1 Logging into the system

```
% login username
% passwd
% logout
```

Home directory `/home/username`

2 System basics

- graphical vs text environment
- changing terminals :

  Alt –F[1,2,...] – changing text terminals = `chvt num`
  Ctrl- Alt- F[1,2,...] – from graphical to text
  Ctrl- Alt- F7 – from text to graphical

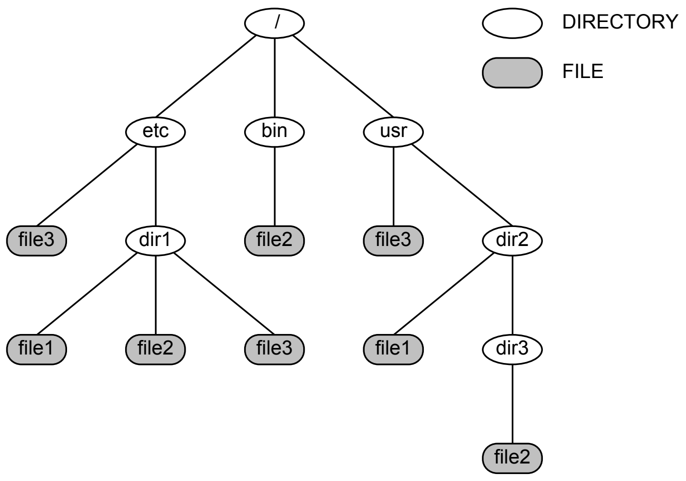
  ↑↓ - latest commands
  Shift –PgUp, PgDn – scrolling the screen forward and backward
  Ctrl –l – emptying the screen

3 Basic information

```
% who am i

% id

% finger username

% logout
```

4 Unix filesystem

The filesystem is constructed in a hierarchical way:

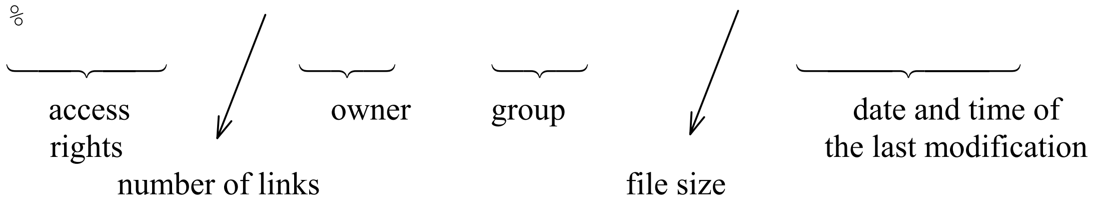


5 File system –basic commands

`pwd`	print working directory – print entire path for current directory on the screen
`mkdir` *dirname*	make a new directory with the name *dirname*
`mkdir –p` *d1/d2/d3*	make a tree of directories
`rmdir` *dirname*	remove the existing empty directory specified by *dirname*
`cd` *dirname*	change the current working directory to *dirname*
`cp` *filename new_destination*	copy *filename* to *new_destination* which can be a name of a copy file or name of an existing directory where the file will be copied with its current name
`mv` *filename new_destination*	moves *filename* to *new_destination*

`ls`	list – list the content of current directory
`ls -l`	list content of current directory in long format

```
% ls -l
total 56
-rw-r--r--   1 darek    student       136 Apr 10 19:16 file1
-rw-r--r--   1 darek    student       136 Apr 10 19:19 file2
-rw-r--r--   1 darek    student       136 Apr 10 19:20 file3
-rw-r--r--   1 darek    student        18 Apr 10 19:25 file4
-rw-r--r--   1 darek    student        13 Apr 10 19:26 script
drwxr-sr-x   2 darek    student       512 Apr 10 19:29 subdir1
drwxr-sr-x   2 darek    student       512 Apr 10 19:30 subdir2
%
```



`ls -l` *filename*	list information about a specified file in long format
`ls` *dirname*	list the content of a directory specified by *dirname*
`ls -al`	list information about all files of the current directory in long format

`rm` *filename*	remove an existing file
`rm -i *`	remove all files in the current directory, but prompt for confirmation before removing any file
`rm -r` *dirname*	remove all files in the specified directory and the directory itself

`%` **`man`** `ls`	manual

```
% LANG=de_DE
% export LANG
```

6 Generalization patterns:

When names of several files have common features then names can be generalized with following patterns:

?	any single sign
*	any sequence of signs, may be empty
[...]	one of signs from braces
[...- ...]	any sequence of signs from the given range
[^...]	every sign except those in brackets

Examples:

?   *   .*   ?[0-9].txt   [aA]*.?   [^0-9]

Exercise:

1. List the content of your `home` directory
2. show content of all files which names end with a number
3. copy directory `subdir2` and files from this directory to `dir2`
4. remove subdirectory `subdir2` from `dir1`
5. remove files which names begin with file and does not contain a number in the name
6. copy `file1` into `file4`

7 Access rights

10 bits :

	user			group			others		
file type	r	w	x	r	w	x	r	w	x

r(read) **4**
w(write) **2**
x(execute) **1**

	File	**Dir**
r	Reading	Coping and listing files
w	Writing	Creating and deleting files
x	Executing	Accessing files

```
chmod   [u g o a ] [ + - = ] [  r w x  ] filename
```

`chmod  - R` changing rights for the directory and its files recursively

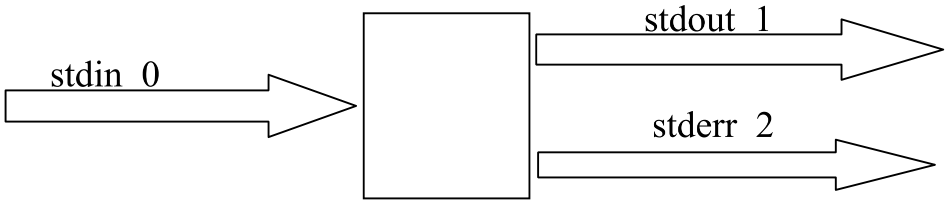
<u>Examples:</u>

```
chmod u+x,g-r file1

chmod 777 file2

chmod ugo-rx file1
```

<u>Exercise:</u>

1. change access rights of `newfile` in such way that the user can write and execute it, group can read it, and others can execute it.

8 Changing output/input



`cat` – types from the keyboard into the screen until ^D

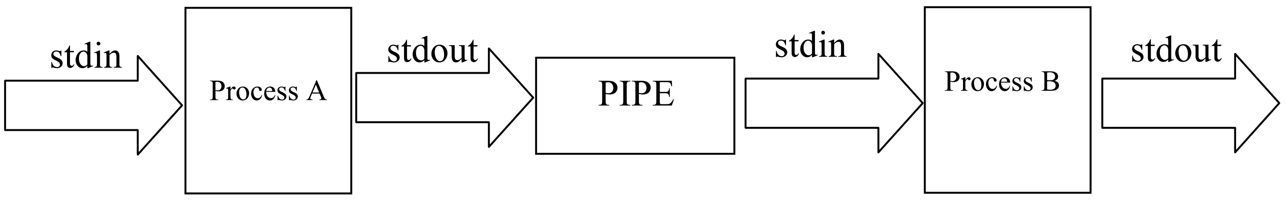
```
cat>filename
cat<filename
cat filename 1 filename 2 filename 3 > outfile
cat < filename 1 > filename 2
cat >> filename
cat<<end
.....
....
..... end   ← does not work
end
^D
```

```
cat f1 f2 f3 2>file_err
cat f1 f2 f3 2> /dev/null
cat f1 f2 f3 2>&1
```

Exercise

1. Show the content of all files from /etc directory, which names begin with s. The information on failures write to the file *error_file*
2. Write into file *f3* the content of files *f1,f2* and the sequence of characters from the keyboard.

9 Pipes



```
%ls -al | more
```
(shows files from the directory side by side)

10 Text file processing commands

`more`	outputs the content of a text file into the terminal screen.

% **`more`** *filename*	– displays the content of the file *filename*
% **`more`** `*txt`	– displays the content all files with names end with `txt`
% **`more`** `-10` *filename*	– displays by 10 lines a screen
% **`more`** `-10` *filename1 filename2*	– as above but subsequently *filename1* and *filename2*
% **`more`** `+40` *filename*	– display begins at line 40

`head`	displays only the beginning of a file content

% **`head`** `-5 *txt`	– displays 5 first lines from each file matching `*txt`

`tail`	displays only the end of a file content

% **`tail`** `-30` *filename*	– displays 30 last lines from the file *filename*

`cat`	displays the content of file/files

% **`cat -s`** *filename*	– gathers following empty lines into single one
% **`cat -n`** *filename*	– numerates all lines
% **`cat -b`** *filename*	– numerates not empty lines

`sort`	sorts data from the file

% **`sort -b`** *filename*	– ignores spaces beginning lines
% **`sort -n`** *filename*	– sorts by numbers
% **`sort -t`** *filename*	– changes separate sign from tab to specified

% **sort -f** *filename* – ignores size of letters
% **sort -r** *filename* – sorts reversely
% **sort +4** *filename* – passes over first 4 columns
% **sort** *filename* **-o** *output_file* –writes results into *output_file*

`uniq` delets recurrent lines from the input data (but does not sort)

% **uniq -u** *filename* – shows unique lines
% **uniq -d** *filename* – shows recurrent lines

`wc` counts words

% **wc -w|c|l** – counts words | charactes | lines

`tr seguence1 seguence2` changes sequence1 into sequence2

% **tr 'a-z' 'A-Z'**
% **tr -d** – truncates the string
% **tr -s ' '** – squeezes signs

\t	tab
\n	new line

% **tr -s ' ' '/t'**

`cut` displays given columns from the text

% **cut -b** *filename* -sign
% **cut -f** *filename* -column
% **cut -d** *filename* -changes separate sign

% **cut -f1,3-5,7 a.txt**

```
% who | sort         (prints sorted list of system users)
% who|cut -f1 -d “  “|sort|uniq|wc -l (???)
% ls -l /usr/bin | sort -bnr +4 | head -5
(???)
```

Exercises:

1) Display the content of file `/etc/passwd` with pages having 5 lines
2) Display 5 first lines of every file in your home directory
3) Display 3rd, 4th and 5th line from file `/etc/passwd`
4) Display the content of `/etc/passwd` file in one line
5) Display the content of a given file in such a way that every word is in new line.
6) Count all files from the directory `/etc` and its `subdirectories`
7) Give the amount of characters from first three lines of file `/etc/passwd`
8) Show files from the current directory, displaying their names in capital letters
9) Show the access rights of files from the current directory, their sizes and names
10) Display the list of files from the current directory sorted by file sizes
11) Give the statistic of access rights (for every access right say how many times it was granted)

11 Process management

`ps` command displays a list of processes executed in current shell

```
% ps
PID   TTY STAT TIME COMMAND
14429  p4 S    0:00 -bash
14431  p4 R    0:00 ps
%
```

terminal   status   time   execut. command   execution

`%ps -l` shows `f`ull information (long format):

```
FLAGS   UID   PID  PPID PRI  NI   SIZE   RSS WCHAN      STA TTY TIME  COMMAND
   100  1002   379   377   0   0   2020   684  c0192be3   S   p0  0:01  -bash
   100  1002  3589  3588   0   0   1924   836  c0192be3   S   p2  0:00  -bash
   100  1002 14429 14427  10   0   1908  1224  c0118060   S   p4  0:00  -bash
100000  1002 14611 14429  11   0    904   516  0          R   p4  0:00  ps -l
%
```

owner   parent process PID   priority   size of text+ data+stack   size in mem.   event for which the process is sleeping   status   terminal   exec. time   execution command

`%ps -ax` `i`nformation about all processes running currently in the system (a – show processes of other users too, x – show processes without controlling terminal)

`kill` command	terminate a process with a given PID sending the `SIGTERM` signal (signal number 15)

```
% kill 14285
% killall console
```

striking ^C key from terminal keyboard - the active shell will send immediately the `SIGINT` signal to all active child processes.

Not all processes can be stopped this way. Some special processes (as shell process) can be killed only by the `SIGKILL` signal (signal number 9)

```
% kill -9 14280
% kill -KILL 14280
```

Exercises:

1) List first 5 users who have the maximal number of running processes
2) Print names of users which have bash process running

12 find directory [criteria]

-name
-type
-size [+ -] n
-user (id lub nazwa)
-group
-newer nazwa_konkretna
-perm

-perm 0060 (exactly 060)
-perm +0060 (those which have either read or write for a group)
-perm –0060 (have read and write for a group)

-a   find –type f –a –size +5 (nie używa się)
-o
        find –name „test2*” –o –name „test3*”
         find \(–name „test2*” –o –name „test3*” \) –type f
- !   find ! –name „test2*”

-exec [ok] ……. {} \;

find –name “test2*” –exec rm –r {} \;

13 grep [options] expression [list of files]

-v	- lines that does not possess the expression
-i	- ignoring small and capital letters
-c	- giving the amount of expression appearance
-n	- prints numbers of lines that possess the pattern
-h	- does not print the name of found files
-r	- recursive search
-l	- shows names of files in which content the expression is found
-L	- shows names of files in which content the expression is not found

expressions:

.	any single sign
[abc]	one of signs from braces
[a-z]	one of signs from range a-z
[^0-9]	every sign except those in brackets
*	repetition (A[a]* stands for A, Aa, Aaa, Aaaaaaaa, itd.)
.*	any sequence of characters
^	line beginning
$	line end