

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

**ОТЧЕТ**  
**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.14 (17)**  
**дисциплины**  
**«Программирование на Python»**

Выполнил:  
Ибрагимов Муса Айнудинович  
2 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем», очная  
форма обучения

---

(подпись)

Руководитель:  
Воронкин Роман Александрович

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

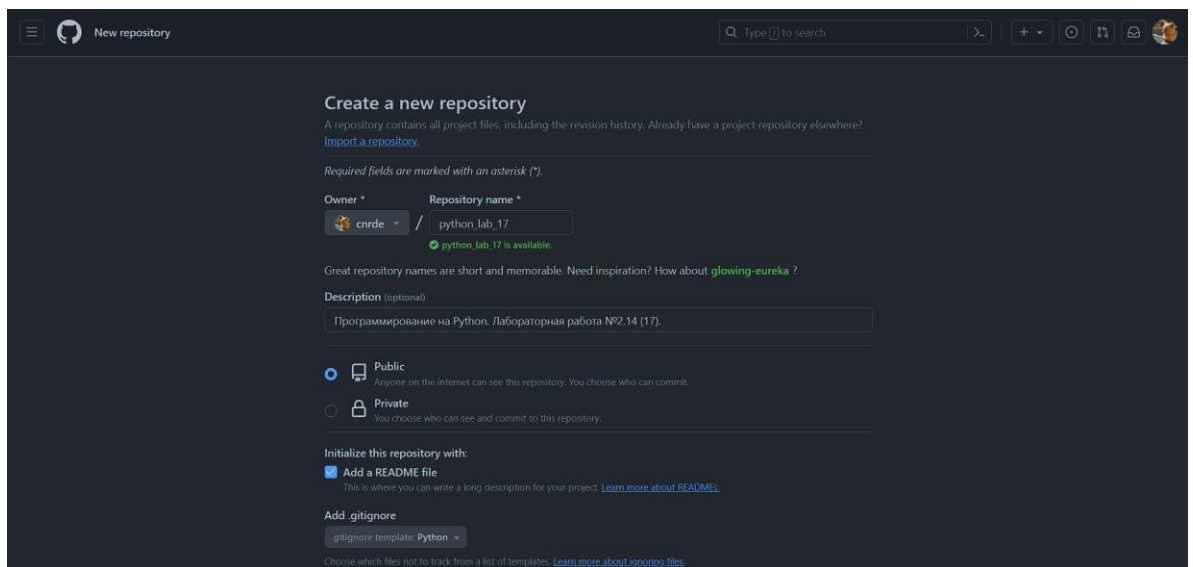
Ставрополь, 2024 г.

Тема: Виртуальное окружение на языке Python.

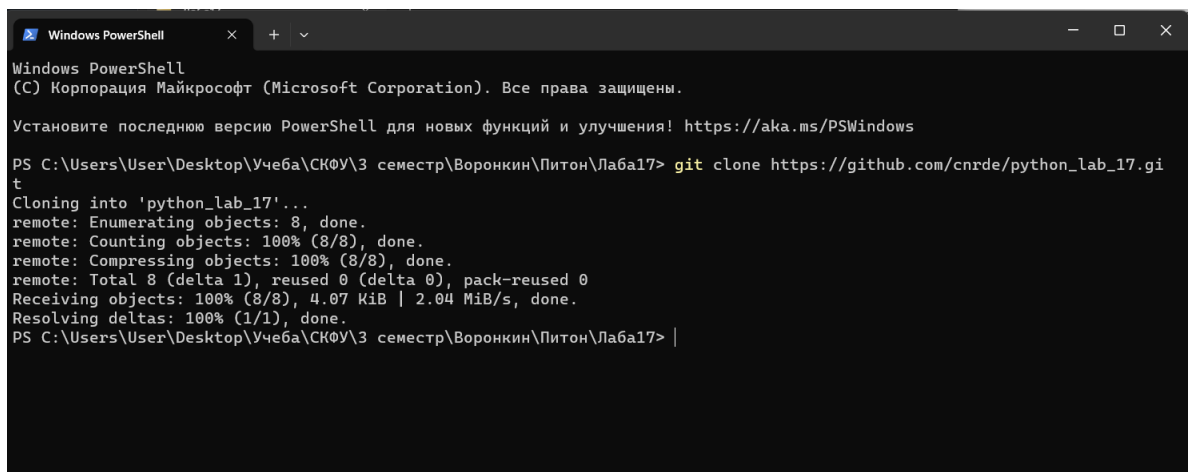
Цель: Приобретение навыков по работа с менеджером пакетов pip и виртуальными окружениями с помощью языка программирования Python версии 3.x.

### Практическая часть:

1. Изучил теоретический материал работы.
2. Создал общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



3. Клонировал репозиторий на ПК.



#### 4. Создал виртуальное окружение.

```
Командная строка
Microsoft Windows [Version 10.0.22621.3007]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\User>pip --version
pip 23.3.1 from C:\Users\User\AppData\Local\Programs\Python\Python311\Lib\site-packages\pip (python 3.11)

C:\Users\User>python -m venv env

C:\Users\User>|
```

#### 5. Установка пакета Black.

```
Командная строка

C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаба17\python_lab_17>pip install black
Collecting black
  Downloading black-24.2.0-cp311-cp311-win_amd64.whl.metadata (74 kB)
    74.6/74.6 kB 815.0 kB/s eta 0:00:00
Collecting click>=8.0.0 (from black)
  Downloading click-8.1.7-py3-none-any.whl.metadata (3.0 kB)
Collecting mypy_extensions>=0.4.3 (from black)
  Downloading mypy_extensions-1.0.0-py3-none-any.whl (4.7 kB)
Requirement already satisfied: packaging>=22.0 in c:\users\user\appdata\local\programs\python\python311\lib\site-packages (from black) (23.1)
Collecting pathspec>=0.9.0 (from black)
  Downloading pathspec-0.12.1-py3-none-any.whl.metadata (21 kB)
Collecting platformdirs>=2 (from black)
  Downloading platformdirs-4.2.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: colorama in c:\users\user\appdata\local\programs\python\python311\lib\site-packages (from click>=8.0.0->black) (0.4.6)
Downloading black-24.2.0-cp311-cp311-win_amd64.whl (1.4 MB)
    1.4/1.4 MB 1.2 MB/s eta 0:00:00
Downloading click-8.1.7-py3-none-any.whl (97 kB)
    97.9/97.9 kB 933.0 kB/s eta 0:00:00
Downloading pathspec-0.12.1-py3-none-any.whl (31 kB)
Downloading platformdirs-4.2.0-py3-none-any.whl (17 kB)
Installing collected packages: platformdirs, pathspec, mypy_extensions, click, black
Successfully installed black-24.2.0 click-8.1.7 mypy_extensions-1.0.0 pathspec-0.12.1 platformdirs-4.2.0

[notice] A new release of pip is available: 23.3.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаба17\python_lab_17>|
```

#### 6. Установка virtualenv.

```
Командная строка

C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаба17\python_lab_17>pip install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.25.0-py3-none-any.whl.metadata (4.5 kB)
Collecting distlib<1,>=0.3.7 (from virtualenv)
  Downloading distlib-0.3.8-py2.py3-none-any.whl.metadata (5.1 kB)
Collecting filelock<4,>=3.12.2 (from virtualenv)
  Downloading filelock-3.13.1-py3-none-any.whl.metadata (2.8 kB)
Requirement already satisfied: platformdirs<5,>=3.9.1 in c:\users\user\appdata\local\programs\python\python311\lib\site-packages (from virtualenv) (4.2.0)
Downloading virtualenv-20.25.0-py3-none-any.whl (3.8 MB)
    3.8/3.8 MB 1.2 MB/s eta 0:00:00
Downloading distlib-0.3.8-py2.py3-none-any.whl (468 kB)
    468.9/468.9 kB 772.1 kB/s eta 0:00:00
Downloading filelock-3.13.1-py3-none-any.whl (11 kB)
Installing collected packages: distlib, filelock, virtualenv
Successfully installed distlib-0.3.8 filelock-3.13.1 virtualenv-20.25.0

[notice] A new release of pip is available: 23.3.1 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip

C:\Users\User\Desktop\Учеба\СКФУ\3 семестр\Воронкин\Питон\Лаба17\python_lab_17>|
```

## 7. Список пакетных зависимость.

```
Командная строка
C:\Users\User\Desktop\Учеба\СКОУ\3 семестр\Воронкин\Питон\Лаба17\python_lab_17>pip freeze
aiohttp==3.8.6
aiosignal==1.3.1
asn1crypto==1.4.0
async-timeout==4.0.3
attrs==23.1.0
base58==2.1.1
bip32utils==0.3.post4
bit==0.7.2
black==24.2.0
certifi==2021.5.30
cffi==1.14.5
chardet==4.0.0
charset-normalizer==3.3.1
click==8.1.7
coincurve==15.0.0
colorama==0.4.6
colored==1.4.2
contourpy==1.1.1
cycler==0.11.0
discord-webhook==1.3.0
distlib==0.3.8
ecdsa==0.16.0
fake_user_agent==2.1.8
filelock==3.13.1
fonttools==4.42.1
frozenlist==1.4.0
helper==2.5.0
idna==2.10
Jinja2==3.1.2
kiwisolver==1.4.5
loguru==0.6.0
lxml==4.9.3
MarkupSafe==2.1.3
matplotlib==3.8.0
mnemonic==0.19
mock==5.1.0
multidict==6.0.4
mypy-extensions==1.0.0
```

## 8. Сохранение списка в файл requirements.txt.

```
requirements
Файл  Изменить  Просмотр

aiohttp==3.8.6
aiosignal==1.3.1
asn1crypto==1.4.0
async-timeout==4.0.3
attrs==23.1.0
base58==2.1.1
bip32utils==0.3.post4
bit==0.7.2
black==24.2.0
certifi==2021.5.30
cffi==1.14.5
chardet==4.0.0
charset-normalizer==3.3.1
click==8.1.7
coincurve==15.0.0
colorama==0.4.6
colored==1.4.2
contourpy==1.1.1
cycler==0.11.0
discord-webhook==1.3.0
distlib==0.3.8
ecdsa==0.16.0
fake_user_agent==2.1.8
filelock==3.13.1
fonttools==4.42.1
frozenlist==1.4.0
helper==2.5.0
idna==2.10
Jinja2==3.1.2
Строка 1, столб 1  100%  Windows (CRLF)  UTF-8
```

## 9. Управление пакетами с помощью Conda.

```
(base) C:\Users\erken\Desktop\python>conda create -n lb2.14 python=3.10
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 23.1.0

Please update conda by running

    $ conda update -n base -c defaults conda

## Package Plan ##

  environment location: C:\Users\erken\anaconda3\envs\lb2.14

  added / updated specs:
    - python=3.10

The following packages will be downloaded:



| package                    | build           | size    |
|----------------------------|-----------------|---------|
| ca-certificates-2023.01.10 | haa95532_0      | 121 KB  |
| certifi-2022.12.7          | py310haa95532_0 | 149 KB  |
| libffi-3.4.2               | hd77b12b_6      | 109 KB  |
| openssl-1.1.1t             | h2bbff1b_0      | 5.5 MB  |
| pip-22.3.1                 | py310haa95532_0 | 2.8 MB  |
| python-3.10.9              | h966fe2a_0      | 15.8 MB |
| setuptools-65.6.3          | py310haa95532_0 | 1.2 MB  |
| sqlite-3.40.1              | h2bbff1b_0      | 889 KB  |
| tzdata-2022g               | h04d1e81_0      | 114 KB  |
| wincertstore-0.2           | py310haa95532_2 | 15 KB   |
| xz-5.2.10                  | h8cc25b3_1      | 520 KB  |
| zlib-1.2.13                | h8cc25b3_0      | 113 KB  |
| Total:                     |                 | 27.2 MB |



The following NEW packages will be INSTALLED:



| package         | source                                                       |
|-----------------|--------------------------------------------------------------|
| bzip2           | pkgs/main/win-64::bzip2-1.0.8-he774522_0 None                |
| ca-certificates | pkgs/main/win-64::ca-certificates-2023.01.10-haa95532_0 None |
| certifi         | pkgs/main/win-64::certifi-2022.12.7-py310haa95532_0 None     |
| libffi          | pkgs/main/win-64::libffi-3.4.2-hd77b12b_6 None               |
| openssl         | pkgs/main/win-64::openssl-1.1.1t-h2bbff1b_0 None             |
| pip             | pkgs/main/win-64::pip-22.3.1-py310haa95532_0 None            |
| python          | pkgs/main/win-64::python-3.10.9-h966fe2a_0 None              |
| setuptools      | pkgs/main/win-64::setuptools-65.6.3-py310haa95532_0 None     |
| sqlite          | pkgs/main/win-64::sqlite-3.40.1-h2bbff1b_0 None              |
| tk              | pkgs/main/win-64::tk-8.6.12-h2bbff1b_0 None                  |
| tzdata          | pkgs/main/noarch::tzdata-2022g-h04d1e81_0 None               |
| vc              | pkgs/main/win-64::vc-14.2-h21ff451_1 None                    |
| vs2015_runtime  | pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2 None |
| wheel           | pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0 None             |
| wincertstore    | pkgs/main/win-64::wincertstore-0.2-py310haa95532_2 None      |
| xz              | pkgs/main/win-64::xz-5.2.10-h8cc25b3_1 None                  |
| zlib            | pkgs/main/win-64::zlib-1.2.13-h8cc25b3_0 None                |


```

## 10. Активация.

```
(base) C:\Users\erken\Desktop\python>conda activate lb2.14

(lb2.14) C:\Users\erken\Desktop\python>
```

## 11. Установка пакетов.

```
==> WARNING: A newer version of conda exists. <==
current version: 22.9.0
latest version: 23.1.0

Please update conda by running

    $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\erken\anaconda3\envs\lb2.14

added / updated specs:
- scipy

The following packages will be downloaded:

package | build | size
-----|-----|-----
brotlipy-0.7.0 | py310h2bbff1b_1002 | 335 KB
cffi-1.15.1 | py310h2bbff1b_3 | 239 KB
cryptography-38.0.4 | py310h21b164f_0 | 1.0 MB
idna-3.4 | py310haa95532_0 | 97 KB
mkl-service-2.4.0 | py310h2bbff1b_0 | 48 KB
mkl_fft-1.3.1 | py310ha0764ea_0 | 136 KB
mkl_random-1.2.2 | py310h4ed8f06_0 | 221 KB
numpy-1.23.5 | py310h60c9a35_0 | 11 KB
numpy-base-1.23.5 | py310h04254f7_0 | 6.0 MB
packaging-22.0 | py310haa95532_0 | 68 KB
pooch-1.4.0 | pyhd3eb1b0_0 | 41 KB
pysocks-1.7.1 | py310haa95532_0 | 28 KB
requests-2.28.1 | py310haa95532_0 | 101 KB
scipy-1.10.0 | py310hb9afe5d_0 | 18.8 MB
urllib3-1.26.14 | py310haa95532_0 | 195 KB
win_inet_pton-1.1.0 | py310haa95532_0 | 9 KB
-----|-----|-----
Total: | 27.3 MB

The following NEW packages will be INSTALLED:

appdirs | pkgs/main/noarch::appdirs-1.4.4-pyhd3eb1b0_0 None
blas | pkgs/main/win-64::blas-1.0-mkl None
brotlipy | pkgs/main/win-64::brotlipy-0.7.0-py310h2bbff1b_1002 None
cffi | pkgs/main/win-64::cffi-1.15.1-py310h2bbff1b_3 None
charset-normalizer | pkgs/main/noarch::charset-normalizer-2.0.4-pyhd3eb1b0_0 None
cryptography | pkgs/main/win-64::cryptography-38.0.4-py310h21b164f_0 None
fftw | pkgs/main/win-64::fftw-3.3.9-h2bbff1b_1 None
icc_rt | pkgs/main/win-64::icc_rt-2022.1.0-h6049295_2 None
idna | pkgs/main/win-64::idna-3.4-py310haa95532_0 None
intel-openmp | pkgs/main/win-64::intel-openmp-2021.4.0-haa95532_3556 None
mkl | pkgs/main/win-64::mkl-2021.4.0-haa95532_640 None
mkl-service | pkgs/main/win-64::mkl-service-2.4.0-py310h2bbff1b_0 None
mkl_fft | pkgs/main/win-64::mkl_fft-1.3.1-py310ha0764ea_0 None
mkl_random | pkgs/main/win-64::mkl_random-1.2.2-py310h4ed8f06_0 None
numpy | pkgs/main/win-64::numpy-1.23.5-py310h60c9a35_0 None
numpy-base | pkgs/main/win-64::numpy-base-1.23.5-py310h04254f7_0 None
packaging | pkgs/main/win-64::packaging-22.0-py310haa95532_0 None
pooch | pkgs/main/noarch::pooch-1.4.0-pyhd3eb1b0_0 None
```



## 12. Установка пакетов TensorFlow.

```
lb2.14) C:\Users\erken\Desktop\python>conda install tensorflow
collecting package metadata (current_repodata.json): done
solving environment: failed with initial frozen solve. Retrying with flexible solve.
solving environment: failed with repodata from current_repodata.json, will retry with next repodata source.
collecting package metadata (repodata.json): done
solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 23.1.0

Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\erken\anaconda3\envs\lb2.14

added / updated specs:
- tensorflow

The following packages will be downloaded:
```

package	build	
_tflow_select-2.3.0	mk1	3 KB
abs1-py-1.3.0	py310haa95532_0	172 KB
aiohttp-3.8.3	py310h2bbff1b_0	418 KB
aiohttp-3.8.3	pyhd3eb1b0_0	12 KB
astunparse-1.6.3	py_0	17 KB
async-timeout-4.0.2	py310haa95532_0	12 KB
attrs-22.1.0	py310haa95532_0	85 KB
blinker-1.4	py310haa95532_0	22 KB
cachetools-4.2.2	pyhd3eb1b0_0	13 KB
click-8.0.4	py310haa95532_0	157 KB
colorama-0.4.6	py310haa95532_0	32 KB
flatbuffers-2.0.0	h6c2663c_0	1.4 MB
flit-core-3.6.0	pyhd3eb1b0_0	42 KB
frozenset-1.3.3	py310h2bbff1b_0	40 KB
gast-0.4.0	pyhd3eb1b0_0	13 KB
giflib-5.2.1	h8cc25b3_1	81 KB
google-auth-2.6.0	pyhd3eb1b0_0	83 KB
google-auth-oauthlib-0.4.4	pyhd3eb1b0_0	18 KB
google-pasta-0.2.0	pyhd3eb1b0_0	46 KB
grpcio-1.42.0	py310hc60d5dd_0	1.7 MB
h5py-3.7.0	py310hfc34f40_0	822 KB
keras-2.10.0	py310haa95532_0	1.6 MB
keras-preprocessing-1.1.2	pyhd3eb1b0_0	35 KB
libcurl-7.87.0	h86230a5_0	324 KB
libprotobuf-3.20.3	h23ce68f_0	2.2 MB
markdown-3.4.1	py310haa95532_0	149 KB
markupsafe-2.1.1	py310h2bbff1b_0	26 KB
multidict-6.0.2	py310h2bbff1b_0	46 KB
oauthlib-3.2.1	py310haa95532_0	195 KB
opt_einsum-3.3.0	pyhd3eb1b0_1	57 KB
protobuf-3.20.3	py310hd77b12b_0	234 KB

### 13. Открыл файл environment.yml.

```
C: > Users > eren > Desktop > python > Ла62.14 > lb2.14 > ! environment.yml
1  name: lb2.14
2  channels:
3    - defaults
4  dependencies:
5    - _tflow_select=2.3.0=mkl
6    - absl-py=1.3.0=py310haa95532_0
7    - aiohttp=3.8.3=py310h2bbff1b_0
8    - aiosignal=1.2.0=pyhd3eb1b0_0
9    - appdirs=1.4.4=pyhd3eb1b0_0
10   - astunparse=1.6.3=py_0
11   - async-timeout=4.0.2=py310haa95532_0
12   - attrs=22.1.0=py310haa95532_0
13   - blas=1.0=mkl
14   - blinker=1.4=py310haa95532_0
15   - brotliipy=0.7.0=py310h2bbff1b_1002
16   - bzip2=1.0.8=he774522_0
17   - ca-certificates=2023.01.10=haa95532_0
18   - cachetools=4.2.2=pyhd3eb1b0_0
19   - certifi=2022.12.7=py310haa95532_0
20   - cffi=1.15.1=py310h2bbff1b_3
21   - charset-normalizer=2.0.4=pyhd3eb1b0_0
22   - click=8.0.4=py310haa95532_0
23   - colorama=0.4.6=py310haa95532_0
24   - cryptography=38.0.4=py310h21b164f_0
25   - fftw=3.3.9=h2bbff1b_1
26   - flatbuffers=2.0.0=h6c2663c_0
27   - flit-core=3.6.0=pyhd3eb1b0_0
28   - frozenlist=1.3.3=py310h2bbff1b_0
29   - gast=0.4.0=pyhd3eb1b0_0
30   - giflib=5.2.1=h8cc25b3_1
31   - google-auth=2.6.0=pyhd3eb1b0_0
32   - google-auth-oauthlib=0.4.4=pyhd3eb1b0_0
33   - google-pasta=0.2.0=pyhd3eb1b0_0
34   - grpcio=1.42.0=py310hc60d5dd_0
35   - h5py=3.7.0=py310hfc34f40_0
36   - hdf5=1.10.6=h1756f20_1
37   - icc_rt=2022.1.0=h6049295_2
```

### Ответы на контрольные вопросы

#### 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. Там также есть возможность выкладывать свои пакеты. Для скачивания и установки используется специальная утилита, которая называется pip.

#### 2. Как осуществить установку менеджера пакетов pip?



При развертывании современной версии Python (начиная с Python 2.7.9 и Python 3.4), `pip` устанавливается автоматически. Но если, по какой-то причине, `pip` не установлен на вашем ПК, то сделать это можно вручную. Будем считать, что Python у вас уже установлен, теперь необходимо установить `pip`. Для того, чтобы это сделать, скачайте скрипт `get-pip.py` `$ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py` и выполните его. `$ python get-pip.py`. При этом, вместе с `pip` будут установлены `setuptools` и `wheels`. `Setuptools` – это набор инструментов для построения пакетов Python.

`Wheels` – это формат дистрибутива для пакета Python. Обсуждение этих составляющих выходит за рамки урока, поэтому мы не будем на них останавливаться.

### **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`?**

```
$ pip install ProjectName
```

#### **5. Как установить заданную версию пакета с помощью `pip`?**

```
$ pip install ProjectName==3.2
```

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

```
$ pip install -e git+https://gitrepo.com/ProjectName.git
```

#### **7. Как установить пакет из локальной директории с помощью `pip`?**

```
$ pip install ./dist/ProjectName.tar.gz
```

#### **8. Как удалить установленный пакет с помощью `pip`?**

```
$ pip uninstall ProjectName
```

#### **9. Как обновить установленный пакет с помощью `pip`?**

```
$ pip install --upgrade ProjectName
```

#### **10. Как отобразить список установленных пакетов с помощью `pip`?**

```
$ pip list
```

## **11. Каковы причины появления виртуальных окружений в языке Python?**

В системе для интерпретатора Python может быть установлена глобально. Если вы уже сталкивались с этой проблемой, то уже задумались, что для каждого проекта нужна своя "песочница", которая изолирует зависимости. Такая "песочница" придумана и называется "виртуальным окружением" или "виртуальной средой". Только одна версия пакета. Это порождает ряд проблем.

## **12. Каковы основные этапы работы с виртуальными окружениями?**

1. Создаём через утилиту новое виртуальное окружение в отдельной папке для выбранной версии интерпретатора Python.
2. Активируем ранее созданное виртуальное окружение для работы.
3. Работаем в виртуальном окружении, а именно управляем пакетами используя `pip` и запускаем выполнение кода.
4. Деактивируем после окончания работы виртуальное окружение.
5. Удаляем папку с виртуальным окружением, если оно нам больше не нужно.

## **13. Как осуществляется работа с виртуальными окружениями с помощью `venv`?**

Для создания виртуального окружения достаточно дать команду в формате:

```
python3 -m venv <путь к папке виртуального окружения>
```

Обычно папку для виртуального окружения называют `env` или `venv`. В описании команды выше явно указан интерпретатор версии 3.x. Под Windows и некоторыми другими операционными системами это будет просто `python`. Чтобы активировать виртуальное окружение под нужно:

```
> env\\Scripts\\activate
```

Просто под Windows мы вызываем скрипт активации напрямую. Чтобы переключиться с одного окружения на другое нам нужно выполнить команду деактивации и команду активации другого виртуального окружения, например, так:

```
$ deactivate
```

```
$ source /home/user/envs/project1_env2/bin/activate
```

#### **14. Как осуществляется работа с виртуальными окружениями с помощью virtualenv?**

Зачем нам нужно уметь работать с утилитой virtualenv? Ведь мы уже научились работать со стандартным модулем Python venv. Просто он очень распространён и поддерживает большее число вариантов и версий интерпретатора Python, например, PyPy и CPython. Для начала пакет нужно установить. Установку можно выполнить командой:

```
# Для python 3
```

```
python3 -m pip install virtualenv
```

```
# Для единственного python
```

```
python -m pip install virtualenv
```

Создание виртуального окружения с утилитой virtualenv отличается от стандартного. Например, создание в текущей папке виртуального окружения для интерпретатора доступного через команду python3 с названием папки окружения env:virtualenv -p python3 env. Активация и деактивация такая же, как у стандартной утилиты Python.

#### **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 погрузка переменных окружения из .env файла После установки pipenv начитается 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?**

Просмотреть список зависимостей мы можем командой: `pip freeze` Что бы его сохранить, нужно перенаправить вывод команды в файл: `pip freeze > requirements.txt` Имя файла хранения зависимостей requirements.txt выбрано не зря. Оно является стандартной договоренностью и используется некоторыми утилитами автоматически. Установка пакетов из файла зависимостей в новом виртуальном окружении так же выполняется одной командой: `pip install -r requirements.txt`

## **17. В чем преимущества пакетного менеджера conda по сравнению с пакетным менеджером pip?**

Основная проблема заключается в том, что `pip`, `easy_install` и `virtualenv` ориентированы на Python. Эти инструменты игнорируют библиотеки зависимостей, реализованные с использованием других языков. Например, XSLT, HDF5, MKL и другие, которые не имеют `setup.py` в исходном коде и не устанавливают файлы в директорию `site-packages`. Conda же 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`). Существуют также некоторые различия, если вы заинтересованы в создании собственных пакетов. Например, `pip` создан на основе `setuptools`, тогда как `conda` использует свой собственный формат, который имеет некоторые преимущества (например, статическая компиляция пакета).

## **18. В какие дистрибутивы Python входит пакетный менеджер conda? Anaconda и Miniconda.**

### **19. Как создать виртуальное окружение conda?**

1. Начиная проект, создайте чистую директорию и дайте ей понятное короткое имя. Для Linux это будет соответствовать набору команд:

```
mkdir $PROJ_NAME  
cd $PROJ_NAME
```

```
touch README.md main.py
```

Для Windows, если используется дистрибутив Anaconda, то необходимо вначале запустить консоль Anaconda Powershell Prompt. Делается это из системного меню, посредством выбора следующих пунктов: Пуск Anaconda3 (64-bit) Anaconda Powershell Prompt (Anaconda3).

Создайте чистое conda-окружение с таким же именем: `conda create -n $PROJ_NAME python=3.7`

## **20. Как активировать и установить пакеты в виртуальное окружение conda?**

```
source activate $PROJ_NAME
```

## **21. Как деактивировать и удалить виртуальное окружение conda?**

```
conda deactivate
```

```
conda remove -n $PROJ_NAME
```

## **22. Каково назначение файла environment.yml? Как создать этот файл?**

Файл environment.yml позволит воссоздать окружение в любой нужный момент.

## **23. Как создать виртуальное окружение conda с помощью файла environment.yml?**

```
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 параметры 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 запуска программы, 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 ОК. 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 нового окружения. Нажимаем на ОК. Далее в созданном окружении устанавливаем 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 создавали 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 какихлибо пакетов в среде как раз и отвечают файлы requirements.txt и environment.yml

**Вывод:** Приобрел навыки по работе с функциями с переменным числом параметров при написании программ помощью языка программирования Python версии 3.x.