

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

Факультет «Информатика и системы управления»

Кафедра ИУ5. Курс «Парадигмы и конструкции языков программирования»

Отчет по лабораторной работе №4
«Функциональные возможности языка Python ч.2»

Выполнил:

Макеева Е. А.
ИУ5-31Б

Подпись и дата:

Проверил:

Нардид А. Н.

Подпись и дата:

Постановка задачи

Задание лабораторной работы состоит из решения нескольких задач.

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

При запуске каждого файла выдаются тестовые результаты выполнения соответствующего задания.

Задача 4

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. Пример:

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

```
Вывод: [123, 100, -100, -30, 30, 4, -4, 1, -1, 0]
```

Необходимо решить задачу двумя способами:

1. С использованием lambda-функции.
2. Без использования lambda-функции.

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

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

```
D:\3_sem\venv\Scripts\python.exe D:/3_sem/lab_python_fp/sort.py
```

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

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

```
Process finished with exit code 0
```

Задача 5

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.

```
def print_result(func):  
    def wrapper(*args, **kwargs):  
        result = func(*args, **kwargs)  
        print(func.__name__)  
  
        if isinstance(result, list):  
            for item in result:  
                print(item)
```

```

        elif isinstance(result, dict):
            for key, value in result.items():
                print(f"{key} = {value}")
        else:
            print(result)

    return result

return wrapper

@print_result
def f_1():
    return 1

@print_result
def f_2():
    return 'iu5'

@print_result
def f_3():
    return {'a': 1, 'b': 2}

@print_result
def f_4():
    return [1, 2]

if __name__ == '__main__':
    print('!!!!!!!')
    f_1()
    f_2()
    f_3()
    f_4()

```



```

D:\3_sem\venv\Scripts\python.exe D:/3_sem/lab_python_fp/print_result.py
!!!!!!!
f_1
1
f_2
iu5
f_3
a = 1
b = 2
f_4
1
2

Process finished with exit code 0

```

Задача 6

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

```
with cm_timer_1():
```

```
    sleep(5.5)
```

После завершения блока кода в консоль должно вывестись `time: 5.5` (реальное время может несколько отличаться).

`cm_timer_1` и `cm_timer_2` реализуют одинаковую функциональность, но должны быть реализованы двумя различными способами (на основе класса и с использованием библиотеки `contextlib`).

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

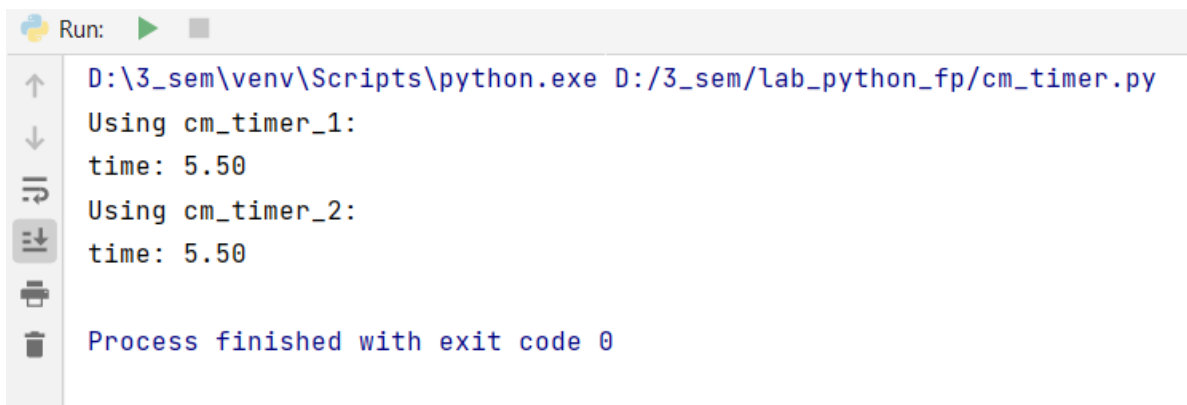
class cm_timer_1:
    def __enter__(self):
        self.start_time = time()
        return self

    def __exit__(self, exception_type, exception_value, traceback):
        self.end_time = time()
        elapsed_time = self.end_time - self.start_time
        print(f'time: {elapsed_time:.2f}')

@contextmanager
def cm_timer_2():
    start_time = time()
    try:
        yield
    finally:
        end_time = time()
        elapsed_time = end_time - start_time
        print(f'time: {elapsed_time:.2f}')

if __name__ == '__main__':
    print("Using cm_timer_1:")
    with cm_timer_1():
        sleep(5.5)

    print("Using cm_timer_2:")
    with cm_timer_2():
        sleep(5.5)
```



```
Run: [icon] [icon]
D:\3_sem\venv\Scripts\python.exe D:/3_sem/lab_python_fp/cm_timer.py
Using cm_timer_1:
time: 5.50
Using cm_timer_2:
time: 5.50

Process finished with exit code 0
```

Задача 7

- 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 повторов (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 С# 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 С# с опытом Python, зарплата 137287 руб. Используйте `zip` для обработки пары специальность — зарплата.

```
import json
import random
from print_result import print_result
from cm_timer import cm_timer_1

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

@print_result
def f1(arg):
    return sorted(set(job['job-name'].strip().lower() for job in arg))

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

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

```

@print_result
def f4(arg):
    salaries = random.sample(range(100000, 200001), len(arg))
    return [f"{profession}, зарплата {salary} руб." for profession, salary in
zip(arg, salaries)]

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
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
агент по недвижимости (стажер)
агент по недвижимости / риэлтор
агент по привлечению юридических лиц
агент по продажам (интернет, тв, телефония) в пао ростелеком в населенных пунктах амурской области: г. благовещенск, г. белогорск, г. свободный, г. шимановск, г. зeya, г. тында

```

И Т. Д.

```

юристконсульт
f2
программист
программист / senior developer
программист 1с
программист с#
программист с++
программист с++/с#/java
программист/ junior developer
программист/ технический специалист
программистр-разработчик информационных систем
f3
программист с опытом Python
программист / senior developer с опытом Python
программист 1с с опытом Python
программист с# с опытом Python
программист с++ с опытом Python
программист с++/с#/java с опытом Python
программист/ junior developer с опытом Python
программист/ технический специалист с опытом Python
программистр-разработчик информационных систем с опытом Python

```

f4

программист с опытом Python, зарплата 155363 руб.

программист / senior developer с опытом Python, зарплата 151519 руб.

программист 1с с опытом Python, зарплата 122858 руб.

программист с# с опытом Python, зарплата 199611 руб.

программист с++ с опытом Python, зарплата 124126 руб.

программист с++/с#/java с опытом Python, зарплата 152317 руб.

программист/ junior developer с опытом Python, зарплата 105905 руб.

программист/ технический специалист с опытом Python, зарплата 187006 руб.

программист-разработчик информационных систем с опытом Python, зарплата 184881 руб.

time: 0.07