Университет им. Н.Э. Баумана

Факультет Радиотехнический

Кафедра РТ5

Курс «Парадигмы и конструкции языков программирования»

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

Выполнил: Кудрявцев Р. В.
Студент группы: РТ5-31Б

Проверяющий: Гапанюк Ю.Е.
Доцент кафедры ИУ5

Москва, 2023 г.

Описание задания

Задание лабораторной работы состоит из решения нескольких задач.

Файлы, содержащие решения отдельных задач, должны располагаться в пакете lab_python_fp. Решение каждой задачи должно раполагаться 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. Пример:

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

`field(goods, 'title')` должен выдавать `'Ковер', 'Диван для отдыха'`

`field(goods, 'title', 'price')` должен выдавать `{'title': 'Ковер', 'price': 2000},`

`{'title': 'Диван для отдыха'}`

- В качестве первого аргумента генератор принимает список словарей, дальше через *args генератор принимает неограниченное количествово 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.

Шаблон для реализации генератора:

```
# goods = [
#    {'title': 'Ковер', 'price': 2000, 'color': 'green'},
#    {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}
# ]
# field(goods, 'title') должен выдавать 'Ковер', 'Диван для отдыха'
# field(goods, 'title', 'price') должен выдавать {'title': 'Ковер', 'price': 2000},
{'title': 'Диван для отдыха', 'price': 5300}
```

```
def field(items, *args):
    assert len(args) > 0
    # Необходимо реализовать генератор
```

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.

Пример:

`gen_random(5, 1, 3)` должен выдать 5 случайных чисел в диапазоне от 1 до 3, например `2, 2, 3, 2, 1`

Шаблон для реализации генератора:

```
# Пример:
# gen_random(5, 1, 3) должен выдать выдать 5 случайных чисел
# в диапазоне от 1 до 3, например 2, 2, 3, 2, 1
# Hint: типовая реализация занимает 2 строки
def gen_random(num_count, begin, end):
    pass
    # Необходимо реализовать генератор
```

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.

Пример:

```
data = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
```

`Unique(data)` будет последовательно возвращать только 1 и 2.

```
data = gen_random(10, 1, 3)
```

`Unique(data)` будет последовательно возвращать только 1, 2 и 3.

```
data = [‘a’, ‘A’, ‘b’, ‘B’, ‘a’, ‘A’, ‘b’, ‘B’]
```

`Unique(data)` будет последовательно возвращать только a, A, b, B.

`Unique(data, ignore_case=True)` будет последовательно возвращать только a, b.

Шаблон для реализации класса-итератора:

```
# Итератор для удаления дубликатов
class Unique(object):
    def __init__(self, items, **kwargs):
        # Нужно реализовать конструктор
        # В качестве ключевого аргумента, конструктор должен принимать bool-параметр
ignore_case,
        # в зависимости от значения которого будут считаться одинаковыми строки в
разном регистре
        # Например: ignore_case = True, Абв и АБВ - разные строки
        #           ignore_case = False, Абв и АБВ - одинаковые строки, одна из
которых удалится
        # По-умолчанию ignore_case = False
        pass

    def __next__(self):
        # Нужно реализовать __next__
        pass

    def __iter__(self):
        return self
```

Задача 4 (файл sort.py)

Дан массив 1, содержащий положительные и отрицательные числа. Необходимо одной строкой кода вывести на экран массив 2, которые 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.

Шаблон реализации:

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

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

    result_with_lambda = ...
    print(result_with_lambda)
```

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 выводить в столбик через знак равенства.

Шаблон реализации:

```
# Здесь должна быть реализация декоратора

@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()
```

Результат выполнения:

```
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. 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.

Шаблон реализации:

```
import json
import sys
# Сделаем другие необходимые импорты

path = None

# Необходимо в переменную path сохранить путь к файлу, который был передан при
запуске сценария

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

# Далее необходимо реализовать все функции по заданию, заменив `raise NotImplemented`
# Предполагается, что функции f1, f2, f3 будут реализованы в одну строку
# В реализации функции f4 может быть до 3 строк

@print_result
def f1(arg):
    raise NotImplemented


@print_result
def f2(arg):
    raise NotImplemented


@print_result
def f3(arg):
    raise NotImplemented


@print_result
def f4(arg):
    raise NotImplemented


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

Текст программы

Задача 1 (файл field.py)

```
def field(items, *args):
    assert len(args) > 0, 'Нужно указать хотя бы 1 ключ'
    if len(args) > 1:
        return [{j: i[j] for j in args if j in i and i[j] != None} for i in
items]
    else:
        return [i[args[0]] for i in items if args[0] in i and i[args[0]] != None]

def main():
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'price': 5300, 'color': 'black'}
    ]
    print(str(field(goods, 'title'))[1:-1])
    print(str(field(goods, 'title', 'price'))[1:-1])

if __name__ == '__main__' :
    main()
```

Задача 2 (файл gen_random.py)

```
import random

def gen_random(c, a, b):
    return [random.randint(a, b) for c in range(0, c)]

def main():
    print(str(gen_random(5, 1, 3))[1:-1])

if __name__ == '__main__' :
    main()
```

Задача 3 (файл unique.py)

```
import gen_random

class Unique(object):
    def __init__(self, items, **kwargs):
        self.index = 0
        if 'ignore_case' in kwargs:
            self.items = list({i.lower() if kwargs['ignore_case'] else i for i in
items})
        else: self.items = list({i for i in items})

    def __next__(self):
        if self.index < len(self.items):
            out = self.items[self.index]
            self.index += 1
            return out
```

```
        else:
            self.index = 0
            raise StopIteration

    def __iter__(self):
        return self

def main():
    data = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]
    _set1 = Unique(data)
    print(', '.join([str(i) for i in _set1]))
    data = gen_random.gen_random(10, 1, 3)
    _set2 = Unique(data)
    print(', '.join([str(i) for i in _set2]))
    data = ['a', 'A', 'b','B', 'a', 'A', 'b', 'B']
    _set3 = Unique(data)
    print(', '.join([str(i) for i in _set3]))
    data = ['a', 'A', 'b','B', 'a', 'A', 'b', 'B']
    _set4 = Unique(data, ignore_case=True)
    print(', '.join([str(i) for i in _set4]))

if __name__ == '__main__' :
    main()
```

Задача 4 (файл sort.py)

```
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), reverse=True)
    print(result_with_lambda)
```

Задача 5 (файл print_result.py)

```
def print_result(func):
    def wrapper(*args, **kwargs):
        func_result = func(*args, **kwargs)
        if type(func_result) == list:
            print(f'{func.__name__}\n' + '\n'.join([str(i) for i in
func_result]))
        elif type(func_result) == dict:
            print(f'{func.__name__}\n' + '\n'.join([f'{i} = {func_result[i]}' for
i in func_result]))
        else:
            print(f'{func.__name__}\n{func_result}')
        return func_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__':
    test_1()
    test_2()
    test_3()
    test_4()
```

Задача 6 (файл cm_timer.py)

```
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):
        print(f'time: {time.time() - self.start_time}')

@contextmanager
def cm_timer_2():
    start_time = time.time()
    yield
    print(f'time: {time.time() - start_time}')

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

Задача 7 (файл process_data.py)

```
import json
import sys
from print_result import print_result
from cm_timer import cm_timer_1
from unique import Unique
```

```python
from field import field
from gen_random import gen_random

path = 'data_light.json'

with open(path, encoding='utf_8') as f:
    data = json.load(f)

@print_result
def f1(arg):
    return [i for i in Unique(field(arg, 'job-name'), 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):
    zipped = list(zip(arg, gen_random(len(arg), 100000, 200000)))
    return [f'{i[0]}, зарплата {i[1]} руб' for i in zipped]


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

Экранные формы с примерами выполнения программы

Задача 1 (файл field.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50256' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\field.py'
'Ковер', 'Диван для отдыха'
{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха', 'price': 5300}
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 2 (файл gen_random.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50271' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\gen_random.py'
3, 2, 2, 1, 1
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 3 (файл unique.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50276' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\unique.py'
1, 2
1, 2, 3
a, A, b, B
a, b
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 4 (файл sort.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50281' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\sort.py'
[123, 100, -100, -30, 4, -4, 1, -1, 0]
[123, 100, -100, -30, 4, -4, 1, -1, 0]
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 5 (файл print_result.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50293' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\print_result.py'
test_1
1
test_2
iu5
test_3
a = 1
b = 2
test_4
1
2
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 6 (файл cm_timer.py)

```
PS D:\учеба\парадигмы программирования\лабы\лаб2>  d:; cd 'd:\учеба\парадигмы программирования\лабы\
лаб2'; & 'C:\Users\Roma\AppData\Local\Programs\Python\Python312\python.exe' 'c:\Users\Roma\.vscode\e
xtensions\ms-python.python-2023.18.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher'
'50298' '--' 'D:\учеба\парадигмы программирования\лабы\лаб2\cm_timer.py'
time: 5.500689268112183
time: 5.500558376312256
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```

Задача 7 (файл process_data.py)

```
f1
упаковщик хлебо - булочных изделий
специалист по кредитным услугам г. новый оскол
продавец-консультант
фельдшер фельдшерско-акушерского пункта
менеджер по работе с людьми
помощник веб-дизайнера
врач-педиатр участковый
ведущий специалист
гибщик судовой
наладчик холодноштамповочного оборудования 4 разряда-4 разряда
врач онколог
аппаратчик пастеризации
электромонтер охранно-пожарной сигнализации и систем видеонаблюдения
врач клинической  лабораторной диагностики
врач педиатр детского отделения
преподаватель (хирург)
специалист группы технической поддержки
врач-отоларинголог
медицинская сестра участковая детской поликлиники
учитель - технологии
помощник бухгалтера
грузчик-наборщик
инженер по качеству 2 категории (класса)
артист-вокалист (солист)
врач-психиатр-нарколог, опо
весовщик
медицинская сестра по массажу, овл
помощник врача-эпидемиолога
зуборезчик
медицинская сестра врача общей практики
инженер-конструктор рэа
товаровед
техник по эксплуатации и ремонту оборудования
инженер-конструктор в наружной рекламе
клиентский менеджер
дежурный по станции - приемосдатчик
консультант территориального отдела в г.дубна московской области
учитель (преподаватель) истории и обществознания
специалист по работе с клиентами
начальник коммерческого отдела
инженер-электронщик (программист асу тп)
```

дизелист
специалист производственной системы
водитель погрузчика 2 разряда-2 разряда
преподаватель робототехники для детей
помощник руководителя
уборщик служебных и производственных помещений
руководитель метрологического центра
юрисконсульт 2 категории
токарь карусельщик
подсобный рабочий
резчик металла на пилах
дизайнер
слесарь по ремонту агрегатов
машинист бульдозера
мойщик автомобилей
врач-бактериолог
ведущий специалист-эксперт отдела бухгалтерского учета и отчетности
оператор ровничного оборудования
ведущий инженер-программист
станочник деревообрабатывающих станков
фасовщица готовых кормов для животных на конвейере (вахта в москве)
менеджер по продажам ит услуг (b2b)
страший специалист 2 разряда отдела по проблемам семьи, материнства и детства
отделочники (ооо "велесстрой")
менеджер по продажам (дилер-менеджер)
требуется уборщица
повар 5 разряда-5 разряда
электромонтер по обслуживанию электрооборудования электростанций
репетитор по технике речи
врач-педиатр, заведующий педиатрическим отдедлением
менеджер по работе с корпоративными клиентами
машинист экскаватора 6 разряда
кухонный рабочий пищеблока
инженер по холодоснабжению
врач-онколог
фермер (жилье в крыму)
инструктор по физической культуре
агрегатчик-топливник komatsu
слесарь - ремонтник
электрогазосварщик
инженер по метрологии i категории отдела физико-химических и оптико-физических си
слесарь по контрольно-измерительным приборам и автоматике
отделочники-универсалы

4-ый механик
дровокол
токарь 4 разряда-6 разряда
инспектор по пожарной безопасности
электрик канатных дорог
педагог-организатор
уборщица склада и офиса
начальник пто
оператор 1с
водитель крановo-манипуляторной установки камаз
медицинский психолог
фрезеровщик
инженер по метрологии i категории лаборатории поверки дозиметрических си отдела измерений ионизирующ
их излучений
терапевт
технический руководитель (в сельском, охотничьем, лесном и рыбном хозяйстве)
старший механик
начальник группы (в прочих отраслях)
составитель поездов - рабочий по станции
операционная медицинская сестра хирургического отделения
повар горячего цеха
менеджер по продажам в департамент долевого строительства
заведующий отделением (в прочих отраслях)
мерчендайзер-консультант в авс-электро
медицинская сестра операционная
продавец тканей
управляющий отделением
инженер опс
инженер - программист асу тп
кухонный работнмк
ведущий специалист по энергетике
водитель экскаватора-погрузчика
зоотехник
начальник производства/участка
слесарь аварийно-восстановительных работ
медицинская сестра в школу
инженер пто/сметчик
инженер-механик по животноводству
инженер по охране труда
дизайнер-консультант
эндокринолог
воспитатель детского сада
плотник

техник в отдел по монтажу и эксплуатации приборов учета
механик по налаживанию оборудования основного производственного цеха
врач-клинической лабораторной диагностики
врач акушер-гинеколог в женскую консультацию
электромонтер по ремонту и обслуживанию оборудования
водитель категории с, д, е
швея - мотористка
монтажник технологичиского трубопровода
мастер леса сосновского участкового лесничества
воспитатель общежития
электромонтер по испытаниям и измерениям 4-6 разряд
врач- рентгенолог
специалист по административному производству
курьер-регистратор
разнорабочий
ведущий экономист бюджетного планироваия
водитель автобетоносмесителя (категория с)
специалист по кадрам (временно)
энтомолог
врач-судмедэксперт
врач-специалист
оленевод 3 разряда
администратор на ресепшен
заливщик компаундами
электрогазосварщик 4 разряда-5 разряда
врач оториноларинголог
тракторист 5 разряда
разработчик мобильных приложений
специалист договорного отдела
сестра медицинская диетическая
руководитель
швея-портной в ателье
инженер-технолог штамповочного производства
инженер-энергетик 2 категории
автослесарь
грузчик на колбасный завод(проживание, питание)
начальник охраны объекта
засольщик мяса и мясопродуктов
ведущий инженер
врач ультразвукой диагностики
начальник электротехнической лаборатории
медицинская сестра палатная (постовая)
токарь-расточник 4-6 разряда по металлу

кровельщик
водитель на краз-255
наладчик технологического оборудования
ведущий специалист отдела экономики
слесарь по ремонту автомобилей
государственный таможенный инспектор - бухгалтер
медицинская сестра процедурной (в отделении патологии новорожденных и недоношенных детей № 1
зам.тех.директора - начальник службы отипб
специалист планово-экономического сектора
слесарь по ремонту оборудования
начальник лаборатории испытаний продукции легкой и текстильной промышленности (орехово-зуевский филиал)
инженер (группа комплектации)
учитель истории
земледел
системный администратор
ведущий инженер-электроник отдела автоматизированной системы управления
шлифовщик механического цеха
оператор колл центра
механик по ремонту строительной техники
программист c++/c#/java
секретарь судебного заседания в аппарате мирового судьи железнодорожного судебного района города ростова-на-дону
управляющий загородным домом
заместитель главного врача по медицинскому обслуживанию населения
начальник смены
врач - анестезиолог-реаниматолог
ведущий бухгалтер
горнорабочий
специалист-землеустроитель
заведующий фап
секретарь суда
ведущий специалист отдела маркетинга и продуктового предложения (направление тарифы и продукты)
главный специалист-эксперт
менеджер проекта
специалист по кредитным услугам пгт томаровка
водитель такси
слесарь механосборочных работ
ведущий специалист в отдел контроля качества в дорожной отрасли
педагог дополнительного образования
врач-нарколог
мастер погрузочного пункта
вальцовщик

вальцовщик
рисовод
руководитель кружка по эстрадному вокалу
водитель грузового самосвала
ведущий экономист по планированию и себестоимости
мобильный мерчендайзер
специалист 1 категории (класса)
водитель автокрана
продавец-консультант-кассир
старший менеджер продаж операторам связи
электромонтажник
медицинская сестра центра здоровья
инженер-схемотехник
секретарь судебного заседания в аппарате мирового судьи волгодонского судебного района ростовской об
ласти
врач-хирург городской поликлиники.
упаковщик-грузчик
заместитель начальника цеха
судебный пристав по обеспечению установленного порядка деятельности судов
государственный инспектор по охране леса
дезинфектор
менеджер по развитию
оператор-приёмщик
машинист подборочно-швейной машины
менеджер по продажам полиграфических услуг
медсестра палатная
слесарь-ремонтник 5 разряда-5 разряда
ведущий эксперт в мостовой отдел
заведующий приёмным отделением
системный программист (c, linux)
бухгалтер
требуются охранники 4 разряда
заместитель (ца) руководителя по общим вопросам
инженер-технолог
начальник отдела (научно-экспериментального)
оператор моечной установки(авто-мойщик)
художественный руководитель
машинист башенного крана
учетчик
крановщик крана
оператор поломоечной машины (м. рыбацкое)
медицинская сестра кардиологического кабинета
заведующий кабинетом медицинской статистики

```
врач-рентгенолог на 0,25 ст.
эксперт по промышленной безопасности оборудования, работающего под давлением
инженер по входному контролю качества
техник по звуку, инженер по звуковому оборудованию
консультант территориального отдела в г.пересвет московской области
токарь
научный сотрудник
агент по привлечению юридических лиц
бухгалтер (по заработной плате)
оператор шинного производства
руководитель направления сервисного обслуживания
сварщик
электромонтер по эксплуатации и ремонту оборудования
слесарь-судоремонтник 4 разряда-5 разряда
инженер-технолог гальванического производства
юрист (специалист по сопровождению международных договоров, английский - разговорный)
врач-невролог детский
формовщик машинной формовки
медицинская сестра кабинета по обслуживанию детей в дошкольных учреждениях
наборщик
участковый уполномоченный полиции
электро и газосварщики
врач  - эндокринолог;
f2
программист c++/c#/java
программист 1c
программист/ технический специалист
программист / senior developer
программист
программист c#
программистр-разработчик  информационных систем
программист c++
программист/ junior developer
f3
программист c++/c#/java с опытом python
программист 1c с опытом python
программист/ технический специалист с опытом python
программист / senior developer с опытом python
программист с опытом python
программист c# с опытом python
программистр-разработчик  информационных систем с опытом python
программист c++ с опытом python
программист/ junior developer с опытом python
f4
программист c++/c#/java с опытом python, зарплата 109309 руб
программист 1c с опытом python, зарплата 128512 руб
программист/ технический специалист с опытом python, зарплата 115470 руб
программист / senior developer с опытом python, зарплата 147322 руб
программист с опытом python, зарплата 106187 руб
программист c# с опытом python, зарплата 108901 руб
программистр-разработчик  информационных систем с опытом python, зарплата 112432 руб
программист c++ с опытом python, зарплата 134174 руб
программист/ junior developer с опытом python, зарплата 165666 руб
time: 0.023420333862304688
PS D:\учеба\парадигмы программирования\лабы\лаб2>
```