**2023/10/30 00:00|Лекция. Практика. Часть 1. (Система регистрации на классах)**

## Практика. Часть 1 (Система регистрации на классах)

Данное занятие начинает серию уроков, посвящённых именно практике. В качестве примера разберем создание системы регистрации пользователя, однако с применением класса.

Для начала давайте определим, какие сущности у нас могут быть для реализации данной системы. Первое, что стоит выделить — это, скорее всего, **пользователь**, поэтому давайте начнём с него. Для реализации пользователя будем использовать уже привычную для нас, скажем так, схему —**это User.** При создании пользователя можем выделить несколько атрибутов: username, password, то есть у пользователя определённый должен логин и пароль(рис.1).

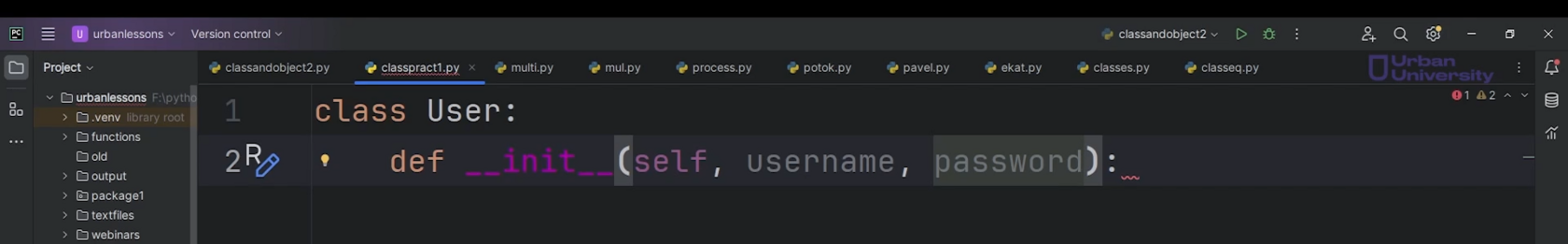


Рис.1

Давайте сделаем так, чтобы эти поля у нас задавались при создании объекта(рис.2). Интерпретатор подсказывает нам email, однако сделаем посложнее. При создании объекта пользователя у нас будет только задаваться логин и пароль, а уже потом, если захотим как-то редактировать профиль пользователя, мы уже будем работать с какими-то методами. Стоит обратить внимание, что одного пароля нам недостаточно. Необходимо ещё подтверждение пароля. Здесь давайте создадим password\_confirm, но добавим условие в конструкторе: если наш пароль будет совпадать с подтверждением пароля, только тогда будем выдавать этот самый password(рис.3). Пользователь в принципе готов.

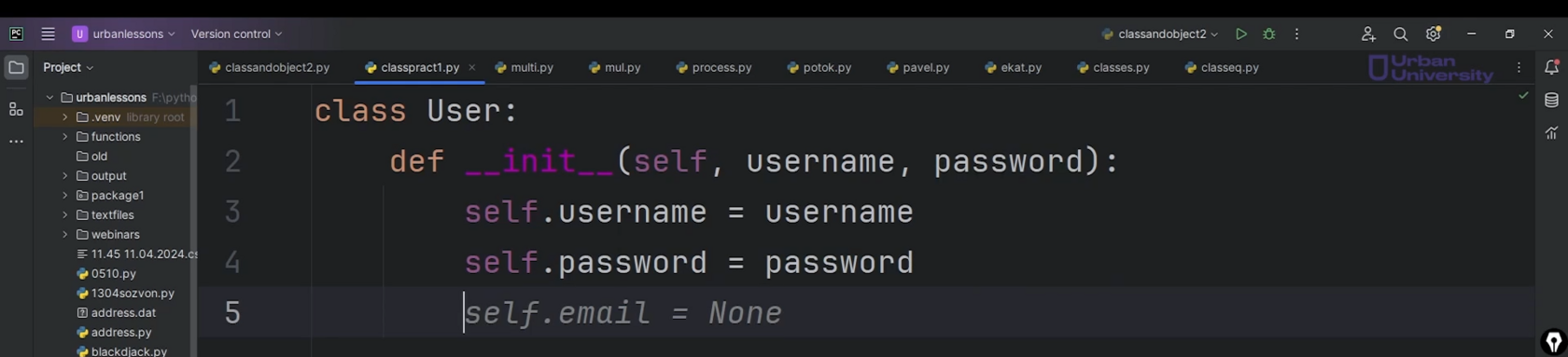


Рис.2

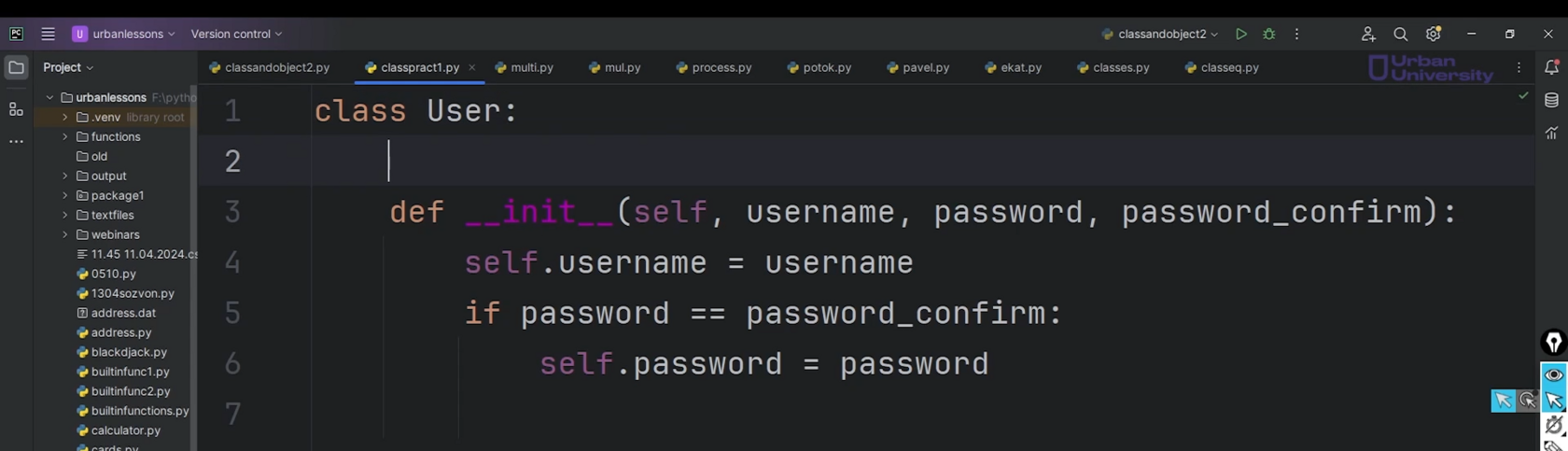


Рис.3

Познакомимся с вами с одной особенностью классов, очень схожей с функциями — **строка документирования.** В ней будем выносить информацию о классах, чтобы потом, если через определённое время откроем код, чтобы нам не пришлось разбираться досконально и могли сразу увидеть какое-то описание этого класса, которое поможет нам в этом разобраться. Когда эти сущности будем передавать кому-то, например, товарищу, коллеге и так далее, чтобы он тоже мог почитать такую коротенькую документацию. Здесь напишем: “Класс пользователя, содержащий атрибуты: логин и пароль”(рис.4).

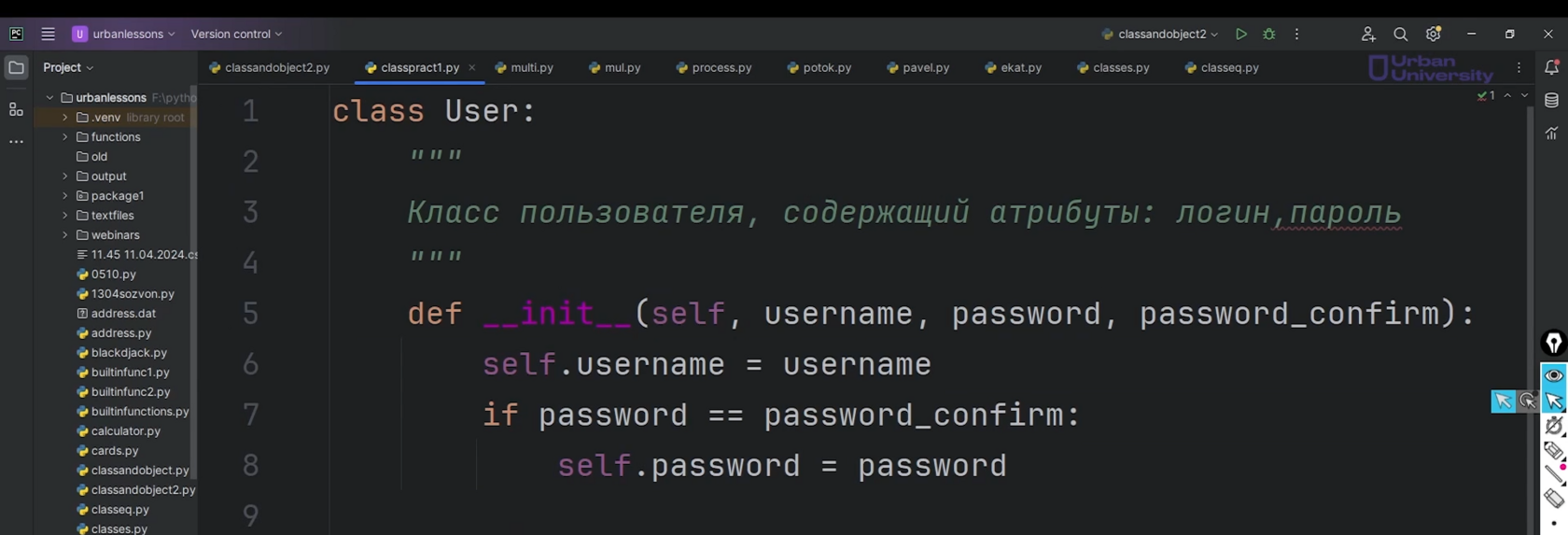


Рис.4

**Давайте посмотрим в интерактивном режиме как обратиться к этой строке.**Запустим. Вот у нас есть наш класс пользователя(рис.5). Обратиться к строке документирования можем, взяв этот класс и обратившись к методу doc(). Здесь мы видим: “Класс пользователя, содержащий атрибуты: логин и пароль” — это и есть наша строка документирования(рис.6).

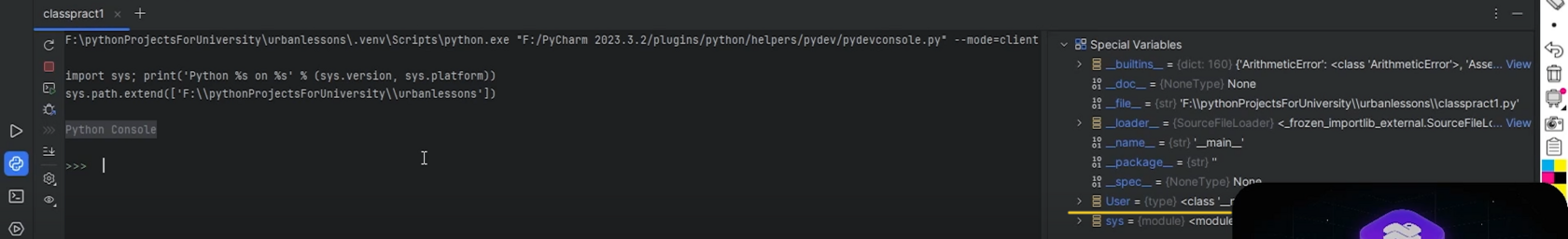


Рис.5

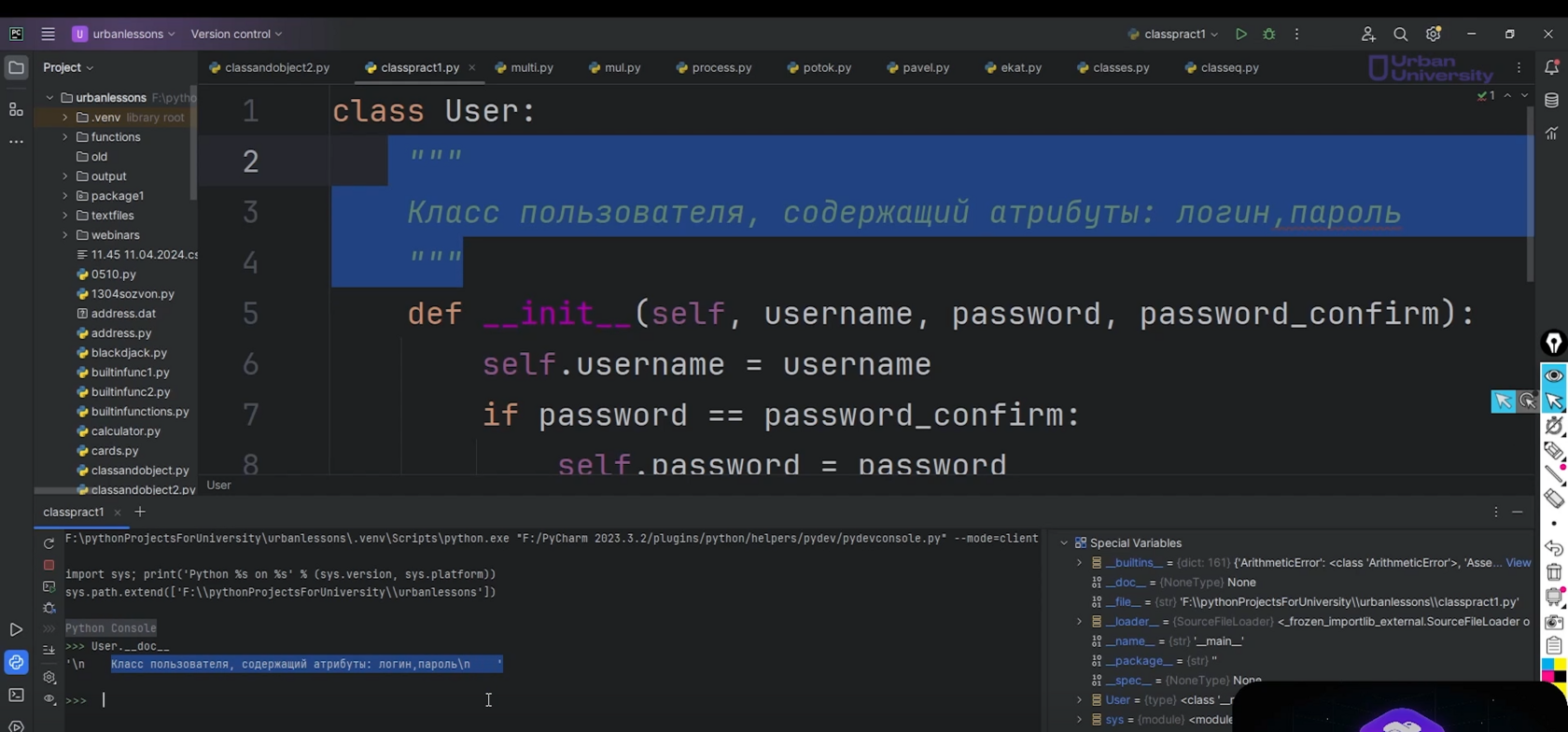


Рис.6

Далее нам понадобится какая-то база данных, поэтому ее будем делать как отдельное представление, отдельная сущность. Она тоже будет создаваться, однако будет иметь атрибут data у экземпляра в виде пустого словаря(рис.7). Также создадим метод add\_user() в базу данных, который будет принимать username и password, брать наш словарик, обращаться по ключу с логином пользователя и сохранять туда значение пароля(рис.8). Это нам пригодится в будущем для проверки.



Рис.7

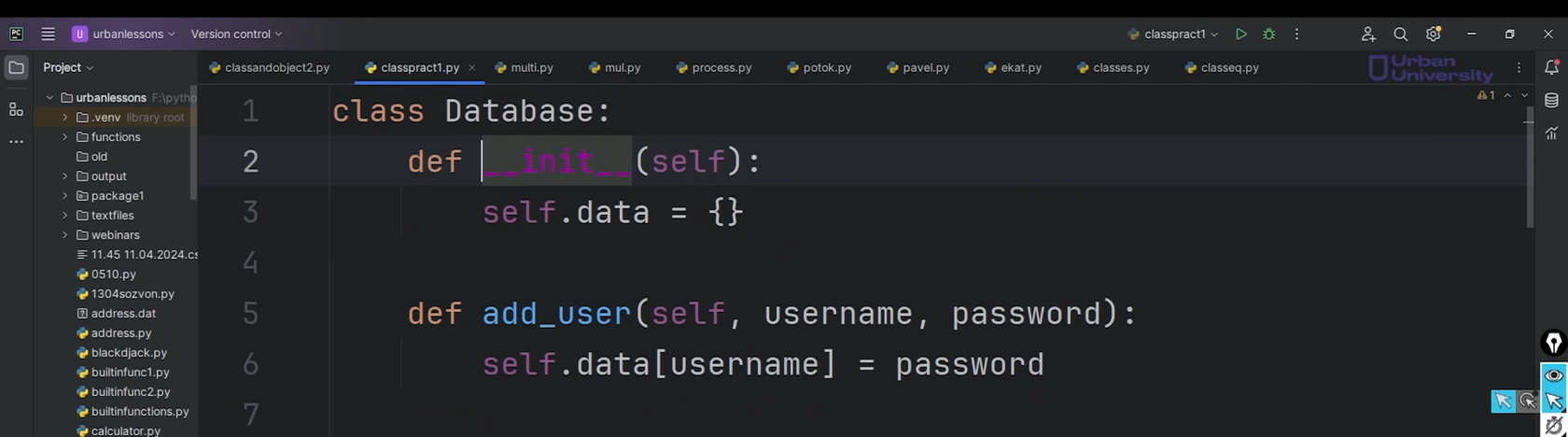


Рис.8

**Попробуем создать пользователя.**Здесь давайте создадим конструкцию if \_\_name\_\_ == ‘\_\_main\_\_’, где создадим базу данных и будем запрашивать ввод пользователя(рис.9). Можем это сделать либо создав отдельные переменные, либо же user = User(), создаём экземпляр класса user и сюда будем вставлять функцию input(), то есть запрашивать пользовательский ввод. Спрашивать пользовательский ввод будем 3 раза, однако здесь делаем пометки: “Введите логин:”, “Введите пароль:”, “Повторите пароль:”(рис.9).

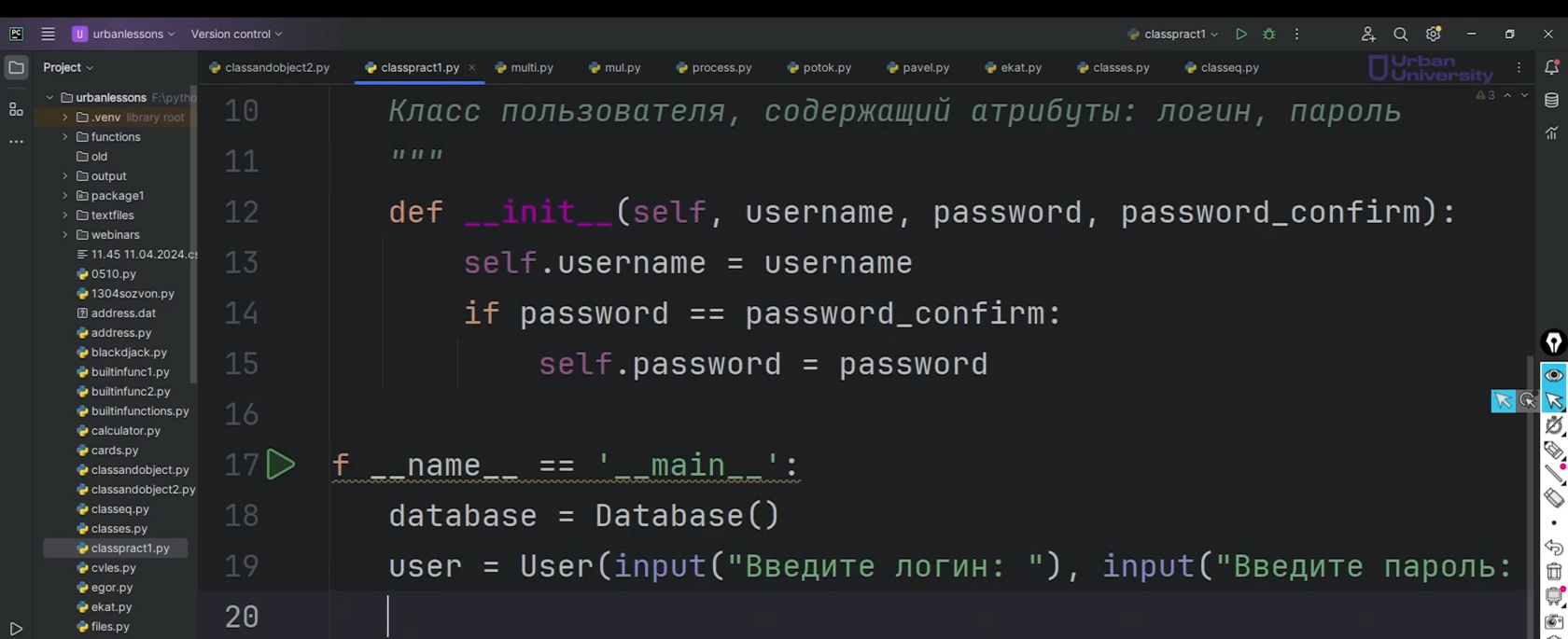


Рис.9

Запускаем нашу программу. Видим “Введите логин”, вводим логин, далее вводим пароль, повторяем пароль(рис.10). Все работает как надо.

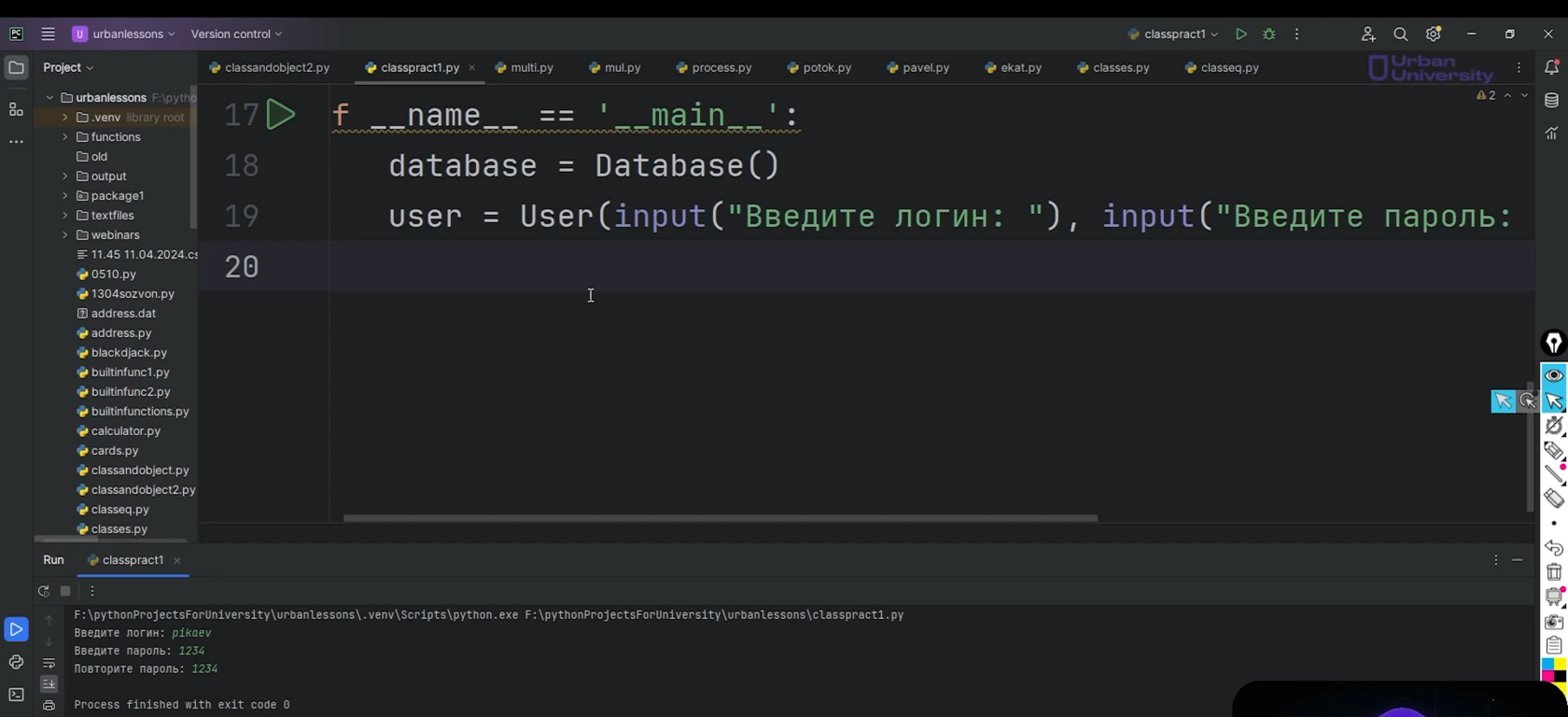


Рис.10

**Попробуем это в интерактивном режиме запустить.** На первом этапе происходит ввод логина, далее пароля и повтор пароля. После всех этих операций видим, что у нас появился экземпляр класса database, содержащий данные, но пока они у нас пустые и экземпляр класса user, который содержит логин пользователя и пароль(рис.11).

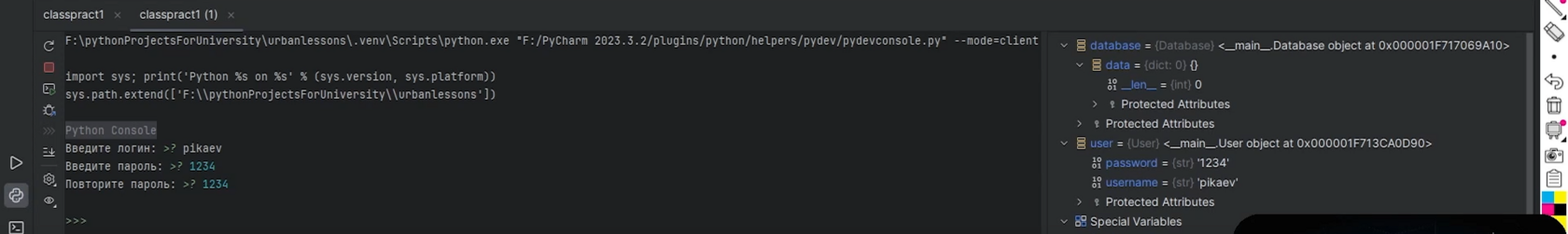


Рис.11

Добавим нашего пользователя в базу данных (database), то есть возьмём наш объект, базу данных и вызовем метод add\_user, куда из user передадим username и пароль(рис.12). Получается словарик. **Запускаем снова нашу программу в интерактивном режиме.** Давайте повторим те же самые действия. Что мы видим? Пользователь остался точно таким же, как и при предыдущем запуске, однако база данных теперь содержит пользователя с логином и паролем(рис.13). В будущем это пригодится для проверки.

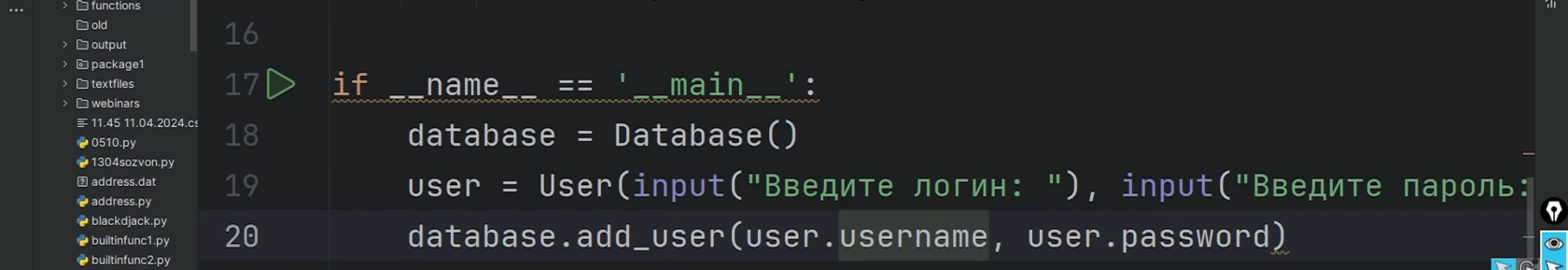


Рис.12

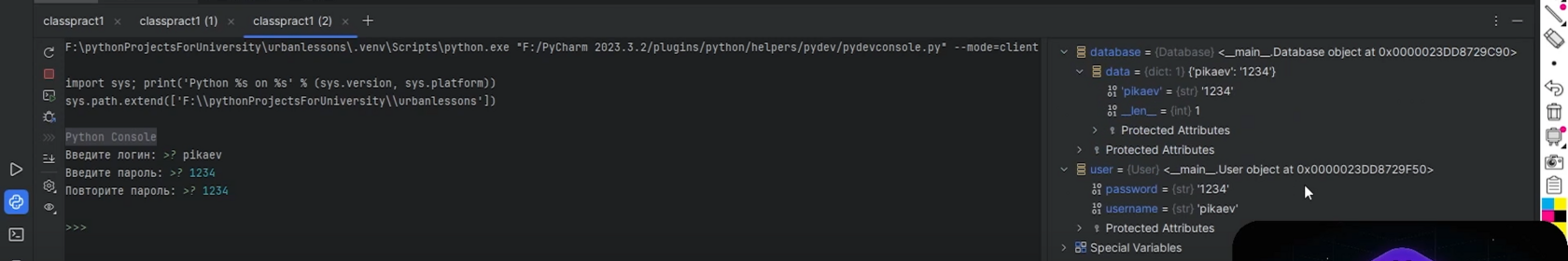


Рис.13

В принципе, на текущем этапе не сильно тяжело, так как приложили мало усилий, потратили мало времени, но у нас уже вырисовывается что-то похожее на систему регистрации пользователя. Как начало — это уже достаточно большой прогресс. Мы используем классы для реализации отдельных сущностей, то есть это: база данных, пользователи. В будущем можно будет навесить различные методы, переопределить, чтобы размер базы данных определялся, чтобы могли проверять, есть ли пользователь в базе данных или нет. Соответственно, потом добавим выборку действий, то есть можно сделать отдельную форму, создать сущность, которая будет организовывать вот эту форму регистрации и попробуем создавать отдельные экземпляры уже внутри другого класса.

Каждый из вас может видоизменять эту систему регистрации, потому что единственного верного решения не существует. Это просто 1 из возможных вариантов реализации такой вот задачки.