**Search pattern in a file and get line numbers of lines containing the pattern with file name.**

**If will go through a simple algorithm without importing any module just for a simple python programs then approach would be**

Accepted arguments – file path and a pattern **()** to lookup for just an approach

Created an empty list of tuples.

Opened the file at the given path in read-only mode.

Iterates over each line in the file one by one.

For each line, checked if it contains the given pattern or not.

If the line contains the given pattern,

Created a tuple of line number & the line and adds that to a list of tuples.

Returned the list of tuples i.e., matched lines along with line numbers.

But according to the question program need to accept list optional parameters which are mutually exclusive.

After importing modules, then approach would be:

USED ASCII ESCAPE COLOR CODES TO HIGHLIGHT MATCHING TEXT .

Red = '\033[91m'

Green = '\033[92m'

Blue = '\033[94m'

Cyan = '\033[96m'

White = '\033[97m'

Yellow = '\033[93m'

Magenta = '\033[95m'

Grey = '\033[90m'

Black = '\033[90m'

Default = '\033[99m'

ENDC = '\033[0m'

BOLD = '\033[1m'

UNDERLINE ='\033[4m'

I am compiling a regular expression, then using it to search for text that matches that regular expression. I passed all the mentioned parameter and printed the file name and the line number for every match.

I am using those names which is in given assignment as I just gave an input as ASCII and not dealing with encoding. here if any lines matches, I am printing with file name and line number for every match.

Mentioned the format .

Used argparse module to parse arguments

I used the optional parameter, -u ,-c and -m .

Previously I have worked on file handing in python selenium but by this assignment I got to know more about re and argparse module and followed this link which gave me details how to use regular expressions <https://www3.ntu.edu.sg/home/ehchua/programming/howto/Regexe.html>.

<http://python-notes.curiousefficiency.org/en/latest/python3/text_file_processing.html>

I put the for and if else statement in a try block and excepting file not found if it’s not present and here I am doing exception handling.

Here if will want to take the file from command line the we could set default to [sys. stdin] while parsing arguments by importing sys module .

APPROACH:

import re

import argparse

class colored:

YELLOW = '\033[93m'

UNDERLINE = '\033[4m'

END = '\033[0m'

def main(regex, file, underscore, color):

    pattern = re.compile(regex)

    try:

        for i, line in enumerate (open(file, ip="ascii")):

            for match in re.finditer(pattern, line):

                message = "Pattern {} found on file: {} in line {}. The line is: ".format(regex, file, i+1)

                if args.color and args.underscore:

                    message = "Pattern found on file: {} in line {}. The line is: ".format(regex, file, i+1)

                    l = len(line)

                    print(message + colored. YELLOW + line + colored.END, end="")

                    print("                                                                " ,"^" \* l)

                    break

                if args.underscore:

                    l = len(line)

                    print(message + line, end="")

                    print("                                                                " ,"^" \* l)

                    break

                if args.color:

                    print(message + colored.YELLOW + line + colored.END, end="")

                    break

                if args.machine:

                    print("{}:{}:{}".format(file, i+1, line), end="")

                    break

                else:

                    print(message + line, end="")

                    break

    except FileNotFoundError:

        print("File not found, please enter ")

        pass

if \_\_name\_\_ == "\_\_main\_\_":

parser = argparse.ArgumentParser(description='Python regex finder', epilog = './python\_parser.py --regex [pattern] --files [file]')

requiredNamed = parser.add\_argument\_group ('required named arguments')

requiredNamed.add\_argument('-r', '--regex', help='regex pattern', required=True)

parser.add\_argument('-f', '--file', help='file to search pattern inside')

parser.add\_argument('-u', '--underscore', action='store\_true', help=''prints "^" under the matching text’')

parser.add\_argument('-c', '--color', action='store\_true', help='highlights matching text.')

parser.add\_argument('-m', '--machine', action='store\_true', help='generates machine readable output’)

args = parser.parse\_args()

main (args.regex, args.file, args.underscore, args.color)