Міністерство освіти і науки України

Національний університет “Львівська політехніка”

Кафедра ЕОМ


# Звіт

З лабораторної роботи №9

Варіант – 9

З дисципліни: «Кросплатформні засоби програмування»

На тему: «ОСНОВИ ОБ’ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ У PYTHON »

Виконав: ст. гр. КІ-305

Заставний Р.А.

Прийняв:

доцент кафедри ЕОМ

Іванов Ю. С.

Львів-2023

**Мета роботи:** оволодіти навиками реалізації парадигм об’єктно-орієнтованого програмування використовуючи засоби мови Python

## ЗАВДАННЯ

1. Написати та налагодити програму на мові Python згідно варіанту. Програма має задовольняти наступним вимогам:
   - класи програми мають розміщуватися в окремих модулях в одному пакеті;
   - точка входу в програму (main) має бути в окремому модулі;
   - мають бути реалізовані базовий і похідний класи предметної області згідно варіанту;
   - програма має містити коментарі.
2. Завантажити код на GitHub згідно методичних вказівок по роботі з GitHub.
3. Скласти звіт про виконану роботу з приведенням тексту програми, результату її виконання та фрагменту згенерованої документації та завантажити його у ВНС.
4. Дати відповідь на контрольні запитання.

## Варіант завдання:

**Базові класи** : 9. Рослина

**Похідні класи** : 9. Дерево

## Код програми:

### Main.py

```
from Plant import Plant

class Tree(Plant):
    def __init__(self, tree_type="Немає"):
        """
        Конструктор класу Tree.

        :param tree_type: Тип дерева, за замовчуванням "Немає".
        """
        super().__init__()
        self.type = tree_type

    def get_type(self):
        """
        Отримати тип дерева.

        :return: Тип дерева.
        """
        print(self.type)
        return self.type

    def get_status(self):
        """
        Отримати статус дерева та вивести інформацію.

        :return: None
        """
        super().get_status()
        print("Тип дерева: " + self.type)

try:
    # Створення та робота з об'єктами Tree
```

```
    tree1 = Tree()
    tree1.set_name("Ялина")
    tree1.set_color("Зелений")
    tree1.set_count_of_leaves(100)
    tree1.set_humidity(70.0)
    tree1.grow_up()
    tree1.get_status()
    tree1.photosynthesis(10.0)
    tree1.death()

    tree2 = Tree("Хвойний")
    tree2.get_status()

except Exception as e:
    print("Помилка:", e)
```

## Fruit.py

```
class Fruit:
    def __init__(self, existence=False, color="Немає", ripeness="Немає"):
        """
        Конструктор класу Fruit.

        Ініціалізує атрибути фрукта рослини.

        Args:
        - existence (bool): Наявність фруктів.
        - color (str): Колір фруктів.
        - ripeness (str): Стиглість фруктів.

        """
        self.existence = existence
        self.color = color
        self.ripeness = ripeness

    def is_existence(self):
        """
        Отримання інформації про наявність фруктів.

        Returns:
        - bool: Наявність фруктів.

        """
        return self.existence

    def set_existence(self, existence):
        """
        Встановлення наявності фруктів.

        Args:
        - existence (bool): Наявність фруктів.

        """
        self.existence = existence

    def get_color(self):
        """
        Отримання кольору фруктів.

        Returns:
        - str: Колір фруктів.

        """
```

```
        return self.color

    def set_color(self, color):
        """
        Встановлення кольору фруктів.

        Args:
        - color (str): Колір фруктів.

        """
        self.color = color

    def get_ripeness(self):
        """
        Отримання стиглості фруктів.

        Returns:
        - str: Стиглість фруктів.

        """
        return self.ripeness

    def set_ripeness(self, ripeness):
        """
        Встановлення стиглості фруктів.

        Args:
        - ripeness (str): Стиглість фруктів.

        """
        self.ripeness = ripeness

    def get_status(self, fout):
        """
        Виведення статусу фруктів рослини.

        Args:
        - fout (File): Файловий об'єкт для запису.

        """
        if self.existence:
            print("Плоди: Є")
            print("Колір: " + self.color)
            print("Стиглість: " + self.ripeness)

            fout.write("Плоди: Є\n")
            fout.write("Колір: " + self.color + "\n")
            fout.write("Стиглість: " + self.ripeness + "\n")
        else:
            print("У рослини немає плодів.")

            fout.write("У рослини немає плодів.\n")
```

## Leaf.py

```
class Leaf:
    def __init__(self, shape="Немає", state=False, color="Немає"):
        """
        Конструктор класу Leaf.

        Ініціалізує атрибути листка рослини.
```

```
    Args:
    - shape (str): Форма листка.
    - state (bool): Стан листка (присутній/відсутній).
    - color (str): Колір листка.

    """
    self.shape = shape
    self.state = state
    self.color = color

def get_shape(self):
    """
    Отримання форми листка.

    Returns:
    - str: Форма листка.

    """
    return self.shape

def set_shape(self, shape):
    """
    Встановлення форми листка.

    Args:
    - shape (str): Форма листка.

    """
    self.shape = shape

def get_state(self):
    """
    Отримання стану листка.

    Returns:
    - bool: Стан листка (присутній/відсутній).

    """
    return self.state

def set_state(self, state):
    """
    Встановлення стану листка.

    Args:
    - state (bool): Стан листка (присутній/відсутній).

    """
    self.state = state

def get_color(self):
    """
    Отримання коліру листка.

    Returns:
    - str: Колір листка.

    """
    return self.color

def set_color(self, color):
    """
    Встановлення коліру листка.
```

```
        Args:
        - color (str): Колір листка.

        """
        self.color = color

    def get_status(self, fout):
        """
        Виведення статусу листка.

        Args:
        - fout (File): Файловий об'єкт для запису.

        """
        if self.state:
            print("Листя: Є")
            fout.write("Листя: Є\n")
        else:
            print("Листя: Немає")
            fout.write("Листя: Немає\n")

        print("Форма:", self.shape)
        print("Колір:", self.color)

        fout.write("Форма: " + self.shape + "\n")
        fout.write("Колір: " + self.color + "\n")
```

## Plant.py

```
import os
from Leaf import Leaf
from Size import Size
from Fruit import Fruit

class Plant:
    def __init__(self):
        """
        Конструктор класу Plant.

        Ініціалізує атрибути рослини.

        Attributes:
        - name (str): Ім'я рослини.
        - size (Size): Об'єкт класу Size, представляє розмір рослини.
        - color (str): Колір рослини.
        - fruit (Fruit): Об'єкт класу Fruit, представляє фрукти рослини.
        - leaf (Leaf): Об'єкт класу Leaf, представляє листя рослини.
        - humidity (float): Вологість рослини.
        - count_of_leaves (int): Кількість листків рослини.
        - fout (File): Файловий об'єкт для ведення логу.

        """
        self.name = "Немає"
        self.size = Size()
        self.color = "Немає"
        self.fruit = Fruit()
        self.leaf = Leaf()
        self.humidity = 0
        self.count_of_leaves = 0
        self.fout = open("./Log.txt", "a")

    def __del__(self):
```

```
        """
        Деструктор класу Plant.

        Закриває відкритий файл і завершує запис у файл логу.

        """
        if hasattr(self, 'fout') and self.fout:
            self.finish()

    def finish(self):
        """
        Завершення запису у файл логу.

        Закриває файл, якщо він відкритий.

        """
        if hasattr(self, 'fout') and self.fout:
            self.fout.close()

    def grow_up(self):
        """
        Збільшення розміру рослини та кількості листя при відповідних умовах.

        Raises:
        - ValueError: Якщо вологість рослини менше або дорівнює 0.

        """
        if self.humidity <= 0:
            raise ValueError("Рослина не може рости: недостатньо вологості.")
        else:
            self.size.change_size_to_grow()
            self.set_count_of_leaves(self.count_of_leaves + 1)

    def set_count_of_leaves(self, count_of_leaves):
        """
        Встановлення кількості листя рослини.

        Args:
        - count_of_leaves (int): Кількість листя.

        Raises:
        - ValueError: Якщо передано від'ємне значення.

        """
        if count_of_leaves < 0:
            raise ValueError("Вказано неправильне значення кількості листків")
        elif count_of_leaves == 0:
            self.count_of_leaves = count_of_leaves
            self.leaf.state = True
        else:
            self.count_of_leaves = count_of_leaves

    def get_status(self):
        """
        Виведення статусу рослини.

        Виводить інформацію про ім'я, колір, вологість, розмір, кількість листя, листя та
фрукти.

        """
        print("Рослина:", self.name)
        print("Колір:", self.color)
        print("Вологість:", self.humidity)
```

```
        self.fout.write("Рослина: " + self.name + "\n")
        self.fout.write("Колір: " + self.color + "\n")
        self.fout.write("Вологість: " + str(self.humidity) + "\n")

        self.size.get_status(self.fout)

        print("Кількість листків:", self.count_of_leaves)

        self.fout.write("Кількість листків: " + str(self.count_of_leaves) + "\n")

        self.leaf.get_status(self.fout)
        self.fruit.get_status(self.fout)
        print()
        self.fout.write("\n")

    def photosynthesis(self, count_light):
        """
        Виконання процесу фотосинтезу рослини.

        Args:
        - count_light (float): Кількість світла.

        Raises:
        - ValueError: Якщо передано від'ємне або нульове значення кількості світла.

        """
        if count_light <= 0:
            raise ValueError("Недостатньо світла для фотосинтезу.")
        elif self.humidity < count_light * 0.1 or self.count_of_leaves <= 0:
            raise ValueError("Недостатньо вологості або листя для фотосинтезу.")
        else:
            self.humidity -= count_light * 0.1
            print("Сфотосинтезовано", count_light * 1.2, "кисню.")

    def death(self):
        """
        Процес вмирання рослини.

        Змінює розмір рослини, колір, кількість листя та вологість.

        """
        self.size.change_size_to_death()
        self.set_color("Немає")
        self.set_count_of_leaves(0)
        self.set_humidity(0)
        self.finish()

    def set_name(self, name):
        """
        Встановлення ім'я рослини.

        Args:
        - name (str): Ім'я рослини.

        """
        self.name = name

    def set_color(self, color):
        """
        Встановлення коліру рослини.

        Args:
        - color (str): Колір рослини.
```

```
        """
        self.color = color

    def set_humidity(self, humidity):
        """
        Встановлення вологості рослини.

        Args:
        - humidity (float): Вологість рослини.

        Raises:
        - ValueError: Якщо передано від'ємне значення вологості.

        """
        if humidity < 0:
            raise ValueError("Вологість не може бути від'ємною.")
        self.humidity = humidity

    def set_fruit(self, fruit):
        """
        Встановлення об'єкта фруктів для рослини.

        Args:
        - fruit (Fruit): Об'єкт класу Fruit.

        """
        self.fruit = fruit

    def set_leaf(self, leaf):
        """
        Встановлення об'єкта листя для рослини.

        Args:
        - leaf (Leaf): Об'єкт класу Leaf.

        """
        self.leaf = leaf

    def set_size(self, size):
        """
        Встановлення об'єкта розміру для рослини.

        Args:
        - size (Size): Об'єкт класу Size.

        """
        self.size = size

    def set_count_of_leaves(self, count_of_leaves):
        """
        Встановлення кількості листя для рослини.

        Args:
        - count_of_leaves (int): Кількість листя.

        """
        self.count_of_leaves = count_of_leaves

    def set_fout(self, fout):
        """
        Встановлення файлового об'єкта для запису у файл логу.

        Args:
        - fout (File): Файловий об'єкт.
```

```
        """
        self.fout = fout


# # Приклад використання:
# try:
#     plant1 = Plant()
#     plant1.set_name("Соняшник")
#     plant1.set_color("Жовтий")
#     plant1.set_count_of_leaves(10)
#     plant1.set_humidity(50.0)
#     plant1.set_leaf(Leaf("n","true","m"))
#     plant1.grow_up()
#     plant1.get_status()
#     plant1.photosynthesis(5.0)
#     plant1.death()
#
# except Exception as e:
#     print("Помилка:", e)
```

## Size.py

```
class Size:
    def __init__(self, width=0, height=0, length=0):
        """
        Конструктор класу Size.

        Ініціалізує атрибути розміру рослини.

        Args:
        - width (float): Ширина рослини.
        - height (float): Висота рослини.
        - length (float): Довжина рослини.

        """
        self.width = width
        self.height = height
        self.length = length

    def get_width(self):
        """
        Отримання ширини рослини.

        Returns:
        - float: Ширина рослини.

        """
        return self.width

    def set_width(self, width):
        """
        Встановлення ширини рослини.

        Args:
        - width (float): Ширина рослини.

        """
        self.width = width

    def get_height(self):
        """
```

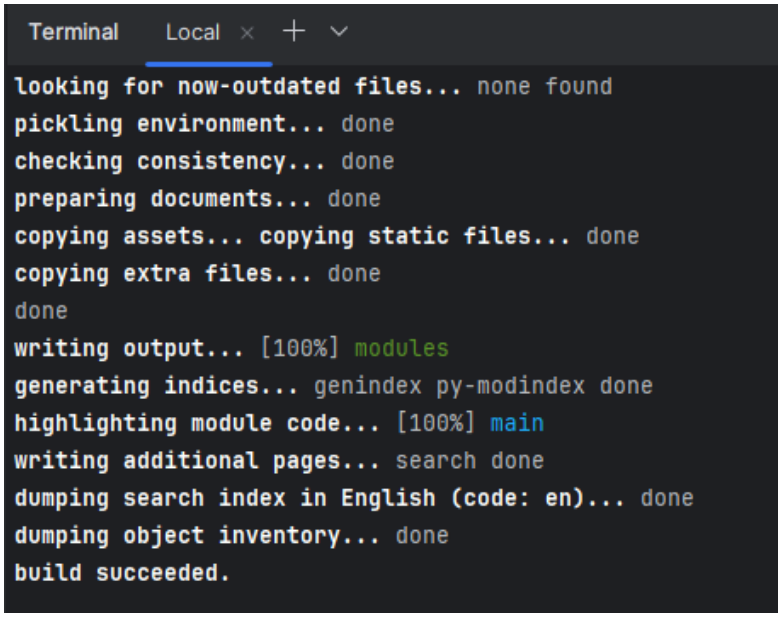
```
    Отримання висоти рослини.

    Returns:
    - float: Висота рослини.

    """
    return self.height

def set_height(self, height):
    """
    Встановлення висоти рослини.

    Args:
    - height (float): Висота рослини.

    """
    self.height = height

def get_length(self):
    """
    Отримання довжини рослини.

    Returns:
    - float: Довжина рослини.

    """
    return self.length

def set_length(self, length):
    """
    Встановлення довжини рослини.

    Args:
    - length (float): Довжина рослини.

    """
    self.length = length

def get_status(self, fout):
    """
    Виведення статусу розміру рослини.

    Args:
    - fout (File): Файловий об'єкт для запису.

    """
    print("Ширина:", self.width)
    print("Довжина:", self.length)
    print("Висота:", self.height)

    fout.write("Ширина: " + str(self.width) + "\n")
    fout.write("Довжина: " + str(self.length) + "\n")
    fout.write("Висота: " + str(self.height) + "\n")

def change_size_to_grow(self):
    """
    Збільшення розміру рослини.

    """
    self.set_width(self.width + 1.1)
    self.set_height(self.height + 1.1)
    self.set_length(self.length + 1.1)

def change_size_to_death(self):
```

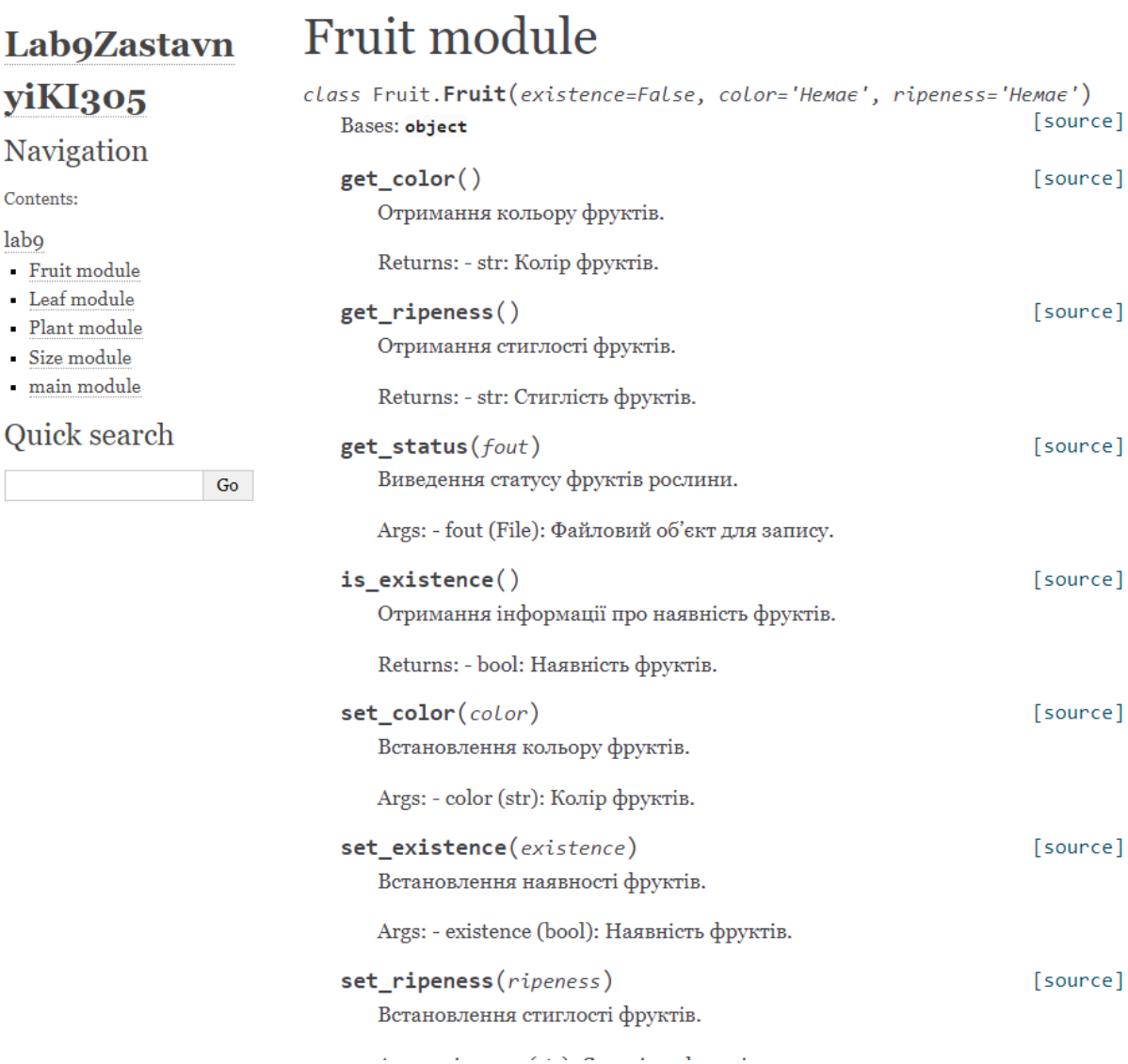
```
    """
    Зміна розміру рослини на нуль (для "смерті").

    """
    self.set_width(0)
    self.set_height(0)
    self.set_length(0)
```

**Результата роботи програми:**

```
C:\Users\fynti\PycharmProjects\pythonProject4\venv\Scripts\python.e
Рослина: Ялина
Колір: Зелений
Вологість: 70.0
Ширина: 1.1
Довжина: 1.1
Висота: 1.1
Кількість листків: 101
Листя: Немає
Форма: Немає
Колір: Немає
У рослини немає плодів.

Тип дерева: Немає
Сфотосинтезовано 12.0 кисню.
Рослина: Немає
Колір: Немає
Вологість: 0
Ширина: 0
Довжина: 0
Висота: 0
Кількість листків: 0
Листя: Немає
Форма: Немає
Колір: Немає
У рослини немає плодів.

Тип дерева: Хвойний

Process finished with exit code 0
```

**Фрагмент згенерованої документації**



Lab9Zastavnyi KI305

Navigation

Contents:

lab9

- Fruit module
- Leaf module
- Plant module
- Size module
- main module

Quick search

Go

# Fruit module

*class* Fruit.**Fruit**(*existence=False, color='Немає', ripeness='Немає'*) [source]

Bases: **object**

**get_color**() [source]

Отримання кольору фруктів.

Returns: - str: Колір фруктів.

**get_ripeness**() [source]

Отримання стиглості фруктів.

Returns: - str: Стиглість фруктів.

**get_status**(*fout*) [source]

Виведення статусу фруктів рослини.

Args: - fout (File): Файловий об'єкт для запису.

**is_existence**() [source]

Отримання інформації про наявність фруктів.

Returns: - bool: Наявність фруктів.

**set_color**(*color*) [source]

Встановлення кольору фруктів.

Args: - color (str): Колір фруктів.

**set_existence**(*existence*) [source]

Встановлення наявності фруктів.

Args: - existence (bool): Наявність фруктів.

**set_ripeness**(*ripeness*) [source]

Встановлення стиглості фруктів.

**Висновок:** я оволодів навиками реалізації парадигм об'єктно-орієнтованого програмування використовуючи засоби мови Python.