Отчет по Рубежному контролю-2

Разработка программы на языке Python

Дисциплина: Парадигмы и конструкции языков программирования

Выполнил: Уйданов Лев

Группа: ИБМ3-34Б

Преподаватель: Гапанюк Ю.Е.

Москва

2025

Вариант 22: Библиотека – язык программирования

Задание:

Вариант Д.

1. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список всех сотрудников, у которых фамилия заканчивается на «ов», и названия их отделов.
2. «Отдел» и «Сотрудник» связаны соотношением один-ко-многим. Выведите список отделов со средней зарплатой сотрудников в каждом отделе, отсортированный по средней зарплате (*отдельной функции вычисления среднего значения в Python нет, нужно использовать комбинацию функций вычисления суммы и количества значений*).
3. «Отдел» и «Сотрудник» связаны соотношением многие-ко-многим. Выведите список всех отделов, у которых название начинается с буквы «А», и список работающих в них сотрудников.

Условия рубежного контроля №2 по курсу ПиК ЯП

Рубежный контроль представляет собой разработку тестов на языке Python.

1) Проведите рефакторинг текста программы рубежного контроля №1 таким образом, чтобы он был пригоден для модульного тестирования.

2) Для текста программы рубежного контроля №1 создайте модульные тесты с применением TDD - фреймворка (3 теста).

Код:

Файл main.py

```
from operator import itemgetter


class Biblioteka:
    def __init__(self, id, nazvanie):
        self.id = id
        self.nazvanie = nazvanie


class Yazyk:
    def __init__(self, id, nazvanie, stroki, bibl_id):
        self.id = id
        self.nazvanie = nazvanie
        self.stroki = stroki
        self.bibl_id = bibl_id


class YazykBiblioteka:
    def __init__(self, yaz_id, bibl_id):
        self.yaz_id = yaz_id
        self.bibl_id = bibl_id


def one_to_many(biblioteki, yazyki):
    return [
```

```
        (y.nazvanie, y.stroki, b.nazvanie)
        for b in biblioteki
        for y in yazyki
        if y.bibl_id == b.id
    ]


def many_to_many(biblioteki, yazyki, svyazi):
    temp = [
        (b.nazvanie, s.yaz_id)
        for b in biblioteki
        for s in svyazi
        if b.id == s.bibl_id
    ]
    return [
        (y.nazvanie, y.stroki, b_name)
        for b_name, y_id in temp
        for y in yazyki
        if y.id == y_id
    ]


def task_d1(biblioteki, yazyki):
    return [
        (b.nazvanie, [y.nazvanie for y in yazyki if y.bibl_id == b.id])
        for b in biblioteki
        if b.nazvanie.endswith("lib")
    ]


def task_d2(biblioteki, one_to_many_rel):
    return sorted(
        [
            (
                b.nazvanie,
                round(
                    sum(x[1] for x in one_to_many_rel if x[2] == b.nazvanie)
/
                    len([x for x in one_to_many_rel if x[2] == b.nazvanie]),
                    2
                )
            )
            for b in biblioteki
            if any(x[2] == b.nazvanie for x in one_to_many_rel)
        ],
        key=itemgetter(1)
    )


def task_d3(biblioteki, many_to_many_rel):
    return {
        b.nazvanie: [x[0] for x in many_to_many_rel if x[2] == b.nazvanie]
        for b in biblioteki
        if b.nazvanie.startswith("P")
    }


if __name__ == "__main__":
    biblioteki = [
        Biblioteka(1, 'NumPy_lib'),
        Biblioteka(2, 'Pandas_lib'),
        Biblioteka(3, 'PyTorch'),
        Biblioteka(4, 'SciPy_lib'),
        Biblioteka(5, 'Plotly')
```

```
    ]

    yazyki = [
        Yazyk(1, 'Python', 15000, 1),
        Yazyk(2, 'C++', 8000, 2),
        Yazyk(3, 'Java', 12000, 3),
        Yazyk(4, 'Rust', 4000, 4),
        Yazyk(5, 'C', 6000, 4),
        Yazyk(6, 'Go', 5000, 5)
    ]

    svyazi = [
        YazykBiblioteka(1, 1),
        YazykBiblioteka(2, 2),
        YazykBiblioteka(3, 3),
        YazykBiblioteka(4, 4),
        YazykBiblioteka(5, 4),
        YazykBiblioteka(6, 5),
        YazykBiblioteka(1, 2),
        YazykBiblioteka(3, 5)
    ]

    otm = one_to_many(biblioteki, yazyki)
    mtm = many_to_many(biblioteki, yazyki, svyazi)

    print("Zadanie D1:", task_d1(biblioteki, yazyki))
    print("Zadanie D2:", task_d2(biblioteki, otm))
    print("Zadanie D3:", task_d3(biblioteki, mtm))
```

Файл test_main.py

```
import unittest
import main


class TestRK2(unittest.TestCase):

    def setUp(self):
        self.biblioteki = [
            main.Biblioteka(1, 'NumPy_lib'),
            main.Biblioteka(2, 'Pandas_lib'),
            main.Biblioteka(3, 'PyTorch'),
            main.Biblioteka(4, 'SciPy_lib'),
            main.Biblioteka(5, 'Plotly')
        ]

        self.yazyki = [
            main.Yazyk(1, 'Python', 15000, 1),
            main.Yazyk(2, 'C++', 8000, 2),
            main.Yazyk(3, 'Java', 12000, 3),
            main.Yazyk(4, 'Rust', 4000, 4),
            main.Yazyk(5, 'C', 6000, 4),
            main.Yazyk(6, 'Go', 5000, 5)
        ]

        self.svyazi = [
            main.YazykBiblioteka(1, 1),
            main.YazykBiblioteka(2, 2),
            main.YazykBiblioteka(3, 3),
            main.YazykBiblioteka(4, 4),
            main.YazykBiblioteka(5, 4),
            main.YazykBiblioteka(6, 5),
```

```
            main.YazykBiblioteka(1, 2),
            main.YazykBiblioteka(3, 5)
        ]

    def test_d1(self):
        res = main.task_d1(self.biblioteki, self.yazyki)
        self.assertIn(('NumPy_lib', ['Python']), res)

    def test_d2(self):
        otm = main.one_to_many(self.biblioteki, self.yazyki)
        res = main.task_d2(self.biblioteki, otm)
        self.assertEqual(res[0][0], 'SciPy_lib')

    def test_d3(self):
        mtm = main.many_to_many(self.biblioteki, self.yazyki, self.svyazi)
        res = main.task_d3(self.biblioteki, mtm)
        self.assertIn('Python', res['Pandas_lib'])


if __name__ == "__main__":
    unittest.main()
```