Week 1: The UNIX terminal and Git

02807 Computational Tools for Data Science

Objective. Today's objective is to get familiar with some of the most basic tools from the UNIX world. If you are already familiar with some of these tools, you should skip the corresponding exercises and spend your time on something more useful[1]. If not, this is the time for you to learn them – many new tools and words will be introduced, so keep calm, and look them up if you do not know what they mean.

Problems? If you encounter problems with installing the needed tools, the exercises, etc. chances are you are not the only one. We recommend you always use the following means to solve your problems in the given order; search the web, lookup in relevant documentation, ask your co-students, ask our helpful teaching assistants during class, ask on Piazza (see first exercise).

Exercises

1 Join Piazza Piazza is a great place to ask questions to co-students when there are no one around to help. So to be prepared for any problems you might encounter, start by joining this course's Piazza page on: `https://piazza.com/dtu.dk/fall2018/02807`.

2 Setting up the UNIX terminal If you are a Mac or Linux user, you already have access to the UNIX terminal. If you are a Windows user, you have different options to get access to a UNIX terminal. You can dual-boot a Linux installation, run a Linux installation in a virtual machine, or use a remote Linux installation[2]. However, we believe the simplest solution needed for our purpose, is to install Ubuntu 18.04 using the Windows Subsystem for Linux - see `https://docs.microsoft.com/en-us/windows/wsl/install-win10` for instructions. In the following we will assume you use either Ubuntu (on Windows) or Mac.

Ensure you have access to the needed tools:

- Open the terminal.
- Verify you can successfully run `python3 --version`. If not, install it - see the following exercise.
- Verify you can successfully run `git --version`. If not, install it - see the following exercise.

3 Installing and using a package In this exercise we will be installing the package `cowsay` using the package manager on your system (if your package manager does not contain this package, choose another package).

- Install `cowsay` using your package manager.
  - Ubuntu: Use `apt-get install cowsay`
  - Mac: Use `brew install cowsay` (if Homebrew is not installed, see `https://brew.sh/`)
- Create a file with some text: `echo What does the cow say? >text.txt`
- Run `cowsay <text.txt` and enjoy the output.
- Try run `cowsay --help` and `man cowsay` to show the help/documentation for the package.
- Using the above, figure out how to change the look of the cow's eyes.

[1] Like, start reading the book, help the beginners, etc.
[2] For instance you can use the gbar: `http://gbar.dtu.dk/faq/53-ssh`

4 Play around with Git At DTU you have access to a GitLab installation using your DTU credentials. In this exercise, you should try to use this Git service.

- Go to `https://gitlab.gbar.dtu.dk` to set up a GIT repository.
- Clone your repository to your local system (use `git clone`).
- Create a few files and stage them (use `git add`).
- Commit your changes (use `git commit`).
- Make some more changes to the files, and show the current status (use `git status`).
- Push your changes to the DTU GitLab repository, and go to the web interface and check the files are now available there (use `git push`).
- Now find another student and invite them to your Git repository. Ask them to clone your repository.
- Both of you, change the contents of the same file. Add, commit and push your changes. The slower one now has to deal with a conflict. Resolve the conflict together.
- Finally ensure both of you have the latest version by pulling them (use `git pull`).

We recommend you use Git to store all files and projects you make in this course in order to get used to Git even if you find it unnecessary. Now would be a good time to create a new and clean repository for this purpose.

5 Creating a bash script When working in the UNIX terminal, it is a great idea to create small scripts that can help solve tasks for you. By making a script file instead of just running the commands directly in the shell, you don't have to remember what you did when you later have to do the same again.

- Create a file called helloworld.sh using your favorite text editor with the content: `echo Hello world`
- Ensure the file can be executed: `chmod +x helloworld.sh`
- Run the script: `./helloworld.sh`

6 Bash challenges Solve each of the following challenges by creating an appropriate bash script. Remember to look in the documentation of the different tools in order to solve the exercises. You can test your solutions on CodeJudge: `https://dtu.codejudge.net/02807-E18/`

6.1 Make a script that given a command line argument X outputs Hello X. Ie. when `./hello.sh World` is executed it prints Hello World (suggested topic to look up: command line arguments).

6.2 Given two command line arguments X and Y, output the number of lines where the text X occurs in the file Y (suggested tools: `grep`, `wc`).

6.3 Given an integer N as command line argument, output the numbers from 1 to N. (suggestion: use a Bash loop)

6.4 Given the filename of a log from an Apache web server as argument, output the most requested addresses in order prefixed by the number of times the address was requested. We base this exercise on the files from `http://www.monitorware.com/en/logsamples/apache.php` (see example data on CodeJudge). (suggested tools: cut, sort, uniq, head)

6.5 Make a script that looks at all files in the current working directory, find and output the name of the one file that contains the string rainbow.