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

Імʼя, Прізвище, група.

Посилання на новий репозиторій.

https://github.com/Danylkko/template\_project

1. 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.**

* + 1. 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 pipenv самостійно на оцінку 80-90**), main.py створимо пізніше. Назву папки віртуального середовища запамʼятайте, ми використаємо її пізніше.

**\*для роботи з pipenv при створенні проєкту**

A screenshot of a computer

Description automatically generated

* + 1. Підготуйте файл .gitignore, щоб папки типу venv або .idea

і.т.п. не потрапили до репозиторію, який призначений суто для коду проєкту.

* + 1. Створіть файл main.py у директорії проєкту, який матиме наступний вигляд:

A screenshot of a computer

Description automatically generated

* + 1. Створіть також новий репозиторій на GitHub (теж підглянути, як це робиться, можете у ПЗ 1).
    2. Об’єднайте локальний та віддалений репозиторії. Залийте зміни на віддалений репозиторій (тут теж можете згадати ПЗ 1). Посилання на нього додайте на початок звіту.

* 1. **Структура проєкту.** 
     1. 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).*

* + 1. Усередині цієї директорії app створити Python Package «io» (скорочено input-output).
    2. У цій директорії io створити два файли: input.py та output.py.
    3. 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.).*
    4. Залити зміни на віддалений репозиторій з відповідним повідомленням у коміті.

* 1. **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

Для Unix/MacOS:

which python

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



1. 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

Для Unix/MacOS:

source nazva\_venv/bin/activate

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

1. Підготуйте pip. Для Windows:

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

Для Unix/MacOS:

python3 -m pip install --upgrade pip python3 -m pip --version

Після цього маєте побачити свіжу версію менеджеру пакетів pip.

1. Встановлюємо пакети через pip.

**Якщо ви працюєте з pipenv, після прочитання цієї статті** [**https://realpython.com/pipenv-guide/**](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](https://pypi.org/) та в пошуку знайти пакет numpy.

A screenshot of a computer

Description automatically generated

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

A blue screen with white text

Description automatically generated

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

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

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

Для Windows:

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

py -m pip install "SomeProject==1.4"

Для Unix/MacOS:

python3 -m pip install "SomeProject==1.4"

де SomeProject – назва бібліотеки для інсталювання, == це визначення для того, яка конкретна версія потрібна і 1.4 – це цифри, що відповідають номерам версії для встановлення. Наприклад,

A screenshot of a computer

Description automatically generated

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/](https://packaging.python.org/en/latest/guides/installing-using-pip-and-virtual-environments/)

1. Тепер пропоную вам встановити самостійно додатково пакети matplotlib та pylint, black.
2. 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 |

Для Unix/MacOS:

python3 -m pip freeze

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

Більше про цей файл та випадки використання можна прочитати тут: <https://pip.pypa.io/en/latest/user_guide/#requirements-files>7. Зробіть commit з відповідним повідомленням.

1. **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.
   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.
   3. Зробіть ще один commit з відповідним повідомленням на цьому кроці.
   4. Створіть docstrings для всіх цих функцій.
   5. 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.
   6. Реалізуйте ці функції.
   7. За потреби, ви можете створити окрему папку для даних (файлів) у кореневій папці проєкту з назвою data. Обовʼязково додайте її до .gitignore. 8. Зробіть commit з відповідним повідомленням.

1. **\*(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. 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/>

1. **Висновки.**

* + 1. Що зробили?

Створив Python Package, додав залежності через pip, вказав версію пакетів pip у requirements.txt. Імплементував кілька io функцій.

* + 1. Що нового дізнались для себе?

Дізнався як виводити текст у файл використовуючи python.

* + 1. Що було корисним? Що б Ви використали в майбутньому?

Як вказувати версії залежностей в python через requirements.txt

* + 1. Що можна було б покращити нам для студентів в цій роботі?

Все чудово!🤘 Окрім того, що ЗГІДНО З правилами української мови, прислівники ЗГІДНО З та ВІДПОВІДНО ДО пишуться так.

1. **Надсилання звіту.**

a. Готовий звіт прикріпити у Мудл згідно дедлайнів.

1. **Наостанок.**

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

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

