M. Hermanowicz, ICM UW November 19, 2025

Demonstrations and Exercises

1. Fundamentals of shell navigation
   - User account and directories (home, data, scratch space), filesystem hierarchy
   - Running programs – syntax, options, arguments
   - Saving command line history
   - Commands: `pwd`, `ls`, `cd`, `mkdir`, `cp`, `mv`, `rm`, `touch`, `cat`, `less`, `wc`, `du`, `free`, `head`, `tail`, `which`, `wget`, `nl`, `sort`, `w`, `finger`, `man`
   - Variables and environment variables (`export`)
   - Input/Output streams and redirections
   - Pipeline `|`
   - `grep` tool
   - System information (CPU, RAM, disks)
   - Exercises
     (a) Create a working directory (`workdir`) in the home directory and `cd` into it
     (b) List the contents of `/etc`, a save the result in `$HOME/workdir/etc.txt` (use stdout redirection, check the `$HOME` variable with `echo`)
     (c) List all the files and directories in the home directory, including hidden ones (check the syntax with `man ls`), and append the result to the existing `etc.txt` file without overwriting it
     (d) Change the `etc.txt` file name into `list` (no extension), and check the file's type with `file list` instrukcji `file`
     (e) Check the `list` file permissions, and display its content (`cat`, `less`)
     (f) Make a copy of `list` with `cp` (or `cat` with stdout redirection)
     (g) How many CPU cores, RAM and disk space is available in the system? Save that information in a file (without using a text editor)
     (h) How many files are there in `/etc`?
     (i) Display 10 files from `/usr/bin/` that are the largest in size, in a descending order
2. `Vim` text editor
   - Modes of work and navigation (`h`, `j`, `k`, `l`, `w`, `b`)
   - Editing (`i`, `a`, `ESC`), zapis pliku (`:w`, `:wq`, `:q!`)
   - Searching (`/`)
   - *Copy/Cut/Paste* operations (`v`, `y`, `d`, `p`)
   - Useful commands: https://vimcheatsheet.com
   - Other text editors: (`nano`, `Emacs`)
   - Exercises:
     (a) Edit the file `list` file to familiarize with basic navigation
     (b) Delete some lines (`dd`) and check the Undo/Redo instructions(`u`, `Ctrl+r`)
     (c) Search the file for the `bash` string
     (`/`, `n`, `Shift+n`)

(d) Experiment with copying, cutting, and pasting
(e) Find and replace (`:s/wzorzec1/wzorzec2`)
(f) Write your changes and exit the editor (`:wq`)
(g) Test the stream editor (`sed`) to find and replace a string with `s/wzorzec1/wzorzec2/g`

3. Batch execution
   - Differences between interactive and batch execution
   - Interpreted vs. compiled programming languages – handling source code files and running programs
   - `#!/bin/bash` construct – meaning and usage
   - Script files - permission to execute
   - A *Hello world* example
   - Passing and processing arguments and simple user interaction with `read` and variables: `$1`, `$@`, `$#`)
   - Using variables in scripts: expressions with “ and ’)
   - Exercises
     (a) Simple automation with shell loops and cron
     (b) Write a script that will accept a user specified argument as a directory name to create, and create it with an appended current date
4. Data archivisation, compression, and backup
5. Remote work: SSH and alternatives to access an HPC system
6. Terminal multiplexers (GNU Screen, Tmux)
7. HPC clusters and parallel execution: Slurm Scheduling System
   (a) Architecture of an HPC cluster
   (b) Running an interactive HPC session with `srun`
   (c) Example Slurm script (batch execution with Slurm):

```
#/bin/bash -l
#SBATCH -J test
#SBATCH --nodes 1
#SBATCH --ntasks-per-node 1
#SBATCH --mem 1G
#SBATCH --time=1:00:00
#SBATCH -A gXX-YYYY
#SBATCH -p partition-name
(...)
```

8. HPC applications
   (a) Environment Modules (`module avail`). Inspecting the environment variables after loading/unloading a module
   (b) Working with self-compiled applications