#2

Text files, Streams, Open Source and Software Management

Agenda

- Working with text files
- Basic bioinformatics file types
  - FASTA, NGS file formats, ALN ClustalW, PDF, GeneBank, SwissProt, Newick
- Streams and filters used to work with sequence files
- Open Source Repositories
  - Sourceforge
  - GitHub
- Software Management

Who am I?

Michał Gałka

Contact:

galka.michal@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Michał Sarna

Contact:

michalsarna@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Marcin Górski

Contact:

mkgorski@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Who am I?

Andrzej Stasiak

Contact:

aandrzej.stasiak@gmail.com (e-mail or hangouts)

Use **[BioIT]** in the e-mail topic.

Working with text files

Looking inside text files

- **`cat` -** concatenate and print files
  - Syntax: `cat [OPTIONS] [FILES]`
  - e.g. `cat -n file1.txt`
- **`tac` -** concatenate and print files in reverse
  - Syntax: `tac [OPTIONS] [FILES]`
  - e.g. `tac file1.txt`
- **`more` -** is a filter for paging through text one screenful at a time
  - Syntax: `more [OPTIONS] [FILES]`
  - e.g. `more file1.txt`
- **`less` -** opposite of more
  - Syntax: `less [OPTIONS] [FILES]`
  - e.g. `less file1.txt`
  - Use ‘q’ to quit

Looking inside text files

- **`tail`** - output the last part of files
  - Syntax: `tail [OPTIONS] [FILES]`
  - e.g. `tail -n 2 file1.txt`
- **`head`** - output the first part of files
  - Syntax: `head [OPTIONS] [FILES]`
  - e.g. `head -n 3 file1.txt`
- **`sort`** - sort lines of text files
  - Syntax: `sort [OPTIONS] [FILES]`
  - e.g. `sort -f file1.txt`
- **`cut`** - remove sections from each line of files
  - Syntax: `cut [OPTIONS] [FILES]`
  - e.g. `cut -c2 file1.txt`

Word counting

- **`wc`** - print newline, word, and byte counts for each file
  - `wc [OPTION] [FILE]`
  - -l - counts lines
  - -w - counts words
  - -m - characters
  - e.g. `wc -l file.txt`

Finding files

- **`find`** - search for files in a directory hierarchy
  - The syntax is: `find [OPTIONS] [STARTPOINT] [EXPRESSION]`
  - e.g. `find . -perm 664`
  - e.g. `find . -name dir`
- **`grep`** - print lines matching a pattern
  - The syntax is: `grep [OPTIONS] [SEARCHPHRASE] [FILENAME]`
  - e.g. `grep -ir test file*`

Text editors

- **`vi`** – programmer's text editor
  - Syntax: `vi [OPTIONS] [FILES]`
  - e.g. `vi -m file1.txt`
- **`nano`** - Nano's ANOther editor, an enhanced free Pico clone
  - Syntax: `nano [OPTIONS] [FILES]`
  - e.g. `nano -v file1.txt`
- **`gedit`** - editor with graphical user interface
  - Syntax: `gedit [OPTIONS] [FILES]`
  - e.g. `gedit -b file1.txt`

STDIN, STDOUT, STDERR

- Since everything is a file You can write to file or read from it.
- Outputting files to terminal is just sending them to STDOUT file.
- Entering commands with keyboard is reading them from STDIN file.
- STDERR file is magic place where all error messages go.
- **echo** - display a line of text
  - The syntax is: `echo [OPTIONS] [TEXT TO SEND TO OUTPUT]`
  - e.g. `echo “Hello World”`
  - e.g. `echo “Hello World” >&2`

|, > and >>

- Since everything is a file we need a way to stream content of one file to another.
- | - is a pipe that can connect one program's output to another's program input
- > - redirects to file (will overwrite file)
- >> - redirects to file (will append to file)

Putting all pieces together

- `cut -d ‘ ‘ -f 3 data.txt >> thirdwords.txt`
  - Removes all but the third word from each line in data.txt file
    - Assuming that “word” is any character sequence between spaces
  - Appends result to thirdwords.txt file
- `cat data.txt | grep -i foo | grep -iv bar`
  - finds all lines in data.txt file that contain “foo” but don’t contain “bar” (case insensitive)
    - “A line with FOO only” will match
    - “A line with foo and bar” will not match
- `tail -n 100 data.txt | sort > sorted.txt`
  - Sorts last 100 lines of the data.txt file and saves result in sorted.txt file

Basic bioinformatics file types

FASTA, NGS file formats, ALN ClustalW, PDF, GeneBank, SwissProt, Newick

Streams and filters used to work with sequence files

Open Source Repositories

Sourceforge

SourceForge

- **Central collection of software projects**
- **Offers support for multiple versioning systems**
  - Git, SVN, Mercurial
- **Most frequently used with a link to an existing project**
- **Beware of advertisements (revenue model)**

GitHub

GitHub

- **The most popular way to host and share code**
  - 14 million users (April 2016)
  - 35 million repositories (April 2016)
- **Free to use for open source repositories**
- **Easy collaboration**
  - Readme files
  - Wiki
  - Bug tracking
- **Based on git version control system**
- **No registration required to download a repository**
- git clone https://github.com/michalsarna/bioit.git

Software Management

Software Management

- Contemporary distributions of Linux install software in pre-compiled packages
  - binaries (applications)
  - configuration
  - dependencies
- There are many package management systems available
  - dpkg
  - RPM
  - opkg
  - and others
- Bio-Linux is based on dpkg
  - we'll use APT (Advanced Package Tool) - a front-end for dpkg
  - almost all commands must be executed with superuser privileges (sudo)
  - be careful - you can damage your system!

Software Management

- Update package database
  - `sudo apt update`
- Upgrade system's software
  - `sudo apt upgrade`
  - run this after `sudo apt update`
- Searching for a package
  - `apt search package-name`
  - e.g. `apt search nano`
- Package information
  - `apt show package-name`
  - e.g. `apt show nano`

Software Management

- Installing a package
  - `sudo apt install package-name [package-name2 ...]`
  - e.g. `sudo apt install nano`
- Remove a package (without configuration)
  - `sudo apt remove package-name`
  - e.g. `sudo apt remove nano`
- Remove a package (along with configuration)
  - `sudo apt purge package-name`
  - e.g. `sudo apt purge nano`

That’s all for today.