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.

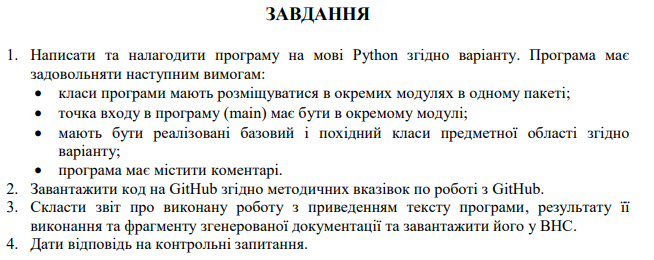
Прийняв:

доцент кафедри ЕОМ

Іванов Ю. С.

Львів-2023

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



**Варіант завдання:**

**:** 

**:**

**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")

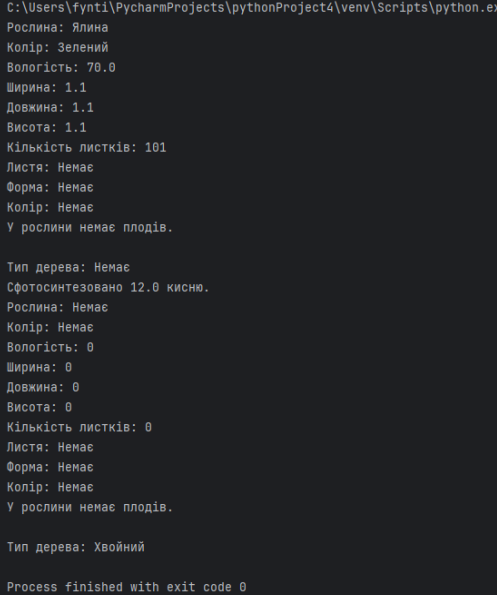
**Plant.py**

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)

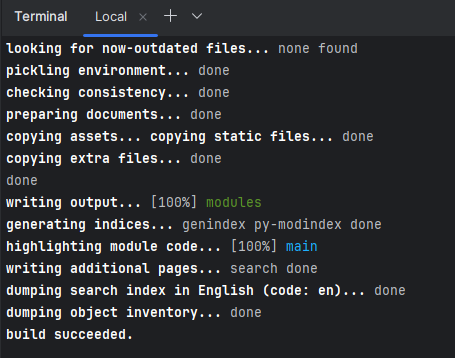
**Size.py**

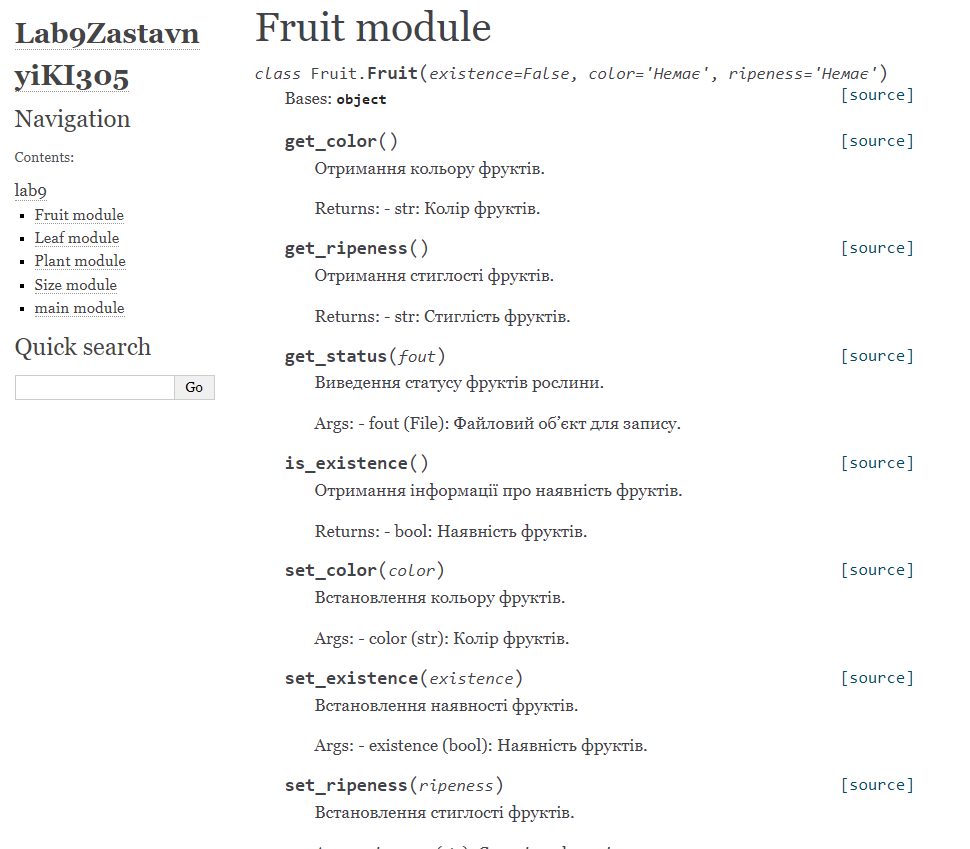
class Size:  
 def \_\_init\_\_(self, width=0, height=0, length=0):  
 *"""  
 Конструктор класу Size.  
  
 Ініціалізує атрибути розміру рослини.  
  
 Args:  
 - width (float): Ширина рослини.  
 - height (float): Висота рослини.  
 - length (float): Довжина рослини.  
  
 """* self.width = width  
 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:**



**Фрагмент згенерованої документації**





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