**2023/11/01 00:00|Лекция. Практика. Часть 3.**

## Практика. Часть 3

Приступаем к заключительной части нашей практики по классам и объектам. **Это завершение создания некой формы для регистрации пользователя.** На предыдущем уроке мы дали пользователю возможность выбирать действие: вход или регистрация. Создали некую переменную, куда пользователь будет вводить свой выбор. Однако логики, которая бы дальше давала нашей программе возможность определять действия нет. Поэтому мы продолжим именно с этого момента.

Пользователь, по нашей задумке, будет вводить номер действия. Так как все готово для регистрации пользователя, давайте начнем с того, что реализуем условие, где пользователь будет выбирать регистрацию. Для этого просто добавим блок if(рис.1) и немного сдвинем код. По сути, уже все готово. Обратите внимание, что код немного вылезает за границы. Так делать не рекомендуется. Можно это перенести на следующую строчку и продолжить(рис.2).

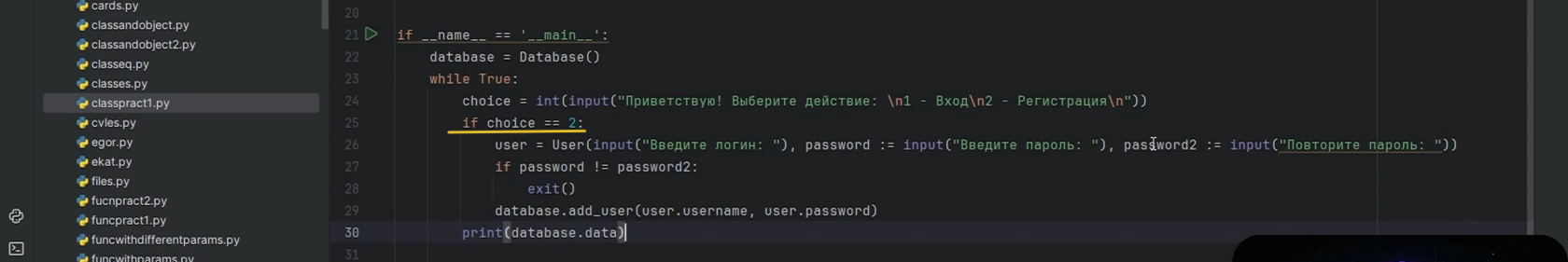


Рис.1

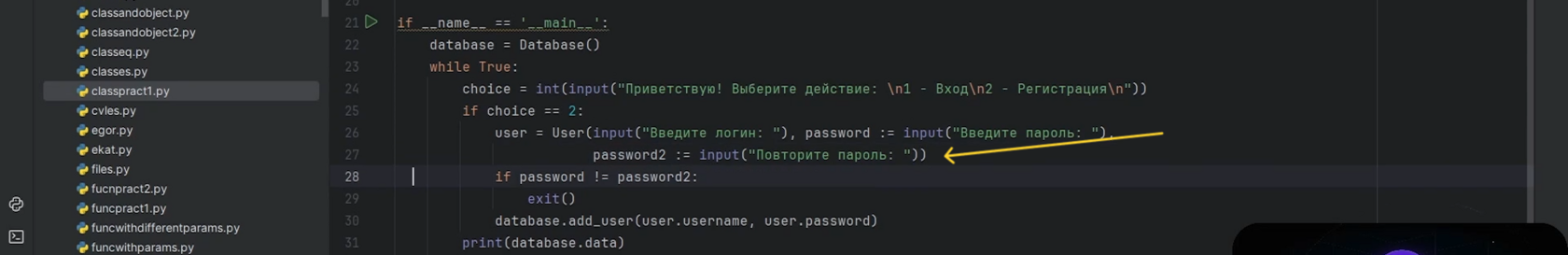


Рис.2

**Дальше приступим уже к реализации логики входа.**По сути, когда пользователь выбирает вход, во-первых, нам нужно дать пользователю возможность ввести логин и пароль, а дальше начинается самое интересное. Если пользователь выбирает действие, связанное с логином, давайте создадим переменную login, куда будем просить ввести логин и переменную password, куда будем просить ввести пароль(рис.3).

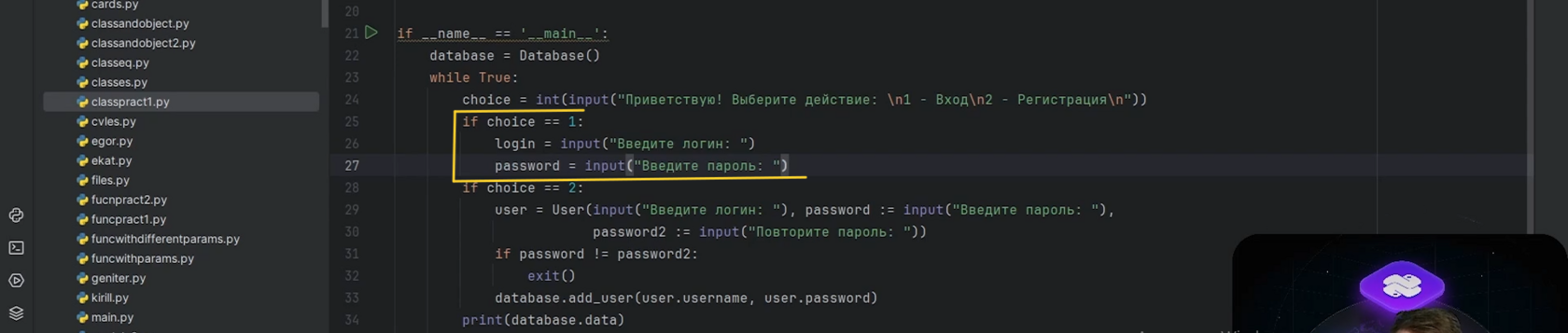


Рис.3

Далее, правильно сначала будет **проверить наличие логина в нашем хранилище данных**, в нашей базе данных, потому что, если введённого логина не окажется, будем получать ошибку. Для этого создадим условие, где будем проверять, что введённый пользователем логин находится в нашем словарике с данными, то будем выводить ‘ok’(рис.4). Это пока для проверки. Пользователь выбирает вход, вводит логин, вводит пароль, но не вышла строчка ‘ok’, то есть у нас ничего не произошло(рис.5). Данные у нас сейчас тоже пустые, потому что условие выбора регистрации у нас не выполнилось.

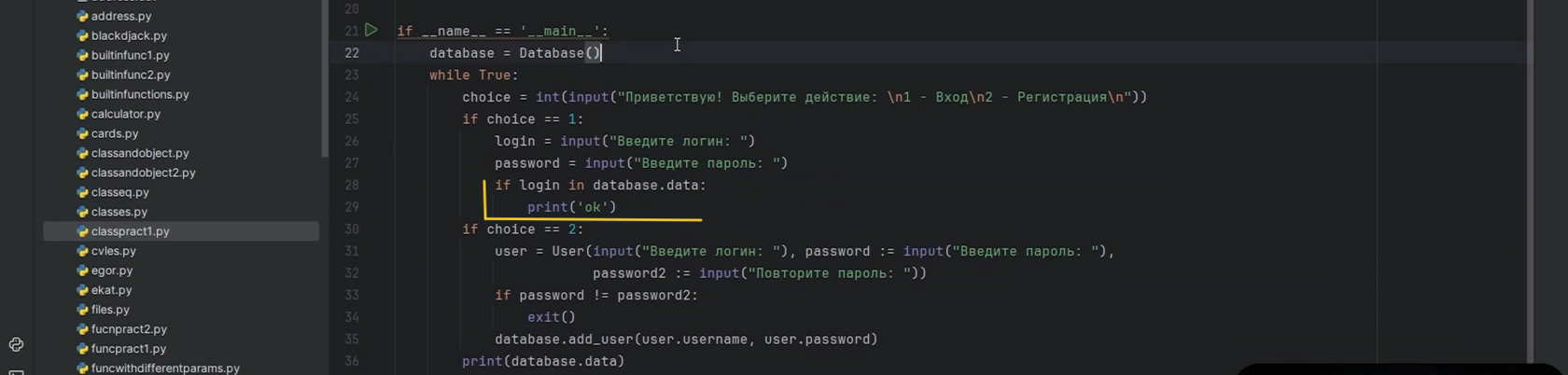


Рис.4

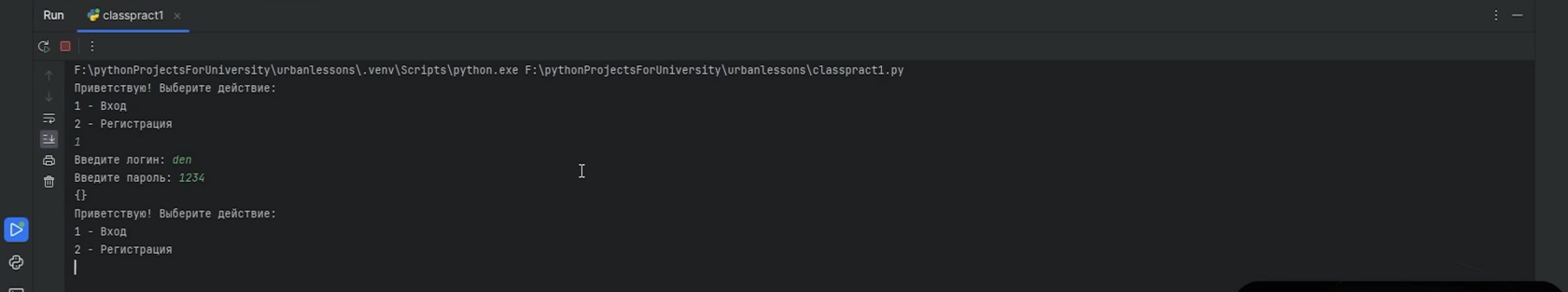


Рис.5

Давайте **сделаем регистрацию**, то есть создадим этого пользователя den, введём ему пароль. Теперь он появился в нашей базе данных(рис.6) и попробуем снова выполнить вход. Введём логин, введём пароль и видим вот этот print(‘ok’)(рис.7), значит, проверка наличия пользователя у нас работает. Однако в 1 выборе добавим условие else, что “Пользователь не найден”. То есть, если человеку придёт действие для входа, и он введет несуществующие в базе данные, то увидим сообщение, что “Пользователь не найден”(рис.8).

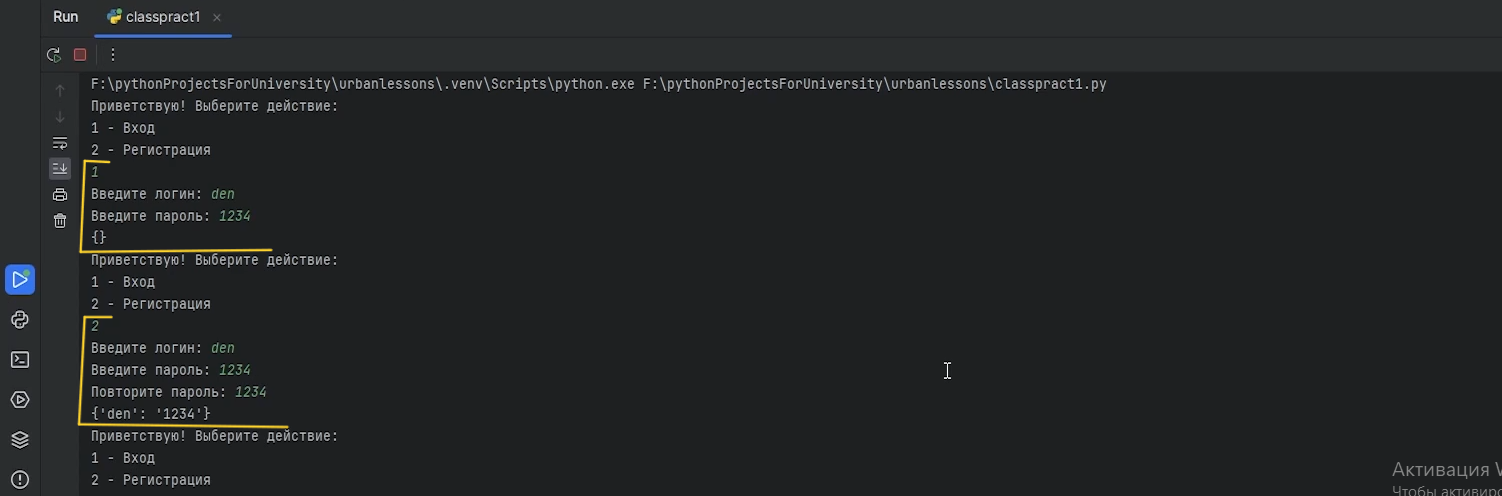


Рис.6



Рис.7

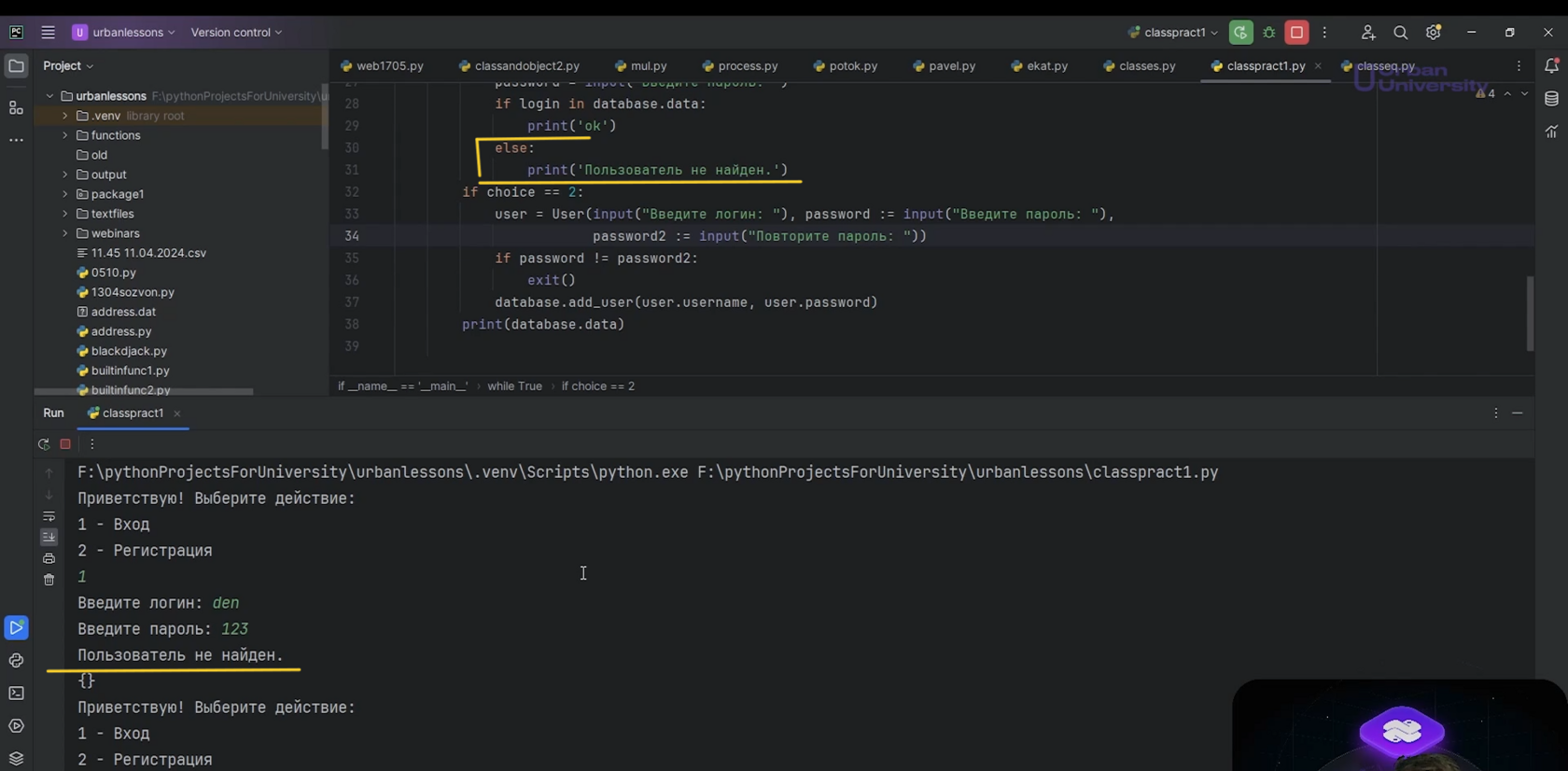


Рис.8

Дальше наша программа возвращается в начало и снова просит выбрать действие. Наша база данных все ещё пустая. У нас пользователь не создаётся. Однако все-таки, если пользователь оказался в нашем словарике, то есть существует такой логин, нам нужно проверить значение, которое будет храниться вместе с заданным логином и сравнить его с введённым паролем. Здесь **будем проверять, что введённый пользователем пароль совпадает с тем паролем, хранящимся под ключом с логином нашего пользователя** (if password = database.data[login])(рис.9). Также выведем информацию о том, что “Вход выполнен” и будем выводить нашего пользователя.

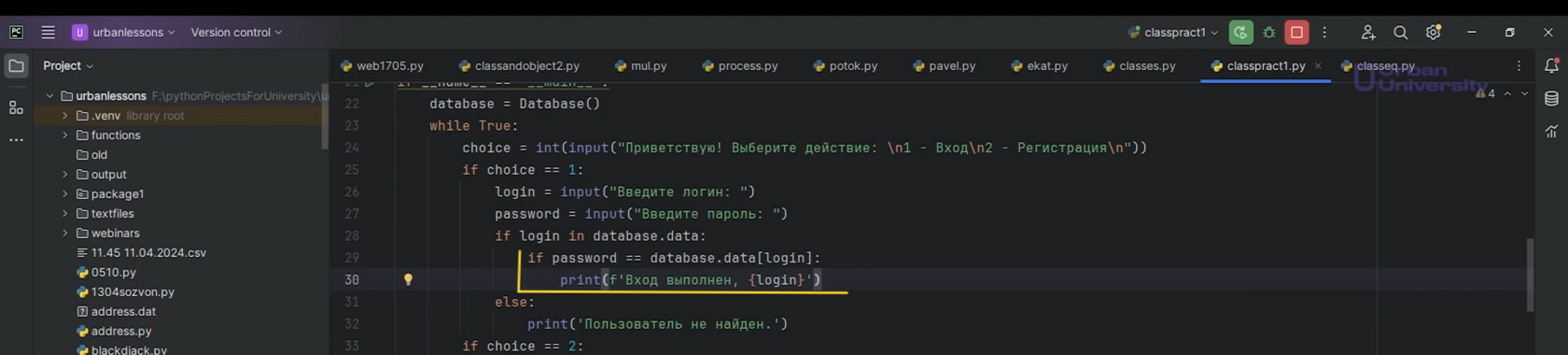


Рис.9

Запускаем, выбираем регистрация, создаём пользователя, есть у нас пользователь den с паролем. Давайте выполним вход, где введём логин den, введём пароль. Видим сообщение “Вход выполнен”(рис.10). После того как вход выполнен можно остановить программу (break)(рис.11), однако это все ещё не совсем корректно. Потому что, если все-таки логин такой есть и пароли совпали, у нас здесь будет выводиться сообщение о том, что “Вход выполнен”. Но если пароли не совпадают, действия у нас нет, поэтому будем говорить, что “Пользователь ввёл неверный пароль”(рис.11).

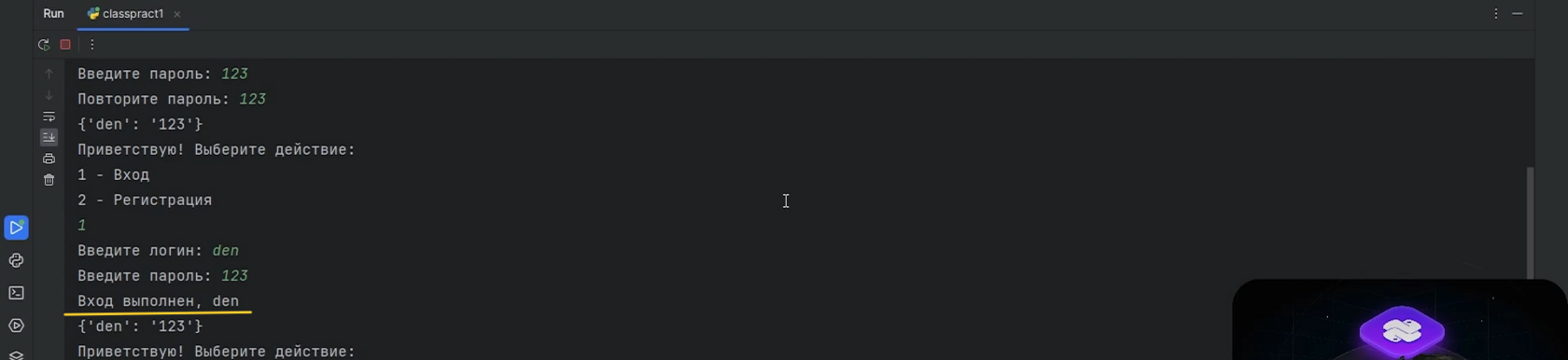


Рис.10

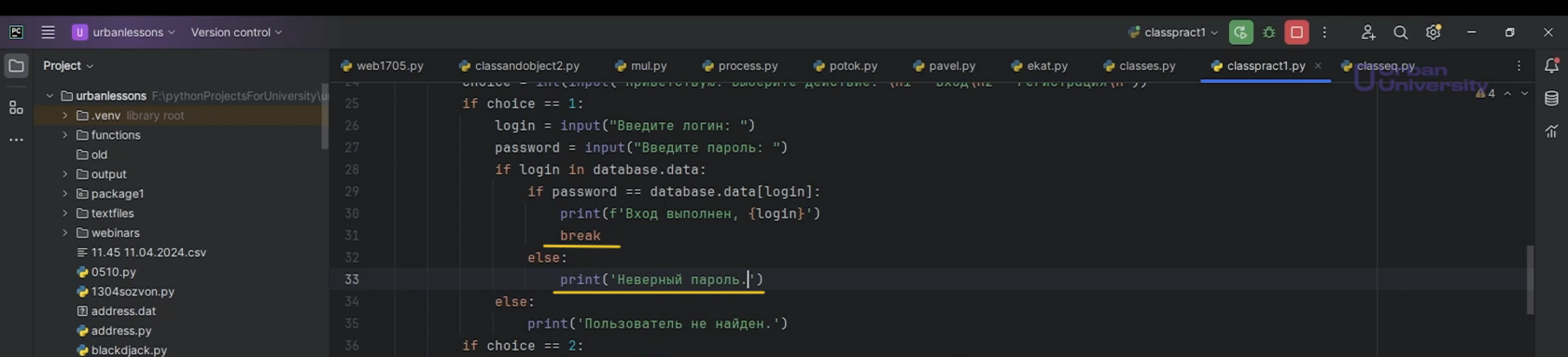


Рис.11

**Давайте поэкспериментируем.** Снова создадим пользователя, введём логин, введём пароль. Видим, что у нас создался пользователь den, и давайте попробуем на него зайти, однако в этот раз введём разные пароли. Вышло сообщение, что наш пароль неверный(рис.12). Вот теперь все работает правильно.



Рис.12

В случае регистрации пользователя, если он ввёл неверный пароль, выход из программы следует заменить на continue(рис.13), чтобы у нас программа не завершалась, а просто переходила к следующему повторению нашего цикла.  
Выбираем регистрацию, вводим логин, вводим пароль, вводим пароль, который не повторяется, и видим нормальную реакцию программы(рис.14). Также следует написать, что “Пароли не совпадают, попробуйте ещё раз”. Вот теперь все будет так, как надо.

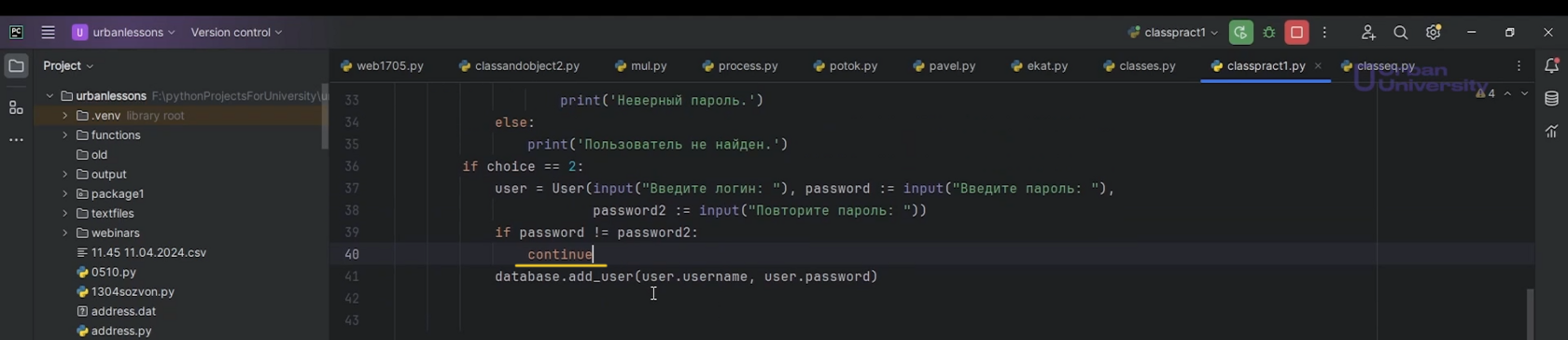


Рис.13

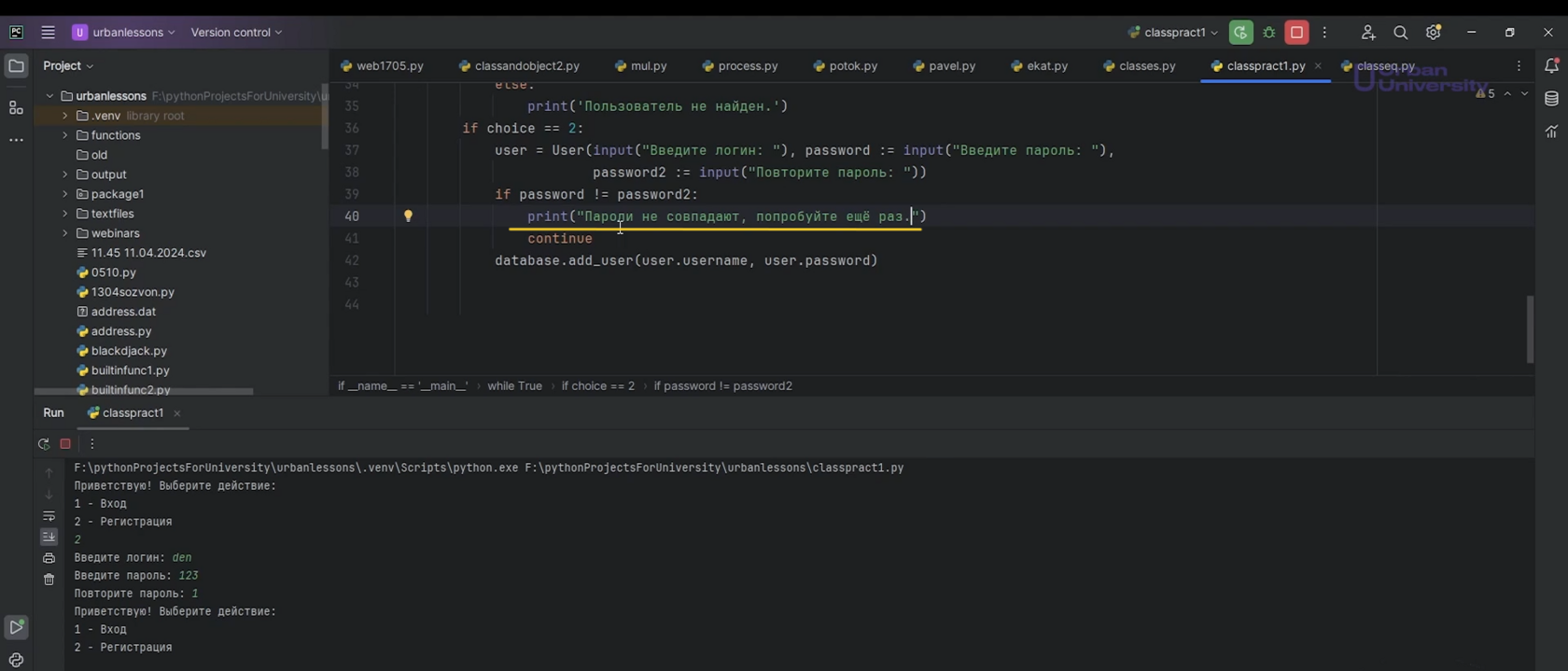


Рис.14

В принципе, это достаточно интересная практика. Данная система регистрации и входа пользователя ни в коем случае не претендует на какое-то реальное применение. Как правило, те инструменты, в которых нам это нужно, имеют уже готовую систему для всей этой истории, однако в ваших Pet-проектах (хобби-проектах) собственных, в качестве практики это вполне можно использовать.

В будущем вы будете уметь работать с файлами, что позволит вам создавать некое хранилище данных. Оно будет работать не в рамках одной сессии, не в рамках одного запуска программы. Это либо создавать какой-нибудь текстовый файл или более удобная и правильная реализация — это использование какой-либо базы данных. На самом деле этот проект, он сильно масштабируемый, то есть вы можете сюда вносить много собственных изменений. Однако наша задача была отработать работу с классами, работу с объектами. Также тут умудрились ещё применить те конструкции, которые осваивали ранее. Единственное, что данная реализация не подразумевает создание объектов одного класса в другом. Здесь она получилась достаточно упрощённая, но, тем не менее, вы для себя можете попробовать завернуть всю эту логику, например, внутрь одного класса. Можете добавить ещё какой-нибудь класс, который будет определять как раз форму ввода, то есть сделать прям все в виде классов, и здесь оставить только создание экземпляров этих классов. Всю логику вы можете вместить, например, в конструктор одного из этих классов.

На протяжении всего вашего пути как программиста вы будете постоянно работать с классами, однако не стоит забывать важность базовых конструкций. Задача состоит в том, чтобы у вас не уходило много времени при продумывании логики программы, и вы не тратили время на размышление, стоит ли применить ту или иную конструкцию, а уже хотя бы заведомо понимали, видели у себя картинку в голове, что вам стоит здесь написать и как это будет работать. На этом наша практика заканчивается.