Exercise Sheet #1

Note: This sheet is meant as a fast introduction to Linux. You do **not** have to hand in this sheet, however we highly recommend to work on the exercises, as they provide the fundamentals for all upcoming exercises.

Introduction to Linux For all the commands below, be sure to check its man page, using `man <command>`. If there is something you want to do, and you don't find a command, you should search the internet, chances are there is a UNIX command that does what you want, or someone already had the same problem.

Online terminal To practice Linux commands, you can also try to use an online terminal, e. g. https://bellard.org/jslinux/vm.html?url=https://bellard.org/jslinux/buildroot-x86.cfg. Mind that for some of the exercises you might miss some directories. As you can never be sure who reads the input to the online console do **never** type your password to any online console. (So you should not do problem №4 online.)

Problem 1 (*File system*)

1. Use the comands `cd`, `pwd` and `ls` to explore the filesystem. Try:
   - `cd <dir>` (change to dir)
   - `pwd` (print working directory)
   - `ls` (list files in current directory)
   - `ls -all` (list all properties)
   - `cd .` (change to here)
   - `cd ..` (change one level down)
   - `cd ~` (change to home directory)
   - `cd` (change to home directory)
   - `cd /` (change to the root directory - the equivalent to "C:" in windows)
   - change to some other user directory, for instance to `username` with `cd ~username`

2. Go to the `/etc` directory and see what is there, check the rest of the filesystem tree using `cd`, `ls`, `pwd` and `cat`. Look in `/bin`, `/usr/bin`, `/sbin`, `/tmp` and `/boot`.

3. Go to your home directory and generate a directory called `uni` and `notuni`, with the `mkdir` command. Change to `notuni` and generate a file with `touch newfile`.

4. Copy the file `newfile` to `copyofnewfile` in the directory `notuni` using the command `cp`. Then rename the file by moving the file with the command `mv`.

5. Now go back one level try to delete both directories using `rm` and its options (check `man rm`).

6. What is the difference between listing the contents with `ls -ltr` and `ls -l`, or `ls` (check some of the options, often denoted *flags* listed in `man ls`).

Problem 2 (*Reading from files*)

Navigate to a directory that contains a text file.

1. Show the content of the text file one page at a time using the `less <filename>` command.

2. Show the first and last ten lines of the file using `head <filename>` and `tail <filename>`, respectively.

3. Use `grep <word> <filename>` to search your file for lines containing a word of your choice.

Problem 3 (*Permissions*)

1. Create a file and a directory with permissions `r--r--r--`. Can you change to the directory you created now? (hint: `man chmod`)

2. Modify the permissions on your home directory to make it completely private. Check with some other user that he can't access your directory. Then put the permissions back to how they were. Choose a directory in your home directory and make all the files on it read only.

Problem 4 (*Connecting to other computers*)

1. Log into another machine in the lab. Find out the name of your lab computer (use `hostname` and `domainname`). Go to another machine and then log to the first machine using ssh: (`ssh username@machine.domain`). You can then work in the other computer.

2. Save one step and run a command in the remote computer via `ssh username@machine.domain <command>`. See what happens if you end the connection while the program is running. Try with a graphical program, for instance *Firefox* (you need an extra option for that, see `man ssh`).

Problem 5 (*Pipe operator and* `xargs`)

The pipe operator `|` is used to hand over the output of one command to another command. Navigate to a directory of your choice.

1. Use the `grep` command and pipe operator to filter the list of files by a specific keyword. For example, if you wanted to filter by files containing some `<word>`, you would use the command `ls | grep <word>`.

2. Pipe the output of `ls -lah` into the `less` command.

If the output of a command contains different arguments that you wish to pipe into another command, the `xargs` utility can be useful.

1. Use `printf "test1.txt\ntest2.txt"` to print out two file names separated by a newline (`\n`).

2. Now pipe the output of that command into `xargs` using the following command: `printf "test1.txt\ntest2.txt" | xargs touch`. The `xargs` utility will, for each line of the output, execute a given command (here `touch`) with that line as an argument. If everything worked, two new files `test1.txt` and `test2.txt` were created. Check if that is the case using `ls`.