Лабораторна Робота 7 – Модулі, пакети у Python.

Мета роботи – вивчити та засвоїти базові навички організації проектів у Python за допомогою модулів та пакетів. Дізнатися про особливості та способи підключення модулів та організації пакетів.

1.1 Базові поняття про модулі

Модулі виконують як мінімум три важливі функції:

- Повторне використання коду: такий код може бути завантажений багато разів у багатьох місцях.
- Управління адресним простором: модуль - це високорівнева організація програм, це пакет імен, який позбавляє вас від конфліктів. Кожен об'єкт «проживає» свій цикл всередині свого модуля, тому модуль - це засіб для угруповання системних компонентів.
- Глобалізація сервісів і даних: для реалізації об'єкта, який використовується в багатьох місцях, досить написати один модуль, який слід імпортувати.

Python дозволяє помістити класи, функції або дані в окремий файл і використовувати їх в інших програмах. Такий файл називається модулем. Об'єкти з модуля можуть бути імпортовані в інші модулі. Файл утворюється шляхом додавання до імені модуля розширення .py. При імпорті модуля інтерпретатор шукає файл з ім'ям my_module.py спочатку в поточному каталозі, потім в каталогах, зазначених у змінній оточення PYTHONPATH, потім в залежних від платформи шляхах за замовчуванням, а також в спеціальних файлах з розширенням '.pth', які лежать в стандартних каталогах. Програміст може внести зміни в PYTHONPATH і в '.pth', додавши туди свій шлях. Каталоги, в яких здійснюється пошук, можна подивитися в змінної sys.path.

Великі програми, як правило, складаються з стартового файлу - файлу верхнього рівня, і набору файлів-модулів. Головний файл займається контролем програми. У той же час модуль - це не тільки фізичний файл. Модуль являє

собою колекцію компонентів. У цьому сенсі модуль - це простір імен, - namespace, і всі імена всередині модуля ще називаються атрибутами - такими, наприклад, як функції і змінні.

1.2 Імпорт модулів

Якщо запустити в каталозі, в якому лежить даний модуль (наприклад, my_module.py), інтерпретатор:

```
>>> python
```

і потім зробити імпорт модуля:

```
>>> import my_module
```

то ми отримуємо доступ до всіх функцій, які в модулі визначені:

```
>>> my_module.func1()
>>> my_module.func2()
...
```

Для більш короткого запису можна створити локальну змінну:

```
>>> f1 = my_module.func1
```

Другий варіант імпорту - взяття безпосередньо імені без імені модуля:

```
>>> from my_module import func1, func2
>>> func1()
```

Третій варіант імпорту - включення всіх імен, визначених в модулі:

```
>>> from my_module import *
>>> func1()
```

Приклад. Імпорт на основі from має таку особливість, що він робить імпортовані атрибути read-only:

```
>>> from small import x, y
>>> x = 42
```

В даному випадку x - це локальна змінна, в той час як змінні x, y в самому модулі small не змінюються:

```
>>> import small
>>> small.x = 42
```

тут **x** - глобальна змінна.

Щоб уникнути непорозумінь import краще без from в тих випадках, коли один і той же модуль використовується в декількох місцях. Оскільки модуль завантажується один раз, для його повторного завантаження можна використовувати функцію reload ().

Кожен модуль має власний простір імен, що є глобальною областю видимості для всіх визначених у ньому функцій. Для того щоб змінні цього модуля не потрапили в конфлікт з іншими глобальними іменами або іншими модулями, потрібно використовувати префікс: _ім'я_модуля _._ ім'я_змінної.

Модулі можуть імпортувати інші модулі. Зазвичай інструкцію import розташовують на початку модуля або програми.

1.3 Створення власного модуля

Створимо файл mymodule.py, в якій визначимо якісь функції:

```
def hello():
    print('Hello, world!')

def fib(n):
    a = b = 1
    for i in range(n - 2):
        a, b = b, a + b
    return b
```

Тепер в цій же папці створимо інший файл, та підключимо у ньому створений модуль.

```
[2]: import My_module
     My_module.hello()
     print(My_module.fib(10))

     Hello, world!
     55
```

1.4 Пакети

Пакети - спосіб структурування просторів імен модулів на основі файлової системи. Пакетна організація дає всі зручності з управління великою кількістю файлів. Пакетний імпорт робить код більш читальним і значно спрощує пошук. Якщо весь код структурований в одному рутовому каталозі, все, що потрібно додати в PYTHONPATH - це рутовий каталог.

Так само, як застосування модулів робить безпечним використання глобального простору імен авторами різних модулів, застосування пакетів робить безпечним використання імен модулів авторами багатомодульних пакетів.

Наприклад, є пакет, який лежить в кореневій папці TCP. У ньому лежать два підкаталогу - Server і Client:

```
TCP/
    _init_.py
    main.py

    Server/
          _init_.py
          tcp.py
          server.py
          lib.py
    Client/
          _init_.py
          tcp.py
          client.py
          lib.py
```

Файл _init_.py необхідний для того, щоб інтерпретатор розпізнав каталог, як що містить пакет. Зазвичай це порожній файл. Тоді імпорт індивідуальних модулів пакета може бути таким:

```
>>> import TCP.Server.lib
>>> import TCP.Client.lib
```

Посилання на функцію повинно бути повним:

```
>>> import TCP.Server.lib.connect()
```

Можна зробити альтернативне завантаження:

```
>>> from TCP.Server import lib as server_lib
>>> from TCP.Client import lib as client_lib
>>> server_lib.connect()
>>> client_lib.connect()
```

Тут замість lib може бути підставлений модуль, підпакету або ім'я, визначене в TCP.Server - тобто це може бути функція, клас чи змінна.

Що стосується варіанту з імпортом:

```
>>> from TCP import *
```

то в кореневому __init__.py може бути визначений список __all__, в якому перераховуються модулі, які імпортуються в цьому випадку. наприклад:

```
__all__ = ["Server", "Client"]
```

2. Варіанти завдання

У даній лабораторній роботі необхідно виконати розбиття функцій з лабораторної роботи №2 на модулі з наступним об'єднанням отриманих модулів в один загальний пакет. Пакет необхідно підключити в окремому файлі програми і продемонструвати роботу з кожним модулем пакета.

Таблиця 1.2 – Варіанти завдань

№	Завдання
1.	Створити функцію для вирішення наступного завдання: Масив X=(x1,x2,...,xn) містить велику кількість нульових елементів. Визначити положення і розмір найбільш довгої серії таких елементів.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 3x * y^2 - 9z$
2.	Створити функцію для вирішення наступного завдання: Заданий масив X=(x1,x2,...,xn), в якому можуть бути однакові числа. Знайти максимальний і мінімальний елементи серед неповторюваних чисел.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 5x * y^3 + 7z$
3.	Створити функцію для вирішення наступного завдання: З масиву чисел X=(x1,x2,...,xn) вилучити всі парні за значенням елементи.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 8x * y^2 - 9z$
4.	Створити функцію для вирішення наступного завдання: У масиві X=(x1,x2,...,xn) поміняти місцями перший і другий негативні елементи, третій і четвертий негативні елементи тощо.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = \frac{3x}{y^4} - 2z$
5.	Створити функцію для вирішення наступного завдання: Елементи масиву X = (x1,x2,...,xn) – це послідовність цифр цілого числа. Переставити цифри числа у зворотному порядку
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = x - y^2 * 4z$

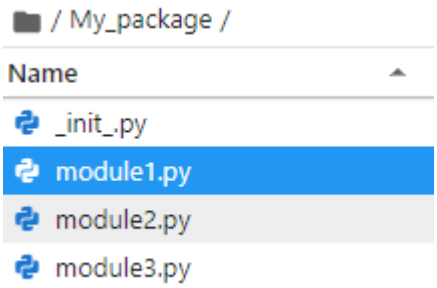
№	Завдання
6.	Створити функцію для вирішення наступного завдання: Відомо, що в целочисельному масиві X=(x1,x2,...,xn) три і тільки три числа, що є рівними між собою. Знайти ці числа
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 2x + y^2 - 5z$
7.	Створити функцію для вирішення наступного завдання: За однократний перегляд масиву знайти його максимальний позитивний елемент X_{max}
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 7x * y^3 * 2z$
8.	Створити функцію для вирішення наступного завдання: Перетворити масив X, розташувавши спочатку його негативні, а потім позитивні елементи, зберігши при цьому в групі негативних та позитивні елементів їх вихідний відносний порядок.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $f = 11x - y^{-2} + 6z$
9.	Створити функцію для вирішення наступного завдання: У масиві X=(x1,x2,...,xn) поміняти місцями перший і другий позитивні елементи, третій і четвертий позитивні елементи тощо.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 6x + y^3 * 8z$
10.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі позитивні числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = \frac{2x}{y^1} + 5z$
11.	Створити функцію для вирішення наступного завдання: Заданий цілочисельний масив X=(x1,x2,...,xn), у якому можуть бути однакові числа. Підрахувати кількість повторюваних чисел у масиві.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $f = 5x + \frac{y^2}{7z}$

№	Завдання
12.	Створити функцію для вирішення наступного завдання: Виконати циклічне зрушення масиву X=(x1,x2,...,xn) на 5 елементів вліво.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $$f = 4x * y^2 - \frac{9}{z}$$
13.	Створити функцію для вирішення наступного завдання: Виконати циклічне зрушення масиву X=(x1,x2,...,xn) на 3 елементів вправо.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 11x * \frac{y^3}{3} + 8z$$
14.	Створити функцію для вирішення наступного завдання: Масив X=(x1,x2,...,xn) містить велику кількість нульових елементів. Визначити положення і розмір найбільш довгої серії ненульових елементів.
	Створити функцію сортування в порядку убування.
	Створити анонімну функцію для обчислення функції: $$f = 2xz * \frac{y^2}{2}$$
15.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі негативні числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 2(x - y) * \frac{z^2}{4}$$
16.	Створити функцію для вирішення наступного завдання: Заданий масив цілих чисел X=(x1,x2,...,xn). Сформувати масив Y=(y1,y2,...,ym), помістивши в нього в порядку убування всі різні (неповторювані) числа, що входять у масив X.
	Створити функцію сортування масиву за зростанням.
	Створити анонімну функцію для обчислення функції: $$f = 2(x - 4z) * \frac{y^{-2}}{5}$$

3. Приклад

Варіант -16.

Структура пакету



Вміст файлу _init.py_

```
1  __all__ = ["module1","module2","module3"]
```

Головний файл програми

```
[20]: from My_package import *
      mas=[1,2,2,4,5,8,4,3,18,11,11] # вхідний масив
      x=5
      y=6
      z=3
      print(module1.func1(mas))
      print(module2.func2(x,y,z))
      print(module3.func3(mas))
```

```
[18, 8, 5, 3, 1]
-0.07777777777777777
[1, 2, 2, 3, 4, 4, 5, 8, 11, 11, 18]
```

4. Контрольні запитання

1. Стандартні модулі у Python, їх види та призначення.
2. Які існують способи звернення до модулів, наведіть приклади.
3. Опишіть процедуру створення власного модуля.
4. Опишіть механізм об’єднання модулів в пакети, сенс, принципи, призначення.
5. Файл _init.py_, для чого створюється та як використовується? Наведіть приклади.