Міністерство освіти і науки України

Національний університет «Львівська політехніка»

Кафедра «Електронних обчислювальних машин»



**Звіт**

з лабораторної роботи № 9

з дисципліни: «Кросплатформенні засоби програмування»

**На тему:** «Основи Об’єктно-орієнтованого програмування у PYTHON»

**Виконав:**

студент групи КІ-306

Бокало П.М.

**Прийняв:**

доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

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

**Завдання(варіант №2):**

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.

**Індивідуальне завдання:**





**Вихідний код програми:**

**Файл Bokalo.KI\_306\_1.Lab9.py:**

from Starship import Starship

from Multi\_Starship import Multi\_Starship

def main():

ms = Multi\_Starship("Boss", "Soreman")

ms.stats()

print("\n")

for i in range(10):

ms.fight()

print("\n")

ms.stats()

print("\n")

ms.repair()

print("\n")

ms.call\_captain()

print("\n")

ms.stats()

print("\n")

ms.call\_captain()

print("\n")

ms.closer()

print("\n")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Файл Starship.py:**

class Starship:

def \_\_init\_\_(self, name="void", c\_name="Noname"):

self.fuel = 100

self.s\_name = name

self.s\_health = 100

self.staff = 10

self.captain = Captain(c\_name)

self.is\_fighting = False

self.fout = None

def closer(self):

if self.fout is not None:

self.fout.close()

self.captain.close()

def call\_captain(self):

self.captain.Captain\_stats()

def move(self):

if self.fuel > 10:

self.fuel -= 10

else:

print("Can't move!!!")

def damage(self):

if self.s\_health > 0 and self.staff > 0:

self.s\_health -= 10

self.staff -= 1

elif self.s\_health == 0:

print("Starship is critically damaged!!!")

def repair(self):

if self.s\_health < 100:

self.staff += 5

self.s\_health += 10

coins = self.captain.getCoins()

coins -= 50

self.captain.setCoins(coins)

else:

print("Repaired")

def fight(self):

self.is\_fighting = True

self.damage()

coins = self.captain.getCoins()

self.captain.setCoins(coins + 25)

def retreat(self):

self.is\_fighting = False

def stats(self):

print("Stats:")

print("Name:", self.s\_name)

print("Fuel:", self.fuel)

print("Health:", self.s\_health)

print("Staff:", self.staff)

print("Fighting:", self.is\_fighting)

def refuel(self):

self.fuel = 100

coins = self.captain.getCoins()

self.captain.setCoins(coins - 25)

def clearShip\_logs(self):

try:

with open("Starship\_log.txt", "w") as clearWriter:

clearWriter.close()

except FileNotFoundError:

print("Error: File 'Starship\_log.txt' cannot be cleared.")

class Captain:

def \_\_init\_\_(self, name="Noname"):

self.name = name

self.coins = 100

self.health = 100

self.id = 0

self.fout = None

def getCoins(self):

return self.coins

def setCoins(self, coins):

self.coins = coins

def Captain\_stats\_file(self):

print("New Captain created:")

print("ID:", self.id)

print("Name:", self.name)

print("Coins:", self.coins)

print("Health:", self.health)

def Captain\_stats(self):

print("Your captain has:")

print("Name:", self.name)

print("Coins:", self.coins)

print("Health:", self.health)

def close(self):

if self.fout is not None:

self.fout.close()

def clearCaptain\_logs(self):

try:

with open("Captain\_log.txt", "w") as clearWriter:

clearWriter.close()

except FileNotFoundError:

print("Error: File 'Captain\_log.txt' cannot be cleared.")

**Файл Multi\_Starship.py:**

from Starship import Starship

class Multi\_Starship(Starship):

def \_\_init\_\_(self, name="void", c\_name="Noname"):

super().\_\_init\_\_(name, c\_name)

self.Life\_points = 5

def Star\_jump(self):

for \_ in range(5):

self.move()

def repair(self):

if self.s\_health == 0:

self.Life\_points = self.Life\_points - 1 if self.Life\_points > 0 else self.Life\_points

super().repair()

elif self.Life\_points == 0:

print("Out of life points!!! Get more")

elif 0 < self.s\_health < 100:

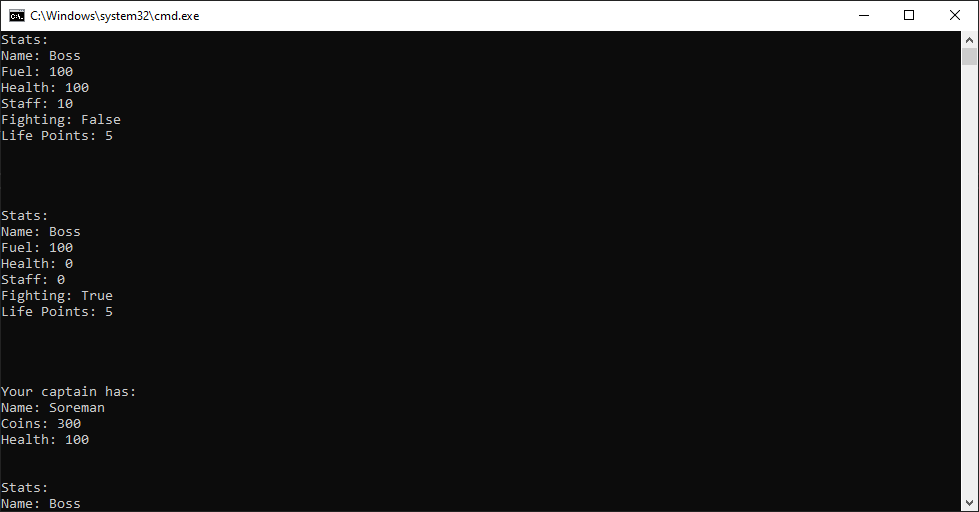
super().repair()

def stats(self):

super().stats()

print("Life Points:", self.Life\_points)

**Результат роботи програми:**



**Відповідь на контрольні питання**

1. ***Що таке модулі?***

Модулі - це файли, які містять код для використання в інших програмах.

1. ***Як імпортувати модуль?***

Імпорт модуля здійснюється за допомогою ключового слова "import" та імені модуля.

1. ***Як оголосити клас?***

Клас оголошується ключовим словом "class", за яким слідує ім'я класу.

1. ***Що може міститися у класі?***

У класі можуть міститися атрибути (змінні) та методи (функції).

1. ***Як називається конструктор класу?***

Конструктор класу називається "init".

1. ***Як здійснити спадкування?***

Спадкування здійснюється за допомогою наслідування від іншого класу.

1. ***Які види спадкування існують?***

Існують одинарне та багатошарове спадкування.

1. ***Які небезпеки є при множинному спадкуванні, як їх уникнути?***

При множинному спадкуванні можуть виникати конфлікти та нерозбіжності, які слід уникнути шляхом коректної організації класів.

1. ***Що таке класи-домішки?***

Класи-домішки - це класи, які містять методи, які можна використовувати у інших класах.

1. ***Яка роль функції super() при спадкуванні?***

Функція super() використовується для виклику методів батьківського класу при спадкуванні.

**Висновок:** на цій лабораторній роботі, я оволодів навичками реалізації парадигм об’єктно-орієнтованого програмування використовуючи засоби мови Python.