

**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 рубежному контролю № 2  
Вариант В-27**

Выполнил:  
студент группы ИУ5-34Б  
Давшиц Е.А.

Проверил:  
преподаватель каф. ИУ5  
Нардид А.Н.

Москва, 2024 г.

## main.py

```
class DataRow:
    """Строка данных"""

    def __init__(self, id, content, table_id):
        self.id = id
        self.content = content
        self.table_id = table_id

class DataTable:
    """Таблица данных"""

    def __init__(self, id, name):
        self.id = id
        self.name = name

class DataRowTable:
    """'Строка данных в таблице' для реализации связи многие-ко-многим"""

    def __init__(self, table_id, row_id):
        self.table_id = table_id
        self.row_id = row_id

# Таблицы данных
tables = [
    DataTable(1, 'Товары'),
    DataTable(2, 'Поставщики'),
    DataTable(3, 'Клиенты'),
]

# Строки данных
rows = [
    DataRow(1, 'Яблоки', 1),
    DataRow(2, 'Груши', 1),
    DataRow(3, 'ООО "Фруктовый Сад"', 2),
    DataRow(4, 'ООО "Свежие Поставки"', 2),
    DataRow(5, 'Иван Иванов', 3),
]

# Связь многие-ко-многим
rows_tables = [
    DataRowTable(1, 1), # Таблица "Товары" - строка "Яблоки"
    DataRowTable(1, 2), # Таблица "Товары" - строка "Груши"
    DataRowTable(2, 3), # Таблица "Поставщики" - строка "ООО 'Фруктовый Сад'"
    DataRowTable(2, 4), # Таблица "Поставщики" - строка "ООО 'Свежие Поставки'"
    DataRowTable(3, 5), # Таблица "Клиенты" - строка "Иван Иванов"
]

def main():
    """Основная функция"""

    # Соединение данных один-ко-многим
    one_to_many = [
        (row.content, row.table_id, table.name)
        for table in tables
        for row in rows
        if row.table_id == table.id
    ]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [
        (table.name, rel.table_id, rel.row_id)
```

```

        for table in tables
        for rel in rows_tables
        if table.id == rel.table_id
    ]

    many_to_many = [
        (row.content, row.table_id, table_name)
        for table_name, _, row_id in many_to_many_temp
        for row in rows if row.id == row_id
    ]

    print('Задание B1')
    # Все строки данных, которые начинаются с буквы "O"
    result_1_temp = list(filter(lambda x: x[0].startswith('O'), one_to_many))
    result_1 = [(row_content, table_name) for row_content, _, table_name in
result_1_temp]
    print(result_1)

    print('\nЗадание B2')
    # Минимальный id для строк в каждой таблице
    result_2_unsorted = []
    for table in tables:
        # Все строки, принадлежащие таблице
        table_rows = list(filter(lambda x: x[2] == table.name, one_to_many))
        if len(table_rows) > 0:
            # Собираем id строк из этой таблицы
            table_row_ids = [row_id for _, row_id, _ in table_rows]
            # Находим минимальный id
            min_row_id = min(table_row_ids)
            result_2_unsorted.append((table.name, min_row_id))

    # Сортировка по минимальному id
    result_2 = sorted(result_2_unsorted, key=lambda x: x[1])
    print(result_2)

    print('\nЗадание B3')
    # Результат многие-ко-многим, отсортированный по названию строки
    result_3_temp = sorted(many_to_many, key=lambda x: x[0])
    result_3 = [(row_content, table_name) for row_content, _, table_name in
result_3_temp]
    print(result_3)

if __name__ == '__main__':
    main()

```

## unit\_test.py

```
import unittest
import main

class TestDataRowTable(unittest.TestCase):
    # Таблицы данных
    tables = [
        main.DataTable(1, 'Товары'),
        main.DataTable(2, 'Поставщики'),
        main.DataTable(3, 'Клиенты'),
    ]

    # Строки данных
    rows = [
        main.DataRow(1, 'Яблоки', 1),
        main.DataRow(2, 'Груши', 1),
        main.DataRow(3, 'ООО "Фруктовый Сад"', 2),
        main.DataRow(4, 'ООО "Свежие Поставки"', 2),
        main.DataRow(5, 'Иван Иванов', 3),
    ]

    # Связи строки с таблицами
    rows_tables = [
        main.DataRowTable(1, 1),
        main.DataRowTable(1, 2),
        main.DataRowTable(2, 3),
        main.DataRowTable(2, 4),
        main.DataRowTable(3, 5)
    ]

    # Построить `one_to_many` и `many_to_many` для тестов
    one_to_many = main.make_one_to_many(rows, tables)
    many_to_many = main.make_many_to_many(rows, tables, rows_tables)

    def test_do_task_one(self):
        result_1 = main.do_task_one(self.one_to_many)
        result_2 = [('ООО "Фруктовый Сад"', 'Поставщики'), ('ООО "Свежие Поставки"', 'Поставщики')]
        self.assertEqual(result_1, result_2)

    def test_do_task_two(self):
        result_1 = main.do_task_two(self.one_to_many, self.tables)
        result_2 = [('Товары', 1), ('Поставщики', 3), ('Клиенты', 5)]
        self.assertEqual(result_1, result_2)

    def test_do_task_three(self):
        result_1 = main.do_task_three(self.many_to_many)
        result_2 = [
            ('Груши', 'Товары'),
            ('Иван Иванов', 'Клиенты'),
            ('ООО "Фруктовый Сад"', 'Поставщики'),
            ('ООО "Свежие Поставки"', 'Поставщики'),
            ('Яблоки', 'Товары'),
        ]
        self.assertEqual(result_1, result_2)

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

## Результат выполнения программы

Задание В1

```
[('ООО "Фруктовый Сад"', 'Поставщики'), ('ООО "Свежие Поставки"', 'Поставщики')]
```

Задание В2

```
[('Товары', 1), ('Поставщики', 2), ('Клиенты', 3)]
```


Задание В3

```
[('Груши', 'Товары'), ('Иван Иванов', 'Клиенты'), ('ООО "Свежие Поставки"', 'Поставщики'), ('ООО "Фруктовый Сад"', 'Поставщики'), ('Яблоки', 'Товары')]
```

```
K1/RK1.py
Задание В1
[('ООО "Фруктовый Сад"', 'Поставщики'), ('ООО "Свежие Поставки"', 'Поставщики')]

Задание В2
[('Товары', 1), ('Поставщики', 2), ('Клиенты', 3)]

Задание В3
[('Груши', 'Товары'), ('Иван Иванов', 'Клиенты'), ('ООО "Свежие Поставки"', 'Поставщики'), ('ООО "Фруктовый Сад"', 'Поставщики'), ('Яблоки', 'Товары')]
PS c:\Users\sdsav\Desktop\Trash\Studies\Pickup>
```

 Командная строка

```
C:\Users\user\Desktop\PCPL_RK2>python unit_test.py
...
-----
Ran 3 tests in 0.001s

OK

C:\Users\user\Desktop\PCPL_RK2>
```