

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

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

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

Выполнил:
Сироткин Сергей
ИУ5-35Б

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.

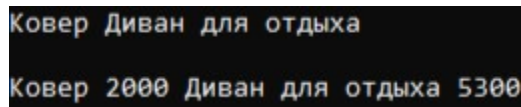
Задача 1 (файл field.py)

Необходимо реализовать генератор field. Генератор field последовательно выдает значения ключей словаря.

```
def field(items, *args):
    assert len(args) > 0
    c_i = len(items)
    c_a = len(args)
    for i in range(c_i):
        for j in range(c_a):
            if args[j] in items[i] and args[j] is not None:
                yield items[i][args[j]]

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

    for ii in field(goods, 'title'):
        print(ii, end=' ')
    print('\n')
    for jj in field(goods, 'title', 'price'):
        print(jj, end=' ')
    print('\n')
```



```
Ковер Диван для отдыха
Ковер 2000 Диван для отдыха 5300
```

Задача 2 (файл gen_random.py)

Необходимо реализовать генератор gen_random(количество, минимум, максимум), который последовательно выдает заданное количество случайных чисел в заданном диапазоне от минимума до максимума, включая границы диапазона.

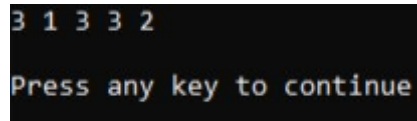
```

import random
def gen_random(num_count, begin, end):
    for i in range(num_count):
        yield random.randint(begin, end)

def main():
    for num in gen_random(5, 1, 3):
        print(num, end=' ')
    print('\n')

    main()

```



```

3 1 3 3 2
Press any key to continue

```

Задача 3 (файл unique.py)

- Необходимо реализовать итератор Unique(данные), который принимает на вход массив или генератор и итерируется по элементам, пропуская дубликаты.
- Конструктор итератора также принимает на вход именованный bool-параметр ignore_case, в зависимости от значения которого будут считаться одинаковыми строки в разном регистре. По умолчанию этот параметр равен False.
- При реализации необходимо использовать конструкцию ****kwargs**.
- Итератор должен поддерживать работу как со списками, так и с генераторами.
- Итератор не должен модифицировать возвращаемые значения.

```

from gen_random import gen_random
class unique(object):
    def __init__(self, items, ignore_case=False, **kwargs):
        self.ignore_case = ignore_case
        self.items = items
        self.index = 0
        self.unique_list = []
        self.seen = set()

    def __next__(self):
        while self.index < len(self.items):
            item = self.items[self.index]
            self.index += 1

            if self.ignore_case:
                item = item.lower()

            if item not in self.seen:
                self.seen.add(item)
                return item

        raise StopIteration()

```

```

def __iter__(self):
    return self
if __name__ == '__main__':
    data1 = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
    unique_data1 = unique(data1)
    for item in unique_data1:
        print(item, end=' ')
    print('\n')
    data2 = []
    for num in gen_random(5, 1, 3):
        data2.append(num)
    unique_data2 = unique(data2)
    for item in unique_data2:
        print(item, end=' ')
    print('\n')
    data3 = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
    unique_data3 = unique(data3, ignore_case=True)
    for item in unique_data3:
        print(item, end=' ')
    print('\n')
    data4 = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
    unique_data4 = unique(data3)
    for item in unique_data4:
        print(item, end=' ')

    print('\n')

```

```

1 2
1
a b
a A b B

```

Задача 4 (файл sort.py)

Дан массив 1, содержащий положительные и отрицательные числа. Необходимо **одной строкой кода** вывести на экран массив 2, который содержит значения массива 1, отсортированные по модулю в порядке убывания. Сортировку необходимо осуществлять с помощью функции `sorted`.

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

```

if __name__ == '__main__':
    def abs_key(number):
        return abs(number)

    result = sorted(data, key = abs_key, reverse = True)
    print(result)

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

```

```

[123, 100, -100, -30, 4, -4, 1, -1, 0]
[123, 100, -100, -30, 4, -4, 1, -1, 0]
Press any key to continue . . .

```

Задача 5 (файл print_result.py)

Необходимо реализовать декоратор `print_result`, который выводит на экран результат выполнения функции.

- Декоратор должен принимать на вход функцию, вызывать её, печатать в консоль имя функции и результат выполнения, после чего возвращать результат выполнения.
- Если функция вернула список (`list`), то значения элементов списка должны выводиться в столбик.
- Если функция вернула словарь (`dict`), то ключи и значения должны выводиться в столбик через знак равенства.

```
import inspect
def print_result(func):
    def wrapper(*args):
        print(func.__name__)
        result = func(*args)
        if isinstance(result, list) or inspect.isgenerator(result):
            for el in result:
                print(el)
        elif isinstance(result, dict) :
            for key in result:
                print(key, "=", result[key])
        else:
            print(result)
    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()
```

```
!!!!!!!
test_1
1
test_2
iu5
test_3
a = 1
b = 2
test_4
1
2
```

Задача 6 (файл cm_timer.py)

Необходимо написать контекстные менеджеры cm_timer_1 и cm_timer_2, которые считают время работы блока кода и выводят его на экран.

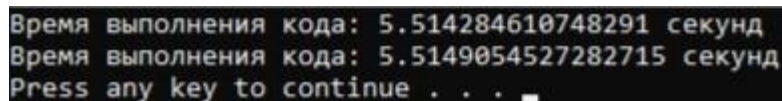
```
import time
from contextlib import contextmanager

class cm_timer_1:
    def __enter__(self):
        self.start_time = time.time()
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        end_time = time.time()
        print("Время выполнения кода:", end_time - self.start_time, "секунд")

@contextmanager
def cm_timer_2():
    start_time = time.time()
    yield
    end_time = time.time()
    print("Время выполнения кода:", end_time - start_time, "секунд")

if __name__ == '__main__':
    with cm_timer_1():
        time.sleep(5.5)
    with cm_timer_2():
        time.sleep(5.5)
```



```
Время выполнения кода: 5.514284610748291 секунд
Время выполнения кода: 5.5149054527282715 секунд
Press any key to continue . . . _
```

Задача 7 (файл process_data.py)

- В предыдущих задачах были написаны все требуемые инструменты для работы с данными. Применим их на реальном примере.
- В файле data_light.json содержится фрагмент списка вакансий.
- Структура данных представляет собой список словарей с множеством полей: название работы, место, уровень зарплаты и т.д.
- Необходимо реализовать 4 функции - f1, f2, f3, f4. Каждая функция вызывается, принимая на вход результат работы предыдущей. За счет декоратора @print_result печатается результат, а контекстный менеджер cm_timer_1 выводит время работы цепочки функций.
- Предполагается, что функции f1, f2, f3 будут реализованы в одну строку. В реализации функции f4 может быть до 3 строк.

- Функция f1 должна вывести отсортированный список профессий без повторений (строки в разном регистре считать равными). Сортировка должна игнорировать регистр. Используйте наработки из предыдущих задач.
- Функция f2 должна фильтровать входной массив и возвращать только те элементы, которые начинаются со слова “программист”. Для фильтрации используйте функцию filter.
- Функция f3 должна модифицировать каждый элемент массива, добавив строку “с опытом Python” (все программисты должны быть знакомы с Python). Пример: Программист C# с опытом Python. Для модификации используйте функцию map.
- Функция f4 должна сгенерировать для каждой специальности зарплату от 100 000 до 200 000 рублей и присоединить её к названию специальности. Пример: Программист C# с опытом Python, зарплата 137287 руб. Используйте zip для обработки пары специальность — зарплата.

```
import json
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 = 'lab3-4/data_light.json'
with open(path, encoding="utf-8") as f:
    data = json.load(f)

@print_result
def f1(arg):
    return sorted(unique(list(field(data, 'job-name'))), True))

@print_result
def f2(arg):
    return list(filter(lambda x: 'программист' in x, field(data, 'job-name')))

@print_result
def f3(arg):
    return list(map(lambda x: x + ' с опытом Python', filter(lambda x: 'программист'
in x, field(data, 'job-name'))))

@print_result
def f4(arg):
    for job in list(filter(lambda x: 'программист' in x, field(data, 'job-name'))):
        salary = list(gen_random(1, 100000, 200000))
        yield job + ", зарплата " + str(salary[0]) + " руб."

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

f1
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
автожестянщик
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
агент по недвижимости (стажер)
агент по недвижимости / риэлтор
агент по привлечению юридических лиц

f2
Системный программист (C, Linux)
Веб-программист
1С программист
программист
Инженер-программист ККТ
инженер - программист
Инженер-программист (Клинский филиал)
Инженер-программист (Орехово-Зуевский филиал)
Ведущий программист
Инженер - программист АСУ ТП
инженер-программист
программист
инженер-программист
Инженер-электронщик (программист АСУ ТП)
Веб-программист
Старший программист
Web-программист
Веб - программист (PHP, JS) / Web разработчик
программист
программист 1С
Инженер-программист 1 категории
Ведущий инженер-программист
Инженер-программист САПОУ (java)
Помощник веб-программиста
веб-программист
веб-программист
педагог программист
Инженер-программист ПЛИС
Инженер-программист

f3

Системный программист (C, Linux) с опытом Python
Веб-программист с опытом Python
1С программист с опытом Python
программист с опытом Python
Инженер-программист ККТ с опытом Python
инженер - программист с опытом Python
Инженер-программист (Клинский филиал) с опытом Python
Инженер-программист (Орехово-Зуевский филиал) с опытом Python
Ведущий программист с опытом Python
Инженер - программист АСУ ТП с опытом Python
инженер-программист с опытом Python
программист с опытом Python
инженер-программист с опытом Python
Инженер-электронщик (программист АСУ ТП) с опытом Python
Веб-программист с опытом Python
Старший программист с опытом Python
Web-программист с опытом Python
Веб - программист (PHP, JS) / Web разработчик с опытом Python
программист с опытом Python
программист 1С с опытом Python
Инженер-программист 1 категории с опытом Python
Ведущий инженер-программист с опытом Python
Инженер-программист САПОУ (java) с опытом Python
Помощник веб-программиста с опытом Python
веб-программист с опытом Python
веб-программист с опытом Python
педагог программист с опытом Python
Инженер-программист ПЛИС с опытом Python
Инженер-программист с опытом Python

f4

Системный программист (C, Linux), зарплата 154336 руб.
Веб-программист, зарплата 171752 руб.
1С программист, зарплата 167729 руб.
программист, зарплата 123976 руб.
Инженер-программист ККТ, зарплата 154783 руб.
инженер - программист, зарплата 107575 руб.
Инженер-программист (Клинский филиал), зарплата 181107 руб.
Инженер-программист (Орехово-Зуевский филиал), зарплата 184461 руб.
Ведущий программист, зарплата 158629 руб.
Инженер - программист АСУ ТП, зарплата 129965 руб.
инженер-программист, зарплата 123873 руб.
программист, зарплата 144496 руб.
инженер-программист, зарплата 192899 руб.
Инженер-электронщик (программист АСУ ТП), зарплата 117848 руб.
Веб-программист, зарплата 120786 руб.
Старший программист, зарплата 177386 руб.
Web-программист, зарплата 157150 руб.
Веб - программист (PHP, JS) / Web разработчик, зарплата 138440 руб.
программист, зарплата 164299 руб.
программист 1С, зарплата 142924 руб.
Инженер-программист 1 категории, зарплата 167756 руб.
Ведущий инженер-программист, зарплата 147066 руб.
Инженер-программист САПОУ (java), зарплата 191274 руб.
Помощник веб-программиста, зарплата 135193 руб.
веб-программист, зарплата 116186 руб.
веб-программист, зарплата 193106 руб.
педагог программист, зарплата 163766 руб.
Инженер-программист ПЛИС, зарплата 108348 руб.