Task documentation SYN: Zvýraznění syntaxe in Python 3.4.3 for IPP 2015/2016
Name and surname: Maroš Kopec
Login: xkopec44

Structure

We decided to divide script into two files. First part `syn.py` which is executed and second part `fun.py` which contains all definitions of functions.

Checking arguments

At first script checks if given arguments are valid and parse them into the corresponding object. Here we used standard library `argparse`. Manual pages are well written and there was no problem with parsing arguments.

Opening files

Then the script tries to open specified files. If the file was not specified the script is considering standard input for input file and standard output for output file. However if format file could not be opened or read or is empty the script prints input to output while trying to add `<br />` at the end of each line if it was desired. Input is read to sting which is then stored for further processing.

Parsing format file

When all arguments and files had been processed, the format file is parsed. This is done when function `parse_format()` is called. This function takes one argument which is string containing whole format file.

It divides lines as items into list and each item or line is consists of dictionary with two keys `regex` and `commands`. Where value of regex key is regular expression stored in string and value of commands key is another list containing all format commands. To create this structure we used standard library re which contains functions and objects for work with regular expressions.

This function checks whether or not the syntax of format file is valid except regular expression validity.

Converting regular expressions

Validity of regular expressions are checked during converting from specified format to python recognized syntax. First we check if negation is used over regular expression describing one character after that the regular expression is converted and then we try to compile it which helps catch all the remaining errors such as incomplete parenthesis.

To convert regular expression to python recognized syntax we created finite state machine which takes one character from string as token.

Matching text and inserting tags

Our first plan was to substitute matched text with text wrapped in tags. But this solution could not be done because in next matching the regular expression started to match inserted tags.

To solve this problem we only stored position at which the tags will be inserted and later insert them. Then the script loops through input character by character while counting on which position in string is located and writing each character to new output string. For each position the list with stored positions of tags is searched and if match is found desired tag is wrote to output string.

As final step the script try to add `<br />` at the end of each line if it was desired.