

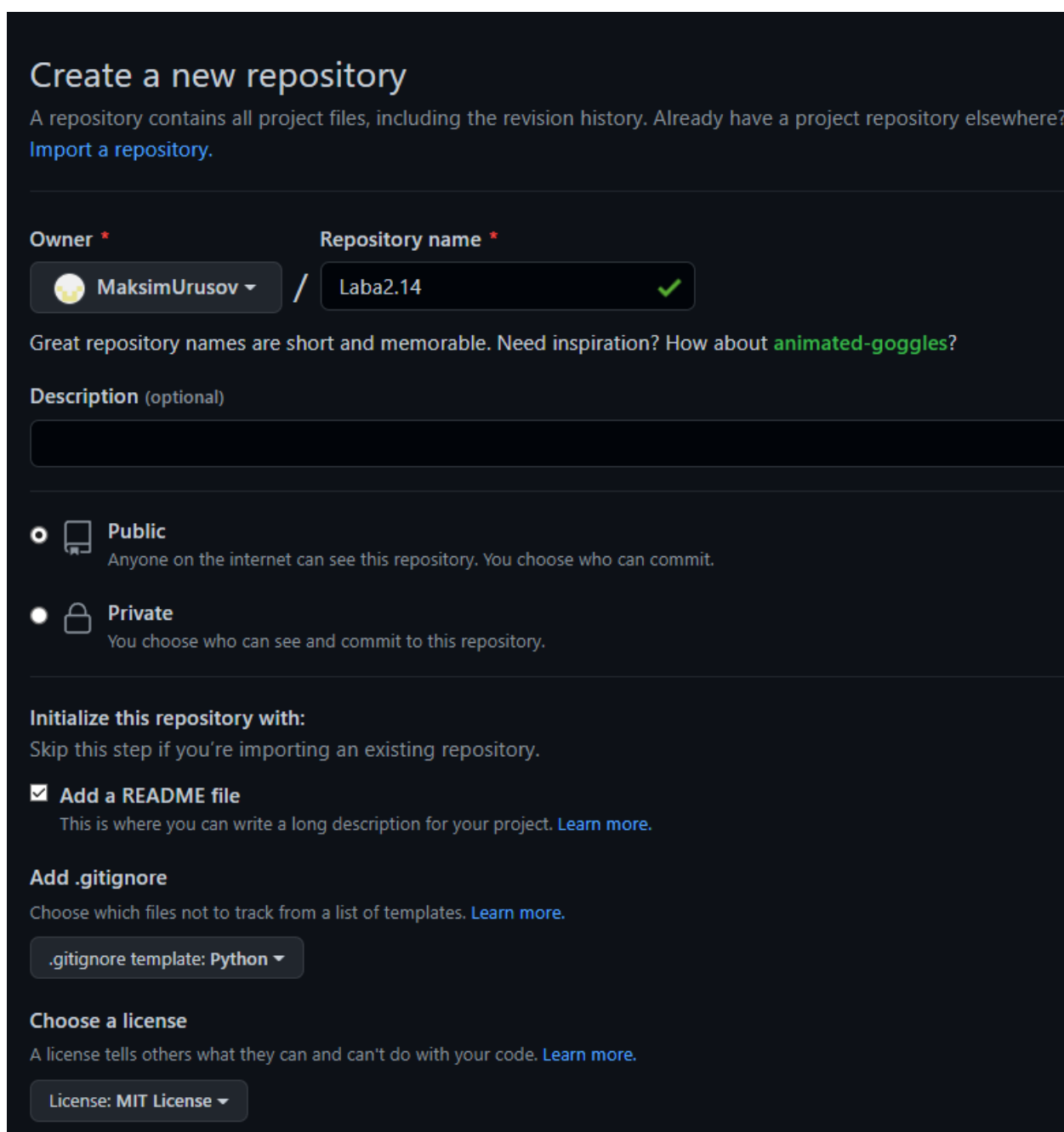
Лабораторная работа №9

Выполнил Эсеналиев Арсен

ИВТ-б-о-21-1

Цель: приобретение навыков по работе с функциями при написании программ с помощью языка программирования Python версии 3.x.

1. Создал общедоступный репозиторий на GitHub с MIT



The screenshot shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation of what a repository is. Below this, there are two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'MaksimUrusov' and the 'Repository name' is 'Laba2.14', which is marked as valid with a green checkmark. A note suggests that repository names should be short and memorable, with an example 'animated-goggles?'. Below the name field is a 'Description (optional)' text area. The next section is for visibility, with 'Public' selected by default and 'Private' as an alternative. The 'Initialize this repository with:' section has three options: 'Add a README file' (checked), 'Add .gitignore' (with a dropdown set to 'Python'), and 'Choose a license' (with a dropdown set to 'MIT License').

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner * Repository name *

MaksimUrusov / Laba2.14 ✓

Great repository names are short and memorable. Need inspiration? How about [animated-goggles?](#)

Description (optional)

☒ Public
Anyone on the internet can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

.gitignore template: Python

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

License: MIT License

2. Выполнил клонирование созданного репозитория.

```
D:\REP9>git clone https://github.com/Arsen445/LB2.14.git
Cloning into 'LB2.14'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

D:\REP9>_
```

3. Дополнил файл .gitignore необходимыми правилами для работы с IDE PyCharm.

```
# Created by https://www.toptal.com/developers/gitignore/api/python,pycharm
# Edit at https://www.toptal.com/developers/gitignore?templates=python,pycharm

### PyCharm ###
# Covers JetBrains IDEs: IntelliJ, RubyMine, PhpStorm, AppCode, PyCharm, CLion, Android Studio
# Reference: https://intellij-support.jetbrains.com/hc/en-us/articles/206544839

# User-specific stuff
.idea/**/workspace.xml
.idea/**/tasks.xml
.idea/**/usage.statistics.xml
.idea/**/dictionaries
.idea/**/shelf

# AWS User-specific
.idea/**/aws.xml

# Generated files
.idea/**/contentModel.xml

# Sensitive or high-churn files
.idea/**/dataSources/
```

4. Организовал репозиторий в соответствии с моделью ветвления git-flow.

```

D:\REP9\LB2.14>git flow init

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? []
Hooks and filters directory? [D:/REP9/LB2.14/.git/hooks]

D:\REP9\LB2.14>

```

5. Проверка версии pip

```

D:\REP9\LB2.14>pip --version
pip 22.0.4 from C:\Python39\lib\site-packages\pip (python 3.9)

D:\REP9\LB2.14>

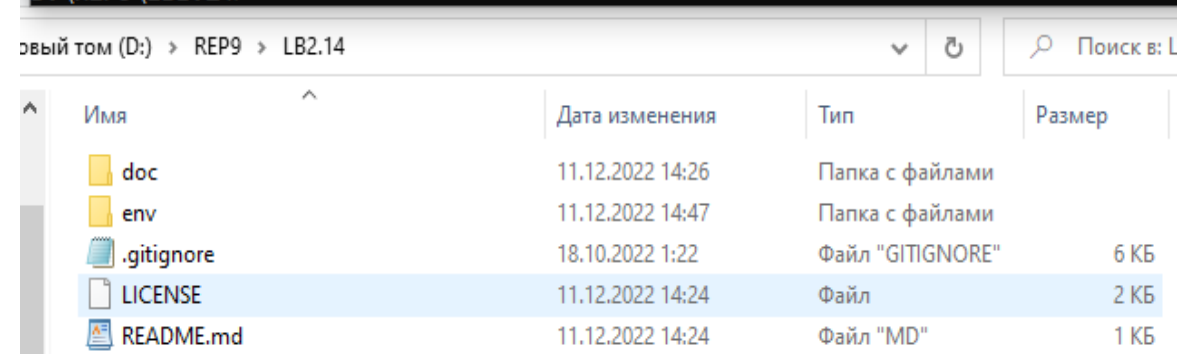
```

6. Создайте виртуальное окружение venv с именем репозитория.

```

D:\REP9\LB2.14>python3 -m venv 2_14
Python
D:\REP9\LB2.14>python3 -m venv env
Python
D:\REP9\LB2.14>python -m venv env
D:\REP9\LB2.14>

```



| Имя | Дата изменения | Тип | Размер |
|------------|------------------|------------------|--------|
| doc | 11.12.2022 14:26 | Папка с файлами | |
| env | 11.12.2022 14:47 | Папка с файлами | |
| .gitignore | 18.10.2022 1:22 | Файл "GITIGNORE" | 6 КБ |
| LICENSE | 11.12.2022 14:24 | Файл | 2 КБ |
| README.md | 11.12.2022 14:24 | Файл "MD" | 1 КБ |

7. Активируем виртуальное окружение

```

D:\REP9\LB2.14>.\env\scripts\activate
(env) D:\REP9\LB2.14>

```

8. Установка пакетов black и flake8

```
(env) D:\REP9\LB2.14>pip install black
Collecting black
  Downloading black-22.12.0-cp39-cp39-win_amd64.whl (1.2 MB)
----- 1.2/1.2 MB 1.9 MB/s eta 0:00:00
Collecting pathspec>=0.9.0
  Downloading pathspec-0.10.3-py3-none-any.whl (29 kB)
Collecting platformdirs>=2
  Downloading platformdirs-2.6.0-py3-none-any.whl (14 kB)
Collecting click>=8.0.0
  Downloading click-8.1.3-py3-none-any.whl (96 kB)
----- 96.6/96.6 KB 5.4 MB/s eta 0:00:00
Collecting mypy-extensions>=0.4.3
  Downloading mypy_extensions-0.4.3-py2.py3-none-any.whl (4.5 kB)
Collecting tomli>=1.1.0
  Downloading tomli-2.0.1-py3-none-any.whl (12 kB)
Collecting typing-extensions>=3.10.0.0
  Downloading typing_extensions-4.4.0-py3-none-any.whl (26 kB)
Collecting colorama
  Downloading colorama-0.4.6-py2.py3-none-any.whl (25 kB)
Installing collected packages: mypy-extensions, typing-extensions, tomli, platformdirs, pathspec, colorama, click, black
Successfully installed black-22.12.0 click-8.1.3 colorama-0.4.6 mypy-extensions-0.4.3 pathspec-0.10.3 platformdirs-2.6.0
tomli-2.0.1 typing-extensions-4.4.0
WARNING: You are using pip version 22.0.4; however, version 22.3.1 is available.
You should consider upgrading via the 'D:\REP9\LB2.14\env\Scripts\python.exe -m pip install --upgrade pip' command.

(env) D:\REP9\LB2.14>D:\REP9\LB2.14\env\Scripts\python.exe -m pip install --upgrade pip
Requirement already satisfied: pip in d:\rep9\lb2.14\env\lib\site-packages (22.0.4)
Collecting pip
  Downloading pip-22.3.1-py3-none-any.whl (2.1 MB)
----- 2.1/2.1 MB 1.5 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 22.0.4
    Uninstalling pip-22.0.4:
      Successfully uninstalled pip-22.0.4
Successfully installed pip-22.3.1

(env) D:\REP9\LB2.14>pip install black
Requirement already satisfied: black in d:\rep9\lb2.14\env\lib\site-packages (22.12.0)
Requirement already satisfied: mypy-extensions>=0.4.3 in d:\rep9\lb2.14\env\lib\site-packages (from black) (0.4.3)
Requirement already satisfied: click>=8.0.0 in d:\rep9\lb2.14\env\lib\site-packages (from black) (8.1.3)
Requirement already satisfied: typing-extensions>=3.10.0.0 in d:\rep9\lb2.14\env\lib\site-packages (from black) (4.4.0)
Requirement already satisfied: tomli>=1.1.0 in d:\rep9\lb2.14\env\lib\site-packages (from black) (2.0.1)
Requirement already satisfied: platformdirs>=2 in d:\rep9\lb2.14\env\lib\site-packages (from black) (2.6.0)
Requirement already satisfied: pathspec>=0.9.0 in d:\rep9\lb2.14\env\lib\site-packages (from black) (0.10.3)
Requirement already satisfied: colorama in d:\rep9\lb2.14\env\lib\site-packages (from click>=8.0.0->black) (0.4.6)

(env) D:\REP9\LB2.14>
```

```
(env) D:\REP9\LB2.14>pip install flake8
Collecting flake8
  Downloading flake8-6.0.0-py2.py3-none-any.whl (57 kB)
----- 57.8/57.8 KB 506.1 KB/s eta 0:00:00
Collecting mccabe<0.8.0,>=0.7.0
  Downloading mccabe-0.7.0-py2.py3-none-any.whl (7.3 kB)
Collecting pycodestyle<2.11.0,>=2.10.0
  Downloading pycodestyle-2.10.0-py2.py3-none-any.whl (41 kB)
----- 41.3/41.3 KB 1.9 MB/s eta 0:00:00
Collecting pyflakes<3.1.0,>=3.0.0
  Downloading pyflakes-3.0.1-py2.py3-none-any.whl (62 kB)
----- 62.8/62.8 KB 1.6 MB/s eta 0:00:00
Installing collected packages: pyflakes, pycodestyle, mccabe, flake8
Successfully installed flake8-6.0.0 mccabe-0.7.0 pycodestyle-2.10.0 pyflakes-3.0.1

(env) D:\REP9\LB2.14>
```

9. Деактивация окружения

```
(env) D:\REP9\LB2.14>deactivate
D:\REP9\LB2.14>
```

10. Установка виртуального окружения **virtualenv**

```
D:\REP9\LB2.14>python -m pip install virtualenv
Collecting virtualenv
  Downloading virtualenv-20.17.1-py3-none-any.whl (8.8 MB)
----- 8.8/8.8 MB 4.5 MB/s eta 0:00:00
Collecting filelock<4,>=3.4.1
  Downloading filelock-3.8.2-py3-none-any.whl (10 kB)
Collecting distlib<1,>=0.3.6
  Downloading distlib-0.3.6-py2.py3-none-any.whl (468 kB)
----- 468.5/468.5 kB 4.9 MB/s eta 0:00:00
Collecting platformdirs<3,>=2.4
  Using cached platformdirs-2.6.0-py3-none-any.whl (14 kB)
Installing collected packages: distlib, platformdirs, filelock, virtualenv
Successfully installed distlib-0.3.6 filelock-3.8.2 platformdirs-2.6.0 virtualenv-20.17.1
D:\REP9\LB2.14>
```

11. Создание 2 виртуального окружения **virtualenv**

```
D:\REP9\LB2.14>virtualenv -p python env
created virtual environment CPython3.9.13.final.0-64 in 2756ms
creator CPython3Windows(dest=D:\REP9\LB2.14\env, clear=False, no_vcs_ignore=False, global=False)
seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=C:\Use
rs\GG_Force\AppData\Local\pypa\virtualenv)
added seed packages: black==22.12.0, click==8.1.3, colorama==0.4.6, flake8==6.0.0, mccabe==0.7.0, mypy_exte
nsions==0.4.3, pathspec==0.10.3, pip==22.3.1, platformdirs==2.6.0, pycodestyle==2.10.0, pyflakes==3.0.1, setupt
ools==65.6.3, tomli==2.0.1, typing_extensions==4.4.0, wheel==0.38.4
activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
D:\REP9\LB2.14>
```

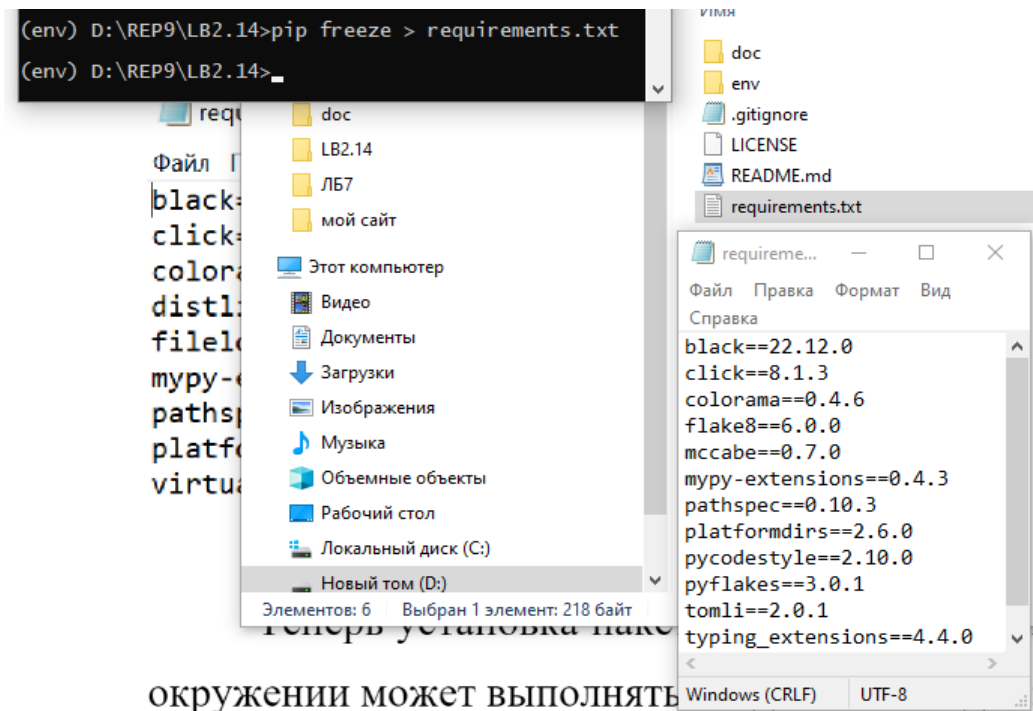
12. Проверка на активацию и деактивацию окружения

```
D:\REP9\LB2.14>.\env\scripts\activate
(env) D:\REP9\LB2.14>deactivate
D:\REP9\LB2.14>
```

13. Просмотр всех установленных пакетов при помощи команды **pip freeze**

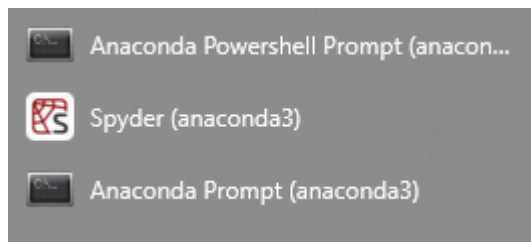
```
(env) D:\REP9\LB2.14>pip freeze
black==22.12.0
click==8.1.3
colorama==0.4.6
flake8==6.0.0
mccabe==0.7.0
mypy-extensions==0.4.3
pathspec==0.10.3
platformdirs==2.6.0
pycodestyle==2.10.0
pyflakes==3.0.1
tomli==2.0.1
typing_extensions==4.4.0
```

14. Сохраняем этот список в txt файл при помощи команды:



15. Создаем виртуальное окружение Anaconda

1. Запускаем Anaconda powershell prompt



2. Чистое виртуальное окружение

```
Anaconda Powershell Prompt (anaconda3)
(base) PS C:\Users\GG_Force> mkdir %LB2.14%

Каталог: C:\Users\GG_Force

Mode                LastWriteTime         Length Name
----                -
d-----          11.12.2022      18:46             %LB2.14%

(base) PS C:\Users\GG_Force>
```

3. Активация

```
(base) PS C:\Users\GG_Force> d:
(base) PS D:\> cd REP9
(base) PS D:\REP9> mkdir %LB2.14%

Каталог: D:\REP9

Mode                LastWriteTime         Length Name
----                -
d-----          11.12.2022      19:37             %LB2.14%

(base) PS D:\REP9> cd %LB2.14%
(base) PS D:\REP9\%LB2.14%> copy NUL > main.py
(base) PS D:\REP9\%LB2.14%> conda activate %LB2.14%
(%LB2.14%) PS D:\REP9>
```

16. Установка пакетов

```
(%LB2.14%) PS D:\REP9> conda install pip, numpy, pandas, scipy
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
1  current version: 22.9.0
   latest version: 22.11.1

Please update conda by running

    $ conda update -n base -c defaults conda

## Package Plan ##

  environment location: C:\Users\GG_Force\.conda\envs\%LB2.14%

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

The following packages will be downloaded:
```

| package | build | |
|-------------------|----------------|---------|
| numexpr-2.8.4 | py39h5b0cc5e_0 | 127 KB |
| numpy-1.23.4 | py39h3b20f71_0 | 11 KB |
| numpy-base-1.23.4 | py39h4da318b_0 | 6.0 MB |
| pandas-1.5.2 | py39hf11a4ad_0 | 10.5 MB |
| scipy-1.9.3 | py39he11b74f_0 | 18.0 MB |
| Total: | | 34.7 MB |

```
The following NEW packages will be INSTALLED:
```

```
blas                pkgs/main/win-64::blas-1.0-mkl None
bottleneck          pkgs/main/win-64::bottleneck-1.3.5-py39h080aedc_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
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-py39h2bbff1b_0 None
mkl_fft             pkgs/main/win-64::mkl_fft-1.3.1-py39h277e83a_0 None
mkl_random          pkgs/main/win-64::mkl_random-1.2.2-py39hf11a4ad_0 None
numexpr             pkgs/main/win-64::numexpr-2.8.4-py39h5b0cc5e_0 None
numpy               pkgs/main/win-64::numpy-1.23.4-py39h3b20f71_0 None
numpy-base          pkgs/main/win-64::numpy-base-1.23.4-py39h4da318b_0 None
packaging            pkgs/main/noarch::packaging-21.3-pyhd3eb1b0_0 None
pandas              pkgs/main/win-64::pandas-1.5.2-py39hf11a4ad_0 None
pyparsing           pkgs/main/win-64::pyparsing-3.0.9-py39haa95532_0 None
python-dateutil     pkgs/main/noarch::python-dateutil-2.8.2-pyhd3eb1b0_0 None
pytz                pkgs/main/win-64::pytz-2022.1-py39haa95532_0 None
scipy               pkgs/main/win-64::scipy-1.9.3-py39he11b74f_0 None
six                 pkgs/main/noarch::six-1.16.0-pyhd3eb1b0_1 None

Proceed ([y]/n)? y

Downloading and Extracting Packages
numpy-base-1.23.4    | 6.0 MB | ##### | 100%
pandas-1.5.2         | 10.5 MB | ##### | 100%
numpy-1.23.4         | 11 KB | ##### | 100%
scipy-1.9.3          | 18.0 MB | ##### | 100%
numexpr-2.8.4        | 127 KB | ##### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
Retrieving notices: ...working... done
(%LB2.14%) PS D:\REP9>
```

17. Установка пакета TensorFlow через Conda


```
(%LB2.14%) PS D:\REP9> conda install tensorflow
Collecting package metadata (current_repodata.json): done
Solving environment: done

--> WARNING: A newer version of conda exists. <==
  current version: 22.9.0
  latest version: 22.11.1
Please update conda by running

  $ conda update -n base -c defaults conda

## Package Plan ##
  environment location: C:\Users\GG_Force\.conda\envs\%LB2.14%
  added / updated specs:
    - tensorflow

The following packages will be downloaded:



| package                      | build          | size   |
|------------------------------|----------------|--------|
| tfio-2.3.0                   | mk1            | 3 KB   |
| abseil-cpp-20211102.0        | h077b12b_0     | 1.7 MB |
| absl-py-1.3.0                | py39haa95532_0 | 171 KB |
| astunparse-1.6.3             | py39h2bbff4b_0 | 415 KB |
| audioread-1.2.0              | pyhd3eb1b0_0   | 12 KB  |
| astunparse-1.6.3             | py39haa95532_0 | 12 KB  |
| async-timeout-4.0.2          | py39haa95532_0 | 12 KB  |
| atrac-1.4                    | py39haa95532_0 | 23 KB  |
| cachetools-4.2.2             | pyhd3eb1b0_0   | 13 KB  |
| certifi-1.5.5                | py39h2bbff4b_3 | 238 KB |
| cryptography-38.0.1          | py39h2bbff4b_0 | 991 KB |
| flatbuffers-2.0.0            | h0c2883c_0     | 1.4 MB |
| flit-core-3.6.0              | pyhd3eb1b0_0   | 42 KB  |
| flit-core-3.6.0              | py39h2bbff4b_0 | 40 KB  |
| gast-0.5.3                   | pyhd3eb1b0_0   | 21 KB  |
| google-auth-2.6.0            | pyhd3eb1b0_0   | 43 KB  |
| google-auth-oauthlib-0.4.4   | pyhd3eb1b0_0   | 18 KB  |
| google-pasta-0.2.0           | py39haa95532_0 | 40 KB  |
| grpcio-1.42.0                | py39haa95532_0 | 1.9 MB |
| idna-3.4                     | py39haa95532_0 | 93 KB  |
| keras-preprocessing-1.1.2    | py39haa95532_0 | 1.5 MB |
| libprotobuf-3.20.1           | py39haa95532_0 | 15 KB  |
| libprotobuf-3.20.1           | h2c4e6ff_0     | 2.0 MB |
| multidict-0.9.2              | py39h2bbff4b_0 | 43 KB  |
| oauthlib-3.2.1               | py39haa95532_0 | 104 KB |
| opt-einsum-3.3.0             | py39haa95532_0 | 12 KB  |
| protobuf-3.20.1              | py39h2bbff4b_3 | 231 KB |
| python-flatbuffers-2.0       | pyhd3eb1b0_0   | 24 KB  |
| requests-oauthlib-1.3.0      | py39haa95532_0 | 23 KB  |
| rsa-4.7.2                    | py39haa95532_0 | 20 KB  |
| tfio-2.3.0                   | py39h2bbff4b_3 | 1.9 MB |
| tensorflow-2.9.0             | py39haa95532_0 | 5.5 MB |
| tensorflow-data-server-0.0.0 | py39haa95532_0 | 17 KB  |
| tensorflow-plugin-win-1.0.1  | py39haa95532_0 | 671 KB |


```

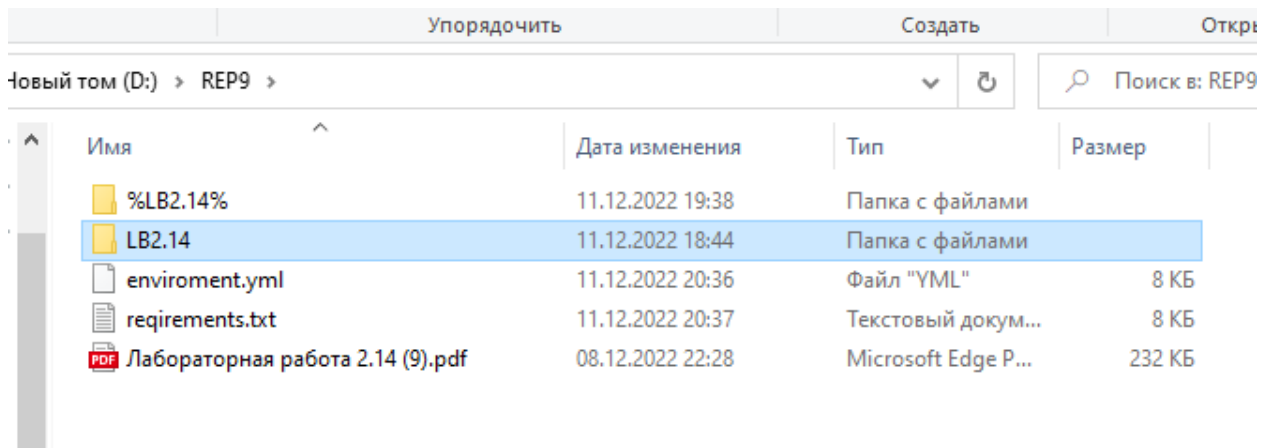
18. Установка пакета TensorFlow через pip

```
(%LB2.14%) PS D:\REP9> pip install tensorflow
Requirement already satisfied: tensorflow in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (2.9.1)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.14.1)
Requirement already satisfied: numpy>=1.20 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.21.5)
Requirement already satisfied: packaging in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (21.3)
Collecting protobuf<3.20,>=3.9.2
  Downloading protobuf-3.19.6-cp39-cp39-win_amd64.whl (895 kB)
    ----- 895.9/895.9 kB 1.7 MB/s eta 0:00:00
Requirement already satisfied: termcolor>=1.1.0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (2.1.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.42.0)
Requirement already satisfied: typing-extensions>=3.6 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (4.4.0)
Requirement already satisfied: keras-preprocessing>=1.1.1 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.1.2)
Requirement already satisfied: tensorflow-estimator<2.10.0,>=2.9.0rc0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (2.9.0)
Collecting gast<0.4.0,>=0.2.1
  Downloading gast-0.4.0-py3-none-any.whl (9.8 kB)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.6.3)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (0.2.0)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (3.3.0)
Collecting flatbuffers<2,>=1.12
  Downloading flatbuffers-1.12-py2.py3-none-any.whl (15 kB)
Collecting libclang>=13.0.0
  Downloading libclang-14.0.6-py2.py3-none-win_amd64.whl (14.2 MB)
    ----- 14.2/14.2 MB 7.0 MB/s eta 0:00:00
Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (2.9.0)
Collecting tensorflow-io-gcs-filesystem>=0.23.1
  Downloading tensorflow_io_gcs_filesystem-0.28.0-cp39-cp39-win_amd64.whl (1.5 MB)
    ----- 1.5/1.5 MB 6.7 MB/s eta 0:00:00
Requirement already satisfied: six>=1.12.0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (1.16.0)
Requirement already satisfied: h5py>=2.0.0 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (3.7.0)
Requirement already satisfied: tensorflow<2.10.0,>=2.9 in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (2.9.0)
Requirement already satisfied: setuptools in c:\users\gg_force\.conda\envs\%LB2.14%\lib\site-packages (from tensorflow) (65.5.0)
```

19. Формируем список нужных пакетов и конфигураций

```
(%LB2.14%) PS D:\REP9> conda env export > environment.yml
(%LB2.14%) PS D:\REP9> conda env export > requirements.txt
(%LB2.14%) PS D:\REP9>
```

TENSORFLOW.



20. Зафиксируйте сделанные изменения в репозитории.

```
D:\REP9\LB2.14>git push --all
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 408 bytes | 408.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/Arsen445/LB2.14/pull/new/develop
remote:
To https://github.com/Arsen445/LB2.14.git
 * [new branch]      develop -> develop
```

21. Выполните слияние ветки для разработки с веткой main/master.

```
D:\REP9\LB2.14>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

D:\REP9\LB2.14>git merge develop
Updating 16686cd..9bda322
Fast-forward
 requirements.txt | 12 ++++++++
 1 file changed, 12 insertions(+)
 create mode 100644 requirements.txt

D:\REP9\LB2.14>
```

Контрольные вопросы:

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.

Как осуществить установку менеджера пакетов pip?

При развертывании современной версии Python, pip устанавливается автоматически. 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.

Откуда менеджер пакетов pip по умолчанию устанавливает пакеты?

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

Как установить последнюю версию пакета с помощью pip?

С помощью команды `$ pip install ProjectName`.

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.

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`

Как установить пакет из локальной директории с помощью pip?

С помощью команды `$ pip install ./dist/ProjectName.tar.gz`

Как удалить установленный пакет с помощью pip?

С помощью команды `$ pip uninstall ProjectName` можно удалить установленный пакет.

Как обновить установленный пакет с помощью pip?

С помощью команды `$ pip install --upgrade ProjectName` можно обновить необходимый пакет.

Как отобразить список установленных пакетов с помощью pip?

Командой `$ pip list` можно отобразить список установленных пакетов.

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 совместимости - некоторые операционные системы, например, Linux и MacOS используют содержащиеся в них предустановленные интерпретаторы Python. Обновив или изменив самостоятельно версию

какого-то установленного глобально пакета, мы можем непреднамеренно сломать 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.

Каковы основные этапы работы с виртуальными окружениями?

Основные этапы: 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.

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.

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 самого интерпретатора, полной стандартной библиотеки, `pip`, и, что самое главное, копии всех необходимых пакетов.

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`.

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 окружение (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.

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`).

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.

Как создать виртуальное окружение conda?

С помощью команды: `conda create -n %PROJ_NAME% python=3.7`

Как активировать и установить пакеты в виртуальное окружение conda?

Чтобы установить пакеты, необходимо воспользоваться командой: – `conda install A` для активации: `conda activate %PROJ_NAME%`

Как деактивировать и удалить виртуальное окружение conda?

Для деактивации использовать команду: `conda deactivate`, а для удаления: `conda remove -n $PROJ_NAME`.

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

Создание файла: `conda env export > environment.yml` Файл `environment.yml` позволит воссоздать окружение в любой нужный момент.

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

Достаточно набрать: `conda env create -f environment.yml`

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 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 нового окружения. Нажимаем на OK. 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.

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: в результате выполнения лабораторной работы были приобретены теоретические сведения и практические навыки для работы с менеджером пакетов pip и виртуальными окружениями с помощью языка программирования Python версии 3.x..