Organisation

- Work in a team of 2 or 3 students
- Duration of this lab is 4 periods (2 weeks)

Pedagogical objectives

- Become familiar with I/O redirection
- Become familiar with Unix command pipelines
- Analyse data in text format

Install compiler and compile on Ubuntu

- Install build environment and compiler `sudo apt install build-essential`
- Compile `g++ out.cpp -o out`, *out* is the executable file

Task 1: Exercises on redirection

Compile the following C++ program, called `out.cpp`, into an executable. The program writes a series of `O` characters on the *stdout* stream and a series of `E` characters on the *stderr* stream:

```
#include <iostream>
#include <cstdlib>

using namespace std;

int main() {
    for (int i = 0; i < 5; ++i) {
        cout << "O";
        cerr << "E";
    }
    return EXIT_SUCCESS;
}
```

Run the executable with `./out`

1. Run the following commands and tell where *stdout* and *stderr* are redirected to.

   a. `./out > file` b. `./out 2> file` c. `./out > file 2>&1` d. `./out 2>&1 > file`
   e. `./out &> file`

2. What do the following commands do?

   a. `cat /usr/share/doc/cron/README | grep -i edit`
   b. `./out 2>&1 | grep –i eeeee`
   c. `./out 2>&1 >/dev/null | grep –i eeeee`

3. Write commands to perform the following tasks:

   a. Produce a recursive listing, using `ls`, of files and directories in your home directory, including hidden files, in the file `/tmp/homefileslist`.

   b. Produce a (non-recursive) listing of all files in your home directory whose names end in `.txt`, `.md` or `.pdf`, in the file `/tmp/homedocumentslist`. The

command must not display an error message if there are no corresponding files.

Task 2: Log analysis

In this task you will use command line pipelines to analyse log data of a website. The website of a course at HEIG-VD is hosted on Amazon S3. When a user requests a page from the site or makes another type of access S3 writes a log entry to the log file. The log entry contains information about who made the access, what type of access it was, what page or resource was accessed, etc.

Log files are typically in text format. Each line of the file corresponds to a log entry. A line contains a sequence of fields with values that are separated by a separator character. All lines contain the same sequence of fields.

Download the log file (https://ads.iict.ch/ads_website.log) using curl. The file has been reformatted a bit for this lab (see below). The format of S3 log files is described in detail in the S3 Server Access Log Format. The following is a summary:

- Each time a client sends an HTTP request to an S3 server a line is written to the log. A request is made when students browse the web site, but also when the professor uploads material to the web site.
- Each line has 18 fields. The fields are separated by tabs. (The fields of the original log produced by S3 are separated by spaces, for this lab they have been reformatted to be separated by tabs.)
- The 3rd field contains the time of the access (date and time).
- The 4th field contains the IP address where the request came from.
- The 7th field contains the S3 operation. S3 operations are an extension of the HTTP methods GET, POST, PUT and DELETE.
- The 9th field contains the URI of the request.
- The 10th field contains the HTTP status code of the server's response.
- The 17th field contains the user agent string which identifies the browser used to make the request.

Verify that the fields are indeed separated by tabs by using the `xxd` command to look at the file (look up `xxd` in the manual).

Answer the following question by using the command line and building a pipeline of commands. You can use `cat`, `grep`, `cut`, `tr`, `wc`, `sort`, `uniq`, `head` and `tail`. For each question give the answer and the pipeline you used to arrive at the answer.

1. How many log entries are in the file?
2. How many accesses were successful (server sends back a status of 200) and how many had an error of "Not Found" (status 404)?
3. What are the URIs that generated a "Not Found" response? Be careful in specifying the correct search criteria: avoid selecting lines that happen to have the character sequence `404` in the URI.
4. How many different days are there in the log file on which requests were made?

5. How many accesses were there on 4th March 2021?
6. Which are the three days with the most accesses? Hint: Create first a pipeline that produces a list of dates preceded by the count of log entries on that date.
7. Which is the user agent string with the most accesses?
8. If a web site is very popular and accessed by many people the user agent strings appearing in the server's log can be used to estimate the relative market share of the users' computers and operating systems. How many accesses were done from browsers that declare that they are running on Windows, Linux and Mac OS X (use three commands)?
9. Read the documentation for the `tee` command. Repeat the analysis of the previous question for browsers running on Windows and insert `tee` into the pipeline such that the user agent strings (including repeats) are written to a file for further analysis (the filename should be `useragents.txt`).

As mentioned previously, the log you are analysing in this task was reformatted so that the fields are separated by tabs. A normal web server log typically uses spaces. You can see an example of such a log in the file access.log (https://ads.iict.ch/access.log).

10. Why is the file `access.log` difficult to analyse, consider for example the analysis of question 7, with the commands you have seen so far?

Task 3: Conversion to CSV

In this task you will use the command line to summarize the log entries and convert the summary into a CSV file that can be read by a spreadsheet program.

A CSV (Comma-Separated Values) file is a file containing a table in text format. It starts with a line containing the column names. Each name is speparated by a comma `,`. The remaining lines contain the rows of the table, one line per row. Each line contains the cells of the corresponding row, again separated by commas. Here is an example of a CSV file:

```
Element, Atomic Mass
H, 1.008
He, 4.002602
Li, 6.94
Be, 9.0121831
```

Note: Spreadsheet software typically tolerates spaces between the values and the commas and removes them. In a spreadsheet the order of columns is not important. Depending on the language of your computer, your spreadsheet may use another character than comma `,` to separate the columns, for example Excel in French expects semicolon `;`.

Produce a CSV file named `accesses.csv` that contains for each day (given by its date) the number of accesses on that day. Transfer that file to your workstation and use spreadsheet software to import the CSV file. Plot the data in a graph and produce a file named `accesses.pdf`.

Notes:

- Make sure the spreadsheet software correctly interprets the date fields as dates and not as text.
- The dates in the file will not be continuous, i.e. there are days without any accesses which will not appear in the file. Choose a type of plot appropriate for this case.