

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 языке Python»

Выполнила:
Быковская Стефания Станиславовна
2 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи», очная
форма обучения

(подпись)

Проверил:

Воронкин Р. А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

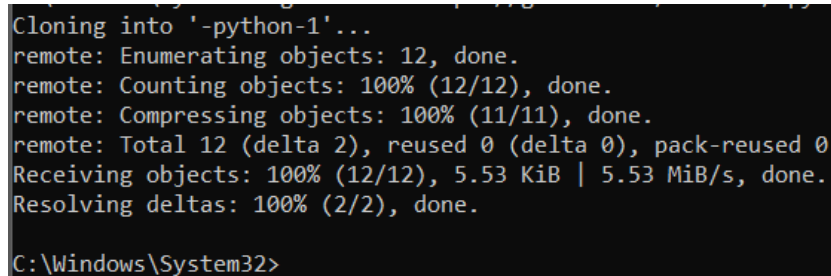
Ставрополь, 2023 г.

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.

Ход работы:

Задание 1. Создала общедоступный репозиторий на GitHub, в котором использована лицензия MIT и язык программирования Python, также добавила файл `.gitignore` с необходимыми правилами. Клонировала свой репозиторий на свой компьютер.

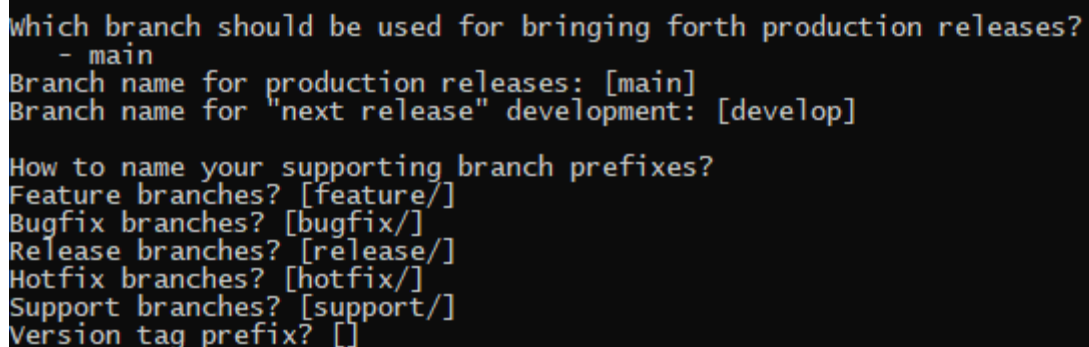


```
Cloning into '-python-1'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 12 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (12/12), 5.53 KiB | 5.53 MiB/s, done.
Resolving deltas: 100% (2/2), done.

C:\Windows\System32>
```

Рисунок 1. Клонирование репозитория

Задание 2. Организовала свой репозиторий в соответствии с моделью ветвления `git-flow`, появилась новая ветка `develop` в которой буду выполнять дальнейшие задачи.



```
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
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? []
```

Рисунок 2. Модель ветвления `git-flow`

Задание 3. Создала виртуальное окружение Anaconda с именем репозитория.

```
(base) PS C:\Users\Gaming-PC\Laba_2.14> conda create -n Laba_2.14 python=3.7
Collecting package metadata (current_repodata.json): done
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: 23.1.0
  latest version: 23.7.3

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=23.7.3

## Package Plan ##

  environment location: C:\Users\Gaming-PC\.conda\envs\Laba_2.14

  added / updated specs:
    - python=3.7

The following packages will be downloaded:



| package                    | build          | size    |
|----------------------------|----------------|---------|
| ca-certificates-2023.08.22 | haa95532_0     | 123 KB  |
| certifi-2022.12.7          | py37haa95532_0 | 149 KB  |
| openssl-1.1.1v             | h2bbff1b_0     | 5.5 MB  |
| pip-22.3.1                 | py37haa95532_0 | 2.7 MB  |
| python-3.7.16              | h6244533_0     | 17.2 MB |
| setuptools-65.6.3          | py37haa95532_0 | 1.1 MB  |
| sqlite-3.41.2              | h2bbff1b_0     | 894 KB  |
| wheel-0.38.4               | py37haa95532_0 | 82 KB   |
| wincertstore-0.2           | py37haa95532_2 | 15 KB   |
| Total:                     |                | 27.8 MB |



The following NEW packages will be INSTALLED:



| package         | source                                                  |
|-----------------|---------------------------------------------------------|
| ca-certificates | pkgs/main/win-64::ca-certificates-2023.08.22-haa95532_0 |
| certifi         | pkgs/main/win-64::certifi-2022.12.7-py37haa95532_0      |
| openssl         | pkgs/main/win-64::openssl-1.1.1v-h2bbff1b_0             |
| pip             | pkgs/main/win-64::pip-22.3.1-py37haa95532_0             |
| python          | pkgs/main/win-64::python-3.7.16-h6244533_0              |
| setuptools      | pkgs/main/win-64::setuptools-65.6.3-py37haa95532_0      |
| sqlite          | pkgs/main/win-64::sqlite-3.41.2-h2bbff1b_0              |
| vc              | pkgs/main/win-64::vc-14.2-h21ff451_1                    |
| vs2015_runtime  | pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2 |
| wheel           | pkgs/main/win-64::wheel-0.38.4-py37haa95532_0           |
| wincertstore    | pkgs/main/win-64::wincertstore-0.2-py37haa95532_2       |


```

Рисунок 3. Виртуальное окружение

```
(base) PS C:\Users\Gaming-PC\Laba_2.14> conda activate Laba_2.14
(Laba_2.14) PS C:\Users\Gaming-PC\Laba_2.14> conda list
# packages in environment at C:\Users\Gaming-PC\.conda\envs\Laba_2.14:
#
# Name                                Version                                Build                                Channel
ca-certificates                       2023.08.22                            haa95532_0                          pkgs/main/win-64
certifi                               2022.12.7                             py37haa95532_0                      pkgs/main/win-64
openssl                               1.1.1v                                h2bbff1b_0                          pkgs/main/win-64
pip                                   22.3.1                                py37haa95532_0                      pkgs/main/win-64
python                                3.7.16                                h6244533_0                          pkgs/main/win-64
setuptools                             65.6.3                                py37haa95532_0                      pkgs/main/win-64
sqlite                                 3.41.2                                h2bbff1b_0                          pkgs/main/win-64
vc                                     14.2                                  h21ff451_1                          pkgs/main/win-64
vs2015_runtime                        14.27.29016                           h5e58377_2                          pkgs/main/win-64
wheel                                  0.38.4                                py37haa95532_0                      pkgs/main/win-64
wincertstore                           0.2                                  py37haa95532_2                      pkgs/main/win-64
```

Рисунок 4. Активация окружения

Задание 4. Установила в виртуальное окружение следующие пакеты:
pip, NumPy, Pandas, SciPy.

```

(Laba_2.14) PS C:\Users\Gaming-PC\Laba_2.14> conda install pip, numpy, pandas, scipy
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: 23.1.0
  latest version: 23.7.3

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=23.7.3

## Package Plan ##

environment location: C:\Users\Gaming-PC\.conda\envs\Laba_2.14

added / updated specs:
- numpy
- pandas
- pip
- scipy

The following packages will be downloaded:



| package           | build          |         |
|-------------------|----------------|---------|
| bottleneck-1.3.5  | py37h080aedc_0 | 105 KB  |
| fftw-3.3.9        | h2bbff1b_1     | 672 KB  |
| mkl-service-2.4.0 | py37h2bbff1b_0 | 49 KB   |
| mkl_fft-1.3.1     | py37h277e83a_0 | 135 KB  |
| mkl_random-1.2.2  | py37hf11a4ad_0 | 216 KB  |
| numexpr-2.8.4     | py37h5b0cc5e_0 | 127 KB  |
| numpy-1.21.5      | py37h7a0a035_3 | 25 KB   |
| numpy-base-1.21.5 | py37hca35cd5_3 | 4.4 MB  |
| packaging-22.0    | py37haa95532_0 | 67 KB   |
| pandas-1.3.5      | py37h6214cd6_0 | 8.4 MB  |
| pytz-2022.7       | py37haa95532_0 | 210 KB  |
| scipy-1.7.3       | py37h7a0a035_2 | 13.8 MB |
| Total:            |                | 28.1 MB |


```

Рисунок 5. Установка пакетов

Задание 5. Установить менеджером пакетов conda пакет TensorFlow. В результате выдало предупреждение о разных версиях.

```

(Laba_2.14) PS C:\Users\Gaming-PC\Laba_2.14> 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: 23.1.0
  latest version: 23.7.3

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=23.7.3

## Package Plan ##

environment location: C:\Users\Gaming-PC\.conda\envs\Laba_2.14

added / updated specs:
- tensorflow

```

Рисунок 6. Установка TensorFlow через conda

После попытки установки пакета TensorFlow через conda, воспользовалась менеджером пакетов pip. После чего выполнила установку данного пакета.

```

tensorboard 2.13.0
tensorboard-data-server 0.7.1

```

Рисунок 7. Установка TensorFlow через pip

Задание 6. Сформировали файл `environment.yml`, определяющий окружение conda, включая название, каналы (источники пакетов) и список зависимостей. Он позволяет указать не только Python-пакеты, но и другие зависимости (например, библиотеки, необходимые для компиляции), а также версию Python.

```

C: > Users > HP > Documents > ИТС-б-о-22-1 > Программирование Python
1  name: Lab_2.14
2  channels:
3  - defaults
4  dependencies:
5  - blas=1.0=mkl
6  - bottleneck=1.3.5=py311h5bb9823_0
7  - bzip2=1.0.8=he774522_0
8  - ca-certificates=2023.08.22=haa95532_0
9  - icc_rt=2022.1.0=h6049295_2
10 - intel-openmp=2023.1.0=h59b6b97_46319
11 - libffi=3.4.4=hd77b12b_0
12 - mkl=2023.1.0=h6b88ed4_46357
13 - mkl-service=2.4.0=py311h2bbff1b_1
14 - mkl_fft=1.3.6=py311hf62ec03_1
15 - mkl_random=1.2.2=py311hf62ec03_1
16 - numexpr=2.8.4=py311h1fcade_1
17 - numpy=1.25.2=py311hdab7c0b_0
18 - numpy-base=1.25.2=py311hd01c5d8_0
19 - openssl=3.0.10=h2bbff1b_2
20 - pandas=2.0.3=py311hf62ec03_0
21 - pip=23.2.1=py311haa95532_0
22 - python=3.11.4=he1021f5_0
23 - python-dateutil=2.8.2=pyhd3eb1b0_0
24 - python-tzdata=2023.3=pyhd3eb1b0_0
25 - pytz=2022.7=py311haa95532_0
26 - scipy=1.11.1=py311hc1ccb85_0
27 - setuptools=68.0.0=py311haa95532_0
28 - six=1.16.0=pyhd3eb1b0_1
29 - sqlite=3.41.2=h2bbff1b_0
30 - tbb=2021.8.0=h59b6b97_0
31 - tk=8.6.12=h2bbff1b_0
32 - tzdata=2023c=h04d1e81_0
33 - vc=14.2=h21ff451_1
34 - vs2015_runtime=14.27.29016=h5e58377_2
35 - wheel=0.38.4=py311haa95532_0
36 - xz=5.4.0=h8cc25b3_0

```


Рисунок 8. Содержимое файла

Далее сформировала файл `requirements.txt` он содержит список зависимостей и их версий в текстовом формате. Каждая строка представляет собой один пакет с указанием версии или диапазона версий.

```
requirements – Блокнот
Файл Правка Формат Вид Справка
abs1-py==1.4.0
astunparse==1.6.3
cachetools==5.3.1
certifi==2023.7.22
charset-normalizer==3.2.0
flatbuffers==23.5.26
gast==0.4.0
google-auth==2.22.0
google-auth-oauthlib==1.0.0
google-pasta==0.2.0
grpcio==1.58.0
h5py==3.9.0
idna==3.4
keras==2.13.1
libclang==16.0.6
Markdown==3.4.4
MarkupSafe==2.1.3
numpy==1.24.3
oauthlib==3.2.2
opt-einsum==3.3.0
packaging==23.1
protobuf==4.24.3
pyasn1==0.5.0
pyasn1-modules==0.3.0
requests==2.31.0
requests-oauthlib==1.3.1
rsa==4.9
six==1.16.0
tensorboard==2.13.0
tensorboard-data-server==0.7.1
tensorflow==2.13.0
tensorflow-estimator==2.13.0
tensorflow-intel==2.13.0
tensorflow-io-gcs-filesystem==0.31.0
termcolor==2.3.0
typing_extensions==4.5.0
urllib3==1.26.16
Werkzeug==2.3.7
wrapt==1.15.0
```






Рисунок 9. Содержимое файла requirements.txt



Задание 7. Закоммитила изменения и прикрепила на GitHub.


-python-1
Общедоступный
Pin-код
Развернуть 1

Главная
1 Ветка
0 Тегов

Добавить файл
Код

Файл	Последнее изменение	Коммитов
 stefa-b environment.yml	6c246f1-4 минуты назад	5 Коммитов
 .gitignore	.gitignore	9 часов назад
 README.md	README.md	9 часов назад
 environment.yml	environment.yml	4 минуты назад
 Лицензия	Лицензия	на прошлой неделе

 **README**


Быковская Стефания Станиславовна ИТС-6-о-22-1 Лабораторная 2.14(1)

Ссылка: <https://github.com/stefa-b/-python-1.git>

Ответы на 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`. Для этого нужно выполнить команду `pip install package_name`

2. Как осуществить установку менеджера пакетов `pip`?

`Pip` обычно устанавливается автоматически вместе с Python. Если он не установлен можно использовать команду `python -m ensurepip --default-pip`

3. Откуда менеджер пакетов `pip` по умолчанию устанавливает пакеты?

По умолчанию, `pip` устанавливает пакеты из Python Package Index (PyPI)

4. Как установить последнюю версию пакета с помощью `pip`?

Для установки последней версии пакета с помощью `pip`, нужно выполнить команду `pip install --upgrade package_name`

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

Для установки заданной версии пакета с помощью `pip`, нужно выполнить команду `pip install package_name==desired_version`

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

Для установки пакета из `git` репозитория, можно использовать команду `pip install git+URL`

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

Для установки пакета из локальной директории, нужно выполнить команду `pip install /path/to/package_directory`

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

Чтобы удалить установленный пакет с помощью `pip`, нужно выполнить команду `pip uninstall package_name`

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

Для обновления установленного пакета с помощью `pip`, нужно использовать команду `pip install --upgrade package_name`

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

Чтобы отобразить список установленных пакетов с помощью `pip`, нужно

выполнить команду `pip list` или `pip freeze`.

11. Каковы причины появления виртуальных окружений в языке Python?

Виртуальные окружения в Python используются для изоляции проектов и их зависимостей. Основные причины их использования включают изоляцию проектов друг от друга, управление версиями зависимостей и предотвращение конфликтов между пакетами.

12. Каковы основные этапы работы с виртуальными окружениями?

1. Создание виртуального окружения.
2. Активация виртуального окружения.
3. Установка необходимых пакетов в виртуальное окружение.
4. Работа с проектом в активированном виртуальном окружении.
5. Деактивация виртуального окружения (по завершении работы).

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

- 1) Создание виртуального окружения: `python -m venv myenv`
- 2) Активация виртуального окружения
- 3) Установка пакетов: `pip install package_name`
- 4) Деактивация: `deactivate`

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

Работа с виртуальными окружениями с помощью `virtualenv` аналогична `venv`. Создаем окружение с помощью `virtualenv myenv`, активируем его, устанавливаем пакеты и деактивируем.

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

- 1) Создание виртуального окружения: `pipenv --python 3.8` (здесь указывается версия Python)
- 2) Установка пакетов: `pipenv install package_name`
- 3) Запуск оболочки в виртуальном окружении: `pipenv shell`

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

Файл `requirements.txt` используется для указания списка зависимостей

проекта. Создать его можно с помощью команды `pip freeze > requirements.txt`, и он будет содержать список установленных пакетов и их версий.

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

Преимущества conda по сравнению с pip включают в себя возможность установки бинарных зависимостей, управление средами и зависимостями, а также поддержку многих языков программирования, не только Python.

18. В какие дистрибутивы Python входит пакетный менеджер conda?

Пакетный менеджер conda входит в дистрибутивы Anaconda и Miniconda. Он также может быть установлен отдельно на другие дистрибутивы Python.

19. Как создать виртуальное окружение conda?

Для создания виртуального окружения conda нужно использовать команду `conda create --name myenv`

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

Для активации и установки пакетов в виртуальное окружение conda, нужно выполнить команду `conda activate myenv`, а затем команду `conda install package_name` для установки пакетов.

21. Как деактивировать и удалить виртуальное окружение conda?

Чтобы деактивировать и удалить виртуальное окружение conda, нужно выполнить команду `conda deactivate` для деактивации, а затем `conda env remove --name myenv` для удаления.

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

Файл `environment.yml` используется для определения окружения conda, включая зависимости. Его можно создать вручную, указав пакеты и их версии, или 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 помощью файла `environment.yml`, нужно использовать 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 среде PyCharm можно создавать и управлять виртуальными окружениями conda через интерфейс IDE. Это делается через "File" -> "Settings" -> "Project: <your project>" -> "Python Interpreter"

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

Файлы requirements.txt и environment.yml содержат информацию о зависимостях проекта и их версиях. Хранение их в репозитории Git позволяет другим разработчикам воссоздать окружение проекта и установить необходимые зависимости для работы с кодом.

Вывод: приобрели навыки по работе с менеджером пакетов pip и виртуальными окружениями с помощью языка программирования Python версии 3.x.