Lab06. Regular expressions

Script Languages (INZ004455)

Wojciech Thomas

14th November 2020

1 Learning goals

After this lab you should be able to:

1. Use regular expressions.

2 Exercises

Artifacts to be uploaded: - file: `app6.py`

1. Download config and log files from ePortal.
2. Develop a function that reads configuration from config file.
    1. Exit application if the config file is not present.
    2. Read the content of the config file. Use regular expressions to analyse each line: recognize if it is section header (e.g. `[Display]`) or section content in the form of `<parameter>=<value>` (e.g. `filename=web20200221.log`).
    3. Put values from particular sections:
        - `[Display]` – into a map with display settings,
        - `[LogFile]` – in a variable with filename,
        - `[Config]` – config logging according to settings.
    4. If any of the settings in the default `lab6.config` file is not present set it to arbitrarily choosen default value.
3. Develop a function that reads the content of the log file into memory. If file does not exist exit the application with proper message. Return a data structure containing all log lines.

4. Develop a function that analyses the content of the log file. Parse the content of each line using a regular expression. Use regular expressions to extract: IP address, timestamp, HTTP request header, HTTP status code, size of the response. Return a list containing tuples with extracted data.

5. Develop a function to print all requests sent from the given IP subnet. Hard code IP addres into your application. IP address can be arbitrarily choosen. IP mask length is evaluated in the following manner: your index number modulo 16 plus 8 (e.g. student's index number: 224538, IP mask length: `224538 % 16 + 8 = 18`). Every number of `lines` (defined in the configuration file) stop and ask user to press `Enter` key.

   Extract a code to check if IP address belongs to the given IP subnet into a separate function.

6. Develop a function that counts a total number of bytes sent in response to requests of type `filter` defined in configuration file. Use regular expressions to identify requests. Print type of a request and a total number of bytes. Separate fields by the separator defined in the configuration file.

7. **Finish all other exercises before this exercise!**

   After finishing all other exercises run `pycodestyle app6.py`. Record the output of the first run. Resolve (fix) all encountered problems. Record the output of the last run.

   Append the first run output and the last run output to `app6.py` at the end as a comment.

   Number of issues solved does not influence final grade!