

# Operating Systems

## Lab 1 Exercise – Files, directories and permissions

Learning goals: practice with the most common commands for interacting with the file system.

Commands: ls, cd, mv, cp, rm, chmod, less, mkdir, wc.

1. Create the following directory tree in your home directory (man `mkdir` for details about how to create a directory).

```
.
├── ex1
│   ├── bin
│   ├── src
│   │   ├── include
│   │   └── lib
│   └── test
│       ├── result
│       └── script
```

Remarks:

- Work on the local disk. Do not use other storage devices;
- Use the TAB key to exploit the auto completion feature of the shell;
- Try `mkdir -p`

2. Download text file `test01.txt` in directory `test`.

1. Copy `test01.txt` to `text01`
2. Create in directory `test` a hard link named `t1` to `text01`
3. Create in directory `bin` a hard link named `t2` to `t1`
4. List all files, and their attributes, in `ex1` and its subdirectories
5. Explain the meaning of the number of links associated to each file or directory.
6. Remove file `text01`, and display the content of `t2`. Does it work? Why?

3. Remove `t1` and `t2`

1. Create in directory `test` a soft link named `t1` to `text01`
2. Create in directory `bin` a soft link named `t2` to `t1`
3. Continue creating soft link `t3` to `t2`, `t4` to `t3`, ... , `ti` to `t(i-1)`. Is there any limit?
4. List all files, and their attributes, in `ex1` and its subdirectories
5. Explain the meaning of the number of links associated to each file.
6. Remove file `text01`, and display the content of `t4`. Does it work? Why?
7. Remove all file `t` in `bin` with a single command

#### 4. Move inside the **src** directory and perform the following actions:

1. Create a simple C program, **main.c**, which displays the number of words of a text file, and copy it in the **src** directory. The filename of the text file must be given as an argument of the command line.
2. Copy the **main.c** file you have just created into directories **test**, **script** and **result** directories staying in the current **src** directory. Once done, verify the existence of the file using the **less** command (**man less** for details about the **less** command) and then delete them without moving from the current directory. Then, move to the parent directory **ex1**, and redo the same actions (copy, verify existence and delete);
3. In 1.1 you used the **cp** command for copying files. What is the difference with the **mv** command? (type **man mv** for more details about the **mv** command)
4. Compile the **main.c** file using the following command line: **gcc -Wall main.c -o myWC**. What's the purpose of the **-Wall** option?
5. Correct any compilation error and verify the existence of the **myWC** executable file.
6. Verify also that the **myWC** permission is set for the file owner. Check that its **x** permission is set.
  1. Run **myWC** without arguments.
  2. Eliminate **myWC x** permission (**man chmod** for details about changing file permissions).
  3. Run **myWC** without arguments. (note the difference)
  4. Restore **x** permission for **myWC**
  5. Once done, move the binary file **myWC** into the **bin** directory;

#### 5. Move inside the **test** directory and perform the following actions:

1. Execute the **myWC** giving as its argument **test01.txt**. The command for running the **myWC** executable is:

```
../bin/myWC test01.txt
```
2. The output should be the number of words in **test01.txt** file. Why do you need to specify a full pathname (**../bin/myWC**) to the executable file in order to execute it?
3. Execute the following command: **wc test01.txt**
4. Compare the output of this command with the output you obtained in step 5.2.  
How we can get **only the number of words** (without the filename, the number of lines and characters) using the command **wc**?