МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Кафедра инфокоммуникаций

Институт цифрового развития

ОТЧЁТ

по лабораторной работе №2.15

Дисциплина: «Основы программной инженерии»

Тема: «Работа с файлами в языке Python»

Выполнил: студент 2 курса

группы Пиж-б-о-21-1

Рязанцев Матвей Денисович

Ставрополь 2023

Цель работы: приобретение навыков по работе с текстовыми файлами при написании программ с помощью языка программирования Python версии 3.x, изучение основных методов модуля os для работы с файловой системой, получение аргументов командной строки

Вариант 25 (4)

Выполнение работы

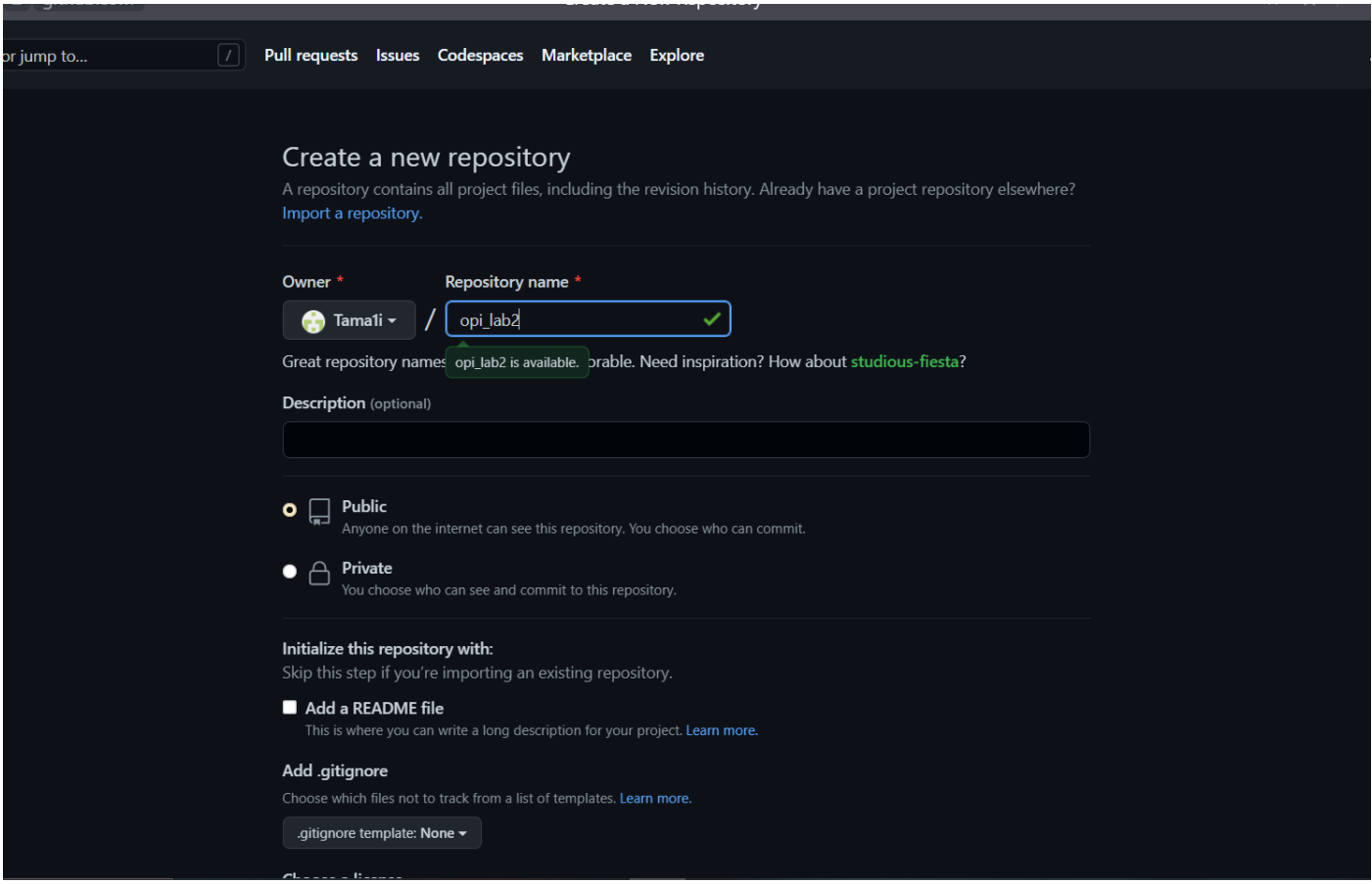

Рисунок 1 -создание репозитория

```
D:\2kurs\!22kurs\opi>git flow init
Initialized empty Git repository in D:/2kurs/!22kurs/opi/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [D:/2kurs/!22kurs/opi/.git/hooks]
```

Рисунок 2 – организация репозитория по модели git flow

```
C:\Users\ryaza>pip --version
pip 22.2.2 from D:\Anaconda\lib\site-packages\pip (python 3.9)

C:\Users\ryaza>
```

Рисунок 3 – версия утилиты pip

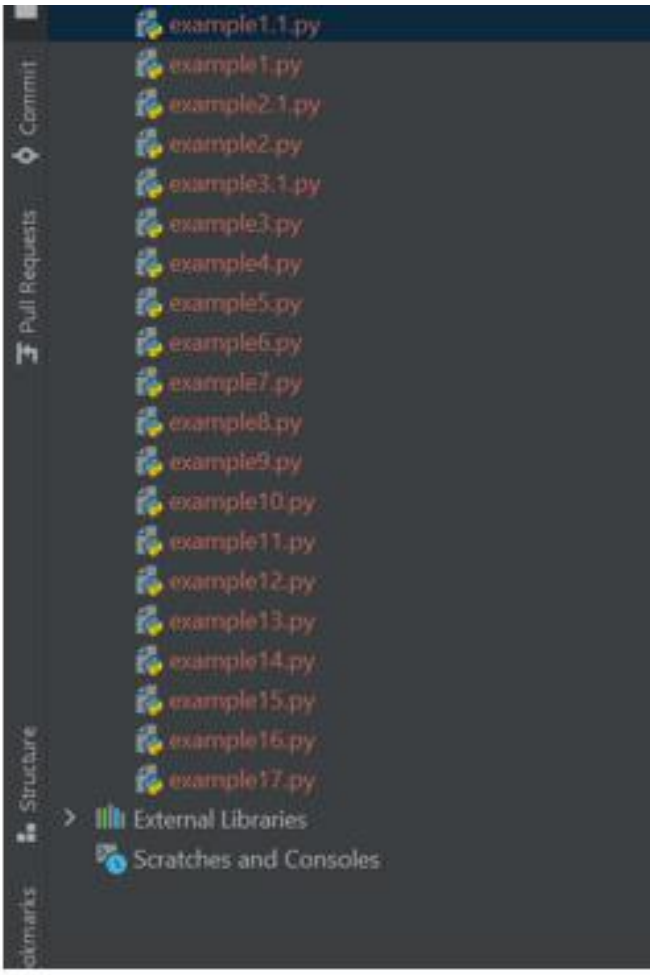

Рисунок 4 – проработанные примеры

```
D:\2kurs\!22kurs\opi_norm\lab2>python -m venv env

D:\2kurs\!22kurs\opi_norm\lab2>env bin activate
'env' is not recognized as an internal or external command,
operable program or batch file.

D:\2kurs\!22kurs\opi_norm\lab2>.\env\Scripts\activate

(env) D:\2kurs\!22kurs\opi_norm\lab2>
```

Рисунок 4 – активация виртуального окружения

```
(env) D:\2kurs\!22kurs\opi_norm\lab2>pip install black
Requirement already satisfied: black in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (23.1.0)
Requirement already satisfied: mypy-extensions>=0.4.3 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from blac
k) (1.0.0)
Requirement already satisfied: pathspec>=0.9.0 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from black) (0.1
1.0)
Requirement already satisfied: packaging>=22.0 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from black) (23.
0)
Requirement already satisfied: typing-extensions>=3.10.0.0 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from
 black) (4.5.0)
Requirement already satisfied: platformdirs>=2 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from black) (3.0
.0)
Requirement already satisfied: click>=8.0.0 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from black) (8.1.3)

Requirement already satisfied: tomli>=1.1.0 in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from black) (2.0.1)

Requirement already satisfied: colorama in d:\2kurs\!22kurs\opi_norm\lab2\env\lib\site-packages (from click>=8.0.0->blac
k) (0.4.6)

(env) D:\2kurs\!22kurs\opi_norm\lab2>
```

Рисунок 5 – установка пакета black

```
(env) D:\2kurs\!22kurs\opi_norm\lab2>deactivate
D:\2kurs\!22kurs\opi_norm\lab2>
```

Рисунок 6 – деактивация виртуального окружения

Индивидуальное задание

4. Написать программу, которая считывает английский текст из файла и выводит на экран слова, начинающиеся с гласных букв.

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
def gl(f):
    if ((f == "a") or (f == "e") or (f == "y") or (f == "u") or
            (f == "i") or (f == "o") or (f == "j")):
        return 1

if __name__ == "__main__":
    file = open("f1.txt", encoding='UTF')
    f = file.read()
    f = f.lower()
    f = " " + f + "\n"
    s = ""
    i = 1
    ot = ""
    for i in range(len(f)-1):
        if (f[i-1] == " ") or (f[i-1] == "\n"):
            k = 1
            if (gl(f[i]) == 1):
                k = 2
        if k == 2:
            s = s + f[i]
            if (f[i] == " ") or (f[i] == "\n"):
                k = 0
                ot = s + " "

    print(ot)
```

Рисунок 5 – код программы

```
D:\2kurs\!22kurs\opi\merim\Scripts\python.exe D:\2kurs\!22kurs\opi_norm\lb2idz\
aespa idle eef ewgwv
```

Рисунок 6 – результат работы программы

4. Напишите программу, которая будет считывать содержимое файла, добавлять к считанным строкам порядковый номер и сохранять их в таком виде в новом файле. Имя исходного файла необходимо запросить у пользователя, так же, как и имя целевого файла. Каждая строка в созданном файле должна начинаться с ее номера, двоеточия и пробела, после чего должен идти текст строки из исходного файла.

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    re = str(input("ishodnic - "))
    wr = str(input("cuda - "))
    with open(re,"r", encoding="utf-8") as file:
        qe = open(wr,"a")
        w = 1
        q = ""
        for i in file:
            q = str(w) + "; " + i
            w += 1
            qe.write(q+"\n")
```

Рисунок 7 – код программы

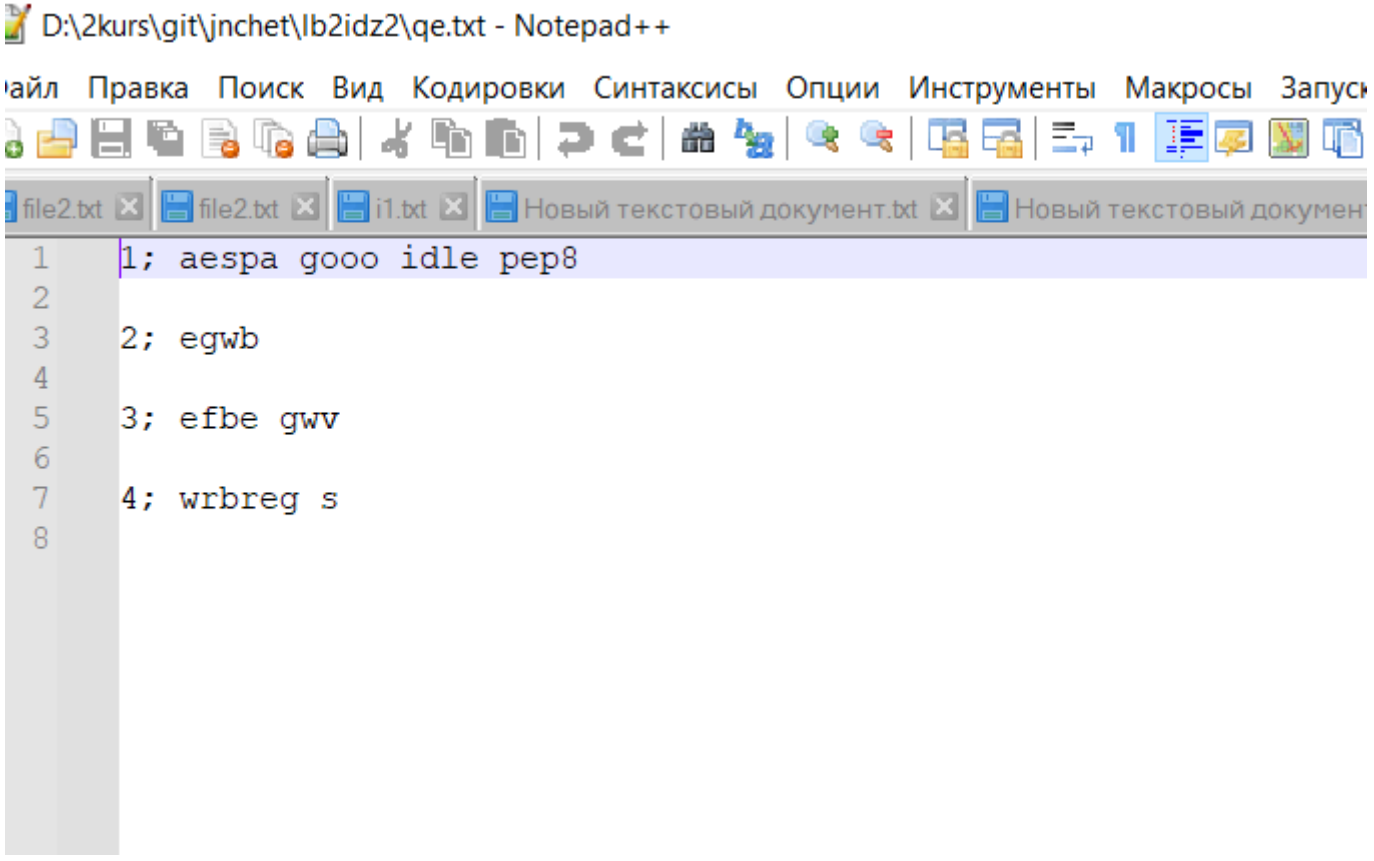

Рисунок 7 – результат работы программы

1. Каким способом можно установить пакет Python, не входящий в стандартную библиотеку?

Существует так называемый Python Package Index (PyPI) – это репозиторий, открытый для всех Python разработчиков, в нем вы можете найти пакеты для решения практически любых задач.

2. Как осуществить установку менеджера пакетов pip?

При развертывании современной версии Python, pip устанавливается ав-томатически. Но если, по какой-то причине, pip не установлен на вашем ПК, то сделать это можно вручную. Чтобы установить pip, нужно скачать скрипт get-pip.py и выполнить его.

3. Откуда менеджер пакетов pip по умолчанию устанавливает пакеты?

По умолчанию менеджер пакетов pip скачивает пакеты из Python Package Index (PyPI).

4. Как установить последнюю версию пакета с помощью pip?С помощью команды $ pip install ProjectName.

5. Как установить заданную версию пакета с помощью pip? С помощью команды $ pip install ProjectName==3.2, где вместо 3.2 необходимо указать нужную версию пакета.

6. Как установить пакет из git репозитория (в том числе GitHub) с помощью pip?

С помощью команды $ pip install e git+https://gitrepo.com/ ProjectName.git

7. Как установить пакет из локальной директории с помощью pip?С помощью команды $ pip install ./dist/ProjectName.tar.gz

8. Как удалить установленный пакет с помощью pip?С помощью команды $ pip uninstall ProjectName можно удалить установленный пакет.

9. Как обновить установленный пакет с помощью pip?С помощью команды $ pip install --upgrade ProjectName можно обновить необходимый пакет.

10. Как отобразить список установленных пакетов с помощью pip?Командой $ pip list можно отобразить список установленных пакетов.

11. Каковы причины появления виртуальных окружений в языке Python?
Существует несколько причин появления виртуальных окружений в языке Python - проблема обратной совместимости и проблема коллективной разработки. Проблема обратной совмести - некоторые операционные системы, например, Linux и MacOs используют содержащиеся в них предустановлен-ные интерпретаторы Python. Обновив или изменив самостоятельно версию ка-когото установленного глобально пакета, мы можем непреднамеренно сло-мать работу утилит и приложений из дистрибутива операционной системы.Проблема коллективной разработки - Если разработчик работает над проектом не один, а с командой, ему нужно передавать и получать список зависимостей, а также обновлять их на своем компьютере таким образом, чтобы не нарушалась работа других его проектов. Значит нам нужен механизм, который вместе с обменом проектами быстро устанавливал бы локально и все необходимые для них пакеты, при этом не мешая работе других проектов.

12. Каковы основные этапы работы с виртуальными окружениями? Основные этапы:Создаём через утилиту новое виртуальное окружение в отдельной папк для выбранной версии интерпретатора Python.Активируем ранее созданное виртуальногоокружения для работы.Работаем в виртуальном окружении, а именно управляем пакетами используя pip и запускаем выполнение кодаДеактивируем после окончания работы виртуальное окружение.Удаляем папку с виртуальным окружением, если оно нам больше не нужно.

13. Как осуществляется работа с виртуальными окружениями с помощью venv? С его помощью можно создать виртуальную среду, в которую можно устанавливать пакеты независимо от основной среды или других виртуальных окружений. Основные действия с виртуальными окружениями с помощью venv: создание виртуального окружения, его активация и деактивация.

14. Как осуществляется работа с виртуальными окружениями с помощью virtualenv?Для начала пакет нужно установить. Установку можно выполнить командой: python3 -m pip install virtualenv Virtualenv позволяет создать абсолютно изолированное

виртуальное окружение для каждой из программ.Окружением является обычная директория, которая содержит копию всего необходимого для запуска определенной программы, включая копию са-мого интерпретатора, полной стандартной библиотеки, pip, и, что самое глав-ное, копии всех необходимых пакетов.

15. Изучите работу с виртуальными окружениями pipenv. Как осуществляется работа с виртуальными окружениями pipenv?Для формирования и развертывания пакетных зависимостей используется утилита pip.

Основные возможности pipenv:

– Создание и управление виртуальным окружением

– Синхронизация пакетов в Pipfile при установке и удалении пакетов

– Автоматическая подгрузка переменных окружения из .env файла

После установки pipenv начинается работа с окружением. Его можно создать в любой папке. Достаточно установить любой пакет внутри папки.
Используем requests, он автоматически установит окружение и создаст Pipfile и Pipfile.lock.

16. Каково назначение файла requirements.txt? Как создать этот файл? Какой он имеет формат?Установить пакеты можно с помощью команды: pip install –r requirements.txt. Также можно использовать команду pip freeze > requirements.txt, которая создаст requirements.txt наполнив его названиями и версиями тех пакетов что используются вами в текущем окружении. Это удобно если вы разработали проект и в текущем окружении все работает, но вы хотите перенести проект в иное окружением (например, заказчику или на сервер). С помощью закрепления зависимостей мы можем быть уверены, что пакеты, установленные в нашей производственной среде, будут точно соответствовать пакетам в нашей среде разработки, чтобы ваш проект неожиданно не ломался.

17. В чем преимущества пакетного менеджера conda по сравнению с пакетным менеджером pip?Conda способна управлять пакетами как для Python, так и для C/ C++, R, Ruby, Lua, Scala и других. Conda устанавливает двоичные файлы, поэтому работу по компиляции пакета самостоятельно выполнять не требуется (по сравнению с pip).

18. В какие дистрибутивы Python входит пакетный менеджер conda?Все чаще среди Python-разработчиков заходит речь о менеджере пакетов conda, включенный в состав дистрибутивов Anaconda и Miniconda. JetBrains включил этот инструмент в

состав PyCharm.

19. Как создать виртуальное окружение conda?С помощью команды: conda create -n %PROJ_NAME% python=3.7

20. Как активировать и установить пакеты в виртуальное окруже-ние conda? Чтобы установить пакеты, необходимо воспользоваться командой: –conda install A для активации: conda activate %PROJ_NAME%

21. Как деактивировать и удалить виртуальное окружение conda?Для деактивации использовать команду: conda deactivate, а для удаления: conda remove -n $PROJ_NAME.

22. Каково назначение файла environment.yml? Как создать этот файл? Создание файла: conda env export > environment.ymlФайл environment.yml позволит воссоздать окружение в любой нужный момент.

23. Как создать виртуальное окружение conda с помощью файла environment.yml? Достаточно набрать: conda env create -f environment.yml

24. Самостоятельно изучите средства IDE PyCharm для работы с виртуальными окружениями conda. Опишите порядок работы с виртуальными окружениями conda в IDE PyCharm.Работа с виртуальными окружениями в PyCharm зависит от способа взаимодействия с виртуальным окружением:Создаём проект со своим собственным виртуальным окружением, куда затем будут устанавливаться необходимые библиотеки. Предварительно создаём виртуальное окружение, куда установим нужные библиотеки. И затем при создании проекта в PyCharm можно будет его выбирать, т.е. использовать для нескольких проектов.Для первого способа ход работы следующий: запускаем PyCharm и в окне приветствия выбираем Create New Project.В мастере создания проекта, указываем в поле Location путь расположения создаваемого проекта. Имя конечной директории также является именем проекта. Далее разворачиваем па-раметры окружения, щелкая по Project Interpreter. И выбираем New environment using Virtualenv. Путь расположения окружения генерируется автоматически. И нажимаем на Create. Теперь установим библиотеки, которые будем использовать в программе. С помощью главного меню переходим в настройки

File → Settings. Где переходим в Project: project_name → Project Interpreter. Выходим из настроек. Для запускапрограммы, необходимо создать профиль с конфигурацией. Для этого в верхнем правом углу нажимаем на кнопку Add Configuration. Откроется окно Run/Debug Configurations, где нажимаем на кнопку с плюсом (Add New Configuration) в

правом верхнем углу и выбираем Python. Далее указываем в поле Name имя конфигурации и в поле Script path расположение Python файла с кодом программы. В завершение нажимаем на Apply, затем на OK. Для второго способа необходимо сделать следующее: на экране приветствия в нижнем правом углу через Configure → Settings переходим в настройки. Затем переходим в раздел Project Interpreter.В верхнем правом углу есть кнопка с шестерёнкой, нажимаем на неё и выбираем Add, создавая новое окружение. И указываем расположение для но-вого окружения. Нажимаем на OK. Далее в созданном окружении устанавли-ваем нужные пакеты. И выходим из настроек. В окне приветствия выбираем Create New Project. В мастере создания проекта, указываем имя расположения проекта в поле Location. Разворачиваем параметры окружения, щелкая по Project Interpreter, где выбираем Existing interpreter и указываем нужное нам окружение. Далее создаем конфигурацию запуска программы, также как со-здавали для раннее. После чего можно выполнить программу.

25. Почему файлы requirements.txt и environment.yml должны храниться в репозитории git? Чтобы пользователи, которые скачивают какие-либо программы, скрипты, модули могли без проблем посмотреть, какие пакеты им нужно установить дополнительно для корректной работы. За описание о наличии каких-либо пакетов в среде как раз и отвечают файлы requirements.txt и environment.yml