Московский государственный технический университет им. Н.Э. Баумана

Факультет РТ
Кафедра РТ5

Курс «Парадигмы и конструкции языков программирования»

Отчет по Лабораторной работе №1

Выполнил:

студент группы РТ5-21Б:

Хашиева Раяна Харомовна

Проверил:

преподаватель каф. ИУ5

Гапанюк Юрий Евгеньевич

Москва, 2025 г.

Постановка задачи

Задача 1 (файл field.py)

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

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, которые содержит значения массива 1, отсортированные по модулю в порядке убывания. Сортировку необходимо осуществлять с помощью функции sorted.

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.

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.

Задача 7 (файл process_data.py)

В предыдущих задачах были написаны все требуемые инструменты для работы с данными. Применим их на реальном примере.

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

field.py:

```
def field(items, *args):
    for item in items:
        if len(args) == 1:
            value = item.get(args[0])
            if value is not None:
                yield value
        else:
            result = {}
            for key in args:
                value = item.get(key)
                if value is not None:
                    result[key] = value

            if result:
                yield result


# Тестовый запуск (по условию — при запуске файла)
if __name__ == "__main__":
    goods = [
        {'title': 'Ковер', 'price': 2000, 'color': 'green'},
        {'title': 'Диван для отдыха', 'color': 'black'}
    ]

    print(list(field(goods, 'title')))
    print(list(field(goods, 'title', 'price')))
```

gen_random.py:

```
import random


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


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

unique.py:

```
class Unique:
    def __init__(self, items, **kwargs):
        self.items = iter(items)
        self.ignore_case = kwargs.get('ignore_case', False)
        self.seen = set()

    def __iter__(self):
        return self

    def __next__(self):
        while True:
            item = next(self.items)

            key = item
```

```
            if self.ignore_case and isinstance(item, str):
                key = item.lower()

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

if __name__ == "__main__":
    data = ['A', 'a', 'B', 'b', 'A', 'c']

    print(list(Unique(data)))
    print(list(Unique(data, ignore_case=True)))
```

sort.py:

```
arr = [3, -1, 2, -10, 5, -7]

print(sorted(arr, key=lambda x: abs(x), reverse=True))
```

print_result.py:

```
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 k, v in result.items():
                print(f"{k} = {v}")
        else:
            print(result)

        return result
    return wrapper


if __name__ == "__main__":
    @print_result
    def test():
        return ["one", "two", "three"]

    test()
```

cm_timer.py:

```
import time
from contextlib import contextmanager


class cm_timer_1:
    def __enter__(self):
        self.start = time.time()

    def __exit__(self, exc_type, exc_val, exc_tb):
        print(f"time: {round(time.time() - self.start, 4)}")


@contextmanager
def cm_timer_2():
    start = time.time()
    yield
    print(f"time: {round(time.time() - start, 4)}")
```

```
if __name__ == "__main__":
    with cm_timer_1():
        time.sleep(1)

    with cm_timer_2():
        time.sleep(1)
```

process_data.py:

```
if __name__ == "__main__"
    from cm_timer import cm_timer_1
    from print_result import print_result
    from unique import Unique
    from gen_random import gen_random
    import json

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

    @print_result
    def f1(data):
        return sorted(
            Unique([x["name"] for x in data], ignore_case=True),
            key=str.lower
        )

    @print_result
    def f2(data):
        return list(filter(lambda x: x.lower().startswith("программист"),
data))

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

    @print_result
    def f4(data):
        salaries = gen_random(len(data), 100000, 200000)
        return [
            f"{s}, зарплата {sal} руб."
            for s, sal in zip(data, salaries)
        ]

    with cm_timer_1():
        f4(f3(f2(f1(data))))
```

Анализ результатов

```
['Ковер', 'Диван для отдыха']
[{'title': 'Ковер', 'price': 2000}, {'title': 'Диван для отдыха'}]
[3, 1, 1, 2, 1]
['A', 'a', 'B', 'b', 'c']
['A', 'B', 'c']
[-10, -7, 5, 3, 2, -1]
test
one
two
three
time: 1.005
time: 1.005
```

Рис.1(Задание1)

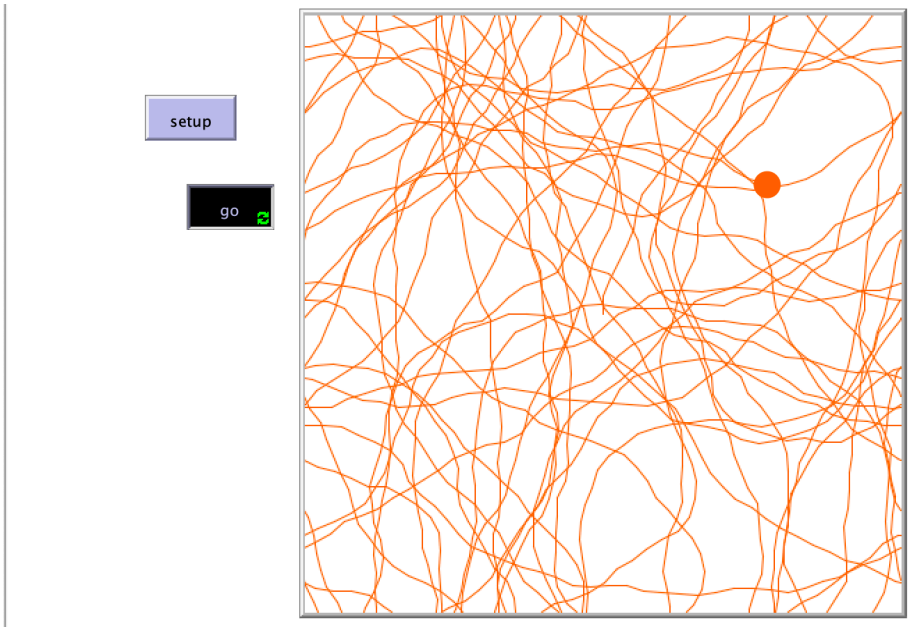

Рис.1(Задание

2)

Использованные источники

- https://iu5edu.ru/wiki/cpp2/docs/labs/lab6/
- https://codelessons.dev/ru/vector-erase-in-cplusplus/
- https://translate.yandex.ru/?source_lang=en&target_lang=ru
- https://hd-rezka.one/filmy/17100-chungkingskiy-ekspress-1994-film-smotret-onlayn.html
- https://rutube.ru/video/private/75f11033a1f49af67e982ae62dde1bbc/?p=_LEEA15KjEjsg_-jN4D9WA