Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

ПРОГРАММНОЕ ПОДКЛЮЧЕНИЕ К СЕРВЕРУ POSTGRESQL

Отчёт о лабораторной работе № 4 по дисциплине «Базы данных»

Выполнил: студент гр. 431-3

Бекиш Е.П.

« » 2023 г.

Проверил: ассистент каф. АСУ

Яблонский Я. В.

« » 2023 г.

Томск 2023

# 1 Цель лабораторной работы

– Разработать программу, которая соответствует условиям индивидуального задания на любом языке программирование.

# 2 Описание требований к отчётам

1. Выполняет подключение к серверу СУБД PostgreSQL.
2. Заносит новые данные в одну из таблиц (по выбору студента).
3. Формирует на экране (или в файле) отчет как показано на рисунке

2.1.



Рисунок 2.1 – успешный результат индивидуального задания

# 3 Листинг

import psycopg2 originall = {

'dbname': 'tusur',

'user': 'postgres',

'password': '123',

}

connect = psycopg2.connect(\*\*originall) cursor = connect.cursor() connect.autocommit = True

query = """

insert into room (rnum, tnum) values

('000', '000000'),

('001', '000001'),

('002', '000002'),

('003', '000003')

;

insert into student (snum, snam, ann, stel, rnum) values

('000000', 'Баранов Алексей Андреевич', '1', '9130000000', '000'),

('000001', 'Барашкин Алексей Андреевич', '1', '9130000001', '000'),

('000002', 'Быков Алексей Андреевич', '1', '9130000002', '000'),

('000003', 'Воловский Алексей Андреевич', '2', '9130000003', '001'),

('000004', 'Гатов Николай Саввич', '3', '9130000004', '002'),

('000005', 'Батов Иван Константинович', '4', '9130000005', '003')

;

insert into "pr-dis" (pnam, dnam) values

('Белоусова София Романовна', 'ПЯВУ'),

('Морозова Майя Андреевна', 'ВУПЯ'),

('Медведев Лев Юрьевич', 'Теоритические основы куроедения'), ('Попов Михаил Кириллович', 'Меланжевое производство'), ('Кулаков Роман Родионович', 'Технология собаковыгуливания'), ('Мальцев Матвей Ильич', 'Технология приготовления яичницы')

;

insert into exams (snum, sem, pnam, ball) values

('000000', '1', 'Белоусова София Романовна', '4'),

('000000', '1', 'Морозова Майя Андреевна', '4'),

('000000', '1', 'Медведев Лев Юрьевич', '4'),

('000001', '1', 'Белоусова София Романовна', '4'),

('000001', '2', 'Морозова Майя Андреевна', '3'),

('000002', '1', 'Морозова Майя Андреевна', '5'),

('000002', '1', 'Медведев Лев Юрьевич', '3'),

('000002', '1', 'Попов Михаил Кириллович', '4'),

('000003', '1', 'Белоусова София Романовна', '3'),

('000003', '2', 'Медведев Лев Юрьевич', '5'),

('000003', '3', 'Кулаков Роман Родионович', '3'),

('000003', '4', 'Морозова Майя Андреевна', '4'),

('000004', '3', 'Белоусова София Романовна', '5'),

('000004', '3', 'Медведев Лев Юрьевич', '5'),

('000004', '3', 'Попов Михаил Кириллович', '5'),

('000005', '1', 'Белоусова София Романовна', '4'),

('000005', '2', 'Морозова Майя Андреевна', '3'),

('000005', '3', 'Кулаков Роман Родионович', '5'),

('000005', '4', 'Попов Михаил Кириллович', '3'),

('000005', '5', 'Медведев Лев Юрьевич', '5'),

('000005', '6', 'Мальцев Матвей Ильич', '3')

;"""

cursor.execute(query) def foo(snum):

query = '''select \* from "pr-dis";''' cursor.execute(query)

pr\_dis = cursor.fetchall()

pr\_dis\_dict = {pr\_dis[i][0]: pr\_dis[i][1] for i in range(len(pr\_dis))} query = f'''select sem, pnam, ball from exams where snum='{snum}';''' cursor.execute(query)

exams = cursor.fetchall()

query = f'''select avg(ball) from exams where snum='{snum}';''' cursor.execute(query)

middle = str(cursor.fetchone()[0])

if sum(list(map(int, middle[2:4]))) == 0: middle = int(middle[:middle.index('.')])

elif middle[3] == '0' or middle[3] != '5': middle = middle[:3]

else:

middle = middle[:4]

s = 'Дисциплина\t\t\t\t\t\tОценка\tСеместр\n' for info in exams:

sem, pnam, ball = info

s += f'\t\t\t\t\t\t\t\t\t {pr\_dis\_dict[pnam]:35}{ball}{sem:10}\n' s += f'\t\t\t\t\t\t\t\t\t\t\t {"Средний балл":25}{middle}'

return s

query = '''select ann from student;''' cursor.execute(query)

ann = set(cursor.fetchall()) file = open('laba4.txt', 'w') for year in ann:

query = f'''select \* from student where ann={year[0]}'''

cursor.execute(query) student\_info = cursor.fetchall() print(f'Год обучения:{year[0]}') for info in student\_info:

snum, snam, ann, tnum, rnum = info print(f'{snam}\t\t\t{foo(snum)}\n')

query = """

delete from exams; delete from student; delete from room; delete from "pr-dis"; """

cursor.execute(query) cursor.close() connect.close()

# 4 Описание процесса реализации

В ходе выполнения лабораторной работы будет использовать язык Python. Создаем файл с расширение .py. Теперь для использования PostgreSQL необходимо импортировать библиотеку psycopg2 как показано на рисунке 4.1.

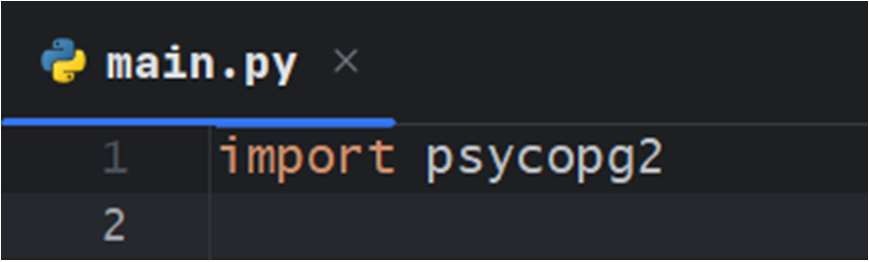


Рисунок 4.1 – импорт библиотеки

Для того, чтобы подключиться к нашей базе данных необходимо использовать метод connect, в котором необходимо указать базу данных, если она существует (по умолчанию подключаемся к postgres), пароль и пользователя (нужно указать postgres). Для удобства создадим словарь, в котором укажем в качестве ключей необходимые название параметров и их

значения. Вообще cсловарь — неупорядоченная структура данных, которая позволяет хранить пары «ключ — значение».

Пример: dictionary = {key: value}

Теперь создаем курсор — область памяти базы, предназначенная для хранения последнего оператора SQL. Иными словами, объект, отвечающий и за отправку запросов, и за получение их результатов. Реальное выполнение команд производится только при выполнении метода conn.commit(). Если же надо, чтобы выражения SQL автоматически выполнялись при каждом вызове метода cursor.execute(), то можно установить автокоммит с помощью свойства connection.autocommit. Где выполнением команды SQL автоматически создается транзакция, в процессе которой можно выполнять различные выражения SQL с помощью методов execute/executemany курсора, но для подтверждения их выполнения необходимо вызывать метод commit() объекта connection.

* execute(query, vars=None): выполняет одну SQL-инструкцию. Через второй параметр в код SQL можно передать набор параметров в виде списка или словаря
* executemany(query, vars\_list): выполняет параметризованное SQL- инструкцию. Через второй параметр принимает наборы значений, которые передаются в выполняемый код SQL.

Результат представлен на рисунке 4.2.

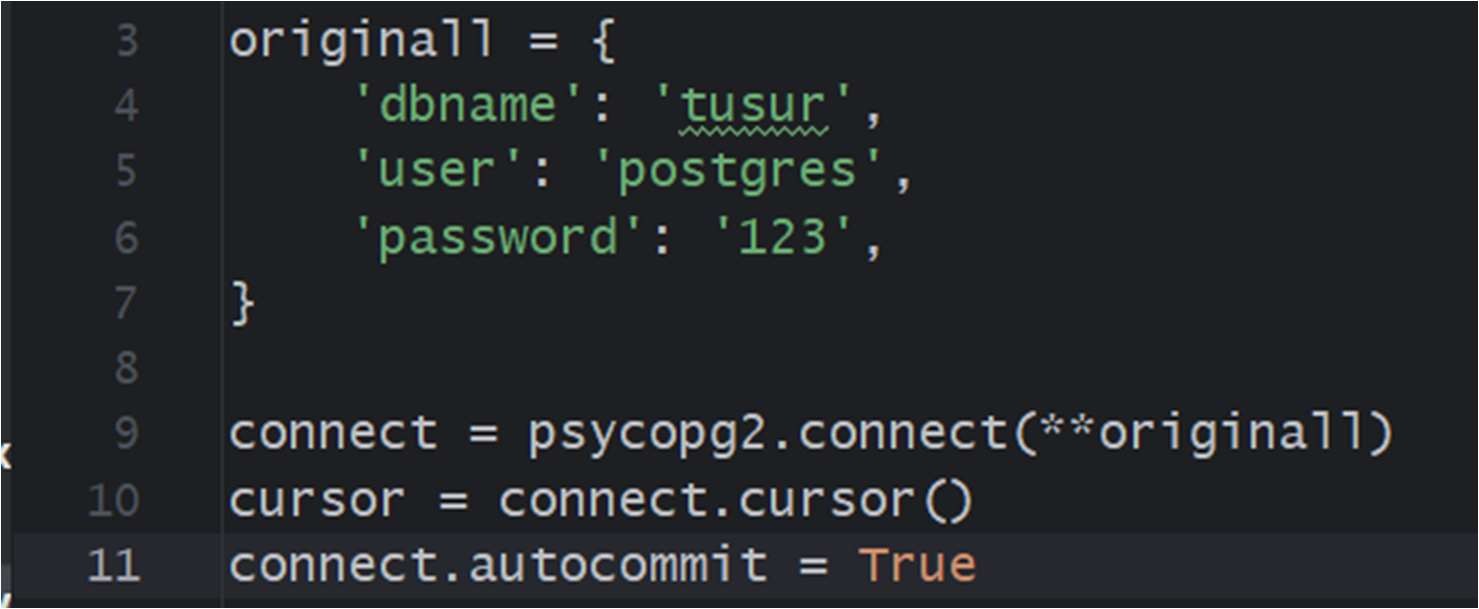


Рисунок 4.2 – авторизация

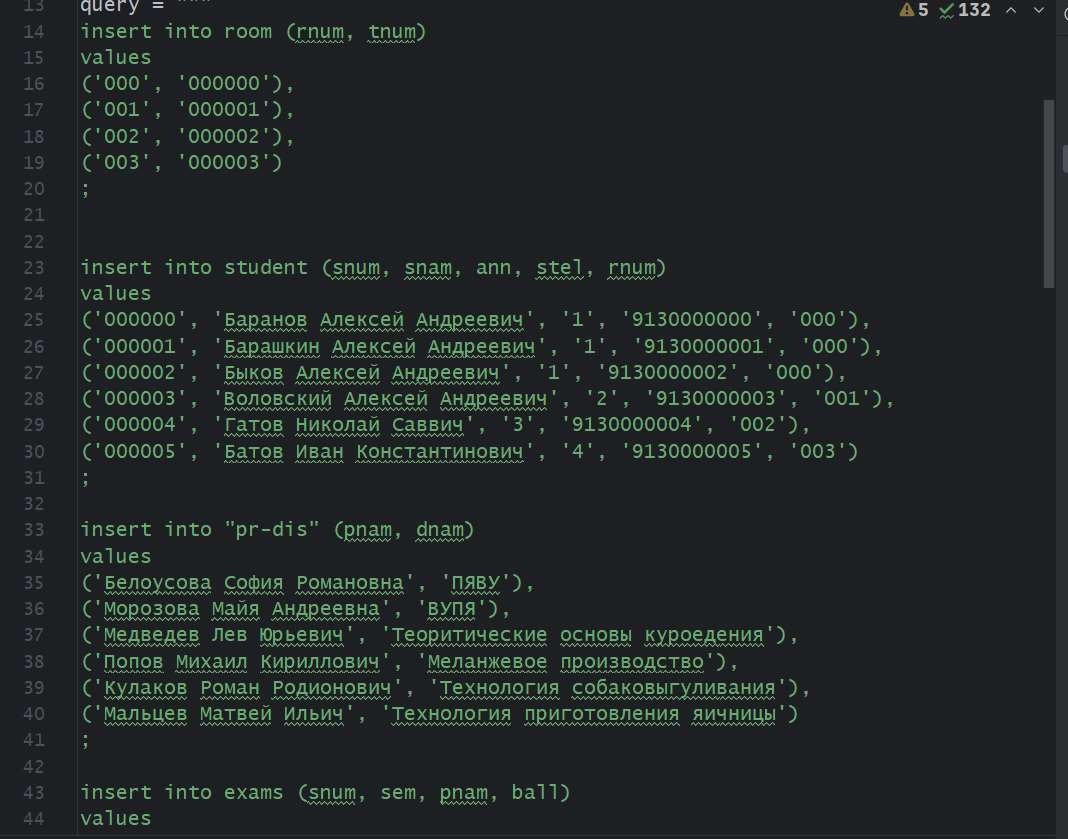
Создадим запрос по добавлению данных в наши созданные таблицы как показано на рисунках 4.3 – 4.4.

Рисунок 4.3 – добавление данных

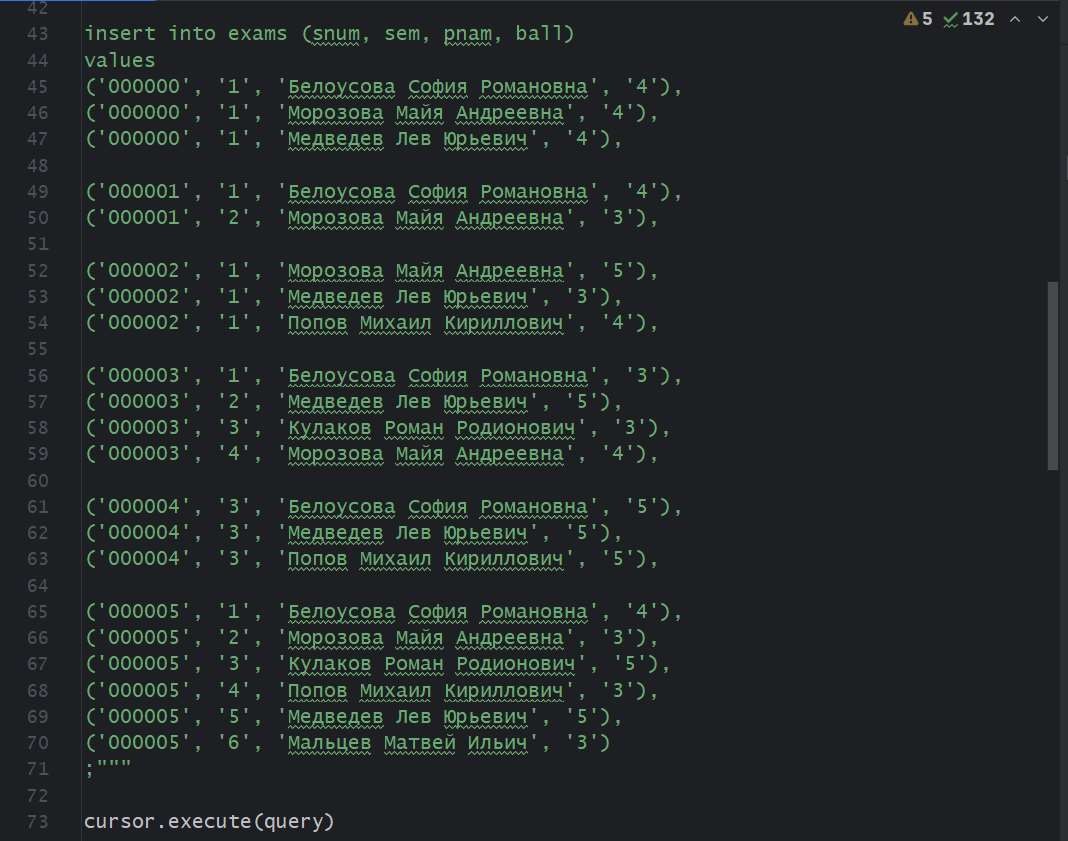


Рисунок 4.4 – добавление данных

После добавления данных создадим функцию, которая будет выдать необходимую информацию об успеваемости. Так как таблицы STUDENT и EXAMS связаны номером студенческого билета, это будет наша переменная, по которой мы сможем брать информацию из данных таблиц. Но в таблице EXAMS хранятся имена преподавателей, поэтому необходимо сформировать словарь по именам преподавателей, которые ведут этот предмет, чтобы имя преподаватель менялось на название дисциплины. Можно это сделать с помощью кортежа словаря, где ключ будет иметь значение имя преподавателя, а значение имеет дисциплину. Результат представлен на рисунке 4.5.

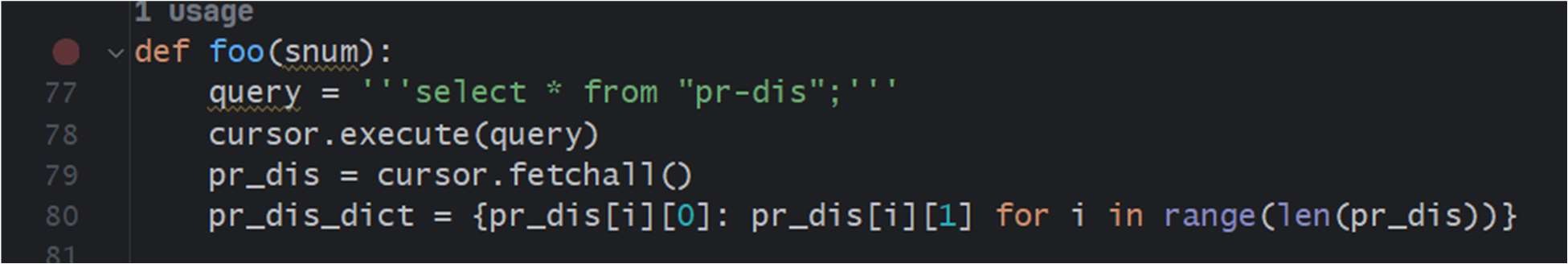


Рисунок 4.5 – словарь дисциплин

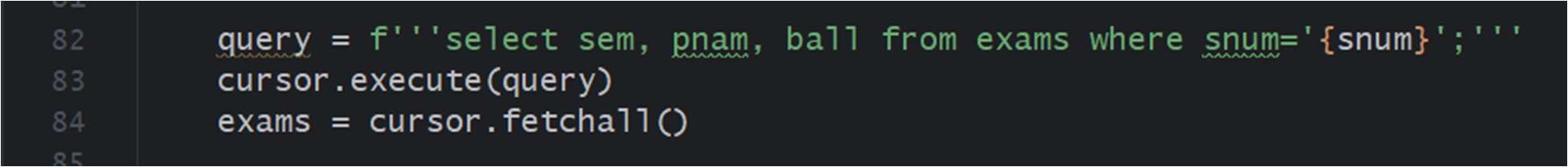
Далее нам необходимо сделать запрос в таблицу EXAMS по студенческому билету. Выбрать информацию о экзаменаторе, семестре, в котором была сдана данная дисциплина и полученная за данную дисциплину оценка (рис. 4.6).

Рисунок 4.6 – информацию о экзамене

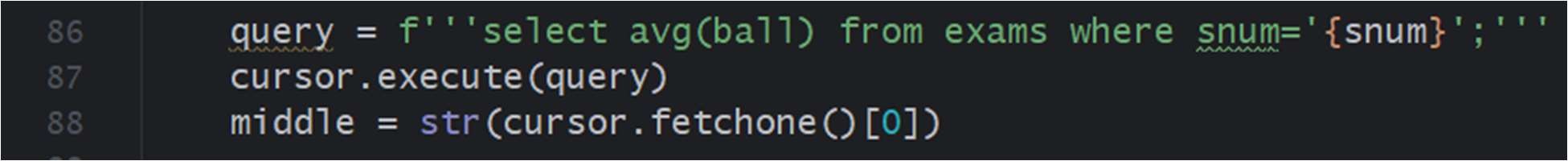
Теперь нам необходимо получить среднее значение за все сданные дисциплины. Это можно сделать с помощью агрегатной функции AVG представленной на рисунке 4.7.

Рисунок 4.7 – информацию о экзамене

После полученного нами среднего числа у нас возникает формат decimal, то есть десятичное число, которая содержит в себе погрешности c шестнадцатью нулей, и чтобы от них избавиться конвертируем decimal в строку. Проверим, что если наши два значения после запятой это два нуля, то мы конвертирует в число, если же второй знак после запятой ноль, либо же не пять, то делаем срез до него, в противном случае срезаем до второго знака включительно (рис. 4.8).

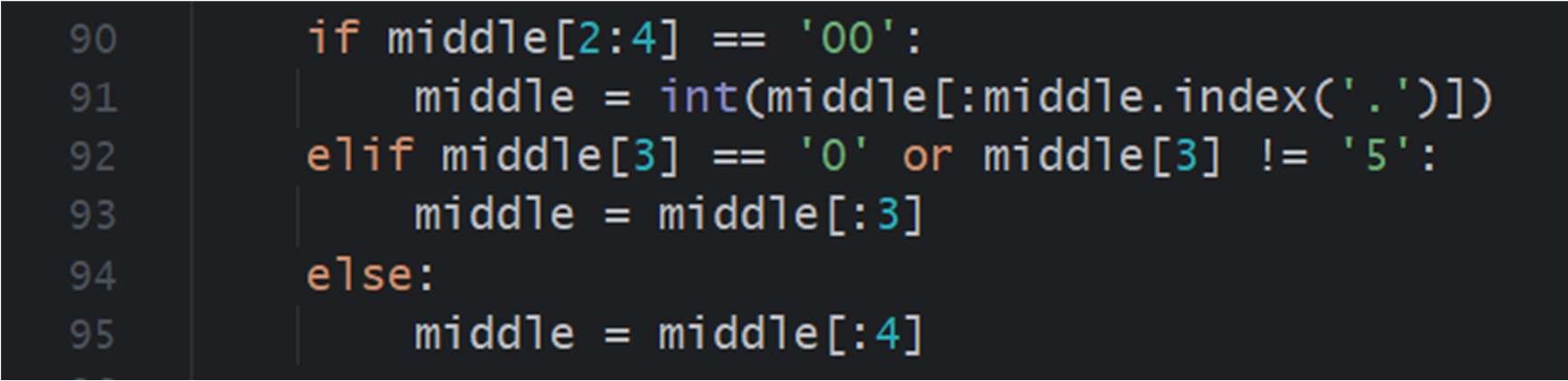


Рисунок 4.8 – средняя оценка

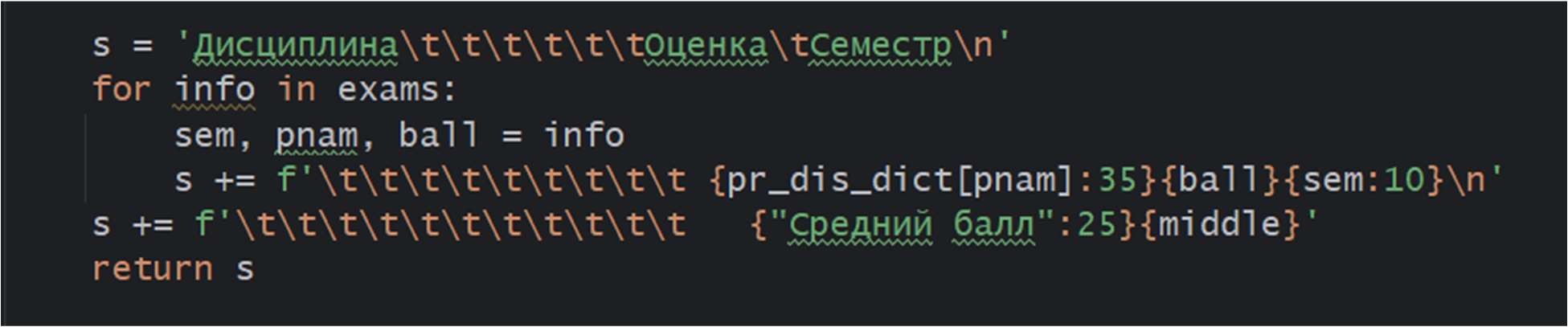
Когда мы получили все необходимые данные, нам нужно вывести всю информацию об студенте по его студенческому билету. Мы формируем строку, в которой будем хоронить всю информацию и ее возвращаем. Результат представлен на рисунке 4.9.

Рисунок 4.9 – получение информации

Теперь необходимо вывести информацию по году обучение, мы делаем запрос в таблицу STUDENT, где выбираем год обучения и формируем множество, потому что в нашей таблице могут быть несколько студентов, которые обладают одним и тем же годом обучение. Дальше создаем цикл, где проходимся по студентам данного года обучения, делаем запрос в таблицу, где выбираем номер студенческого билета, чтобы добыть информацию о данных дисциплины и его имя, после этого выводим год обучения и ту информацию, которую мы получили из нашей функции и закрываем нашу базу данных (рис. 4.10).

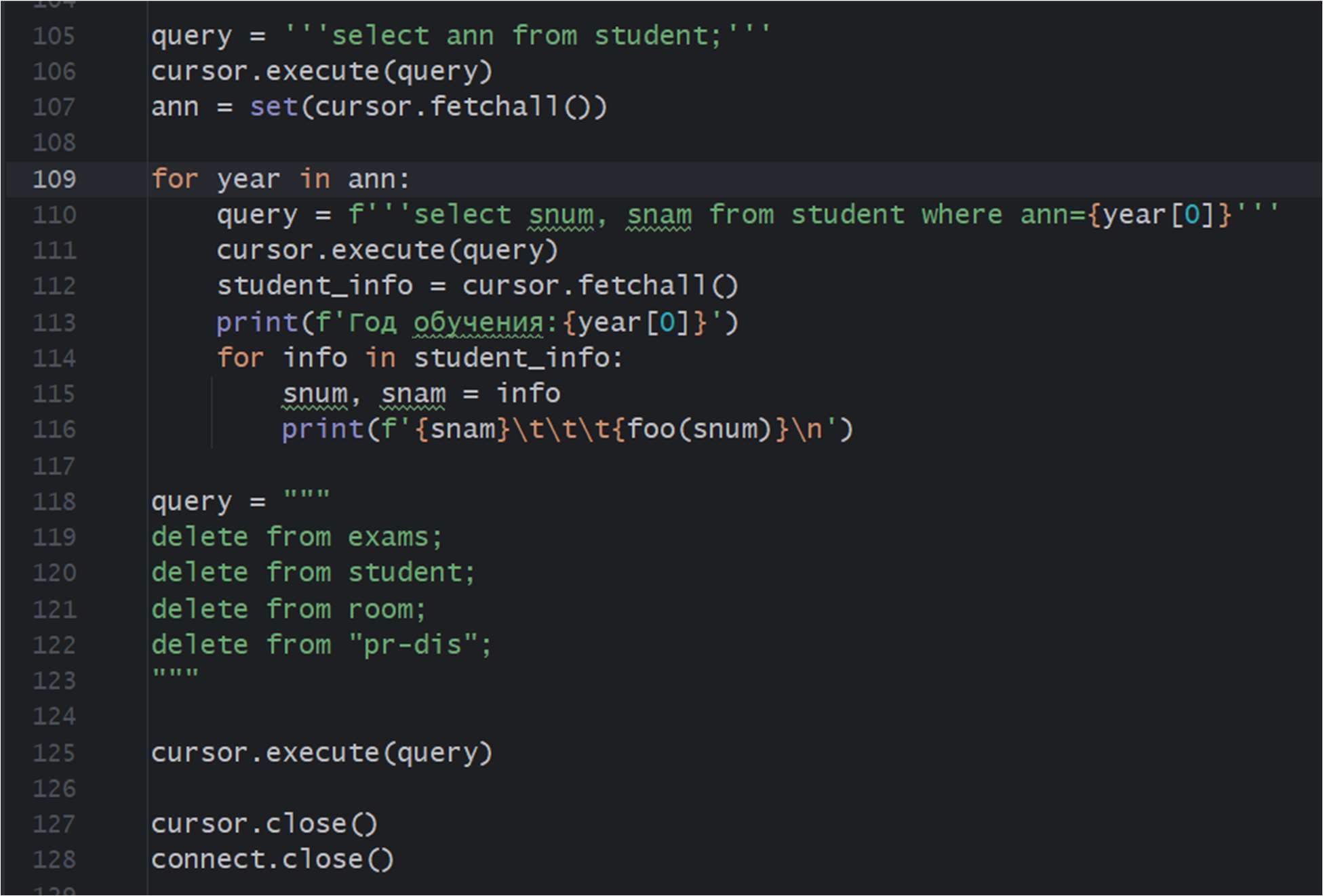


Рисунок 4.10 – вывод необходимой информации Результат работы все программы представлен на рисунке 4.11.

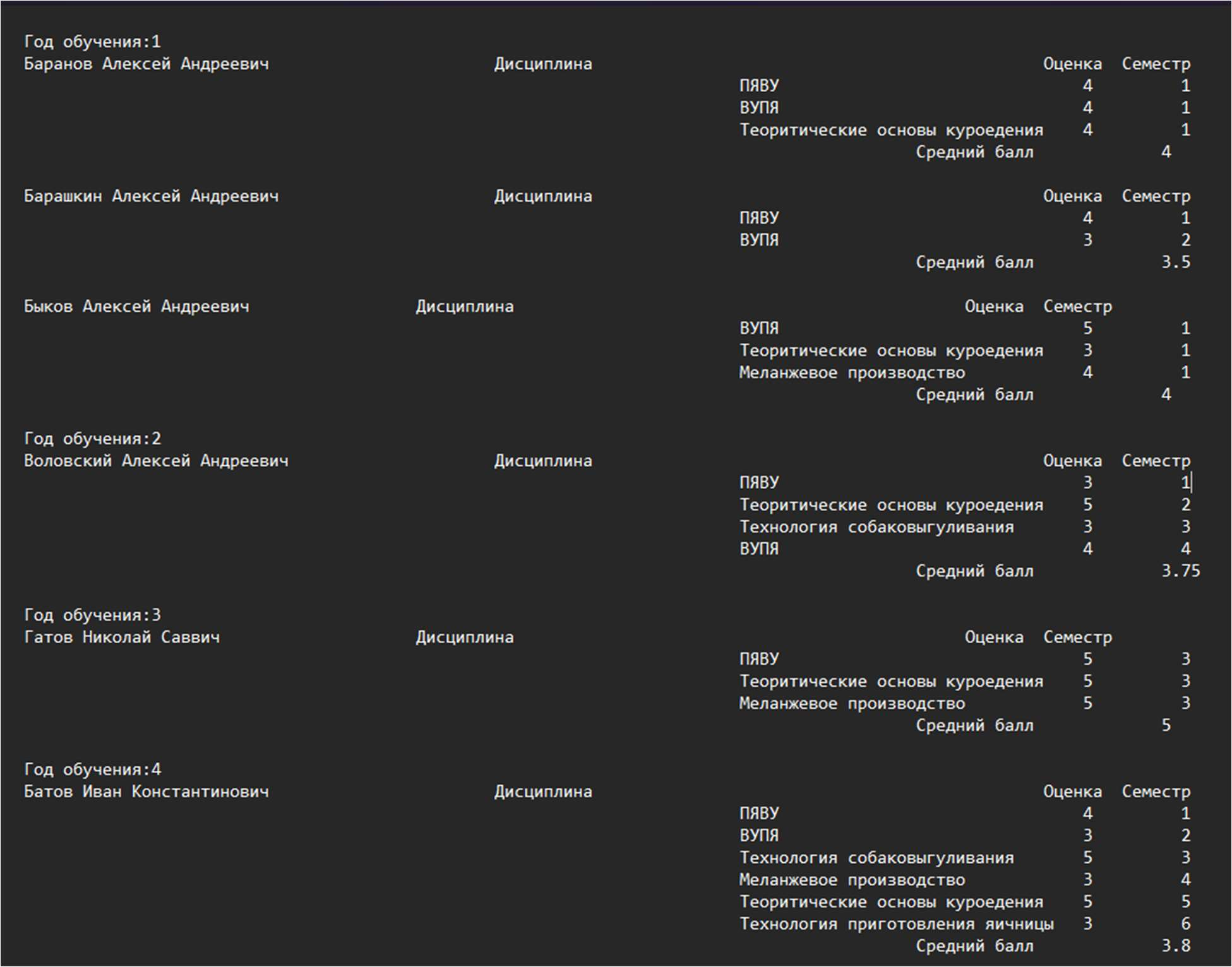


Рисунок 4.11 – успешное выполнение программы

# 5 Выводы

В ходе выполнение лабораторной работы я научился работать с языком Python, подключаться через библиотеку к базе данных, получать данные из таблиц, заносить данные в таблицы.