Домашнее задание № 7. Оболочка и скрипты

29 декабря 2023 г.

Бойко Роман, ИУ9-12Б

Цель работы

Скриптовый язык, на котором будет выполняться лабораторная работа, студентом выбирается самостоятельно. Примеры возможных скриптовых языков: JavaScipt (Node.js), Python, Ruby, Lua, Perl, Racket и т.д.

Реализация

Файл `tree.py`:

```
#!/usr/bin/env python3

import argparse
import os
import sys


def draw_tree(directory, output, indent=''):
    files = os.listdir(directory)
    count = 0
    for file in files:
        count += 1
        path = os.path.join(directory, file)
        if os.path.isdir(path):
            output.write(f'{indent}├── {file}\n')
            if count == len(files):
                draw_tree(path, output, indent + '    ')
            else:
                draw_tree(path, output, indent + '│   ')
        else:
            if count == len(files):
                output.write(f'{indent}└── {file}\n')
            else:
                output.write(f'{indent}├── {file}\n')


def tree(directory, output):
    with (open(output, 'w') if output is not None else sys.stdout) as file:
        file.write(f'{directory}\n')
        draw_tree(directory, file, indent='')


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('-o', '--output_file')
    parser.add_argument('-d', '--directory')
    args = parser.parse_args()

    directory = args.directory if args.directory is not None else '.'
    output = args.output_file if args.output_file is not None else None
    tree(directory, output)


if __name__ == '__main__':
    main()
```

Файл grep.py:

```
#!/usr/bin/env python3

import argparse
import re
import sys


def grep(pattern, files, ignore_case, max_count, line_number):
    count = 0
    for file in files:
        try:
            with (open(file, 'r') if file != sys.stdin else sys.stdin) as f:
                for i, line in enumerate(f, start=1):
                    if ignore_case:
                        if re.search(pattern, line, re.IGNORECASE):
                            count += 1
                            if line_number:
                                print(f'{i}:{line}', end='')
                            else:
                                print(f'{line}', end='')
                    else:
                        if re.search(pattern, line):
```

```
                        count += 1
                        if line_number:
                            print(f'{i}:{line}', end='')
                        else:
                            print(f'{line}', end='')

                if max_count and count >= max_count:
                    return
        except IOError as e:
            print(f'Error: {e}', file=sys.stderr)


def main():
    parser = argparse.ArgumentParser(description='Personal grep utility')
    parser.add_argument('pattern')
    parser.add_argument('-e', dest='pattern_as_expression',
                        action='store_true')
    parser.add_argument('-i', dest='ignore_case', action='store_true')
    parser.add_argument('-m', dest='max_count', type=int)
    parser.add_argument('-n', dest='line_number', action='store_true')
    parser.add_argument('files', nargs='*')

    args = parser.parse_args()

    if args.pattern_as_expression:
        pattern = args.pattern
    else:
        pattern = re.escape(args.pattern)

    files = args.files if args.files else [sys.stdin]
    ignore_case = args.ignore_case
    max_count = args.max_count
    line_number = args.line_number

    grep(pattern, files, ignore_case, max_count, line_number)


if __name__ == '__main__':
    main()
```

Файл `wc.py`:

```
#!/usr/bin/env python3

import argparse
import sys
```

```python
def count_characters(data):
    return len(data)


def count_words(data):
    words = data.split()
    return len(words)


def count_lines(data):
    lines = data.split('n')
    return len(lines)


def count_bytes(data):
    return len(data.encode('utf-8'))


def wc(file):
    with (open(file, 'r') if file != sys.stdin else sys.stdin) as f:
        data = f.read()

    count = {
        'c': count_characters(data),
        'w': count_words(data),
        'l': count_lines(data),
        'm': count_bytes(data)
    }

    return count


def main():
    parser = argparse.ArgumentParser(
        description='Word, character, line, byte count.'
    )
    parser.add_argument('file', nargs='*', help='input file(s)')

    group = parser.add_mutually_exclusive_group()
    group.add_argument('-c', action='store_true', help='print character count')
    group.add_argument('-w', action='store_true', help='print word count')
    group.add_argument('-l', action='store_true', help='print line count')
    group.add_argument('-m', action='store_true', help='print byte count')

    args = parser.parse_args()
    if not args.c and not args.w and not args.l and not args.m:
        args.c = True

    if not args.file:
        count = wc(sys.stdin)
        print_count(count, args)
    else:
        for file in args.file:
            try:
                count = wc(file)
                print_count(count, args)
            except FileNotFoundError:
                print(f'wc: {file}: No such file or directory',
                      file=sys.stderr)


def print_count(count, args):
    if args.c:
        print(count['c'], end=' ')
    if args.w:
        print(count['w'], end=' ')
    if args.l:
        print(count['l'], end=' ')
    if args.m:
        print(count['m'], end=' ')
    print()


if __name__ == '__main__':
    main()
```

Фвйл speller.py:

```
#!/usr/bin/env python3

import argparse


def load_dictionary(file):
    with open(file, 'r', encoding='utf-8') as f:
        dictionary = set(word for line in f for word in line.split())
    return dictionary


def detect_misspelled_words(dictionary, text):
    misspelled_words = []
    lines = text.split('\n')
    for i, line in enumerate(lines):
        words = line.strip().split()
        for j, word in enumerate(words):
            if word.lower() not in dictionary:
                misspelled_words.append((i+1, j+1, word))
    return misspelled_words


def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('dictionary', help='dictionary file')
    parser.add_argument('text', help='text file')
    args = parser.parse_args()

    dictionary = load_dictionary(args.dictionary)

    with open(args.text, 'r', encoding='utf-8') as f:
        text = f.read()

    misspelled_words = detect_misspelled_words(dictionary, text)

    for line, col, word in misspelled_words:
        print(f'{line}, {col}\t{word}')


if __name__ == '__main__':
    main()
```

Тестирование

```
nick@MacBook-Air-Roman hw7 % chmod +x tree.py
nick@MacBook-Air-Roman hw7 % ./tree.py
.
├── speller.py
├── tree.py
├── wc.py
├── example-missprint.txt
├── rep.md
├── dictionary.txt
├── grep.py
├── readme.md
└── tree.txt
```

```
nick@MacBook-Air-Roman hw7 % chmod +x grep.py
nick@MacBook-Air-Roman hw7 % ./grep.py -n -l if *.py
12:        with (open(file, 'r') if file != sys.stdin else sys.stdin) as f:
14:                    if ignore_case:
15:                     if re.search(pattern, line, re.IGNORECASE):
17:                            if line_number:
22:                        if re.search(pattern, line):
24:                            if line_number:
29:                    if max_count and count >= max_count:
47:    if args.pattern_as_expression:
52:    files = args.files if args.files else [sys.stdin]
60:if __name__ == '__main__':
18:            if word.lower() not in dictionary:
40:if __name__ == '__main__':
14:        if os.path.isdir(path):
16:            if count == len(files):
21:            if count == len(files):
28:   with (open(output, 'w') if output is not None else sys.stdout) as file:
39:   directory = args.directory if args.directory is not None else '.'
40:   output = args.output_file if args.output_file is not None else None
44:if __name__ == '__main__':
26:   with (open(file, 'r') if file != sys.stdin else sys.stdin) as f:
53:    if not args.c and not args.w and not args.l and not args.m:
56:    if not args.file:
70:    if args.c:
72:    if args.w:
74:    if args.l:
76:    if args.m:
81:if __name__ == '__main__':
```

```
nick@MacBook-Air-Roman hw7 % chmod +x wc.py
nick@MacBook-Air-Roman hw7 % ./wc.py wc.py
1889
nick@MacBook-Air-Roman hw7 % ./wc.py tree.py -l
45

nick@MacBook-Air-Roman hw7 % chmod +x speller.py
nick@MacBook-Air-Roman hw7 % ./speller.py dictionary.txt example-
missprint.txt
1, 1    genral
1, 2    clawn
2, 1    spoel
```

Вывод

Разобрался с некотороми командными утилитами, понял, как их писать, как они работают. Углубил свои знания в скриптовых языках.