Московский государственный техническийуниверситет им. Н.Э Баумана

Факультет «Информатика и системы управления»
Кафедра ИУ5 «Системы обработки информации и управления»

Курс «Разработка интернет приложений»

Лабораторной работе №3
Вариант 18

Выполнил:
студент группы ИУ5-53Б
Сергеев М.Ю.

Проверил:
преподаватель каф. ИУ5
Гапанюк Ю.Е.

Москва, 2021 г.

Задание:

Задание лабораторной работы состоит из решения нескольких задач.

Файлы, содержащие решения отдельных задач, должны располагаться в пакете lab_python_fp. Решение каждой задачи должно раполагаться в отдельном файле.

При запуске каждого файла выдаются тестовые результаты выполнения соответствующего задания.

Текст программы:

```
from time import sleep
from lab_python_fp.field import field
from lab_python_fp.cm_timer import cm_timer_1, cm_timer_2
from lab_python_fp.gen_random import gen_random
from lab_python_fp.uniq import Unique
import lab_python_fp.sorted

def test_field():
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'color': 'black'}
    ]
    result = field(goods, 'title')
    print(result)
    result = field(goods, 'title', 'price')
    print(result)

def test_gen_random():
    print(gen_random(5, 1, 4))

def test_uniq():
    uniq = Unique([1, 1, 1, 1, 1, 2, 2, 2, 2, 2])
    print([x for x in uniq])

    uniq = Unique(['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B'], ignore_case=True)
    print([x for x in uniq])

    uniq = Unique(['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B'])
    print([x for x in uniq])
```

```
def test_timer():
    with cm_timer_1():
        sleep(5.5)

    with cm_timer_1():
        sleep(3.6)

def main():
   test_field()
   test_gen_random()
   test_uniq()
   test_timer()

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

```python
import time
from contextlib import contextmanager

class cm_timer_1:

    def __init__(self):
        pass

    def __enter__(self):
        self.start_time = time.time()

    def __exit__(self, exc_type, exc_val, exc_tb):
        print("time:", time.time() - self.start_time, "ms")

@contextmanager
def cm_timer_2():
    try:
        start_time = time.time()
        yield start_time
    except OSError:
        print("We had an error!")
    finally:
        print("time:", time.time() - start_time, "ms")
```

```python
def field(items, *args):
    assert len(args) > 0
    result = []

    if len(args) == 1:
        key = args[0]
        for item in items:
            if key in item:
                result.append(item[key])
    else:
        for item in items:
            result.append({key: value for key, value in item.items() if key in args})
    return result
```

```python
import random

def gen_random(num_count, min, max):
    return [random.randint(min, max) for x in range(num_count)]
```

```python
def print_result(func):
    def wrapped(*args, **kwargs):
        print (func.__name__)
        result = func(*args, **kwargs)
        if isinstance(result, dict):
            for key, val in result.items():
                print (key, '=', val)
        elif isinstance(result, list):
            for val in result:
                print (val)
        else:
            print (result)
        return result

    return wrapped

@print_result
def test_1():
    return 1


@print_result
def test_2():
    return 'iu5'


@print_result
def test_3():
    return {'a': 1, 'b': 2}
```

```python
import json
import sys
from field import field
from cm_timer import cm_timer_1, cm_timer_2
from gen_random import gen_random
from uniq import Unique
from print_result import print_result

path = 'data_light.json'

@print_result
def f1(arg):
    return sorted(Unique(field(arg, 'job-name'), ignore_case=True))


@print_result
def f2(arg):
    return list(filter(lambda x: x.startswith(('программист', 'Программист')), arg))


@print_result
def f3(arg):
    return list(map(lambda x: x + ' с опытом Python', arg))


@print_result
def f4(arg):
    return list(map(lambda x: ''.join(x), list(zip(arg, [', зарплата ' + str(x) + ' руб.' for x in gen_random(len(arg
```

```python
data = [4, -30, 100, -100, 123, 1, 0, -1, -4]

if __name__ == '__main__':
    result = sorted(data, reverse=True, key=abs)
    print(result)

    result_with_lambda = sorted(data, reverse=True, key=(lambda x: abs(x)))
    print(result_with_lambda)
```

```python
class Unique(object):
    def __init__(self, items, **kwargs):
        self.checked = []
        self.iter = iter(items)

        if 'ignore_case' in kwargs:
            self.ignore_case = kwargs['ignore_case']
        else:
            self.ignore_case = False

        pass

    def __next__(self):
        current_element = next(self.iter)
        while True:
            stop_for = True
            for element in self.checked:
                if self.ignore_case:
                    try:
                        stop_for = not (element.lower() == current_element.lower())
                    except:
                        stop_for = not (element == current_element)
                else:
                    stop_for = not (element == current_element)
                if not stop_for:
                    break

            if stop_for:
                break
            else:
```

Результат выполнения:

```
['Ковер', 'Диван для отдыха']
[{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха'}]
[3, 2, 1, 1, 2]
[1, 2]
['a', 'b']
['a', 'A', 'b', 'B']
time: 5.500204563140869 ms
time: 3.6001691818237305 ms
```