

Факультет Радиотехнический

Кафедра ИУ5 Системы обработки информации и управления

**Отчет по лабораторной работе №3 по курсу
Базовые компоненты интернет-технологий
" Функциональные возможности языка Python "**

9

(количество листов)

Вариант № 16

Исполнитель

студент группы РТ5-316

Нижаметдинов М.Ш.

“ 16 ” сентября 2021 г.

Проверил

Доцент кафедры ИУ5

Гапанюк Ю.Е.

“ ____ ” _____ 2021 г.

Москва, 2021 г.

Задание

Задание лабораторной работы состоит из решения нескольких задач.

Файлы, содержащие решения отдельных задач, должны располагаться в пакете lab_python_fr. Решение каждой задачи должно располагаться в отдельном файле. При запуске каждого файла выдаются тестовые результаты выполнения соответствующего задания.

Текст программы

Файл «field.py»

```
def field(items, *args):
    assert len(args) > 0

    if len(args) > 1:
        for item in items:
            current_dict = dict()
            for arg in args:
                if item.get(arg, None):
                    current_dict[arg] = item[arg]
            if len(current_dict) > 0:
                yield current_dict
    else:
        for item in items:
            if item.get(args[0], None):
                yield item[args[0]]

def print_generator_output(lst):
    size = len(lst)
    for counter in range(size):
        if counter < size - 1:
            print(lst[counter], end=', ')
        else:
            print(lst[counter])

if __name__ == '__main__':
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}
    ]

    field_gen = field(goods, 'title')
    print_generator_output([item for item in field_gen])

    field_gen = field(goods, 'title', 'price')
    print_generator_output([item for item in field_gen])
```

Файл «gen_random.py»

```
import random

def gen_random(num_count, begin, end):
    random.seed(version=2)
    for i in range(num_count):
        yield random.randint(begin, end)
```

```

if __name__ == '__main__':
    random_gen = gen_random(5, 1, 3)
    for item in random_gen:
        print(item, end=' ')

```

Файл «unique.py»

```

# Итератор для удаления дубликатов
class Unique(object):
    def __init__(self, items, **kwargs):
        self.items = items
        self.items_len = len(items)
        self.current_element = 0

        if len(kwargs) == 0:
            self.flag = False
        else:
            self.flag = kwargs['ignore_case']

        self.no_duplicates = list()

    def __next__(self):
        for i in range(self.current_element, self.items_len):
            if not(self.items[i] in self.no_duplicates or
                str(self.items[i]).lower() in self.no_duplicates and self.flag):
                self.no_duplicates.append(self.items[i])
                return self.no_duplicates[-1]
            self.current_element = i + 1

        if self.current_element == self.items_len:
            raise StopIteration

    def __iter__(self):
        return self

if __name__ == '__main__':
    data_1 = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
    data_2 = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']

    for unique in Unique(data_1):
        print(unique, end=' ')
    print()

    for unique in Unique(data_2, ignore_case=False):
        print(unique, end=' ')
    print()

    for unique in Unique(data_2, ignore_case=True):
        print(unique, end=' ')
    print()

```

Файл «sort.py»

```

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

if __name__ == '__main__':
    result = ...
    print(result)

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

```

Файл «print_result.py»

```

def print_result(func_to_decorate):

    def decorated_func():
        print(func_to_decorate.__name__)
        func_product = func_to_decorate()

        if type(func_product) == list:
            for item in func_product:
                print(item)
        elif type(func_product) == dict:
            for key in func_product.keys():
                print('{} = {}'.format(key, func_product[key]))
        else:
            print(func_product)

    return decorated_func

@print_result
def test_1():
    return 1

@print_result
def test_2():
    return 'iu5'

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

@print_result
def test_4():
    return [1, 2]

if __name__ == '__main__':
    print('!!!!!!!!!!')
    test_1()
    test_2()
    test_3()
    test_4()

```

Файл «cm_timer.py»

```

import time
from contextlib import contextmanager

class cm_timer_1:
    def __enter__(self):
        self.current_time = time.time()

    def __exit__(self, exp_type, exp_value, traceback):
        if exp_type is not None:
            print(exp_type, exp_value, traceback)
        else:
            print('time:', time.time() - self.current_time)

@contextmanager
def cm_timer_2():
    current_time = time.time()
    yield time.time() - current_time

```

```

        print('time:', time.time() - current_time)

if __name__ == '__main__':
    with cm_timer_1():
        time.sleep(1.3)

    with cm_timer_2():
        time.sleep(1.3)

```

Файл «process_data.py»

```

import json
from cm_timer import cm_timer_1
from unique import Unique
from gen_random import gen_random

path = 'data_light.json'
with open(path) as f:
    data = json.load(f)

def print_result(func_to_decorate):

    def decorated_func(arg):
        print(func_to_decorate(arg))
        return func_to_decorate(arg)

    return decorated_func

@print_result
def f1(arg):
    return sorted([unique for unique in Unique([job["job-name"] for job in arg],
ignore_case=True)])

@print_result
def f2(arg):
    return list(filter(lambda x: x.split()[0] == "программист", arg))

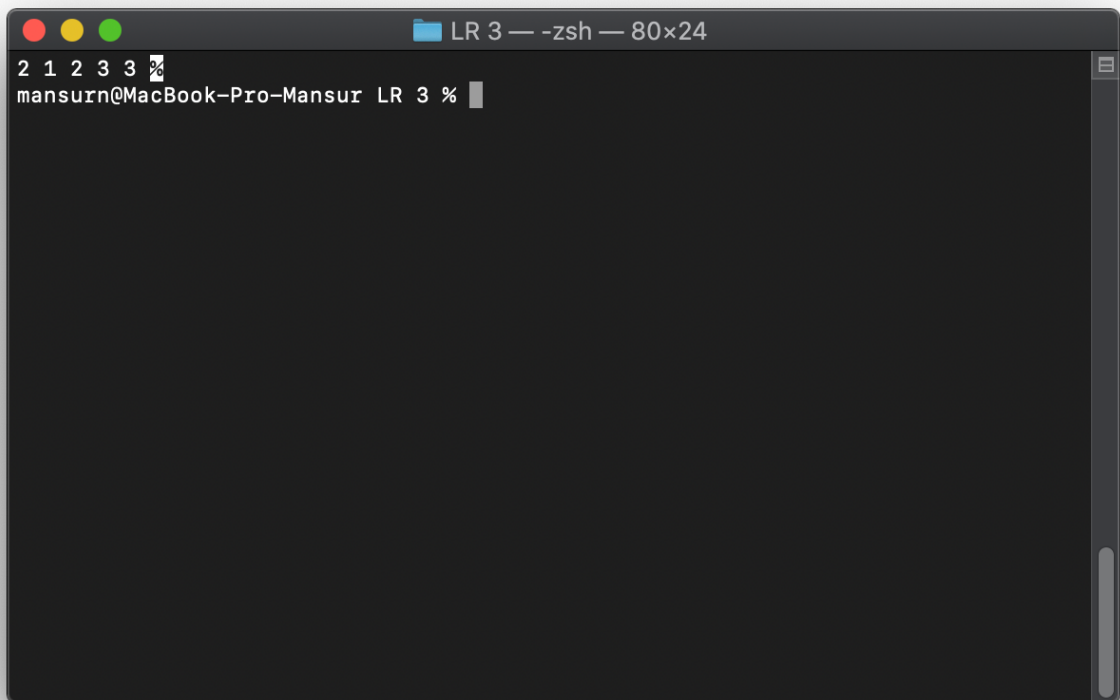
@print_result
def f3(arg):
    return list(map(lambda x: "{} с опытом Python".format(x), arg))

@print_result
def f4(arg):
    return list(map(lambda x: "{} зарплата {} руб.".format(x[0], x[1]),
list(zip(arg, gen_random(len(arg), 100000, 200000)))))

if __name__ == '__main__':
    with cm_timer_1():
        f4(f3(f2(f1(data))))

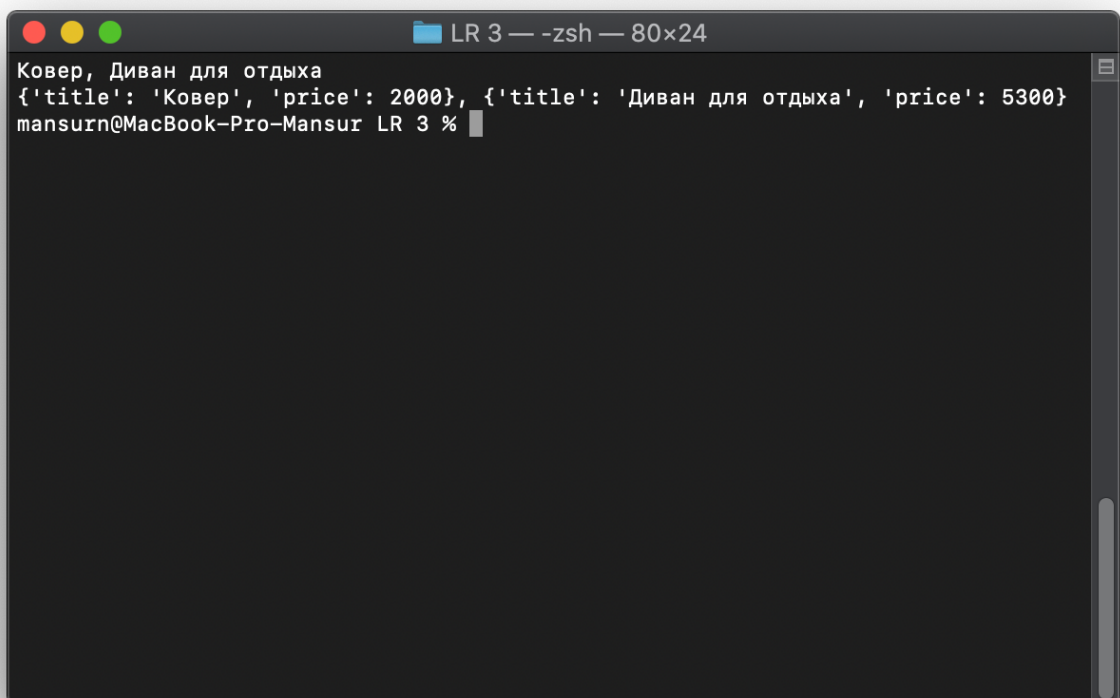
```

Экранные формы с примерами выполнения программы



```
LR 3 — -zsh — 80x24
2 1 2 3 3 %
mansurn@MacBook-Pro-Mansur LR 3 %
```

A terminal window titled "LR 3 — -zsh — 80x24" with a dark background. The prompt shows the command "2 1 2 3 3 %" followed by a cursor. Below it, the prompt "mansurn@MacBook-Pro-Mansur LR 3 %" is visible.



```
LR 3 — -zsh — 80x24
Ковер, Диван для отдыха
{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха', 'price': 5300}
mansurn@MacBook-Pro-Mansur LR 3 %
```

A terminal window titled "LR 3 — -zsh — 80x24" with a dark background. The prompt shows the command "Ковер, Диван для отдыха" followed by a cursor. Below it, the prompt "mansurn@MacBook-Pro-Mansur LR 3 %" is visible. The output of the command is a JSON array: [{"title": "Ковер", "price": 2000}, {"title": "Диван для отдыха", "price": 5300}].

```
LR 3 — -zsh — 80x24
1 2
a A b B
a b
mansurn@MacBook-Pro-Mansur LR 3 %
```

```
LR 3 — -zsh — 80x24
Ellipsis
[123, 100, -100, -30, 4, -4, 1, -1, 0]
mansurn@MacBook-Pro-Mansur LR 3 %
```

```
LR 3 — -zsh — 80x24
!!!!!!!
test_1
1
test_2
iu5
test_3
a = 1
b = 2
test_4
1
2
mansurn@MacBook-Pro-Mansur LR 3 %
```

```
LR 3 — -zsh — 80x24
time: 1.303210973739624
time: 1.3001489639282227
mansurn@MacBook-Pro-Mansur LR 3 %
```



```
LR 3 — -zsh — 110x29
ристав по обеспечению установленного порядка деятельности судов', 'теплотехник', 'тестовод', 'технический спец
иалист', 'ткач', 'токарь', 'токарь 4 разряда', 'токарь-карусельщик', 'токарь-расточник', 'торговый представите
ль', 'тракторист', 'требуются рабочие', 'тренер по конному спорту', 'тренер-преподаватель по плаванию', 'трене
р-преподаватель по спорту', 'трубопроводчик судовой', 'уборщик', 'уборщик мест общего пользования', 'уборщик п
омещения', 'уборщик производственных и служебных помещений', 'уборщик производственных помещений', 'уборщик сл
ужебных и производственных помещений', 'уборщик служебных помещений', 'уборщик территории', 'уборщица', 'упако
вщик', 'участковый врач терапевт', 'участковый геолог', 'учитель обществознания, право', 'учитель - логопед',
'учитель - технологии', 'учитель английского языка', 'учитель английского языка и информатики', 'учитель биол
огии и географии', 'учитель иностранного языка (английский)', 'учитель иностранных языков', 'учитель информати
ки', 'учитель информатики и физики (или математика+ информатика)', 'учитель истории', 'учитель истории и общес
твознания', 'учитель математики', 'учитель математики, информатики', 'учитель математики, информатики и ИКТ',
'учитель начальных классов', 'учитель русского языка и литературы', 'учитель технологии', 'учитель технологии
(мальчики), ОБЖ, английского языка', 'учитель физики и информатики', 'учитель химии', 'Федеральное государствен
ное унитарное предприятие "Российская телевизионная и радиовещательная сеть', 'фельдшер', 'фельдшер - лаборан
т бактериологической лаборатории', 'фельдшер ФАП', 'фельдшер отделения скорой медицинской помощи', 'фельдшер п
оликлиники', 'фельдшер скорой медицинской помощи', 'фельдшер скорой помощи', 'фельдшер фельдшерско - акушерско
го пункта', 'фельдшер-лаборант', 'физик-эксперт', 'формовщик', 'фрезеровщик', 'фтизиатрия', 'химик', 'химик-эк
сперт', 'художественный руководитель', 'художник', 'художник-постановщик', 'швея - мотористка', 'шеф-повар', '
шиномонтаж', 'шлифовщик 5 разряда', 'шлифовщик механического цеха', 'эколог', 'экономист', 'электрик', 'электр
огазосварщик', 'электромонтер', 'электромонтер -линейщик по монтажу воздушных линий высокого напряжения и конт
актной сети', 'электромонтер по испытаниям и измерениям 4-6 разряд', 'электромонтер по ремонту и обслуживанию
электрооборудования', 'электромонтер станционного телевизионного оборудования', 'электросварщик', 'электросвар
щик на автоматических и полуавтоматических машинах', 'энтомолог', 'юрисконсульт', 'юрисконсульт 2 категории',
'юрист']
['программист', 'программист 1С']
['программист с опытом Python', 'программист 1С с опытом Python']
['программист с опытом Python, зарплата 185835 руб.', 'программист 1С с опытом Python, зарплата 173022 руб.']
time: 0.27234411239624023
mansurn@MacBook-Pro-Mansur LR 3 %
```