

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**  
**дисциплины «Программирование на Python»**

Выполнил:  
Магдаев Даламбек Магомедович  
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 практики:  
Воронкин Р.А., доцент кафедры  
инфокоммуникаций

---

(подпись)

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

Ставрополь, 2023 г.

## Тема: Установка пакетов в Python. Виртуальные окружения

**Цель:** приобретение навыков по работе с менеджером пакетов `pip` и виртуальными окружениями с помощью языка программирования Python версии 3.x.

### Порядок выполнения работы:

1. Создал новый репозиторий, клонировал его, в нем создал ветку `developer` и перешел на нее.
2. Создал виртуальное окружение Anaconda с именем репозитория:

```
ПРОБЛЕМЫ  ВЫХОДНЫЕ ДАННЫЕ  КОНСОЛЬ ОТЛАДКИ  ТЕРМИНАЛ  ПОРТЫ

PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python\14\main.py"
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> conda create -n Lab2.14 python=3
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
  current version: 23.7.4
  latest version: 23.11.0

Please update conda by running

  $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

  conda install conda=23.11.0

## Package Plan ##

  environment location: C:\Users\dalam\anaconda3\envs\Lab2.14

added / updated specs:
- python=3

The following packages will be downloaded:

  package | build | size
  -----|-----|-----
  ca-certificates-2023.12.12 | haa95532_0 | 127 KB
  expat-2.5.0 | hd77b12b_0 | 225 KB
  openssl-3.0.12 | h2bbff1b_0 | 7.4 MB
  pip-23.3.1 | py312haa95532_0 | 2.9 MB
  python-3.12.0 | h1d929f7_0 | 16.2 MB
  setuptools-68.2.2 | py312haa95532_0 | 1.2 MB
  wheel-0.41.2 | py312haa95532_0 | 150 KB
  xz-5.4.5 | h8cc25b3_0 | 593 KB
  -----|-----|-----
  Total: | | 28.8 MB

The following NEW packages will be INSTALLED:

  bzip2 | pkgs/main/win-64::bzip2-1.0.8-he774522_0
  ca-certificates | pkgs/main/win-64::ca-certificates-2023.12.12-haa95532_0
  expat | pkgs/main/win-64::expat-2.5.0-hd77b12b_0
  libffi | pkgs/main/win-64::libffi-3.4.4-hd77b12b_0
  openssl | pkgs/main/win-64::openssl-3.0.12-h2bbff1b_0
  pip | pkgs/main/win-64::pip-23.3.1-py312haa95532_0
  python | pkgs/main/win-64::python-3.12.0-h1d929f7_0
  setuptools | pkgs/main/win-64::setuptools-68.2.2-py312haa95532_0
```

Рисунок 1. Создание conda окружения

```

xz 5.4.2 h8cc25b3_0
y-py 0.5.9 py311hb6bf4ef_0
yaml 0.2.5 he774522_0
yaml-cpp 0.7.0 hd77b12b_1
yapf 0.31.0 pyhd3eb1b0_0
yarl 1.8.1 py311h2bbff1b_0
ypy-websocket 0.8.2 py311haa95532_0
zeromq 4.3.4 hd77b12b_0
zfp 1.0.0 hd77b12b_0
zict 2.2.0 py311haa95532_0
zipp 3.11.0 py311haa95532_0
zlib 1.2.13 h8cc25b3_0
zlib-ng 2.0.7 h2bbff1b_0
zope 1.0 py311haa95532_1
zope.interface 5.4.0 py311h2bbff1b_0
zstandard 0.19.0 py311h2bbff1b_0
zstd 1.5.5 hd43e919_0
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> pip list
Package Version
-----
certifi 2023.11.17
charset-normalizer 3.3.2
contourpy 1.1.1
cyclor 0.11.0
fonttools 4.42.1
idna 3.6
kiwisolver 1.4.5
matplotlib 3.8.0
numpy 1.26.0
packaging 23.1
Pillow 10.0.1
pip 23.3.2
pyparsing 3.1.1
python-dateutil 2.8.2
requests 2.31.0
scipy 1.11.4
setuptools 65.5.0
six 1.16.0
urllib3 2.1.0

```

Рисунок 2. Предустановленные пакеты

3. Установил пакеты с помощью conda install:

```

PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> conda install numpy pandas scipy
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
current version: 23.7.4
latest version: 23.11.0

Please update conda by running

    $ conda update -n base -c defaults conda

Or to minimize the number of packages updated during conda update use

    conda install conda=23.11.0

## Package Plan ##

environment location: C:\Users\dalam\anaconda3

added / updated specs:
- numpy
- pandas
- scipy

The following packages will be downloaded:



| package            | build           |         |
|--------------------|-----------------|---------|
| certifi-2023.11.17 | py311haa95532_0 | 160 KB  |
| pandas-2.1.4       | py311hf62ec03_0 | 13.6 MB |
| scipy-1.11.4       | py311hc1ccb85_0 | 20.9 MB |
| Total:             |                 | 34.6 MB |



The following packages will be UPDATED:



|                 |                                                          |
|-----------------|----------------------------------------------------------|
| ca-certificates | 2023.08.22-haa95532_0 --> 2023.12.12-haa95532_0          |
| certifi         | 2023.7.22-py311haa95532_0 --> 2023.11.17-py311haa95532_0 |
| openssl         | 3.0.10-h2bbff1b_2 --> 3.0.12-h2bbff1b_0                  |
| pandas          | 2.0.3-py311hf62ec03_0 --> 2.1.4-py311hf62ec03_0          |
| scipy           | 1.11.1-py311hc1ccb85_0 --> 1.11.4-py311hc1ccb85_0        |



Proceed ([y]/n)? y

Downloading and Extracting Packages

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python>

```

Рисунок 3. Установка пакетов

```

widgetsnbextension      4.0.5      py311haa95532_0
win_inet_pton           1.1.0      py311haa95532_0
winpty                  0.4.3      4
wrappt                  1.14.1     py311h2bbff1b_0
xarray                  2023.6.0   py311haa95532_0
xlwings                 0.29.1     py311haa95532_0
xxhash                  0.8.0      h2bbff1b_3
xyzservices             2022.9.0   py311haa95532_1
xz                      5.4.2      h8cc25b3_0
y-py                    0.5.9      py311hb6bf4ef_0
yaml                    0.2.5      he774522_0
yaml-cpp                0.7.0      hd77b12b_1
yapf                    0.31.0     pyhd3eb1b0_0
yarl                    1.8.1      py311h2bbff1b_0
ypy-websocket           0.8.2      py311haa95532_0
zeromq                  4.3.4      hd77b12b_0
zfp                     1.0.0      hd77b12b_0
zict                    2.2.0      py311haa95532_0
zipp                    3.11.0     py311haa95532_0
zlib                    1.2.13     h8cc25b3_0
zlib-ng                 2.0.7      h2bbff1b_0
zope                    1.0         py311haa95532_1
zope.interface          5.4.0      py311h2bbff1b_0
zstandard               0.19.0     py311h2bbff1b_0
zstd                    1.5.5      hd43e919_0
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> pip list
Package      Version
-----
certifi      2023.11.17
charset-normalizer 3.3.2
contourpy    1.1.1
cycler       0.11.0
fonttools    4.42.1
idna         3.6
kiwisolver   1.4.5
matplotlib   3.8.0
numpy        1.26.0
packaging    23.1
Pillow       10.0.1
pip          23.3.2
pyparsing    3.1.1
python-dateutil 2.8.2
requests     2.31.0
scipy        1.11.4
setuptools   65.5.0
six          1.16.0
urllib3      2.1.0

```

Рисунок 4. Пакеты установились

4. Попробуйте установить менеджером пакетов conda пакет TensorFlow. Возникает ли при этом ошибка? Попробуйте выявить и укажите причину этой ошибки:

```

PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> conda install tensorflow
Collecting package metadata (current_repodata.json): done
Solving environment: unsuccessful initial attempt using frozen solve. Retrying with flexible solve.
Solving environment: unsuccessful attempt using repodata from current_repodata.json, retrying with next repodata source.
Collecting package metadata (repodata.json): done
Solving environment: unsuccessful initial attempt using frozen solve. Retrying with flexible solve.
Solving environment: /
Found conflicts! Looking for incompatible packages.
This can take several minutes. Press CTRL-C to abort.
failed

UnsatisfiableError: The following specifications were found
to be incompatible with the existing python installation in your environment:

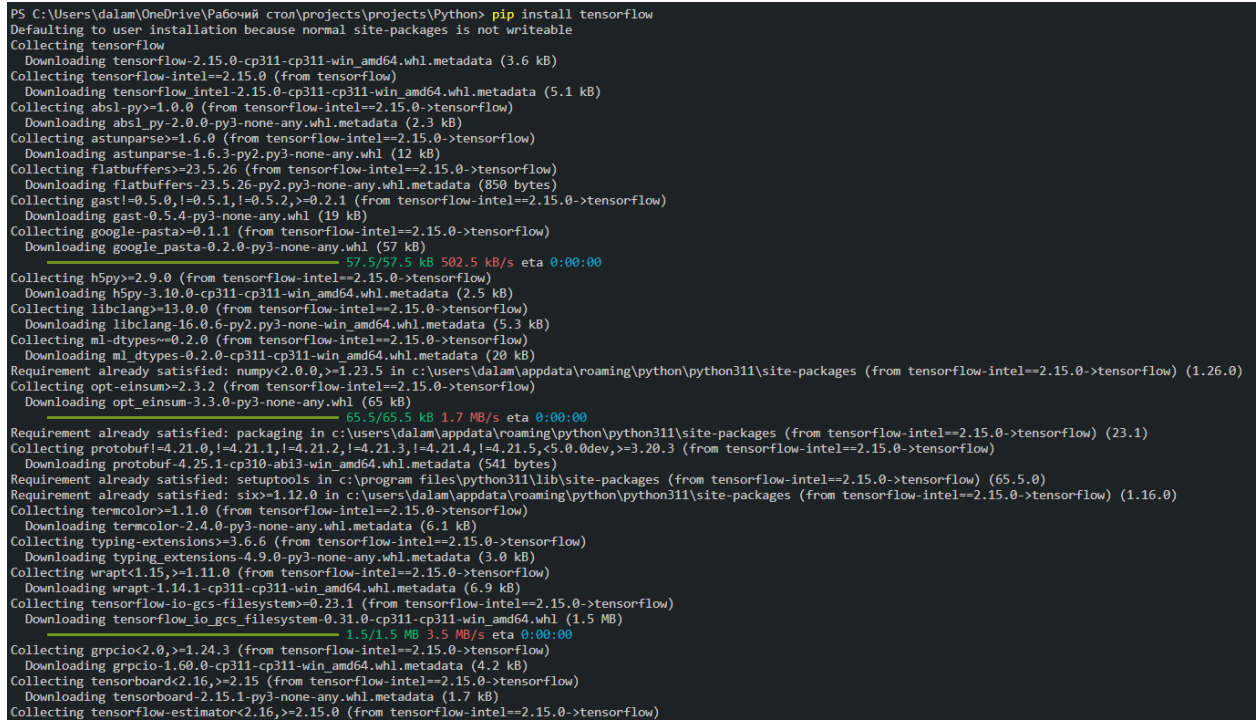
```

Рисунок 5. Попытка установки пакета TensorFlow

Пакет TensorFlow совместим с Python 3.8, для решения возможной проблемы необходимо понизить версию python с помощью команды:

```
conda install python=3.7
```

## 5. Установил пакет TensorFlow с помощью менеджера пакетов pip:

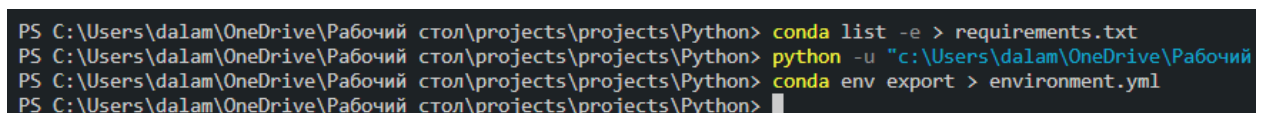


```
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> pip install tensorflow
Defaulting to user installation because normal site-packages is not writeable
Collecting tensorflow
  Downloading tensorflow-2.15.0-cp311-cp311-win_amd64.whl.metadata (3.6 kB)
Collecting tensorflow-intel==2.15.0 (from tensorflow)
  Downloading tensorflow_intel-2.15.0-cp311-cp311-win_amd64.whl.metadata (5.1 kB)
Collecting absl-py==1.0.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading absl_py-2.0.0-py3-none-any.whl.metadata (2.3 kB)
Collecting astunparse==1.6.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading astunparse-1.6.3-py2.py3-none-any.whl (12 kB)
Collecting flatbuffers==23.5.26 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading flatbuffers-23.5.26-py2.py3-none-any.whl.metadata (850 bytes)
Collecting gast==0.5.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading gast-0.5.4-py3-none-any.whl (19 kB)
Collecting google-pasta==0.1.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading google_pasta-0.2.0-py3-none-any.whl (57 kB)
Collecting h5py==2.9.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading h5py-3.10.0-cp311-cp311-win_amd64.whl.metadata (2.5 kB)
Collecting libclang==13.0.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading libclang-16.0.6-py2.py3-none-win_amd64.whl.metadata (5.3 kB)
Collecting ml-dtypes==0.2.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading ml_dtypes-0.2.0-cp311-cp311-win_amd64.whl.metadata (20 kB)
Requirement already satisfied: numpy<2.0.0, >=1.23.5 in c:\users\dalam\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.15.0->tensorflow) (1.26.0)
Collecting opt_einsum==2.3.2 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading opt_einsum-3.3.0-py3-none-any.whl (65 kB)
Requirement already satisfied: packaging in c:\users\dalam\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.15.0->tensorflow) (23.1)
Collecting protobuf==4.25.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading protobuf-4.25.1-cp310-abi3-win_amd64.whl.metadata (541 bytes)
Requirement already satisfied: setuptools in c:\program files\python311\lib\site-packages (from tensorflow-intel==2.15.0->tensorflow) (65.5.0)
Requirement already satisfied: six>=1.12.0 in c:\users\dalam\appdata\roaming\python\python311\site-packages (from tensorflow-intel==2.15.0->tensorflow) (1.16.0)
Collecting termcolor==1.1.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading termcolor-2.4.0-py3-none-any.whl.metadata (6.1 kB)
Collecting typing_extensions==3.6.6 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading typing_extensions-4.9.0-py3-none-any.whl.metadata (3.0 kB)
Collecting wrapt==1.11.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading wrapt-1.14.1-cp311-cp311-win_amd64.whl.metadata (6.9 kB)
Collecting tensorflow-io-gcs-filesystem==0.23.1 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading tensorflow_io_gcs_filesystem-0.31.0-cp311-cp311-win_amd64.whl (1.5 MB)
Collecting grpcio==2.0.3 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading grpcio-1.60.0-cp311-cp311-win_amd64.whl.metadata (4.2 kB)
Collecting tensorflow-estimator==2.15.0 (from tensorflow-intel==2.15.0->tensorflow)
  Downloading tensorflow-estimator-2.15.1-py3-none-any.whl.metadata (1.7 kB)
```

Рисунок 6. Установка TensorFlow при помощи pip

6. Сформируйте файлы requirements.txt и environment.yml.

Проанализируйте содержимое этих файлов:



```
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> conda list -e > requirements.txt
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> python -u "c:\Users\dalam\OneDrive\Рабочий
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> conda env export > environment.yml
PS C:\Users\dalam\OneDrive\Рабочий стол\projects\projects\Python> |
```

Рисунок 7. Формирование файлов requirements.txt и environment.yml

```
requirements.txt
1  # This file may be used to create an environment using:
2  # $ conda create --name <env> --file <this file>
3  # platform: win-64
4  _anaconda_depends=2023.09=py311_mkl_1
5  abseil-cpp=20211102.0=hd77b12b_0
6  aiobotocore=2.5.0=py311haa95532_0
7  aiofiles=22.1.0=py311haa95532_0
8  aiohttp=3.8.5=py311h2bbff1b_0
9  aioitertools=0.7.1=pyhd3eb1b0_0
10 aiosignal=1.2.0=pyhd3eb1b0_0
11aiosqlite=0.18.0=py311haa95532_0
12alabaster=0.7.12=pyhd3eb1b0_0
13anaconda-anon-usage=0.4.2=py311hfc23b7f_0
14anaconda-catalogs=0.2.0=py311haa95532_0
15anaconda-client=1.12.1=py311haa95532_0
16anaconda-cloud-auth=0.1.3=py311haa95532_0
17anaconda-navigator=2.5.0=py311haa95532_0
18anaconda-project=0.11.1=py311haa95532_0
19anyio=3.5.0=py311haa95532_0
20aom=3.6.0=hd77b12b_0
21appdirs=1.4.4=pyhd3eb1b0_0
22argon2-cffi=21.3.0=pyhd3eb1b0_0
23argon2-cffi-bindings=21.2.0=py311h2bbff1b_0
24arrow=1.2.3=py311haa95532_1
25arrow-cpp=11.0.0=ha81ea56_2
26astroid=2.14.2=py311haa95532_0
27astropy=5.1=py311h5bb9823_0
28asttokens=2.0.5=pyhd3eb1b0_0
29async-timeout=4.0.2=py311haa95532_0
30atomicwrites=1.4.0=py_0
31attrs=22.1.0=py311haa95532_0
32automat=20.2.0=py_0
33autopep8=1.6.0=pyhd3eb1b0_1
34aws-c-common=0.6.8=h2bbff1b_1
35aws-c-event-stream=0.1.6=hd77b12b_6
36aws-checksums=0.1.11=h2bbff1b_2
37aws-sdk-cpp=1.8.185=hd77b12b_1
38babel=2.11.0=py311haa95532_0
39backcall=0.2.0=pyhd3eb1b0_0
40backports=1.1=pyhd3eb1b0_0
41backports.functools_lru_cache=1.6.4=pyhd3eb1b0_0
42backports.tempfile=1.0=pyhd3eb1b0_1
43backports.weakref=1.0.post1=py_1
44bcrypt=3.2.0=py311h2bbff1b_1
45beautifulsoup4=4.12.2=py311haa95532_0
```

Рисунок 8. Файл requirements.txt

```

environment.yml
1  name: base
2  channels:
3    - defaults
4  dependencies:
5    - _anaconda_depends=2023.09=py311_mkl_1
6    - abseil-cpp=20211102.0=hd77b12b_0
7    - aiobotocore=2.5.0=py311haa95532_0
8    - aiofiles=22.1.0=py311haa95532_0
9    - aiohttp=3.8.5=py311h2bbff1b_0
10   - aioitertools=0.7.1=pyhd3eb1b0_0
11   - aiosignal=1.2.0=pyhd3eb1b0_0
12   - aiosqlite=0.18.0=py311haa95532_0
13   - alabaster=0.7.12=pyhd3eb1b0_0
14   - anaconda-anon-usage=0.4.2=py311hfc23b7f_0
15   - anaconda-catalogs=0.2.0=py311haa95532_0
16   - anaconda-client=1.12.1=py311haa95532_0
17   - anaconda-cloud-auth=0.1.3=py311haa95532_0
18   - anaconda-navigator=2.5.0=py311haa95532_0
19   - anaconda-project=0.11.1=py311haa95532_0
20   - anyio=3.5.0=py311haa95532_0
21   - aom=3.6.0=hd77b12b_0
22   - appdirs=1.4.4=pyhd3eb1b0_0
23   - argon2-cffi=21.3.0=pyhd3eb1b0_0
24   - argon2-cffi-bindings=21.2.0=py311h2bbff1b_0
25   - arrow=1.2.3=py311haa95532_1
26   - arrow-cpp=11.0.0=ha81ea56_2
27   - astroid=2.14.2=py311haa95532_0
28   - astropy=5.1=py311h5bb9823_0
29   - asttokens=2.0.5=pyhd3eb1b0_0
30   - async-timeout=4.0.2=py311haa95532_0
31   - atomicwrites=1.4.0=py_0
32   - attrs=22.1.0=py311haa95532_0
33   - automat=20.2.0=py_0
34   - autopep8=1.6.0=pyhd3eb1b0_1
35   - aws-c-common=0.6.8=h2bbff1b_1
36   - aws-c-event-stream=0.1.6=hd77b12b_6
37   - aws-checksums=0.1.11=h2bbff1b_2
38   - aws-sdk-cpp=1.8.185=hd77b12b_1
39   - babel=2.11.0=py311haa95532_0
40   - backcall=0.2.0=pyhd3eb1b0_0
41   - backports=1.1=pyhd3eb1b0_0
42   - backports.functools_lru_cache=1.6.4=pyhd3eb1b0_0
43   - backports.tempfile=1.0=pyhd3eb1b0_1
44   - backports.weakref=1.0.post1=py_1
45   - bcrypt=3.2.0=py311h2bbff1b_1

```

Рисунок 9. Файл environment.yml

В файле requirements.txt хранятся зависимости созданные pip, а в файле environment.yml хранятся параметры окружения conda.

### 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.

## 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==*
```

## 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 только одна версия пакета. Это порождает ряд проблем: проблема обратной совместимости и проблема коллективной разработки. Получается, что для каждого проекта нужна своя "песочница", которая изолирует зависимости. Такая "песочница" придумана и называется "виртуальным окружением" или "виртуальной средой".

12. Каковы основные этапы работы с виртуальными окружениями?

- Создаём через утилиту новое виртуальное окружение в отдельной папке для выбранной версии интерпретатора Python.
- Активируем ранее созданное виртуальное окружение для работы.
- Работаем в виртуальном окружении, а именно управляем пакетами используя pip и запускаем выполнение кода.
- Деактивируем 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?

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 нужно:

```
$ source env/bin/activate
```

В Windows мы вызываем скрипт активации напрямую.

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

Чтобы переключиться с одного окружения на другое нам нужно выполнить команду деактивации и команду активации другого виртуального окружения, например, так:

```
$ deactivate
```

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

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`?

`pipenv install` – Создание виртуального окружения

`pipenv install <package>` – Установка определённого пакета и добавление его в `Pipfile`.

`pipenv uninstall <package>` – Удаление установленного пакета и его исключение из `Pipfile`.

`pipenv shell` – Активация виртуального окружения.

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 договорённостью 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?

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
```

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?

```
conda activate $PROJ_NAME
```

```
conda install $PACKAGE_NAME
```

21. Как деактивировать и удалить виртуальное окружение conda?

```
conda deactivate
```

```
conda remove -n $PACKAGE_NAME
```

22. Каково назначение файла `environment.yml`? Как создать этот файл?  
Файл `environment.yml` позволит воссоздать окружение в любой нужный момент.

```
conda env export > environment.yml
```

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

24. Самостоятельно изучите средства IDE PyCharm для работы с виртуальными окружениями conda. Опишите порядок работы с виртуальными окружениями conda в IDE PyCharm.

Необходимо установить Anaconda или Miniconda.

В Pycharm необходимо настроить интерпретатор Python:

Нужно перейти в `File > Settings` (для Windows/Linux) или `PyCharm > Preferences` (для macOS).

В левой части окна настроек выбрать `Project: ваш_проект > Python Interpreter`.

Нажать на шестерёнку справа от списка интерпретаторов и выбрать `Add`.  
В открывшемся окне добавления интерпретатора выбрать `Conda Environment`.

Можно либо создать новое окружение, выбрав `New environment`, либо использовать существующее, выбрав `Existing environment`.

Создание нового окружения Conda:

Необходимо указать имя окружения, версию Python и нажать кнопку `OK`.

PyCharm автоматически создаст новое окружение Conda и установит в него выбранную версию Python.

Использование существующего окружения Conda:

Нужно нажать на кнопку с тремя точками и найти путь к существующему окружению Conda.

Активация окружения Conda:

При использовании терминала в PyCharm окружение Conda должно активироваться автоматически. Если этого не произошло, его можно активировать вручную, введя команду `conda activate имя_окружения` в терминале.

Работа с проектом:

После настройки окружения Conda можно работать с проектом в PyCharm, как обычно.

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`.

**Вывод:** в результате выполнения работы были приобретены навыки по работе с менеджером пакетов `pip` и виртуальными окружениями с помощью языка программирования Python версии 3.x.