

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

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

Курс «Парадигмы и конструкции языков программирования»

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

Выполнил:

студент группы ИУ5-31Б
Орлова Анна

Подпись и дата:

Проверил:

преподаватель каф. ИУ5

Подпись и дата:

Москва, 2024 г.

Задание:

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.

- В качестве первого аргумента генератор принимает список словарей, дальше через *args генератор принимает неограниченное количество аргументов.
- Если передан один аргумент, генератор последовательно выдает только значения полей, если значение поля равно None, то элемент пропускается.
- Если передано несколько аргументов, то последовательно выдаются словари, содержащие данные элементы. Если поле равно None, то оно пропускается. Если все поля содержат значения None, то пропускается элемент целиком.

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.

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.

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.

Необходимо решить задачу двумя способами:

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

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.

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. 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`).

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.

Текст программы и экранные формы с примерами выполнения программы.

Задача 1

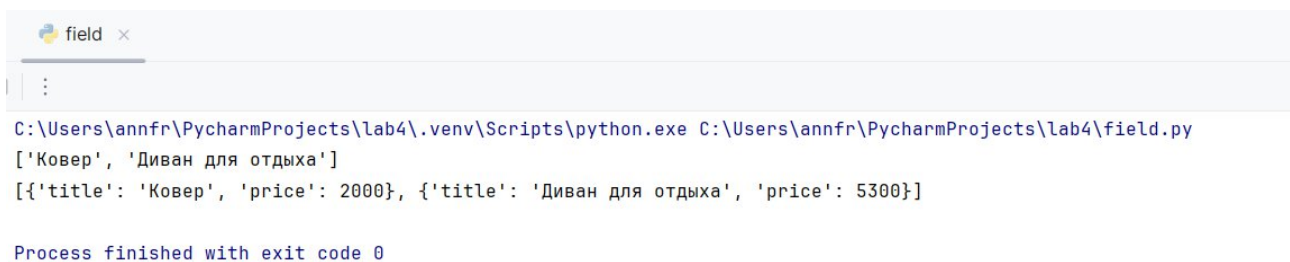
```
def field(items, *args):
    assert len(args) > 0
    for j in items:
        if len(args) == 1:
            if j[args[0]] is not None:
                yield j[args[0]]
        else:
            ans = {}
            for i in args:
                if j[i] is not None:
                    ans[i] = j[i]
            if ans:
                yield ans
```

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

```
titles = list(field(goods, 'title')) # ['Ковер', 'Диван для отдыха']
```

```
prices = list(
    field(goods, 'title', 'price')) # [{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха', 'price': 5300}]
```

```
print(titles)
print(prices)
```



```
field x
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\field.py
['Ковер', 'Диван для отдыха']
[{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха', 'price': 5300}]

Process finished with exit code 0
```

Задача 2

```
# Пример:
# должен выдать 5 случайных чисел
# в диапазоне от 1 до 3, например 2, 2, 3, 2, 1
# Hint: типовая реализация занимает 2 строки
import random
def gen_random(num_count, begin, end):
    for kol in range(num_count):
        yield random.randint(begin, end)
```

```
if __name__ == "__main__":
    a=list(gen_random(5, 1, 3))
    print(a)
```

```
gen_random x
:
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\gen_random.py
[2, 2, 3, 2, 3]

Process finished with exit code 0
```

Задача 3

Итератор для удаления дубликатов

```
class Unique(object):
    def __init__(self, items, **kwargs):
        self.elem = set()
        self.data = items
        self.index = 0
        self.ignore_case = kwargs.get('ignore_case', False) # меняем true-false местами

    def __next__(self):
        while self.index < len(self.data):
            item = self.data[self.index]
            item = item.lower() if self.ignore_case else item
            if item not in self.elem:
                self.elem.add(item)
                return item

            self.index += 1

        raise StopIteration() # Завершаем итерацию, если все элементы обработаны

    def __iter__(self):
        return self
```

```
if __name__ == "__main__":
    data = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
    unique_iterator = Unique(data)
    print(list(unique_iterator)) # Вывод: [1, 2]

    data = ['a', 'A', 'b', 'B', 'a', 'A', 'b', 'B']
    unique_iterator = Unique(data, ignore_case=True)
    print(list(unique_iterator)) # Вывод: ['a', 'b']
```

```
unique x
:
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\unique.py
[1, 2]
['a', 'b']

Process finished with exit code 0
```

Задача 4

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

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



```
sort x  
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\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

Здесь должна быть реализация декоратора

Исправленная реализация декоратора

```
def print_result(func):  
    def wrapper(*args, **kwargs): # Принимаем произвольные аргументы  
        print("Функция:", func.__name__)  
        res = func(*args, **kwargs) # Передаем аргументы в оригинальную функцию  
        if isinstance(res, list):  
            for i in res:  
                print(i)  
        elif isinstance(res, dict):  
            for i, j in res.items():  
                print(i, "=", j)  
        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

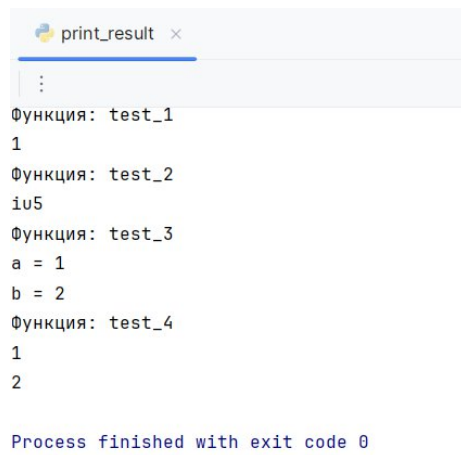
```
def test_4():  
    return [1, 2]
```

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

```

print('!!!!!!!')
test_1()
test_2()
test_3()
test_4()

```



```

print_result x
:
Функция: test_1
1
Функция: test_2
iu5
Функция: test_3
a = 1
b = 2
Функция: test_4
1
2

Process finished with exit code 0

```

Задача 6

```

from contextlib import contextmanager
import time

```

```

class cm_timer_1:

```

```

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

    def __exit__(self, *_):
        self.finish = time.time()
        self.work_time = self.finish - self.start
        print(self.work_time)

```

```

@contextmanager

```

```

def cm_timer_2():
    start = time.time()
    try:
        yield
    finally:
        finish = time.time()
        work_time = finish - start
        print(work_time)

```

```

if __name__ == "__main__":
    with cm_timer_1():
        time.sleep(3)

```

```

    with cm_timer_2():
        time.sleep(3)

```

```
cm_timer x
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\cm_timer.py
3.0010886192321777
3.000288963317871

Process finished with exit code 0
```

Задача 7

```
import json
import cm_timer
from print_result import print_result
from gen_random import gen_random
from unique import Unique

path = "C:/Users/annfr/PycharmProjects/lab4/data_light.json"

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

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

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

@print_result
def f4(arg):
    salaries = list(gen_random(len(arg), 100000, 200000))
    result = [
        f'{specialty}, зарплата {salary} руб.'
        for specialty, salary in zip(arg, salaries)
    ]
    return result

if __name__ == '__main__':
    with cm_timer.cm_timer_1():
        with open(path, encoding='utf-8') as f:
            data = json.load(f)
            f4(f3(f2(f1(data))))
```



```
process_data x
C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\process_data.py
Функция: 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
```

C:\Users\annfr\PycharmProjects\lab4\.venv\Scripts\python.exe C:\Users\annfr\PycharmProjects\lab4\process_data.py

Функция: 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
агент торговый
агрегатчик-топливник komatsu
агроном
агроном по защите растений
агроном-полевод
агрохимик почвовед
администратор
администратор (удаленно)
администратор active directory
администратор в парикмахерский салон
администратор зала (предприятий общественного питания)
администратор кофейни
администратор на ресепшен
администратор на телефоне
администратор по информационной безопасности
администратор ресторана
администратор сайта
администратор ярмарок выходного дня
администратор-кассир
аккомпаниатор на 0,5 ст.
аккумуляторщик 4 разряда
акушерка
акушерка в родильное отделение
акушерка женской консультации
акушерка лысогогорская врачебная амбулатория
акушерка фап
акушерка, ао
акушерка, вл
альпинист промышленный
аналитик
анестезиолог - реаниматолог
анестезиолог-реаниматолог
анестезиолог-реаниматолог детский
аниматор
антенщик-мачтовик 4 разряда
аппаратчик обработки зерна

аппаратчик обработки зерна 5 разряда
аппаратчик пастеризации
аппаратчик установки опытного производства
аппаратчик химводоочистки
арматурщик
арматурщик кузовного цеха
арматурщики
артист (кукловод) театра кукол
артист оркестра
артист отдела социально - культурной деятельности районного цнк
артист хора
артист(кукловод) театра кукол
артист-вокалист (солист)
артист-кукловод
архивариус
архивариус (орехово-зюевский филиал)
архитектор, картограф, инженер-проектировщик
ассистент главы отделения
ассистент отдела продаж
ассистент режиссера
балетмейстер-постановщик
бармен
бармен-кассир в кафе
бармен-официант
бетонщик
бетонщик (на срубку свай)
бетонщик - арматурщик
бетонщик-монолитчик
библиограф
библиотекарь
библиотекарь отдела абонементов
биолог
боец скота
бригадир в животноводстве
бригадир животноводства
бригадир мобильной бригады
бригадир технического обслуживания газоиспользующего оборудования
бригадир, производитель работ
брокер коммерческой недвижимости
брошюрщик
бухгалтер
бухгалтер (по заработной плате)
бухгалтер 2 категории
бухгалтер на группу "обработка первичной документации"
бухгалтер по ведению первичной документации
бухгалтер по заработной плате
бухгалтер по мтп и гсм
бухгалтер по начислению заработной платы
бухгалтер по расчету заработной платы
бухгалтер по расчету калькуляции
бухгалтер, ведущий
бухгалтер,ведущий бухгалтер
бухгалтер-делопроизводитель
бухгалтер-кассир
бухгалтер-кассир 1 категории
бухгалтер-материалист
бухгалтер-ревизор
бухгалтер-экономист
вальцовщик
варщик зефира
варщик мармеладных изделий
вахта
вахтер
вахтёр
веб - программист (php, js) / web разработчик
веб-дизайнер
веб-программист
ведущий агрохимик лаборатории полевых изысканий отдела агроэкологического мониторинга почв
ведущий библиотекарь отдела книгохранения
ведущий бухгалтер
ведущий бухгалтер, экономист
ведущий инженер
ведущий инженер в сметно-договорной отдел
ведущий инженер внедрения кпп и а
ведущий инженер конструктор окб
ведущий инженер лаборатории испытаний и экспертизы
ведущий инженер отдела главного метролога
ведущий инженер отдела оценки компетентности испытательных лабораторий и физико-химических измерений (сергиево-посадский филиал)
ведущий инженер отдела эксплуатации (сергиево-посадский филиал)
ведущий инженер по метрологии лаборатории аттестации методик выполнения измерений (сергиево-посадский филиал)
ведущий инженер по метрологии отдела испытаний, поверки и калибровки си механических и геометрических величин (сергиево-посадский филиал)
ведущий инженер по метрологии отдела испытаний, поверки и калибровки си оптико-физических, теплотехнических, температурных величин, давления и вакуума, физико-химического состава и свойств веществ (сергиево-посадский филиал)
ведущий инженер по метрологии отдела радиоакустических измерений
ведущий инженер-конструктор
ведущий инженер-программист
ведущий инженер-технолог
ведущий инженер-технолог (химик)
ведущий инженер-электроник (схемотехник)
ведущий инженер-электроник отдела автоматизированной системы управления
ведущий консультант по внедрению
ведущий менеджер по продажам
ведущий методист
ведущий научный сотрудник
ведущий программист
ведущий специалист
ведущий специалист 3-го разряда
ведущий специалист в отдел контроля качества в дорожной отрасли
ведущий специалист гражданской обороны
ведущий специалист информационной безопасности
ведущий специалист отдела предоставления мер социальной поддержки
ведущий специалист отдела компенсаций и социальных гарантий
ведущий специалист отдела маркетинга и продуктового предложения (направление тарифы и продукты)
ведущий специалист отдела отчетности
ведущий специалист отдела расчетов и управления задолженностью

ведущий специалист отдела учета дебиторской и кредиторской задолженности
ведущий специалист отдела экономики
ведущий специалист охране труда и технике безопасности
ведущий специалист по лабораторному контролю
ведущий специалист по обучению и развитию персонала
ведущий специалист по охране труда и технике безопасности
ведущий специалист по проектной деятельности
ведущий специалист по энергетике
ведущий специалист финансового отдела
ведущий специалист-эксперт
ведущий специалист-эксперт отдела бухгалтерского учета и отчетности
ведущий экономист
ведущий экономист бюджетного планирования
ведущий экономист по организации и оплате труда
ведущий экономист по планированию и себестоимости
ведущий эксперт в мостовой отдел
ведущий эксперт в отдел ремонта автомобильных дорог
верификатор кредитных заявок/ кредитный эксперт
весовщик
ветврач
ветеринарный врач
ветеринарный лаборант
ветеринарный фельдшер по противоэпизоотическим мероприятиям
вечерний секретер
взрывник открытых горных работ
водитель
водитель - механик
водитель gett/гетт и yandex/яндекс такси на личном автомобиле
водитель автобетоносмесителя
водитель автобетоносмесителя (категория с)
водитель автобуса
водитель автобуса паз
водитель автокрана
водитель автомобиля
водитель автомобиля (коломенский филиал)
водитель автомобиля волчанского участкового лесничества
водитель автомобиля группы тылового обеспечения
водитель автомобиля кат.с (камаз)
водитель автомобиля кыплымского участкового лесничества
водитель атз(топливо заправщики)
водитель бульдозера

.....

Функция: f2

программист

программист / senior developer

программист 1с

программист с#

программист с++

программист с++/с#/java

программист/ junior developer

программист/ технический специалист

программист-разработчик информационных систем

Функция: 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

программист-разработчик информационных систем с опытом Python

Функция: f4

программист с опытом Python, зарплата 131966 руб.

программист / senior developer с опытом Python, зарплата 129008 руб.

программист 1с с опытом Python, зарплата 165637 руб.

программист с# с опытом Python, зарплата 110407 руб.

программист с++ с опытом Python, зарплата 160763 руб.

программист с++/с#/java с опытом Python, зарплата 107067 руб.

программист/ junior developer с опытом Python, зарплата 162323 руб.

программист/ технический специалист с опытом Python, зарплата 102036 руб.

программист-разработчик информационных систем с опытом Python, зарплата 122010 руб.

0.2401118278503418

Process finished with exit code 0