

**Московский государственный технический
университет им. Н.Э. Баумана**

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

Курс «Базовые компоненты интернет-технологий»

Отчет по лабораторной работе №3-4
«Функциональные возможности языка Python.»

Выполнил:
студент группы ИУ5-34Б

Новиков Богдан

Проверил:
преподаватель каф.
ИУ5

Гапанюк Ю. Е.

2022 г.

Задание:

1. Задание лабораторной работы состоит из решения нескольких задач.
2. Файлы, содержащие решения отдельных задач, должны располагаться в пакете lab_python_fr. Решение каждой задачи должно располагаться в отдельном файле.
3. При запуске каждого файла выдаются тестовые результаты выполнения соответствующего задания.

Текст программы.

Cm_timer.py

```
from time import sleep, time
from contextlib import contextmanager

class cm_timer_1:
    def __init__(self):
        self.start = -1
        self.stop = -1
        self.diff = -1

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

    def __exit__(self, exc_type, exc_val, exc_tb):
        self.stop = time()
        self.diff = self.stop - self.start

    print('time:', self.diff)

@contextmanager
def cm_timer_2():
    start = time()

    yield

    print('time:', time() - start)

with cm_timer_1():
    sleep(1.5)

with cm_timer_2():
```

```
sleep(1.5)
```

field.py

```
def field(items, *args):
```

```
    assert len(args) > 0
```

```
    for item in items:
```

```
        d = {arg: item.get(arg) for arg in args if item.get(arg)}
```

```
        if len(d) == 0:
```

```
            continue
```

```
        if len(args) == 1:
```

```
            yield d[args[0]]
```

```
        else:
```

```
            yield d
```

gen_random.py

```
from random import randint
```

```
def gen_random(num_count, begin, end):
```

```
    return (randint(begin, end) for _ in range(num_count))
```

print_result.py

```
def print_result(func):
```

```
    def wrapper(*args, **kwargs):
```

```
        res = func(*args, **kwargs)
```

```
        if type(res) == list:
```

```
            print(*res, sep='\n')
```

```
        elif type(res) == dict:
```

```
            [print(key, '=', value) for key, value in res.items()]
```

```
        else:
```

```
            print(res)
```

```
    return res
```

```
    return wrapper
```

```
@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()
```

sort.py

```
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 el: abs(el))
print(result_with_lambda)
```

unique.py

```
class Unique(object):
    def __init__(self, items, ignore_case=False, **kwargs):
        if ignore_case == True:
            copy_items = [item for item in items]
            lower_items = [item.lower() for item in copy_items]
```

```

        self.all_items = iter(lower_items)
    else:
        self.all_items = iter(items)

    self.uniq_items = set()
    self.ignore_case = ignore_case

    def __next__(self):
        while True:
            next_item = next(self.all_items)
            if next_item not in self.uniq_items:
                self.uniq_items.add(next_item)
                return next_item

    def __iter__(self):
        return self

```

process_data.py

```

import json
import sys
from field import field
from gen_random import gen_random
from unique import Unique
from print_result import print_result
from cm_timer import cm_timer_1

path = sys.argv[1]

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

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

@print_result
def f2(jobs):

```

```
return filter(lambda job: job.startswith('программист'), jobs)
```

```
@print_result
```

```
def f3(jobs):
```

```
    return map(lambda job: job + ' с опытом Python', jobs)
```

```
@print_result
```

```
def f4(jobs):
```

```
    jobs = list(jobs)
```

```
    salary = gen_random(len(jobs), 100000, 200000)
```

```
    return [job + f', зарплата {salary} руб' for salary, job in zip(salary, jobs)]
```

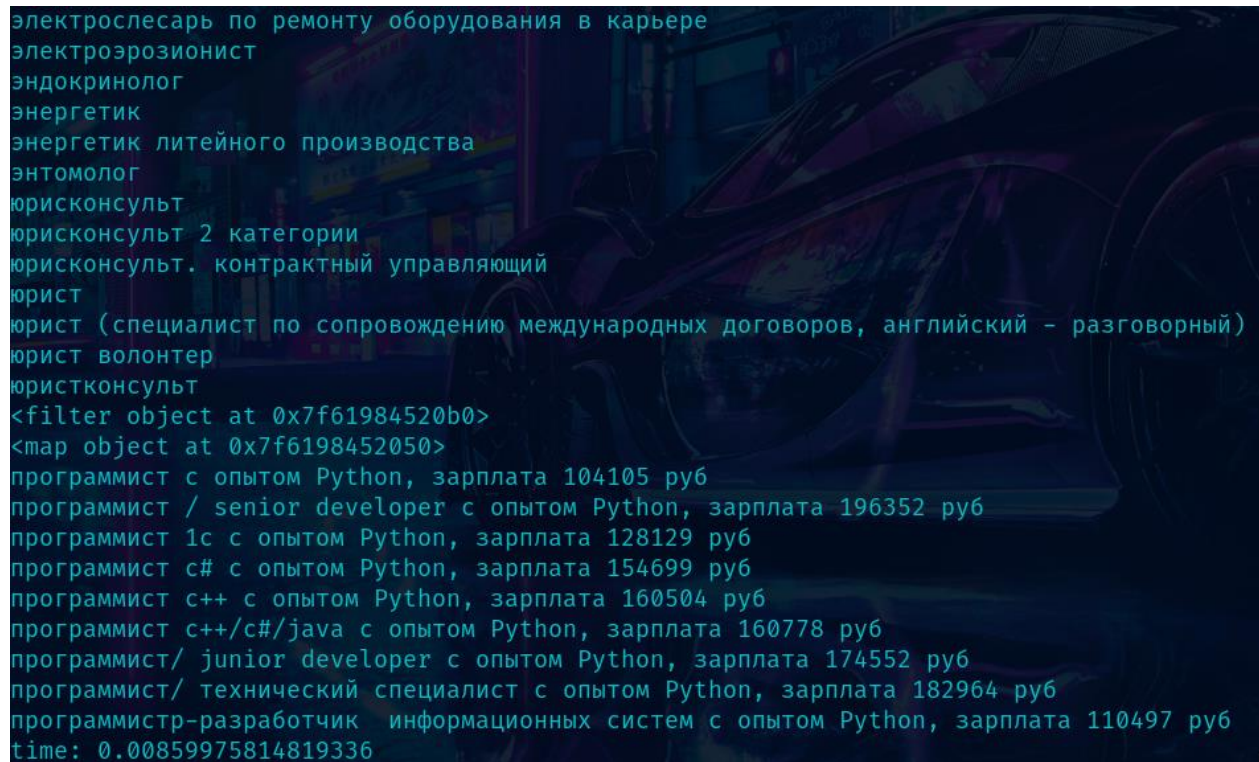
```
if __name__ == '__main__':
```

```
    with cm_timer_1():
```

```
        f4(f3(f2(f1(data))))
```

Результаты выполнения программы:

```
→ lab3-4 git:(lab3-4) x python lab_python_fp/process_data.py lab_python_fp/data_light.json
time: 1.5015370845794678
time: 1.5014197826385498
1с программист
2-ой механик
3-ий механик
4-ый механик
4-ый электромеханик
[химик-эксперт
asic специалист
javascript разработчик
rtl специалист
web-программист
web-разработчик
автожестящик
автоинструктор
автомаляр
автомойщик
автор студенческих работ по различным дисциплинам
автослесарь
автослесарь - моторист
автоэлектрик
агент
агент банка
агент нпф
агент по гос. закупкам недвижимости
агент по недвижимости
```



электрослесарь по ремонту оборудования в карьере
электроэрозионист
эндокринолог
энергетик
энергетик литейного производства
энтомолог
юрисконсульт
юрисконсульт 2 категории
юрисконсульт. контрактный управляющий
юрист
юрист (специалист по сопровождению международных договоров, английский – разговорный)
юрист волонтер
юристконсульт
<filter object at 0x7f61984520b0>
<map object at 0x7f6198452050>
программист с опытом Python, зарплата 104105 руб
программист / senior developer с опытом Python, зарплата 196352 руб
программист 1с с опытом Python, зарплата 128129 руб
программист с# с опытом Python, зарплата 154699 руб
программист с++ с опытом Python, зарплата 160504 руб
программист с++/с#/java с опытом Python, зарплата 160778 руб
программист/ junior developer с опытом Python, зарплата 174552 руб
программист/ технический специалист с опытом Python, зарплата 182964 руб
программист-разработчик информационных систем с опытом Python, зарплата 110497 руб
time: 0.00859975814819336