Биомолекула<https://biomolecula.ru/articles/ia-b-v-bioinformatiki-poshiol-pust-menia-nauchat> по мотивам статьи (<https://www.nature.com/articles/nbt.2740>)

<https://zenodo.org/record/1462325#.XWaiT-gzY2z> – книжка линукс для биоинформатиков

**Unix tutorial**

**Listing files and directories**

**ls (short for list)**

$ ls

To list all files in your home directory including those whose names begin with a dot

$ ls –a

with additional information:

$ ls –la

## **Making Directories**

### **mkdir (make directory)**

$ mkdir bioinformatics

$ ls

## **Changing to a different directory**

### **cd (change directory)**

$ cd bioinformatics

Type ls to see the contents (which should be empty)

### **Exercise**

Make another directory inside the **bioinformatics** directory called **seminar1.** Navigate through

## **The directories . and ..**

Still in the **seminar1** directory, type

$ ls -a

As you can see, in the **seminar1** directory (and in all other directories), there are two special directories called (**.**) and (**..**)

In UNIX, (**.**) means the current directory, so typing

$ cd .

NOTE: there is a space between cd and the dot

means stay where you are (the **seminar1** directory).

(**..**) means the parent of the current directory, so typing

$ cd ..

## **Pathnames**

### **pwd (print working directory)**

Pathnames enable you to work out where you are in relation to the whole file-system

$ pwd

The full pathname will look something like this -

**/Users/mariap**

which means that mariap (your home directory) is in the directory Users (the directory containing all user directories), which is located in the root directory (the root directory is indicated by the leading slash '/').

### **Exercise**

Use the commands ls, pwd and cd to explore the file system. (Remember, if you get lost, type cd by itself to return to your home-directory)

### **~ (your home directory)**

$ ls ~/bioinformtics

$ ls ~

$ ls ~/..

## **Copying Files**

### **cp (copy)**

cp *file1 file2*

$ cd ~/bioinformatics

$ touch dna.txt

$ cp dna.txt protein.txt

## **Moving files**

### **mv (move)**

mv *file1 file2* moves (or renames) **file1** to **file2**

$ mkdir rna

$ cp dna.txt rna.txt

$ mv rna.txt rna/.

$ cp dna.txt dna1.txt

$ mv dna1.txt rna.txt

Type ls and ls/rna to see if it has worked.

## **Removing files and directories**

### **rm (remove), rmdir (remove directory)**

$ rm rna.txt

$ ls (to check if it has deleted the file)

$ rmdir rna

## **Displaying the contents of a file on the screen**

### **clear (clear screen)**

$ clear

### **DOWNLOAD A FILE TO PLAY WITH**

### 16SRNA.txt

### **cat, less, more, head, tail**

The command cat can be used to display the contents of a file on the screen. Type:

$ cat 16SRNA.txt

The command less writes the contents of a file onto the screen a page at a time. Type

$ less 16SRNA.txt

Press the [space-bar] if you want to see another page, type [q] if you want to quit reading. As you can see, less is used in preference to cat for long files.

$ head 16SRNA.txt

$ head -5 16SRNA.txt

$ tail 16SRNA.txt

## **Searching the contents of a file**

### Взять файл geneticCode.txt

### **Simple searching using less**

Using less, you can search though a text file for a keyword (pattern). For example, to search through **science.txt** for the word 'science', type

$ less geneticCode.txt

then, still in less (i.e. don't press [q] to quit), type a forward slash [/] followed by the word to search

/CCC

/Glycine

As you can see, less finds and highlights the keyword. Type [n] to search for the next occurrence of the word.

### **grep (globally search a regular expression and print)**

grep is one of many standard UNIX utilities. It searches files for specified words or patterns. First clear the screen, then type

$ grep Glycine geneticCode.txt

To ignore upper/lower case distinctions, use the -i option, i.e. type

$ grep –i glycine geneticCode.txt

Some of the other options of grep are:

-v display those lines that do NOT match

-n precede each matching line with the line number

-c print only the total count of matched lines

$ grep –ivc glycine geneticCode.txt

### **wc (word count)**

A handy little utility is the wc command, short for word count. To do a word count on **science.txt**, type

$ wc -w geneticCode.txt

To find out how many lines the file has, type

$ wc -l geneticCode.txt

## **Redirection**

**cat**

Standart input – from the keyboard, standart output – to the terminal screen.

If you run the cat command without specifying a file to read, it reads the standard input (the keyboard), and on receiving the 'end of file' (^D), copies it to the standard output (the screen).

$ cat

Then type a few words on the keyboard and press the [Return] key.

Finally ^D to end the input.

If you run the cat command without specifying a file to read, it reads the standard input (the keyboard), and on receiving the 'end of file' (^D), copies it to the standard output (the screen).

In UNIX, we can redirect both the input and the output of commands.

### **Redirecting the Output**

We use the **>** symbol to redirect the output of a command. Create a file called **nucleotides** containing a list of nucleotides, type

$ cat > nucleotides

Type:

adenine

cytosine

guanine

thymine

^D (Control D to stop)

$ cat nucleotides1

**>>** - appends standard output to a file

Exercise: create file **nucleotides2** with 4 of them, than append one more (uracil) to the file

We will now use the cat command to join (concatenate) **nucleotides1** and **nucleotides2** into a new file called **nucleotides3**. Type

$ cat nucleotides1 nucleotides2 > nucleotides3

$ cat nucleotides3

## **Redirecting the Input**

< - to redirect the input of a command.

**sort** - alphabetically or numerically sorts a list

$ sort

Then type in the names of some nucleotides. Press [Return] after each one.

adenine

cytosine

guanine

thymine

^D (control d to stop)

$ sort < nucleotides1

output the sorted list to a file:

$ sort < nucleotides1 > sorted\_list

$ cat sorted\_list

**Pipes**

### $ cat nucleotides1 | sort

### **Exercise**

Using pipes, print all lines of **nucleotides1** and **nucleotides2** containing the letter 'u', sort the result, and save to the **list**.

(answer: $ cat nucleotides1 nucleotides2 | grep u | sort > list)

remove this files: $ rm list nucleotides1 nucleotides2

## **Wildcards**

### **\* and ?**

### \* - none or more character(s) in a file (or directory) name

? - exactly one character

$ ls \*.txt

$ ls ?eneticCode.tx?

## **Getting Help**

**man (manual)**

$ man cat

**apropos** - when you are not sure of the exact name of a command

$ apropos copy

## **File system security (access rights)**

$ ls -l

first column - access rights

the left group of 3 gives the file permissions for the user that owns the file (or directory) (sburns in the above example)

the middle group gives the permissions for the group of people to whom the file (or directory) belongs (sburns in the above example);

the rightmost group gives the permissions for all others.

### **Access rights on files.**

### r (or -), indicates read permission (or otherwise), that is, the presence or absence of permission to read and copy the file

### w (or -), indicates write permission (or otherwise), that is, the permission (or otherwise) to change a file

### x (or -), indicates execution permission (or otherwise), that is, the permission to execute a file, where appropriate

### **Access rights on directories**

### r allows users to list files in the directory;

### w means that users may delete files from the directory or move files into it;

### x means the right to access files in the directory. This implies that you may read files in the directory provided you have read permission on the individual files.

## **Changing access rights**

### **chmod (changing a file mode)**

## **Processes and Jobs**

ps (process) - information about your processes

$ ps

**&** at the end of the command line - background process

### Backgrounding a current foreground process

### $ sleep 100

### **^Z** Then to put it in the background, type

### $ bg

jobs - see list of running processes with status

$ jobs

to restart process

$ fg %jobnumber

fg with no job number foregrounds the last suspended process

kill process

$ kill %jobnumber

or

$ kill (PID\_number)

if refuses

$ kill -9 (PID\_number)

## **Other useful UNIX commands**

### **df** - reports the space left on the file system

**du** - the amount disk space used by each subdirectory

the depth of the listing with the -d switch

$ du -d 1

**zip**

$ zip geneticCode geneticCode.txt

$ unzip geneticCode.txt.zip

**history** - show command history list

**file** - classifies the named files according to the type of data they contain

$ file \*

**Pattern Search**

$ cd /usr/share/common-licenses/

Поиск встречаемости паттерна

$ grep "GNU" GPL-3

Поиск паттерна не обращая внимание на размер символа

$ grep -i "license" GPL-3

Найдем строки не содержащие BSD

$ grep -v "the" BSD

В дополнение к предыдущему выведем еще номер строки

$ grep -vn "the" BSD

Регулярные выражения

[ ]: Matches any one of a set characters

[ ] with hyphen: Matches any one of a range characters

^: The pattern following it must occur at the beginning of each line

^ with [ ] : The pattern must not contain any character in the set specified

$: The pattern preceding it must occur at the end of each line

. (dot): Matches any one character

\ (backslash): Ignores the special meaning of the character following it

\*: zero or more occurrences of the previous character

(dot).\*: Nothing or any numbers of characters.

Строки где слова начинаются с GNU и что угодно может идти после

grep "^GNU" GPL-3

Найти любую строку, которая заканчивается на "and"

$ grep "and$" GPL-3

Найти "cpt" с любыми двумя символами перед этим

$ grep "..cept" GPL-3

Выражения в скобках для поиска с использованием регулярных выражений

Найдет too или two

$ grep "t[wo]o" GPL-3

Начинается с "с" и имеет "ode" после

$ grep "[^c]ode" GPL-3

Найдем все слова когда строка начинается с заглавных букв от A до Z

$ grep "^[A-Z]" GPL-3

Аналогичное можно сделать, используя POSIX стандарт

$ grep "^[[:upper:]]" GPL-3

Ищем слова в скобках с возможностью пробела между ними

$ grep "([A-Za-z ]\*)" GPL-3

Найти любую строку, которая начинается с заглавной буквы и заканчивается точкой

$ grep "^[A-Z].\*\.$" GPL-3

Найти либо GPL либо General Public License

Внимание -Е означает группировку (расширенные регулярные выражения)

$ grep -E "(GPL|General Public License)" GPL-3

Найти либо copy либо copyright либо right

$ grep -E "(copy)?right" GPL-3

Любые три символа из AEIOUaeiou идут рядом

$ grep -E "[AEIOUaeiou]{3}" GPL-3

**Bash-скрипты:**

$ #!/bin/bash

#!/bin/bash

# This is a comment

pwd

whoami

Cохранить в файл, увидеть ошибку доступа. Добавить право на исполнение.

$ chmod +x ./myscript

./myscript

Добавить вывод сообщений

#!/bin/bash

# our comment is here

echo "The current directory is:"

pwd

echo "The user logged in is:"

whoami

#!/bin/bash

# display user home

echo "Home for the current user is: $HOME"

Извлечение информации из вывода команд и назначение её переменным

Значок обратного апострофа «`»

Конструкция $()

mydir=`pwd`

mydir=$(pwd)

#!/bin/bash

mydir=$(pwd)

echo $mydir

**if-then**

if команда

then

команды

fi

#!/bin/bash

if pwd

then

echo "It works"

fi

#!/bin/bash

user=student

if grep $user /etc/passwd

then

echo "The user $user Exists"

fi

if команда

then

команды

else

команды

fi

#!/bin/bash

user=anotherUser

if grep $user /etc/passwd

then

echo "The user $user Exists"

else

echo "The user $user doesn’t exist"

fi

#!/bin/bash

user=anotherUser

if grep $user /etc/passwd

then

echo "The user $user Exists"

elif ls /home

then

echo "The user doesn’t exist but anyway there is a directory under /home"

fi

Сравнение чисел

n1 -eq n2 Возвращает истинное значение, если n1 равно n2.

n1 -ge n2 Возвращает истинное значение, если n1больше или равно n2.

n1 -gt n2 Возвращает истинное значение, если n1 больше n2.

n1 -le n2 Возвращает истинное значение, если n1меньше или равно n2.

n1 -lt n2 Возвращает истинное значение, если n1 меньше n2.

n1 -ne n2 Возвращает истинное значение, если n1не равно n2.

Сравнение строк

str1 = str2 Проверяет строки на равенство, возвращает истину, если строки идентичны.

str1 != str2 Возвращает истину, если строки не идентичны.

str1 < str2 Возвращает истину, если str1 меньше, чем str2.

str1 > str2 Возвращает истину, если str1 больше, чем str2.

-n str1 Возвращает истину, если длина str1 больше нуля.

-z str1 Возвращает истину, если длина str1 равна нулю.

#!/bin/bash

val1=”ATCTGACT”

val2="AATT"

if [ $val1 \> $val2 ]

then

echo "$val1 is greater than $val2"

else

echo "$val1 is less than $val2"

fi

Проверка файлов

-d file Проверяет, существует ли файл, и является ли он директорией.

-e file Проверяет, существует ли файл.

-f file Проверяет, существует ли файл, и является ли он файлом.

-r file Проверяет, существует ли файл, и доступен ли он для чтения.

-s file Проверяет, существует ли файл, и не является ли он пустым.

-w file Проверяет, существует ли файл, и доступен ли он для записи.

-x file Проверяет, существует ли файл, и является ли он исполняемым.

file1 -nt file2 Проверяет, новее ли file1, чем file2.

file1 -ot file2 Проверяет, старше ли file1, чем file2.

-O file Проверяет, существует ли файл, и является ли его владельцем текущий пользователь.

-G file Проверяет, существует ли файл, и соответствует ли его идентификатор группы идентификатору группы текущего пользователя.

#!/bin/bash

mydir=/home/likegeeks

if [ -d $mydir ]

then

echo "The $mydir directory exists"

cd $ mydir

ls

else

echo "The $mydir directory does not exist"

fi

**Циклы for**

for var in list

do

команды

done

#!/bin/bash

for var in first second third fourth fifth

do

echo The $var item

done

#!/bin/bash

for var in first "the second" "the third" "I’ll do it"

do

echo "This is: $var"

done

#!/bin/bash

file="myfile"

for var in $(cat $file)

do

echo " $var"

done

**IFS (Internal Field Separator) -** переменная окружения

#!/bin/bash

file="/etc/passwd"

IFS=$'\n'

for var in $(cat $file)

do

echo " $var"

done

**Вывод списка файлов и папок**

#!/bin/bash

for file in /home\*

do

if [ -d "$file" ]

then

echo "$file is a directory"

elif [ -f "$file" ]

then

echo "$file is a file"

fi

done

**Стандартный цикл for**

#!/bin/bash

for (( i=1; i <= 10; i++ ))

do

echo "number is $i"

done

Цикл **while**

#!/bin/bash

var1=5

while [ $var1 -gt 0 ]

do

echo $var1

var1=$[ $var1 - 1 ]

done

**Обработка содержимого файла**

#!/bin/bash

IFS=$'\n'

for entry in $(cat /etc/passwd)

do

echo "Values in $entry –"

IFS=:

for value in $entry

do

echo " $value"

done

done

Есть **break** и **continue**

**Поиск исполняемых файлов**

#!/bin/bash

IFS=:

for folder in $PATH

do

echo "$folder:"

for file in $folder/\*

do

if [ -x $file ]

then

echo " $file"

fi

done

done

**Ввод и вывод**

* 0, STDIN — стандартный поток ввода.
* 1, STDOUT — стандартный поток вывода.
* 2, STDERR — стандартный поток ошибок.

$ pwd >> myfile

Появится сообщение об ошибке, и ничего не сделается:

$ ls –l xfile > myfile

Сообщение об ошибке теперь попадёт в файл myfile:

$ ls -l xfile 2>myfile

$ cat ./myfile

ls –l myfile xfile anotherfile 2> errorcontent 1> correctcontent

В bash-cкрипте

Амперсанд перед номером дескриптора

#!/bin/bash

echo "This is an error" >&2

echo "This is normal output"

Обычный вывод делается в консоль, а сообщения об ошибках попадают в файл:

./myscript 2> myfile

#!/bin/bash

exec 1>outfile

echo "This is a test of redirecting all output"

echo "from a shell script to another file."

echo "without having to redirect every line"

#!/bin/bash

exec 2>myerror

echo "This is the start of the script"

echo "now redirecting all output to another location"

exec 1>myfile

echo "This should go to the myfile file"

echo "and this should go to the myerror file" >&2

Ввод файла построчно:

#!/bin/bash

exec 0< testfile

count=1

while read line

do

echo "Line #$count: $line"

count=$(( $count + 1 ))

done

**Создадим 10 файлов**

for (( i=1; i <= 10; i++ )) do

touch file$i.fasta

done

**Сменить расширение всех файлов в директории:**

for f in \*.fasta; do mv $f `basename $f .fasta`.fa; done;

**Добавить расширение ко всем файлам:**

for f in \*; do mv $f `basename $f `.txt; done;

**Удалить расширение из всех файлов**

for f in \*.txt; do mv $f `basename $f .txt`; done;

**Summary**

ls list files and directories

ls -a list all files and directories

mkdir make a directory

cd directory change to named directory

cd change to home-directory

cd ~ change to home-directory

cd .. change to parent directory

pwd display the path of the current directory

cp file1 file2 copy file1 and call it file2

mv file1 file2 move or rename file1 to file2

rm file remove a file

rmdir directory remove a directory

cat file display a file

more file display a file a page at a time

head file display the first few lines of a file

tail file display the last few lines of a file

grep 'keyword' file search a file for keywords

wc file count number of lines/words/characters in file

command > file redirect standard output to a file

command >> file append standard output to a file

command < file redirect standard input from a file

command1 | command2 pipe the output of command1 to the input of command2

cat file1 file2 > file0 concatenate file1 and file2 to file0

sort sort data

who list users currently logged in

lpr -Pprinter filename print a text or PDF file to named printer

\* match any number of characters

? match one character

man command read the online manual page for a command

whatis command brief description of a command

apropos keyword match commands with keyword in their man pages

ls -la list access rights for all files

chmod [options] file change access rights for named file

command & run command in background

^C kill the job running in the foreground

^Z suspend the job running in the foreground

bg background the suspended job

jobs list current jobs

fg %1 foreground job number 1

kill %1 kill job number 1

ps list current processes

kill 26152 kill process number 26152