Lab07. Regular expressions.

Script Languages (INZ002025)

Wojciech Thomas

Spring 2023

1 Learning goals

After this lab you should be able to:

1. Use regular expressions.

2 Exercises

Artefacts to be uploaded:

- file: `app7.py`

3 Tasks

1. Download config and log files from ePortal.
2. Develop a function that reads configuration from the downloaded config file. Function should:
    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]` – into a variable with the filename,
        - `[Config]` – config logging according to settings, without storing it in any additional variables.
    4. If any of the settings in the default `lab.config` file is not present, set it to the arbitrarily chosen default value.
3. Develop a function that reads the content of the log file into memory.
    1. If the log file does not exist exit the application with the proper message.
    2. Return a data structure containing all log lines.
4. Create a function to parse a single line of the log. Return the single object representing request.
    1. Use regular expressions to extract fields: IP address, timestamp, HTTP request header, HTTP status code, size of the response.
    2. Convert each field to the appropriate data type (eg. int) if needed.
5. Develop a function that analyses all lines of the log file.

   1. Take a list of lines as an argument.
   2. Return a list of objects representing log entries.

6. Develop a function to print all requests sent from the given IP subnet.

   1. Hardcode arbitrarily chosen IP address of the subnet into your application.

   2. IP address mask length should be evaluated in the following manner:

      Your student index number modulo 16 plus 8 (e.g. student's index number: 224538, IP mask length: `224538 % 16 + 8 = 18`).

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

   4. Every number of `lines` (a value defined in the configuration file) ask a user to press the `Enter` key.

7. Develop a function to print all requests issued by browser of your choice (Chrome, Firefox, Safari etc.).

8. Develop a function that prints a total number of bytes sent in response to requests of the type defined in a configuration file (field `filter`).

   1. Use regular expressions to identify the type of request using the HTTP request header.

   2. Print a type of request and the total number of bytes sent. Separate fields by the separator defined in the configuration file.

9. **Do not continue until your application fulfils all requirements 1-6!**

10. Run `pycodestyle app7.py`. *Save the output of the first run to the text file.*

11. Resolve (fix) all encountered problems. Run `pycodestyle` until there is no error or warning.

12. Save the output of the last run in the text file.

13. Paste both text files at the end of `app7.py` in a comment block.

    The number of issues solved **does not** affect final grade!