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 ПРАКТИЧЕСКОЙ РАБОТЕ №2.14
дисциплины «Программирование на Python»

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

_____
(подпись)

Руководитель практики:
Воронкин Р.А., канд. техн. наук,
доцент, 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. Создаю виртуальное окружение Anaconda с именем репозитория

```
(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17>mkdir lab2.14.17

(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17>cd lab2.14.17

(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>touch README.md main.py
"touch" не является внутренней или внешней
командой, исполняемой программой или пакетным файлом.

(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>copy NUL > main.py
```

Рисунок 1 – Создание чистой директории с именем репозитория

```
(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>conda create -n lab2.14.17 python=3.11
Collecting package metadata (current_repodata.json): -
```

Рисунок 2 – Создание conda-окружения

```
(base) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>conda activate lab2.14.17

(lab2.14.17) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>
```

Рисунок 3 – Активация conda-окружения

2. Устанавливаю пакеты: pip, NumPy, Pandas, SciPy.

```
(lab2.14.17) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>con
# packages in environment at D:\conda\envs\lab2.14.17:
#
# Name                    Version                   Build  Channel
bzip2                     1.0.8                he774522_0
ca-certificates           2023.08.22           haa95532_0
libffi                    3.4.4                hd77b12b_0
numpy                     1.26.2                   pypi_0    pypi
openssl                   3.0.12               h2bbff1b_0
pandas                    2.1.4                    pypi_0    pypi
pip                       23.3.1          py311haa95532_0
python                    3.11.5               he1021f5_0
python-dateutil           2.8.2                    pypi_0    pypi
pytz                      2023.3.post1             pypi_0    pypi
scipy                     1.11.4                   pypi_0    pypi
setuptools                68.0.0          py311haa95532_0
six                       1.16.0                   pypi_0    pypi
sqlite                    3.41.2               h2bbff1b_0
tk                        8.6.12               h2bbff1b_0
tzdata                    2023.3                   pypi_0    pypi
vc                        14.2                 h21ff451_1
vs2015_runtime            14.27.29016          h5e58377_2
wheel                     0.41.2          py311haa95532_0
xz                        5.4.5                h8cc25b3_0
zlib                      1.2.13               h8cc25b3_0
```

Рисунок 4 – Список установленных пакетов

3. Установка пакета TensorFlow менеджером пакетов conda

```
(lab2.14.17) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>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 sour
ce.
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:

Specifications:

  - tensorflow -> python[version='3.10.*|3.9.*|3.8.*|3.7.*|3.6.*|3.5.*']

Your python: python=3.11

If python is on the left-most side of the chain, that's the version you've asked for.
When python appears to the right, that indicates that the thing on the left is somehow
not available for the python version you are constrained to. Note that conda will not
change your python version to a different minor version unless you explicitly specify
that.
```

Рисунок 5 – Неудачная установка пакета TensorFlow

Причина: слишком большая версия и решение заморожено

4. Установка пакета TensorFlow менеджером пакетов pip

```
Successfully installed MarkupSafe-2.1.3 TensorFlow-2.15.0 absl-py-2.0.0 astunparse-1.6.3 cachetools-5.3.2 certifi-202
3.11.17 charset-normalizer-3.3.2 flatbuffers-23.5.26 gast-0.5.4 google-auth-2.25.2 google-auth-oauthlib-1.1.0 google-
pasta-0.2.0 grpcio-1.60.0 h5py-3.10.0 idna-3.6 keras-2.15.0 libclang-16.0.6 markdown-3.5.1 ml-dtypes-0.2.0 oauthlib-3
.2.2 opt-einsum-3.3.0 packaging-23.2 protobuf-4.23.4 pyasn1-0.5.1 pyasn1-modules-0.3.0 requests-2.31.0 requests-oauth
lib-1.3.1 rsa-4.9 tensorboard-2.15.1 tensorboard-data-server-0.7.2 tensorflow-estimator-2.15.0 tensorflow-intel-2.15.
0 tensorflow-io-gcs-filesystem-0.31.0 termcolor-2.4.0 typing-extensions-4.9.0 urllib3-2.1.0 werkzeug-3.0.1 wrapt-1.14
.1
```

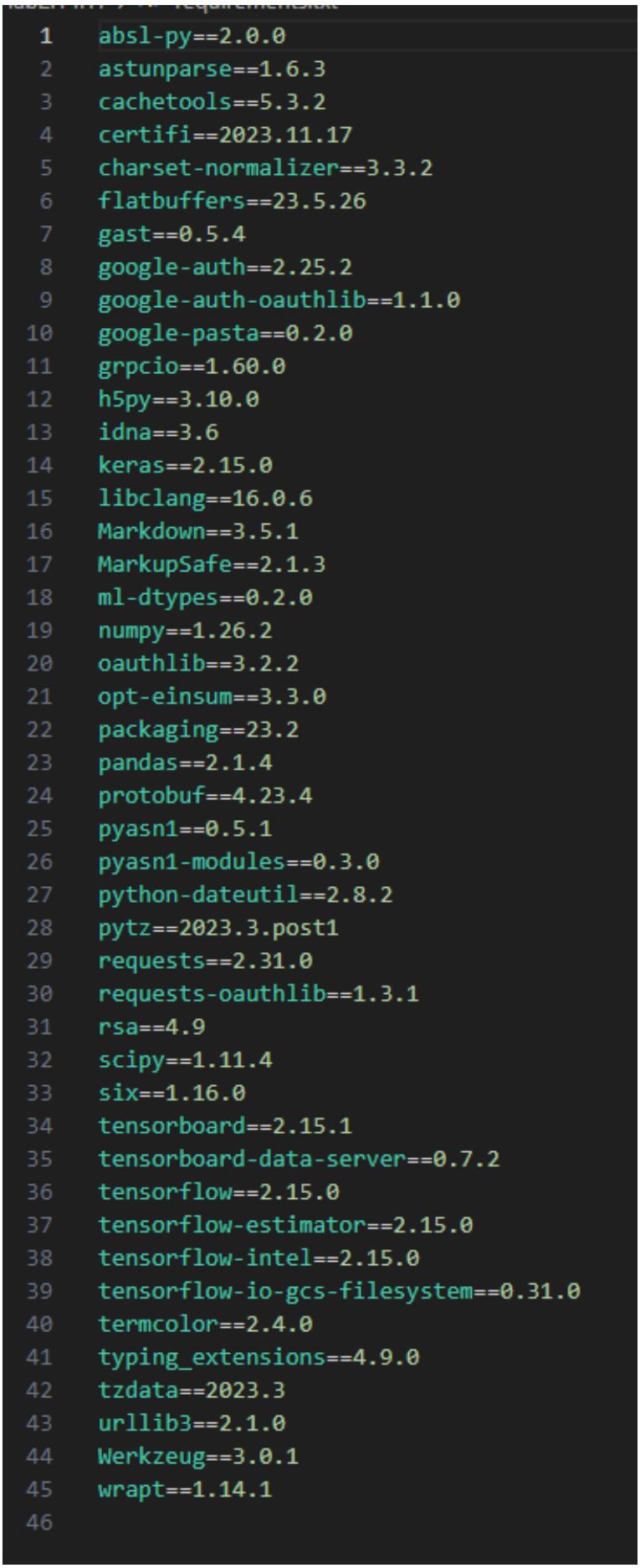
Рисунок 6 – Успешная установка пакета TensorFlow

5. Формирую файлы requirements.txt и environment.yml

```
lab2.14.17) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>conda env export > environment.yml

lab2.14.17) C:\Users\Лео\Desktop\Python-labs\lab2.14.17\lab2.14.17>pip freeze > requirements.txt
```

Рисунок 7 – Команды для формирования файлов



```
absl-py==2.0.0
astunparse==1.6.3
cachetools==5.3.2
certifi==2023.11.17
charset-normalizer==3.3.2
flatbuffers==23.5.26
gast==0.5.4
google-auth==2.25.2
google-auth-oauthlib==1.1.0
google-pasta==0.2.0
grpcio==1.60.0
h5py==3.10.0
idna==3.6
keras==2.15.0
libclang==16.0.6
Markdown==3.5.1
MarkupSafe==2.1.3
ml-dtypes==0.2.0
numpy==1.26.2
oauthlib==3.2.2
opt-einsum==3.3.0
packaging==23.2
pandas==2.1.4
protobuf==4.23.4
pyasn1==0.5.1
pyasn1-modules==0.3.0
python-dateutil==2.8.2
pytz==2023.3.post1
requests==2.31.0
requests-oauthlib==1.3.1
rsa==4.9
scipy==1.11.4
six==1.16.0
tensorboard==2.15.1
tensorboard-data-server==0.7.2
tensorflow==2.15.0
tensorflow-estimator==2.15.0
tensorflow-intel==2.15.0
tensorflow-io-gcs-filesystem==0.31.0
termcolor==2.4.0
typing_extensions==4.9.0
tzdata==2023.3
urllib3==2.1.0
Werkzeug==3.0.1
wrapt==1.14.1

```

Рисунок 8 – Содержимое файла requirements.txt

Таким образом, requirements.txt содержит все пакеты, которые были установлены перед выполнением команды и предположительно использованы в каком-либо проекте.

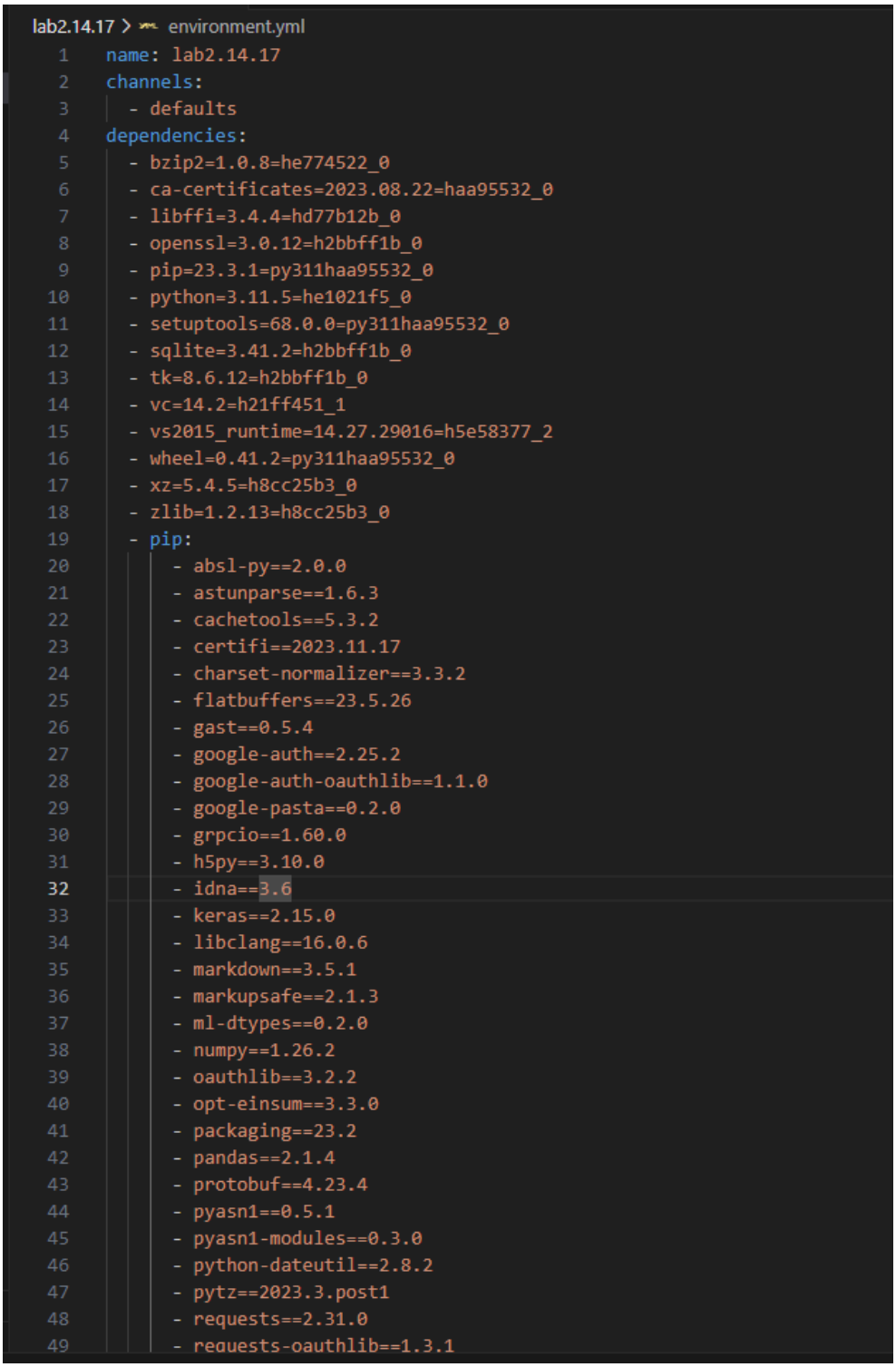


Рисунок 9 – Содержимое файла environment.yml

Файл environment.yml содержит параметры окружения, данный файл используется для воссоздания окружения в любой нужный момент времени.

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

Контрольные вопросы

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

Для установки пакета Python, не входящего в стандартную библиотеку, можно воспользоваться менеджером пакетов pip. Например, для установки пакета "requests" выполните команду: pip install requests

2. Как осуществить установку менеджера пакетов pip?

Для установки менеджера пакетов pip, обычно он устанавливается вместе с Python. Если он не установлен, можно воспользоваться инструкцией по установке pip для вашей операционной системы.

3. Откуда менеджер пакетов pip по умолчанию устанавливает пакеты?

По умолчанию pip устанавливает пакеты из Python Package Index (PyPI), но также может устанавливать их из других источников, таких как Git репозитории.

4. Как установить последнюю версию пакета с помощью pip?

Для установки последней версии пакета с помощью pip, используйте команду: pip install --upgrade package_name

5. Как установить заданную версию пакета с помощью pip?

Для установки заданной версии пакета с помощью pip, используйте команду: pip install package_name==version_number

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

Для установки пакета из git репозитория с помощью pip, используйте команду: pip install git+https://github.com/username/repository.git

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

Для установки пакета из локальной директории с помощью pip, используйте команду: pip install /path/to/local/directory

8. Как удалить установленный пакет с помощью pip?

Для удаления установленного пакета с помощью pip, используйте команду: pip uninstall package_name

9. Как обновить установленный пакет с помощью pip?

Для обновления установленного пакета с помощью pip, используйте команду: pip install --upgrade package_name

10. Как отобразить список установленных пакетов с помощью pip?

Для отображения списка установленных пакетов с помощью pip, используйте команду: pip list

11. Каковы причины появления виртуальных окружений в языке Python?

Виртуальные окружения в Python используются для изоляции проектов и их зависимостей, чтобы избежать конфликтов между различными версиями пакетов.

12. Каковы основные этапы работы с виртуальными окружениями?

Основные этапы работы с виртуальными окружениями включают создание, активацию, деактивацию и удаление виртуальных окружений.

13. Как осуществляется работа с виртуальными окружениями с помощью venv?

Для работы с виртуальными окружениями с помощью venv, используйте стандартную библиотеку Python для создания и управления виртуальными окружениями.

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

Virtualenv предоставляет инструменты для создания изолированных виртуальных окружений Python.

15. Изучите работу с виртуальными окружениями pipenv. Как осуществляется работа с виртуальными окружениями pipenv?

Pipenv предоставляет удобный способ управления зависимостями и виртуальными окружениями для проектов Python.

16. Каково назначение файла requirements.txt? Как создать этот файл? Какой он имеет формат?

Файл requirements.txt используется для хранения списка зависимостей проекта, что позволяет легко установить их на другой системе. Файл создается вручную и содержит список пакетов и их версий.

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

Conda позволяет управлять не только Python-пакетами, но и библиотеками, написанными на других языках. Он также умеет устанавливать библиотеки, которые содержат бинарные зависимости, что делает его более гибким по сравнению с pip.

18. В какие дистрибутивы Python входит пакетный менеджер conda?

Conda входит в дистрибутив Anaconda и Miniconda, которые предоставляют широкий выбор пакетов для научных вычислений и анализа данных.

19. Как создать виртуальное окружение conda?

Для создания виртуального окружения с помощью conda используется команда conda create --name myenv для создания нового окружения с именем "myenv".

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

Для активации виртуального окружения conda используйте команду conda activate myenv, а для установки пакетов в это окружение используйте conda install package_name.

21. Как деактивировать и удалить виртуальное окружение conda?

Для деактивации виртуального окружения conda используйте команду conda deactivate, а для удаления окружения используйте conda remove --name myenv --all.

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

Файл environment.yml используется для описания окружения conda, включая список пакетов и их версий. Этот файл можно создать вручную, указав необходимые пакеты и их версии, или сгенерировать автоматически с помощью команды conda env export > environment.yml.

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

Для создания виртуального окружения conda с использованием файла environment.yml, выполните команду 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.

В PyCharm можно работать с виртуальными окружениями conda, создавая и активируя их через интерфейс пользователя. Для этого необходимо установить плагин Conda, после чего можно создавать, активировать и управлять виртуальными окружениями через интерфейс PyCharm.

25. Почему файлы requirements.txt и environment.yml должны храниться в репозитории git?

Файлы requirements.txt и environment.yml должны храниться в репозитории git, чтобы другие разработчики могли легко воссоздать окружение проекта на своих системах. Это позволяет обеспечить консистентность окружения и упростить процесс развертывания проекта.