

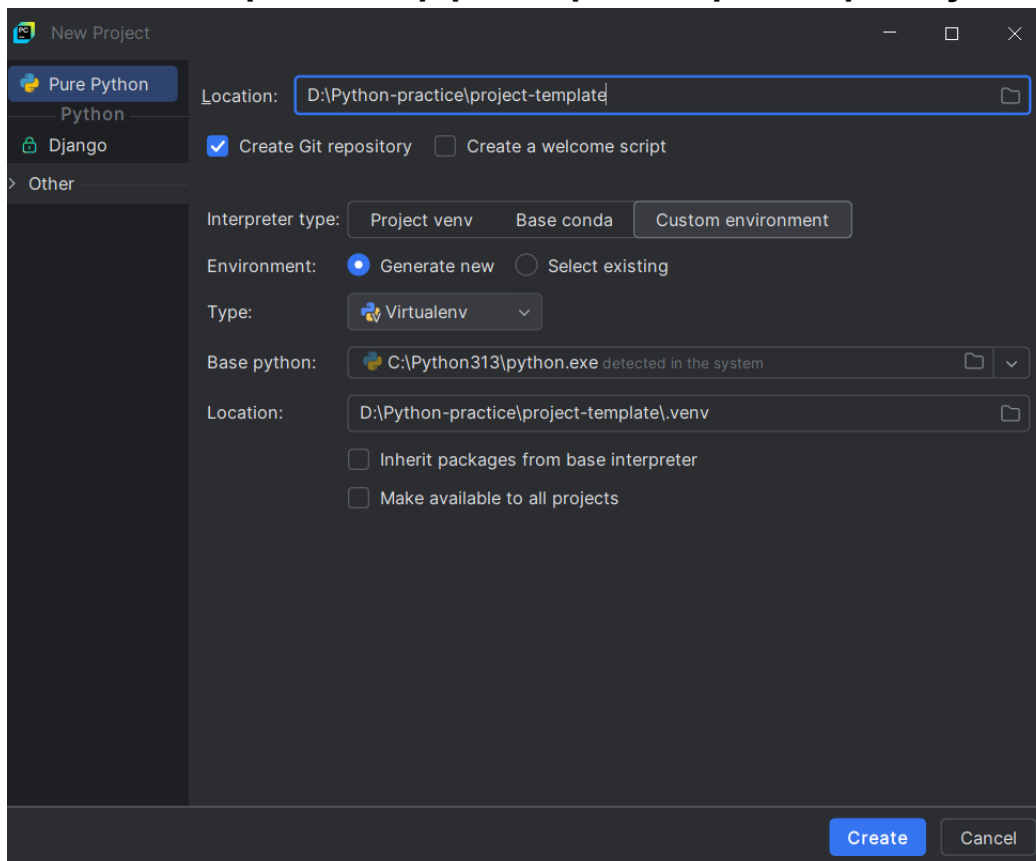
Практична робота №7. Проєкт, модулі, імпорт бібліотек, рір. Робота з файлами у Python.

Єлизавета Петренко, 6та група.

https://github.com/Yelyzavetx/python-practice-project_template.git

0. Підготувати звіт, де в репозиторії та скріншотах відображається кожен етап, який пізніше Ви зможете прикріпити у мудл. Цей файл використовувати в якості шаблону для звіту. **1. Створення нового проєкту.**
 - a. Створити новий проєкт локально у PyCharm або VSCode (можна частково використовувати інструкції з ПЗ 1). При створенні проєкта, назвіть його «project_template» та оберіть створення віртуального середовища venv (***робота з рірenv самостійно на оцінку 80-90**), main.py створимо пізніше. Назву папки віртуального середовища запам'ятайте, ми використаємо її пізніше.

***для роботи з рірenv при створенні проєкту**



- b. Підготуйте файл `.gitignore`, щоб папки типу `venv` або `.idea` і.т.п. не потрапили до репозиторію, який призначений суто для коду проекту.



```
1 # created by virtualenv automatically ✓
2 # Python specific
3 __pycache__/
4 *.py[cod]
5 *.pyo
6 *.pyd
7 *.pdb
8 *.egg-info/
9 dist/
10 build/
11 eggs/
12 *.egg
13
14 # Virtual environment
15 venv/
16 env/
17 ENV/
18 .venv/
19 .venv.*
20
21 # IDE specific
22 .idea/
23
24 .DS_Store
25
26 Thumbs.db
27 Desktop.ini
28
29 *.iml
30
```

- c. Створіть файл `main.py` у директорії проекту, який матиме наступний вигляд:



```
1 def main(): 1 usage new *
2     pass
3
4 if __name__ == "__main__":
5     main()
```

- d. Створіть також новий репозиторій на GitHub (теж підглянути, як це робиться, можете у ПЗ 1).

master

1 Branch 0 Tags

Go to file

t

Add file

<> Code

- е. Об'єднайте локальний та віддалений репозиторії. Залийте зміни на віддалений репозиторій (тут теж можете згадати ПЗ 1). Посилання на нього додайте на початок звіту.

master

1 Branch 0 Tags

Go to file

t

Add file

<> Code

Yelyzavetx .gitignore was updated and main.py was created

be57547 · 3 minutes ago

2 Commits

| | | |
|---------|--|---------------|
| .idea | .gitignore was updated and main.py was created | 3 minutes ago |
| .venv | .gitignore was updated and main.py was created | 3 minutes ago |
| main.py | .gitignore was updated and main.py was created | 5 minutes ago |

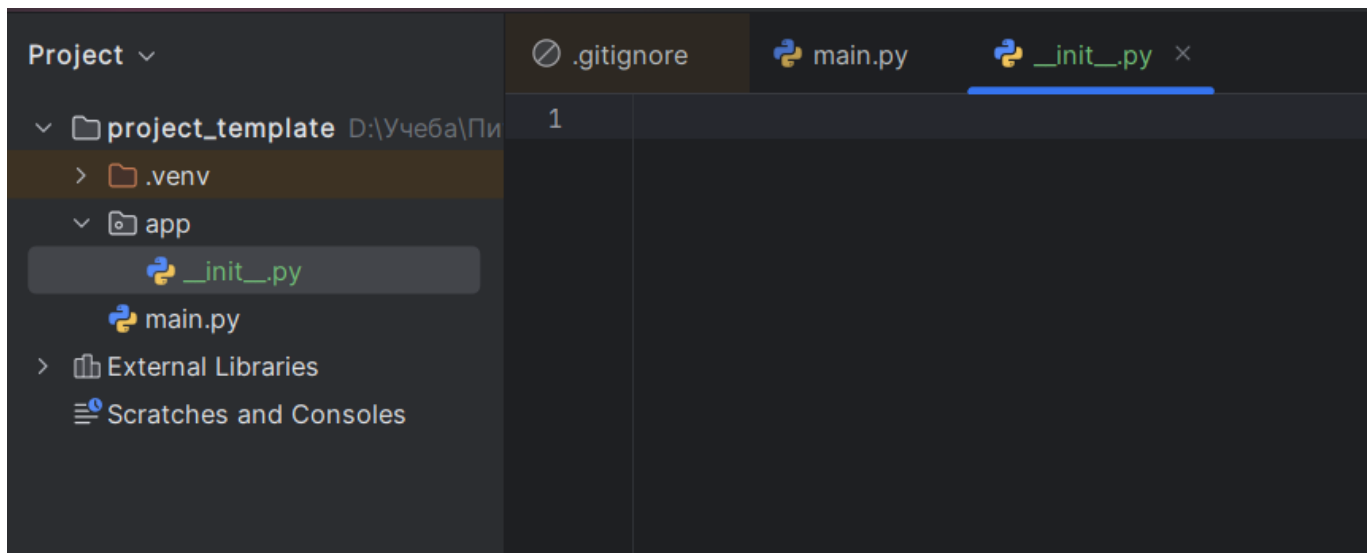
README

Pushed master to new branch
origin/master

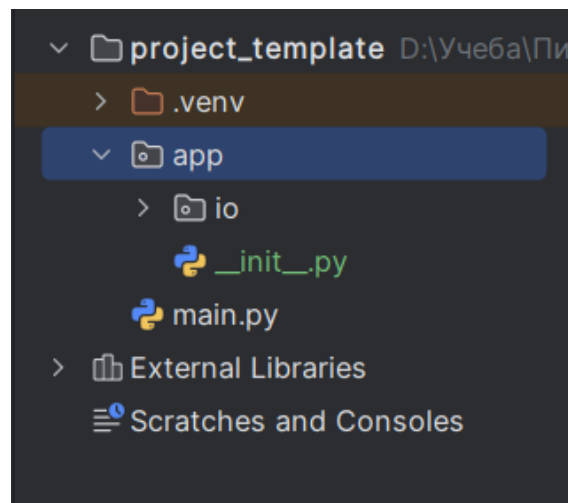
2. Структура проєкту.

- а. Створити в директорії проєкту нову папку (Python Package – директорія, яка має одразу пустий файл `__init__.py`) і назвати її «app».

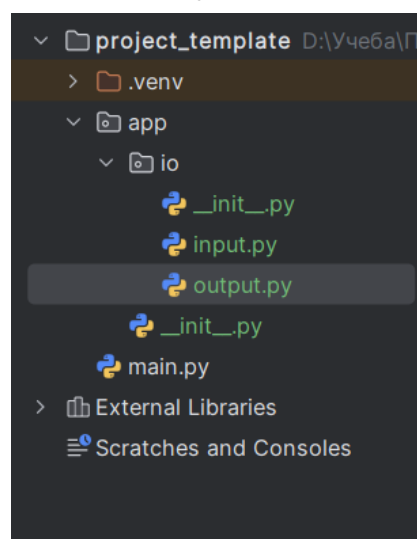
Це є місце, де структуровано зберігаються модулі проєкту з кодом, який безпосередньо бере участь у запуску та виконанні задач застосунку. Тобто це код, який запускається користувачем (у його ролі може бути як людина, що на кнопку на фронтенді натиснула, так і інша система, яка, наприклад, використовує результати поточної).



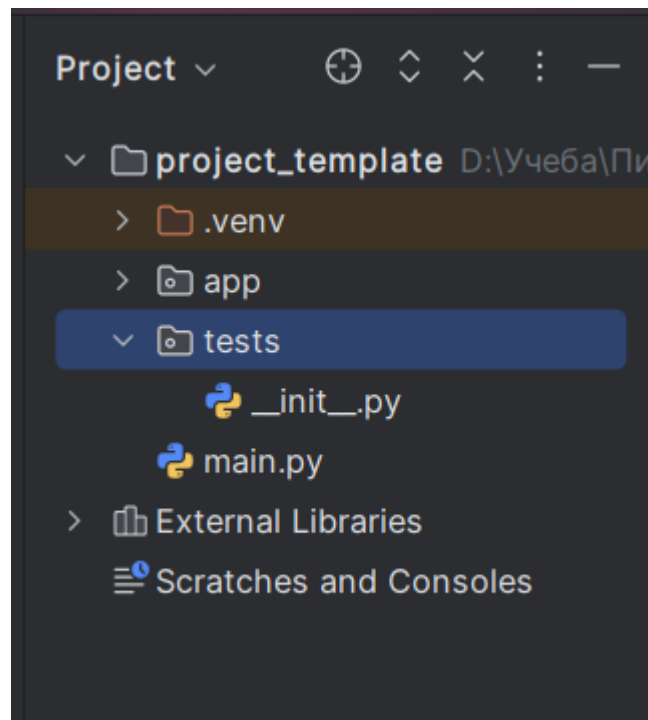
b. У середині цієї директорії app створити Python Package «io» (скорочено input-output).



c. У цій директорії io створити два файли: input.py та output.py.



- d. Створити ще один Python Package і назвати його «tests». Це є директорія, що містить unit тести, та буде дзеркальною для app (тобто, наприклад, файл `test_input.py` у `tests` відповідатиме файлу `input.py` у `app`, і те саме для піддиректорії `io` у `app` та `test_io` у `tests` і т.д.).



- е. Залити зміни на віддалений репозиторій з відповідним повідомленням у коміті.

| | | |
|---|--|----------------|
| Yelyzavetx Project structure was added 49c52b8 · 1 minute ago 3 Commits | | |
| .idea | .gitignore was updated and main.py was created | 20 minutes ago |
| .venv | .gitignore was updated and main.py was created | 20 minutes ago |
| app | Project structure was added | 1 minute ago |
| tests | Project structure was added | 1 minute ago |
| main.py | .gitignore was updated and main.py was created | 22 minutes ago |

3. Робота з модулями.

1. Якщо ви працюєте з `pipenv`, перейдіть до кроку 3.

Переконайтесь, що ваше віртуальне середовище активовано. Якщо ні, переходьте до кроку 2. Щоби перевірити, що середовище активовано, використайте відповідну команду, яка покаже, який інтерпретатор використовується в даний момент у проєкті.

Для Windows:

```
where python
```

Маєте побачити повний шлях до віртуального середовища у проєкті. Наприклад:

```
(.venv) PS D:\Учеба\Питон\project_template> where python
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe
C:\Python313\python.exe
C:\Users\2025\AppData\Local\Microsoft\WindowsApps\python.exe
(.venv) PS D:\Учеба\Питон\project_template> 
```

2. Активуйте його самостійно за допомогою наступних команд у терміналі у директорії проєкту можна переключитись за допомогою команди
- ```
cd path/to/proj_dir)
```

Для Windows:

```
nazva_venv\Scripts\activate
```

де `nazva_venv` – це назва папки з віртуальним середовищем при створенні у вашому проєкті, скоріше за все, вона має назву `venv`.

```
(.venv) PS D:\Учеба\Питон\project_template> where python
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe
C:\Python313\python.exe
C:\Users\2025\AppData\Local\Microsoft\WindowsApps\python.exe
(.venv) PS D:\Учеба\Питон\project_template>
```

### 3. Підготуйте pip. Для Windows:

```
py -m pip install --upgrade pip
py -m pip --version
```

Після цього маєте побачити свіжу версію менеджера пакетів pip.

```
(.venv) PS D:\Учеба\Питон\project_template> py -m pip install --upgrade pip
py -m pip --version

Usage:
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe -m pip install [options] <requirement specifier> [package-index-options] ...
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe -m pip install [options] -r <requirements file> [package-index-options] ...
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe -m pip install [options] [-e] <vcs project url> ...
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe -m pip install [options] [-e] <local project path> ...
D:\Учеба\Питон\project_template\.venv\Scripts\python.exe -m pip install [options] <archive url/path> ...
```

### 4. Встановлюємо пакети через pip.

**Якщо ви працюєте з `pipenv`, після прочитання цієї статті <https://realpython.com/pipenv-guide/> виконайте аналогічні для `pipenv` інструкції нижче (мається на увазі не виконання 1-в-1, а знаходження інструкцій, як зробити ту ж саму логіку, але через `pipenv`).**

#### 4.а. Встановлення останньої версії пакету.

Для цього рекомендую вам перейти на сайт <https://pypi.org> та в пошуку знайти пакет `numpy`.

Фильтр по [классификатору](#).

10 000+ проектов по запросу «numpy»

Сортировать по

Соответствию

- Framework
- Topic
- Development Status
- License
- Programming Language
- Operating System
- Environment
- Intended Audience
- Natural Language
- Typing



**numpy**

Fundamental package for array computing in Python

16 мар. 2025 г.



**faster-numpy**

26 сент. 2023 г.



**funcnodes-numpy**

implementations of numpy for funcnodes

31 янв. 2025 г.



**guillotina-numpy**

22 сент. 2020 г.



**hypothesis-numpy2**

Provides strategies for generating various `numpy` objects

12 апр. 2020 г.



**idx2numpy**

9 окт. 2020 г.

## Опис проекту



powered by NumFOCUS PyPI downloads 237M/month Conda downloads 71M stackoverflow Ask questions  
DOI 10.1038/s41592-019-0686-2 openssf scorecard 8.7

NumPy is the fundamental package for scientific computing with Python.

Скопіюйте цю команду з верхньої частини сторінки та запустіть її у терміналі.

```
pip install numpy
```

Після цього ви маєте бачити повідомлення про успішну інсталяцію пакету numpy та його dependencies (залежностей - пакетів).



```
(.venv) PS D:\Учеба\Питон\project_template> pip install numpy
Collecting numpy
 Obtaining dependency information for numpy from https://files.pythonhosted.org/packages/01/96/9e959196d60c2345829916cd89b147343574d457327dbf735e0eb960c/numpy-2.2.4-cp313-cp313-win_amd64.whl.metadata
 Downloading numpy-2.2.4-cp313-cp313-win_amd64.whl.metadata (60 kB)
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 60.8/60.8 kB 502.0 kB/s eta 0:00:00
 Downloading numpy-2.2.4-cp313-cp313-win_amd64.whl (12.6 MB)
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.6/12.6 MB 25.1 MB/s eta 0:00:00
Installing collected packages: numpy
Successfully installed numpy-2.2.4

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS D:\Учеба\Питон\project_template>
```

4.b. Встановлення конкретної версії пакету (рекомендований спосіб для подальшого використання).

Тепер знайдіть у рурі бібліотеку pandas, в історії версій (релізів) знайдіть **передостанню** версію та введіть у терміналі команду, щоб встановити його з відповідною версією:



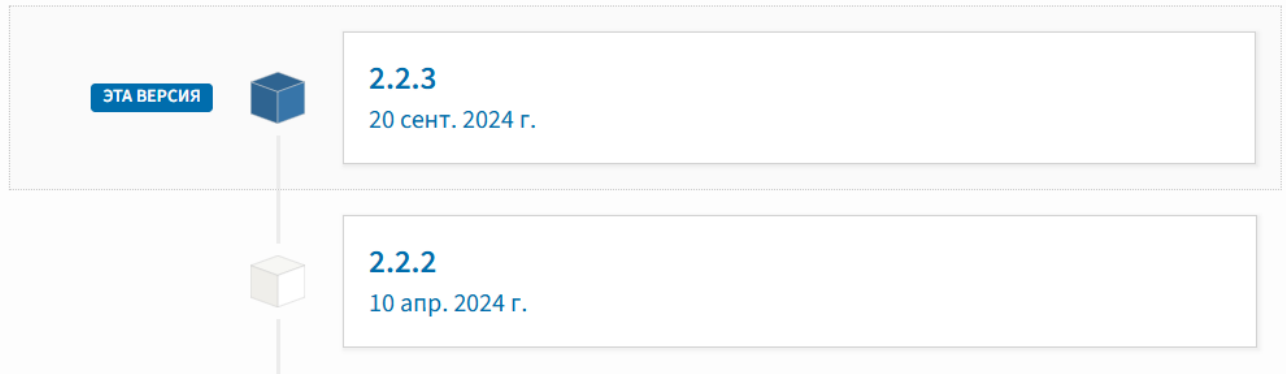
pandas: powerful Python data analysis toolkit

|         |                                                                                                                                                                   |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Testing | Unit Tests <span>passing</span> codecov <span>85%</span>                                                                                                          |
| Package | <a href="#">pypi v2.2.3</a> <a href="#">PyPI downloads 301M/month</a> <a href="#">Anaconda.org 2.2.3</a> <a href="#">Conda downloads 62M</a>                      |
| Meta    | <a href="#">powered by NumFOCUS</a> <a href="#">DOI 10.5281/zenodo.3509134</a> <a href="#">license BSD</a> <a href="#">join Slack</a> <a href="#">information</a> |

What is it?

## История выпусков

[Уведомления о выпусках](#) | [Лента RSS](#)



```
(.venv) PS D:\Python-practice\project-template> py -m pip install "pandas==2.2.3"
Requirement already satisfied: pandas==2.2.3 in d:\python-practice\project-template\.venv\lib\site-packages (2.2.3)
Requirement already satisfied: numpy>=1.26.0 in d:\python-practice\project-template\.venv\lib\site-packages (from pandas==2.2.3) (2.2.4)
Requirement already satisfied: python-dateutil>=2.8.2 in d:\python-practice\project-template\.venv\lib\site-packages (from pandas==2.2.3) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in d:\python-practice\project-template\.venv\lib\site-packages (from pandas==2.2.3) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in d:\python-practice\project-template\.venv\lib\site-packages (from pandas==2.2.3) (2025.2)
Requirement already satisfied: six>=1.5 in d:\python-practice\project-template\.venv\lib\site-packages (from python-dateutil>=2.8.2->pandas==2.2.3) (1.17.0)

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS D:\Python-practice\project-template>
```

Для Windows:

```
python -m pip install "SomeProject==1.4" або
py -m pip install "SomeProject==1.4"
```

seaborn 0.13.2

pip install seaborn

Опубліковано: 25 січ. 2024 р.

Історія випусків

0.13.2  
25 січ. 2024 р.

0.13.1  
1 січ. 2024 р.

```
python -m pip install "seaborn==0.13.1"
```

Більше про встановлення бібліотек можете прочитати тут:

<https://packaging.python.org/en/latest/guides/installing-using-pip-andvirtual-environments/>

5. Тепер пропоную вам встановити самостійно додатково пакети matplotlib та pylint, black.

pypi v3.10.1 conda-forge v3.10.1 downloads 85M/month powered by NumFOCUS

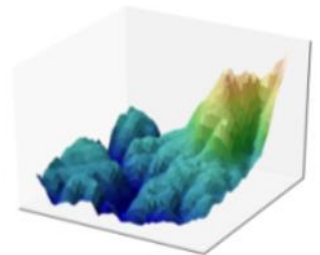
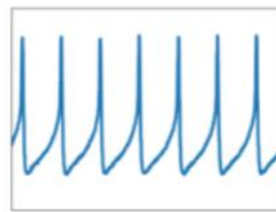
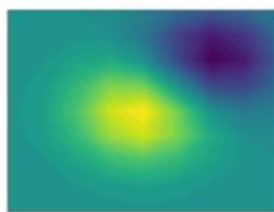
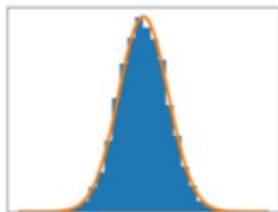
help forum discourse chat on gitter issue tracking github PR Welcome

Tests passing Azure Pipelines succeeded build unknown codecov 87% version scheme EffVer

# matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.

Check out our [home page](#) for more information.



```
Downloading pyparsing-3.2.3-py3-none-any.whl (111 kB)
111.1/111.1 kB 20.6 MB/s eta 0:00:00
Installing collected packages: pyparsing, pillow, packaging, kiwisolver, fonttools, cycler, contourpy, matplotlib
Successfully installed contourpy-1.3.1 cycler-0.12.1 fonttools-4.56.0 kiwisolver-1.4.8 matplotlib-3.10.1 packaging-24.2 pillow-11.1.0 pyparsing-3.2.3

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS D:\Python-practice\project-template>
```

Tests passing codecov 96% pypi v3.3.6 docs passing code style black linting pylint pre-commit.ci passed  
openssf best practices passing openssf scorecard 6.4 chat 43 online

## What is Pylint?

Pylint is a [static code analyser](#) for Python 2 or 3. The latest version supports Python 3.9.0 and above.

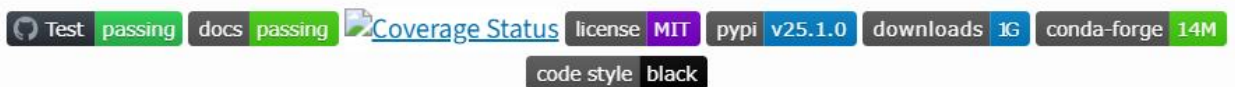
Pylint analyses your code without actually running it. It checks for errors, enforces a coding standard, looks for [code smells](#), and can make suggestions about how the code could be refactored.

```
Installing collected packages: tomlkit, platformdirs, mccabe, isort, dill, colorama, astroid, pylint
Successfully installed astroid-3.3.9 colorama-0.4.6 dill-0.3.9 isort-6.0.1 mccabe-0.7.0 platformdirs-4.3.7 pylint-3.3.6 tomlkit-0.13.2

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS D:\Python-practice\project-template>
```



## The Uncompromising Code Formatter



*"Any color you like."*

```
Successfully installed black-25.1.0 click-8.1.8 mpy-extensions-1.0.0 pathspec-0.12.1

[notice] A new release of pip is available: 23.2.1 -> 25.0.1
[notice] To update, run: python.exe -m pip install --upgrade pip
(.venv) PS D:\Python-practice\project-template>
```

6. Після цього утворимо список з усіма пакетами та їхніми версіями для зручнішої роботи у команді. Зазвичай це робиться через файл `requirements.txt` або `pipfile` при роботі з `pipenv`. Отже, якщо ви робите цю роботу з `pipenv`, вам необхідно додати до репозиторію `pipfile` та `pipfile.lock`, а при використанні `venv` – `requirements.txt`.

Для `venv`:

Для Windows:

```
python -m pip freeze
або
py -m pip freeze
```

```
(.venv) PS D:\Python-practice\project-template> python -m pip freeze
astroid==3.3.9
black==25.1.0
click==8.1.8
colorama==0.4.6
contourpy==1.3.1
cycller==0.12.1
dill==0.3.9
fonttools==4.56.0
isort==6.0.1
kiwisolver==1.4.8
matplotlib==3.10.1
mccabe==0.7.0
mypy-extensions==1.0.0
numpy==2.2.4
packaging==24.2
pandas==2.2.3
pathspec==0.12.1
pillow==11.1.0
platformdirs==4.3.7
pylint==3.3.6
pyparsing==3.2.3
python-dateutil==2.9.0.post0
pytz==2025.2
six==1.17.0
tomlkit==0.13.2
tzdata==2025.2
(.venv) PS D:\Python-practice\project-template>
```

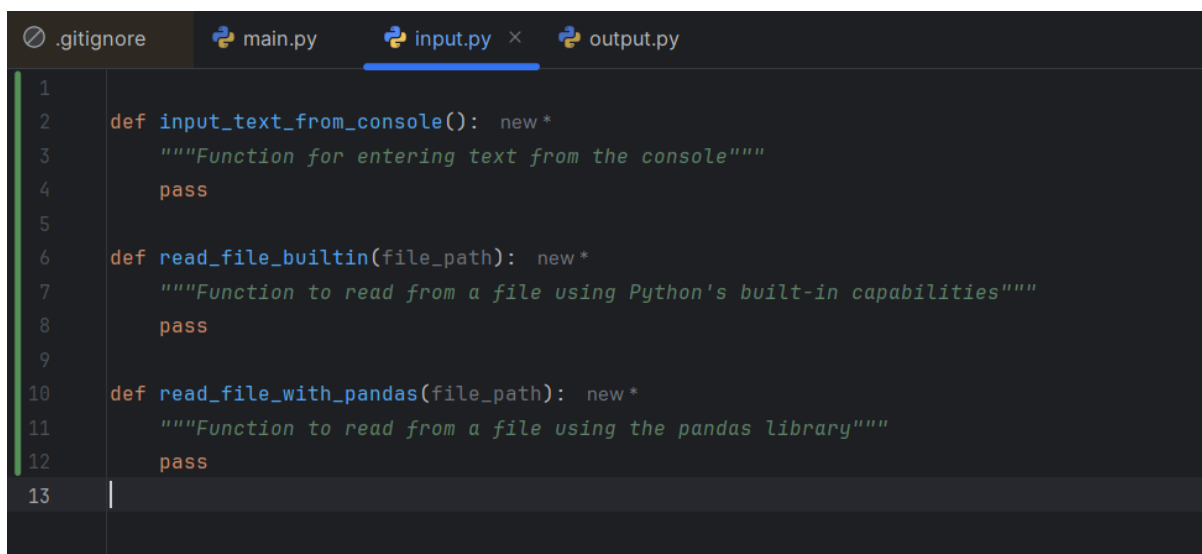
Тепер інші розробники, маючи цей файл можуть автоматично інсталювати всі ті самі пакети та версії за

допомогою команди `python -m pip install -r requirements.txt`

7. Зробіть commit з відповідним повідомленням.

#### 4. Робота з файлами.

1. У файлі `input.py` створіть пусті 3 функції: 1) для вводу тексту з консолі, 2) для зчитування з файлу за допомогою вбудованих можливостей python, 3) для зчитування з файлу за допомогою бібліотеки `pandas`.



```
.gitignore main.py input.py x output.py
1
2 def input_text_from_console(): new *
3 """Function for entering text from the console"""
4 pass
5
6 def read_file_builtin(file_path): new *
7 """Function to read from a file using Python's built-in capabilities"""
8 pass
9
10 def read_file_with_pandas(file_path): new *
11 """Function to read from a file using the pandas library"""
12 pass
13
```

2. У файлі `output.py` створіть пусті 3 функції: 1) для виводу тексту у консоль, 2) для запису до файлу за допомогою вбудованих можливостей python.



```
.gitignore main.py input.py output.py x
1
2 def output_text_to_console(text): new *
3 """Function for outputting text to the console"""
4 pass
5
6 def write_to_file_builtin(file_path, content): new *
7 """Function to write to a file using Python's built-in capabilities"""
8 pass
9
10 def write_to_file_with_pandas(file_path, content): new *
11 """Function to write to a file using the pandas library"""
12 pass
13
```

- Зробіть ще один commit з відповідним повідомленням на цьому кроці.
- Створіть docstrings для всіх цих функцій.
- У main.py у функції main() доповніть її тіло викликами виществорених функцій так, щоб текстові результати, що повертаються функціями 4.1.1) , 4.1.2) та 4.1.3) були виведені у консоль, а також записані до файлу через вбудовані можливості python.
- Реалізуйте ці функції.

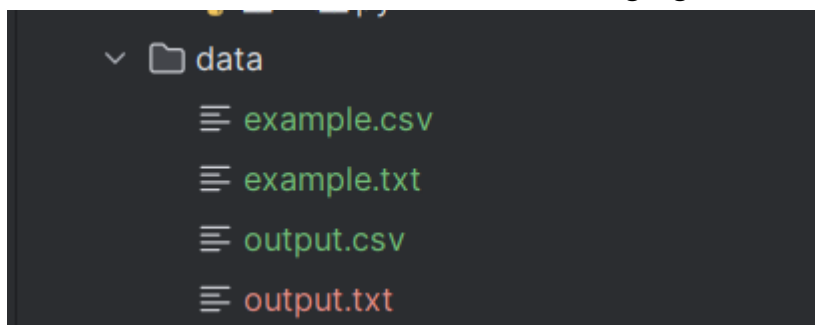
```
.gitignore main.py input.py output.py
1 import pandas as pd
2
3 def input_text_from_console(): 2 usages Yelyzaveta Petrenko *
4 """Function for entering text from the console"""
5 return input("Input text: ")
6
7 def read_file_builtin(file_path): 2 usages Yelyzaveta Petrenko *
8 """Function to read from a file using Python's built-in capabilities"""
9 with open(file_path, 'r') as file:
10 return file.read()
11
12 def read_file_with_pandas(file_path): 2 usages Yelyzaveta Petrenko *
13 """Function to read from a file using the pandas library"""
14 return pd.read_csv(file_path)
15
```

```
.gitignore main.py input.py output.py
1 import pandas as pd
2
3 def output_text_to_console(text): 4 usages Yelyzaveta Petrenko *
4 """Function for outputting text to the console"""
5 print(text)
6
7 def write_to_file_builtin(file_path, content): 2 usages Yelyzaveta Petrenko *
8 """Function to write to a file using Python's built-in capabilities"""
9 with open(file_path, 'w') as file:
10 file.write(content)
11
12 def write_to_file_with_pandas(file_path, content): 2 usages Yelyzaveta Petrenko *
13 """Function to write to a file using the pandas library"""
14 content.to_csv(file_path, index=False)
15
```

```
.gitignore main.py _init_.py example.csv output.csv example.txt input.py output.py

1 from app.io.input import input_text_from_console, read_file_builtin, read_file_with_pandas
2 from app.io.output import output_text_to_console, write_to_file_builtin, write_to_file_with_pandas
3
4
5 def main(): 1 usage Yelyzaveta Petrenko *
6
7 text = input_text_from_console()
8 file_data_builtin = read_file_builtin("data/example.txt")
9 file_data_pandas = read_file_with_pandas("data/example.csv")
10
11 #Output
12 output_text_to_console(text)
13 output_text_to_console(file_data_builtin)
14 output_text_to_console(file_data_pandas)
15
16
17 write_to_file_builtin(file_path: "data/output.txt", text)
18 write_to_file_with_pandas(file_path: "data/output.csv", file_data_pandas)
19
20
21 if __name__ == "__main__":
22 main()
23
```

7. За потреби, ви можете створити окрему папку для даних (файлів) у кореневій папці проєкту з назвою data. Обов'язково додайте її до .gitignore.



```
D:\Python-practice\project-template\.venv\Scripts\python.exe D:\Python-practice\project-template\main.py
Input text: python
python
abcdefghijkl
1234567890
 Name Age
0 Alice 30
1 Bob 25
2 Charlie 22

Process finished with exit code 0
```

8. Зробіть commit з відповідним повідомленням.



## 5. \*(На оцінку 90+). Написання тестів.

Використовуючи пакети unittest або pytest на ваш вибір, напишіть по три тести до функцій 2 та 3 (зчитування з файлів) з файлу input.py.

```
main.py test_input.py × mocked_empty_file.csv mocked_file.csv mocked_empty_f

1 import unittest
2 from unittest.mock import mock_open, patch
3 import pandas as pd
4 from app.io.input import read_file_builtint, read_file_with_pandas
5
6 class TestFileReading(unittest.TestCase): new *
7
8 # Tests for the second function read_file_builtint
9 @patch("builtins.open", mock_open(read_data="Hello, world!")) new *
10 def test_read_file_builtint_success(self):
11 """Test for successful reading from a file"""
12 result = read_file_builtint("data/mock_file.txt")
13 self.assertEqual(result, second: "Hello, world!")
14
15 @patch("builtins.open", mock_open(read_data="")) new *
16 def test_read_file_builtint_empty(self):
17 """Test reading from an empty file"""
18 result = read_file_builtint("data/mock_empty_file.txt")
19 self.assertEqual(result, second: "")
20
21 @patch(target: "builtins.open", side_effect=FileNotFoundError) new *
22 def test_read_file_builtint_file_not_found(self, mock_file):
23 """Test for trying to read a file that doesn't exist"""
24 with self.assertRaises(FileNotFoundError):
25 read_file_builtint("data/non_existent_file.txt")
```

```
27 # Tests for the third function read_file_with_pandas
28 @patch(target="pandas.read_csv", return_value=pd.DataFrame({"Name": ["Alice", "Bob"], "Age": [25, 30]})) new *
29 def test_read_file_with_pandas_success(self, mock_read_csv):
30 """Test for successful reading of CSV file with pandas"""
31 result = read_file_with_pandas("data/mocked_file.csv")
32 expected_result = pd.DataFrame({"Name": ["Alice", "Bob"], "Age": [25, 30]})
33 pd.testing.assert_frame_equal(result, expected_result)
34
35 @patch(target="pandas.read_csv", return_value=pd.DataFrame()) new *
36 def test_read_file_with_pandas_empty(self, mock_read_csv):
37 """Test reading an empty CSV file"""
38 result = read_file_with_pandas("data/mocked_empty_file.csv")
39 expected_result = pd.DataFrame()
40 pd.testing.assert_frame_equal(result, expected_result)
41
42 @patch(target="pandas.read_csv", side_effect=FileNotFoundError) new *
43 def test_read_file_with_pandas_file_not_found(self, mock_file):
44 """Test for trying to read a non-existent CSV file"""
45 with self.assertRaises(FileNotFoundError):
46 read_file_with_pandas("data/non_existent_file.csv")
47
✓ Tests passed: 6 of 6 tests – 6 ms

D:\Python-practice\project-template\.venv\Scripts\python.exe "C:/Program Files/JetBrains/PyCharm Community Edition 2024.3.3/plugins/python-
Testing started at 21:32 ...
Launching unittests with arguments python -m unittest D:\Python-practice\project-template\tests\data\test_input.py in D:\Python-practice\pr

Ran 6 tests in 0.011s

OK

Process finished with exit code 0
```

Після написання тестів для кожної окремої функції дуже рекомендую робити commit.

Ресурси, які можуть вам бути корисні:

<https://docs.python.org/3/library/unittest.html>

<https://docs.pytest.org/en/7.4.x/getting-started.html>

<https://realpython.com/python-testing/> <https://www.dataquest.io/blog/unit-tests-python/>

## 6. Висновки.

а. Що зробили?

- Я створила функції для зчитування і запису даних у файли, використовуючи вбудовані можливості Python і бібліотеку pandas.
- Написала тести для перевірки цих функцій за допомогою пакета unittest. Тести перевіряли коректність роботи функцій, зокрема зчитування файлів та обробку помилок.
- Ініціювала роботу з тестами, налаштувала запуск тестів за допомогою команд unittest і використовуючи моки для симуляції роботи з файлами.
- Попрацювала з Git, зробила коміти, додала файли у .gitignore.

**b. Що нового дізнались для себе?**

Поглибила знання Python, включаючи різні способи зчитування та запису даних:

- Як зчитувати і записувати дані за допомогою стандартних можливостей Python.
- Як використовувати бібліотеку pandas для роботи з CSV файлами.

Робота з Git:

- Як правильно організувати структуру .gitignore.
- Як підтримувати чистоту репозиторію, виключаючи непотрібні файли.

**c. Що було корисним? Що б Ви використали в майбутньому?**

- Мокування у тестуванні
- Робота з pandas для обробки даних
- Використання unittest для тестування
- Використання Git

**d. Що можна було б покращити нам для студентів в цій роботі?**

**d. Що можна було б покращити нам для студентів в цій роботі?**

У процесі інсталювання передостанньої версії бібліотеки pandas в мене виникла помилка суміщення версій з python. Тому я інсталювала останню версію. Можливо, для практики встановлення якоїсь конкретної версії потрібно використовувати якусь іншу бібліотеку.

Також, на першому етапі виконання завдання пояснення щодо використання рірепν та венν було не зрозумілим. Думаючи, що зробивши проєкт з венν, отримаю більше балів, почала робити з ним. Під кінець практичної вже дізналась, що навпаки - більше балів можна отримати з використанням рірепν.

### **Надсилання звіту.**

- а. Готовий звіт прикріпити у Мудл згідно дедлайнів.

### **7. Наостанок.**

***Похваліть себе, Ви дуже багато зусиль доклали! Побалуйте себе відпочинком або якимось смаколиком.***

***Дякую, що доклали зусиль, у Вас вийшло!***

