OXE5g1Z6Vu
ZLpjYjWAZAjhelp fAEFHYQ4rO7bFS
9qPAKjaYfUFosjNaEgIIjqcxhelpUC
rXMAdAeVM1N6S1wIXMhelp ri5Hzua    ==============
Z1wBOjpWKjii3Vl9YNTXhelp AZyGD    Next exercise
BRdO1AiB9TGVw6bhTohelpLvzl7ncH    ==============
PVW6VrjQ help pkWdQ6V8MUlBb4mm
M48fL93YMhelpYdAsUIvJ49XZeu6Em    For the next exercise, you will need
to open
Y5xGjZqOSOr0KgGv1dBhelpqwLgkag    grep's documentation and identify tw
o options:
RUkLYcn67CO8vdJP help dgdyAbkr
uk4qm4NtrDwxoifclE5zuhelp QZBj    - the option for case-insensitive se
arches
f9kFTlhQQAmhelp qWaQWrvPWGz eL    - the option for counting
cfGAsfmlGnBaDu4YdoRMhelpSY0oUv      the number of matching lines,
Ij3Fvhelphbk MeRlx 7tydUBcaCznjF      instead of printing them to the sc
reen.
helpeoAiKHpCi9jT 4iNxf9UucLfG0
EJVr lhelp jkJcVXqoiITMrAgC00c    Once you have identified these optio
ns,
hXissBPNstqPwJrLhelp eb7g0BOHl    do a case-insensitive search on this
file for the word
y3CwhA0YH3v help 0huOOdCCSiWyb    "ignorecase", then count the number
 of lines.
pQun5zSme6 help 80vwKy oy4Ti X
AA6helpovE0Iw4f1CeWrTzC1nwpb4u
```

In [6]:

```
# If we do a search for "ignorecase" without any option, we only get some of the li
# You can notice that the cow is not properly displayed :-)
grep ignorecase 2_searching_patterns.txt
```

```
J0WGignorecaseqrq 9hak97vkCYL0
PWPsSOpcgCPMignorecasevzY7sDxG
GZVfLsGkT7vOAeofqOignorecaseKo    Remember that, to continue with the
 exercise,
Wc37PlL5LsjqvNrgoignorecase7jW    you need to do a case-insensitive se
arch for the word
y3CwhA0YH3v help 0huOOdCCSiWyb    "ignorecase", then count the number
 of lines.
Ei4dDJignorecaseL64bJfMudEsq5M
ZO3JWgignorecaseAgQb0mUonH3Snj    | itive        |
rhUDhIJsyq4mYhcPFQoignorecaseN    \ search       /
```

In [7]:

```
# The -i option allows to do a case-insensitive search.
# As you can see, some lines contain upper case characters:
grep -i ignorecase 2_searching_patterns.txt
```

```
J0WGignorecaseqrq 9hak97vkCYL0
PWPsSOpcgCPMignorecasevzY7sDxG
GZVfLsGkT7vOAeofqOignorecaseKo    Remember that, to continue with the
 exercise,
Wc37PlL5LsjqvNrgoignorecase7jW    you need to do a case-insensitive se
arch for the word
y3CwhA0YH3v help 0huOOdCCSiWyb    "ignorecase", then count the number
 of lines.
Ei4dDJignorecaseL64bJfMudEsq5M
yX6cvAZsIGNORECASEhu3lxg5OTkk9     _____________
YBgY HtVe6lUjUbhYGIGNORECASESd    / Good Job!    \
fCkzDFRyUIignOrecaseNjeVrQaL V    | You did a    |
DjP4Qa ignOrecase60rD76lrkJlOP    | case-insens  |
ZO3JWgignorecaseAgQb0mUonH3Snj    | itive        |
rhUDhIJsyq4mYhcPFQoignorecaseN    \ search       /
eIb0cbignOrecaserqT99cHXsbazNE     -------------
dth3agTCwYH0ouignOrecase86d908           \   ^__^
n ignOrecase8JdVs9NCgrFTRXELXJ            \  (oo)\_______
Il7nHbcIGNORECASEEgUF5IeSoZfT1               (__)\       )\/\
E0IGNORECASEQrSNWGlfJITjVutE0q                   ||----w |
0CSMFb4OUMoaeSjSIGNORECASEBCxr                   ||     ||
H3FDmXPjSt0IgnorEcaseCB7Lfrtoh
zjZwttL5RQignOrecaseePhoKUN6lz
LvX4RShx95emIgnorEcaseQkJNFiqY
```

In [8]:

```
# To solve the exercise, we also have to count the number of output lines.
# This can be done with the "-c" option:
grep -i -c ignorecase 2_searching_patterns.txt
```

```
21
```

In [9]:

```
# solution: how to find the instructions for the next exercise
grep 21 2_searching_patterns.txt
```

```
XvQkpnuGE76p6e9OIDJDWZocYQq21z
ZzNpBzsof8oCSIXw013oC21FgzjmoY
WBOYltyyUdP821rAGsMzaMqQhXWdlI     ____________________
21SEAWynnbhHdSgg7t4jb3bTehrDD8    / Congrats! Yes       \
X8ETxBuWCJwnU5SF4eufjXGD0Ol213    | the answer to the   |
snaAFxRROH21yfMk4J9py5rHbOdkrp    | case-insensitive    |
487lJUgZMGuA8xGYFPYMMKPVqp9218    | and count question  |
CJIWv21yJkeMIAw6WgteBotAN1v8Sy    | is 21.              /
LvPMXrbgI4QG95gZaFCLg21JPMs9TK     --------------------
NoLOe21xUJXFNVcNnwA6MI3HyBrwng            \   ^__^
AxsMe0L0RI5Pt21QGXsExNM3ZUQfwn             \  (oo)\_______
nsg RCxVtQZMzn keLaTzhm kVn21X                (__)\       )\/\
qvTkY57v8rSIa5W21a 0NqNZ jWgCv                    ||----w |
LjkILyX3X21A98SQrMbRBsKig8AGg9                    ||     ||
sZWBKGyJ8IurleKipjixQ21j5euj15
UlNkBb 9JIJ7gPFE8a7qcfwSx21jlP
IwTSAEw8MLcsEofkQDRsi21YIaA rM    =============
Q21AFnH5GzPHHCirwoIWJkZdfknt40    Next exercise
VN6olNG8cw9rw5rcxJeaHckVuztB21    =============
nj8lvbvU8oDlZbnGUMsC7aw6zGw21M
BdQhorRoOCggPSLI4bfMZ211DwDf4p    Searching in multiple files
IKdlDTv6cpaLYISjZO21m6oqckkl6i
N21wFpJGlRg969qfL kbveWG0slXqe    Grep can search the same pattern
7NgLAwqDPsMQQxM5TNFXpU30xJ21O     in more than one file at the same ti
me.
 JB921UtTm9Hdx9OWgIyyDf9NVFA0Y
aEhY210ci75TGjYdmXtylKyDEy7Kfl    The folder data/multiplefiles/ conta
ins hundreds of different files.
9sW5VuKAnkBUS5tE7PL50Grq21wL4B
7oHcn60lBwS QPfile321mtDvIWy5u    Next Exercise
qE21lJ1GcXfMkE9Ze8NIYsmV7BWBfL    Can you identify the file containing
the word "regex"?
mwvlebf216dNweaaVwOv6tpp5GXl4t
g3521Rg3t3wHZxke5exFZaiByZJvvi
```

Searching multiple files

Grep is useful to search over multiple files in a single command.

The folder data/multiplefiles/ contains 50 randomly generated files. You can see their contents with head data/multiplefiles/* or with less.

One of these files contains the word "regex" in it. Are you able to find it?

In [11]:

```
# solution: you can use the "*" character to specify multiple files:
grep 'regex' multiplefiles/*
```

```
multiplefiles/file32.txt:5gsumFTKbKEJv9dD8W94FhoEQU8qf8RMUcregexR
multiplefiles/file32.txt:YgDiqkA C1oregex9giqI66c3sOwfLirOsgPpSuq
multiplefiles/file32.txt:IsXSnp 8U8pKR0LsVuKregexO5GFegOtV4GW4fNQ    G
ood! You've found the
multiplefiles/file32.txt:l 4px8KhPRmfEJgi5uTuVO1XahG3H1sYregex4wt    f
ile containing the word "regex"
multiplefiles/file32.txt:yz8P5 HC6N5D XRHPncZjTAeMregexT9bQUoZdsh
multiplefiles/file32.txt:eWUd18s0MVx5YYrEK KCKeF5hvOregexIiZbIGUX    T
o continue,
multiplefiles/file32.txt:MLXiKZJ8KyHMou9lYsz4ZjFYJSfB 14tregextpJ    g
rep file32.txt exercises/2_searching_patterns.txt
multiplefiles/file32.txt:veFQUregexfnQxwQw6POJRNvvAeYwToX6ptvN39m
multiplefiles/file32.txt:cHoNvregexiGjHkmptPVTjOzvWVGbrGoHoywV4Vy
```

Searching multiple patterns and the Unix piping system

How can we search that contain two or more patterns?

One solution is to use the Unix piping system, executing one grep command, and then another grep on the output.

This can be done using the pipe "|" symbol, like the following:

```
$: grep (first pattern) myfile.txt | grep (second pattern)
```

Press space or the down key for some examples.

The file data/genes/mgat_genes.gb is a genbank file. Notice how this format is well suited for grep searches:

In [12]:

```
head genes/mgat_genes.gb
```

```
LOCUS       HUMUDPCNA               4705 bp    DNA     linear   PRI 19
-SEP-1995
DEFINITION  Human alpha-1,3-mannosyl-glycoprotein beta-1,
            2-N-acetylglucosaminyltransferase (MGAT) gene, complete cd
s.
ACCESSION   M61829
VERSION     M61829.1  GI:340075
KEYWORDS    alpha-1,3-mannosyl-glycoprotein beta-1,2-N-acetylglucosami
nyltrae.
SOURCE      Homo sapiens (human)
  ORGANISM  Homo sapiens
            Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Eutele
ostomi;
            Mammalia; Eutheria; Euarchontoglires; Primates; Haplorrhin
i;
```

Let's say we want to search all the lines where "ORGANISM" is "Homo sapiens".

We can do it with two grep commands:

```
    grep ORGANISM genes/mgat_genes.gb | grep 'Homo sapiens'
```

Notice that searching for "Homo sapiens" alone would not be enough, as there are other lines where the word "Homo sapiens" is present.

In [13]:

```
grep ORGANISM genes/mgat_genes.gb | grep 'Homo sapiens'
```

```
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
  ORGANISM  Homo sapiens
```

The file contains sequences from two other organisms apart from Homo sapiens. Can you guess which one to search for the next exercise?

In [19]:

```
# Solution: grep for "bos taurus":
grep ORGANISM genes/mgat_genes.gb | grep taurus
```

```
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus                     _______________
  ORGANISM  Bos taurus                    < Good Guess! >
  ORGANISM  Bos taurus                     ---------------
  ORGANISM  Bos taurus                                  \   ^__^
  ORGANISM  Bos taurus                                   \  (oo)\_____
__
  ORGANISM  Bos taurus                                      (__)\
)\/\
  ORGANISM  Bos taurus                                          ||----
w |
  ORGANISM  Bos taurus                                          ||
||
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus                          ===============
  ORGANISM  Bos taurus                          Next Exercise
  ORGANISM  Bos taurus                          ===============
  ORGANISM  Bos taurus
  ORGANISM  Bos taurus                          To continue, grep
  ORGANISM  Bos taurus                          "regex" in
  ORGANISM  Bos taurus                          data/exercise1_grep.tx
t
  ORGANISM  Bos taurus
```

The Unix Philosophy and Approach to

programming

Earlier this morning we mentioned the Unix Philosophy:

- Make each program do one thing well.
- Expect the output of every program to become the input to another, as yet unknown, program.
- Work on file streams, reading one line at a time.

All the commands we saw today follow the Unix philosophy:

- **ls** is for listing files
- **grep** is for searching patterns in files
- **less**, **head**, and others are for seeing the contents of a file

Applying the Unix Philosophy to your programming

In the following days, when you will learn the basics of Python and R programming, you will be faced with the dilemma of how to organize your scripts.

You can use the Unix Philosophy as a guideline to have more versatile files and scripts.

Example of Unix Approach

Let's immagine we want to plot the CG content of a genome:

- **Approach 1 (non-Unix)**: write a single script that downloads the genome, calculates the GC content, and draws a plot
- **Approach 2 (Unix)**: write three separate scripts or functions, then pipe them together using the Unix Pipe or with a wrapper script.

The Unix Approach requires a bit of extra work, but it is more versatile in case you want to reuse the same scripts.

Regular Expressions

Regular expressions allow to search for more complex patterns.

Here are some simple regular expression examples:

regex	description
.	matches any character
[A-Za-z]	matches any of the characters within parenthesis
.*	matches any character, any number of times

Regular Expression exercise

Let's have a look at the file data/genes/sequences.fasta:

In [15]:

```
head genes/sequences.fasta
```

```
>seq000 sequence description
CGNTTTNTAATTATATANNTAGCGTGATCC
>seq001 sequence description
NNCGNANAGCTNGACCTAGTTGAAATTGTG
>seq002 sequence description
CGGTATGCAGCCCNNGCGTNGCNTNAATNA
>seq003 sequence description
CGCNTCNTNCTTCACCACGCCAGCTTTANC
>seq004 sequence description
CGTNGCGNNCGNGAGCCATTANTAGNNCCT
```

Can you use grep to identify all the sequences containing three As, followed by any two characters, followed by three Ts?

In [16]:

```
grep 'AAA..TTT' genes/sequences.fasta
```

```
TCGACCTGCNNANGGCTNTGTAAAccTTTG
TNCGTCCGGCCAAAAAAgtTTTNGNGCTTG
NAAGTTTAAGAAAaaTTTCATGAGCNCGAG
NCTGCGCNGCGAGCCACAACNAAAgtTTTG
TAGNGTACTCCTTNTNAAAaaTTTTTCTCG
ACTNGACGCTTCNCTNAAAccTTTGCNTNT
GGGNAAAaaTTTCGGCTGNNTGCNCNNTNN
TNGAACCCNGTNCGGTCAAAccTTTTCNTA
NGGTACAAAGCNCCCANNNTTAAAgtTTTC
AAAccTTTNCNGCCNTNCTCANNCGTGNTT
NTTCACTCAAActTTTNGNGNATTNGGAAT
GGATGAAAaaTTTNCCAGCCAANNCTTGNA
```

Bonus: if we use the -B 1 grep option, we can retrieve the names of these sequences:

In [17]:

```
grep -B1 'AAA..TTT' genes/sequences.fasta
```

```
>seq009 sequence description
TCGACCTGCNNANGGCTNTGTAAAccTTTG
--
>seq012 sequence description    ________________
TNCGTCCGGCCAAAAAAgtTTTNGNGCTTG
--
>seq024 sequence description   / Congrats! This \
NAAGTTTAAGAAAAaaTTTCATGAGCNCGAG
>seq025 sequence description   | was the last   |
NCTGCGCNGCGAGCCACAACNAAAgtTTTG
>seq026 sequence description   \ grep exercise /
TAGNGTACTCCTTNTNAAAAaaTTTTTCTCG
--
>seq030 sequence description    ----------------
ACTNGACGCTTCNCTNAAAccTTTGCNTNT
--
>seq032 sequence description       \   ^__^
GGGNAAAAaaTTTCGGCTGNNTGCNCNNTNN
>seq033 sequence description        \  (oo)\_______
TNGAACCCNGTNCGGTCAAAccTTTTCNTA
--
>seq038 sequence description           (__)\       )\/\
NGGTACAAAGCNCCCANNNTTAAAgtTTTC
--
>seq043 sequence description               ||----w |
AAAccTTTNCNGCCNTNCTCANNCGTGNTT
--
>seq046 sequence description               ||     ||
NTTCACTCAAActTTTNGNGNATTNGGAAT
--
>seq049 sequence description
GGATGAAAAaaTTTNCCAGCCAANNCTTGNA
```

In [18]:

```
# Bonus: pipe an additional grep '>' to see a cow:
grep -B1 'AAA..TTT' genes/sequences.fasta  | grep '>'
```

```
>seq009 sequence description
>seq012 sequence description    ________________
>seq024 sequence description   / Congrats! This \
>seq025 sequence description   | was the last   |
>seq026 sequence description   \ grep exercise /
>seq030 sequence description    ----------------
>seq032 sequence description       \   ^__^
>seq033 sequence description        \  (oo)\_______
>seq038 sequence description           (__)\       )\/\
>seq043 sequence description               ||----w |
>seq046 sequence description               ||     ||
>seq049 sequence description
```

Time for a break!

In [ ]:

```

```