**ДВНЗ «ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ»**

**Факультет комп’ютерно-інформаційних технологій та автоматизації**

**ЗВІТ до практичної РОБОТИ №5**

з дисципліни «Методи та системи штучного інтелекту»

за темою: «ПРОЕКТУВАННЯ КЛАСІВ ТА ЇХ ІЄРАРХІЙ»

Варіант №7

Виконав:

Студент гр. КІ-21

Іван ДАЦЕНКО

Перевірив:

Андрій НІКІТЕНКО

Луцьк – 2024

**Мета роботи:** вивчити та засвоїти базові навички роботи з класами та їх спадкуванням у Python. Ознайомитися з поняттями конструктор, деструктор, метод класу, рівні доступу, перевизначення методу, ієрархія класів, доповнення методу та отримати практичні навички їх застосування.

**Завдання**

1) Створити клас згідно з варіантом (табл. 5.1). У програмі необхідно створити декілька об’єктів визначеного класу та продемонструвати роботу кожного з методів класу.

Зображення, що містить текст, знімок екрана, Шрифт, число

Автоматично згенерований опис

2) У таблиці 6.2 задано сімейство об'єктів, що мають деяку схожість (загальні ознаки). Необхідно виділити найбільш загальні риси об'єктів, на основі яких скласти базовий клас. На основі базового класу розробити ієрархію класівнащадків, кінцевими гілками в якій будуть задані об'єкти. Ієрархія класів повинна бути мінімум трирівнева, тобто між базовим класом і класами, що описують задані об'єкти, повинен бути хоча б один проміжний рівень ієрархії. Продемонструвати роботу з об'єктами, заданими за таблицею 6.2. У звіті привести діаграму класів.



**Лістинг програми:**

**1 завдання – lab5.1.py**

class Nomenclature:  
 def \_\_init\_\_(self, name, wholesale\_price, retail\_markup, quantity):  
 self.name = name  
 self.wholesale\_price = wholesale\_price  
 self.retail\_markup = retail\_markup  
 self.quantity = quantity  
  
 def calculate\_profit(self, selling\_price):  
 total\_cost = self.wholesale\_price + (self.wholesale\_price \* self.retail\_markup / 100)  
 profit = (selling\_price - total\_cost) \* self.quantity  
 return profit  
  
 def display\_info(self):  
 print("\nProduct Name:", self.name)  
 print("Wholesale Price:", self.wholesale\_price)  
 print("Retail Markup (%):", self.retail\_markup)  
 print("Quantity:", self.quantity)  
  
  
#Example  
product1 = Nomenclature("Chair", 50, 30, 100)  
product1.display\_info()  
profit1 = product1.calculate\_profit(100)  
print("Possible Profit:", profit1)  
  
product2 = Nomenclature("Table", 150, 25, 50)  
product2.display\_info()  
profit2 = product2.calculate\_profit(250)  
print("Possible Profit:", profit2)

**2 завдання – lab5.2.py.**

class Weapon:  
 def \_\_init\_\_(self, name, material, damage):  
 self.name = name  
 self.material = material  
 self.damage = damage  
  
 def display\_info(self):  
 print("Name:", self.name)  
 print("Material:", self.material)  
 print("Damage:", self.damage)  
  
  
class BladeWeapon(Weapon):  
 def \_\_init\_\_(self, name, material, damage, blade\_length):  
 super().\_\_init\_\_(name, material, damage)  
 self.blade\_length = blade\_length # Blade length attribute  
  
  
class ThrowingWeapon(Weapon):  
 def \_\_init\_\_(self, name, material, damage, quantity):  
 super().\_\_init\_\_(name, material, damage)  
 self.quantity = quantity # Quantity attribute  
  
  
class PoleWeapon(Weapon):  
 def \_\_init\_\_(self, name, material, damage, length):  
 super().\_\_init\_\_(name, material, damage)  
 self.length = length # Length attribute  
  
  
class Sword(BladeWeapon):  
 def \_\_init\_\_(self, name, material, damage, blade\_length):  
 super().\_\_init\_\_(name, material, damage, blade\_length)  
  
 def swing(self):  
 print("Swinging the sword...") # Swing method  
  
  
class Dagger(BladeWeapon):  
 def \_\_init\_\_(self, name, material, damage, blade\_length):  
 super().\_\_init\_\_(name, material, damage, blade\_length)  
  
 def stab(self):  
 print("Stabbing with the dagger...") # Stab method  
  
  
class Rapier(BladeWeapon):  
 def \_\_init\_\_(self, name, material, damage, blade\_length, flexibility):  
 super().\_\_init\_\_(name, material, damage, blade\_length)  
 self.flexibility = flexibility # Flexibility attribute  
  
 def parry(self):  
 print("Parrying with the rapier...") # Parry method  
  
  
class Kindjal(BladeWeapon):  
 def \_\_init\_\_(self, name, material, damage, blade\_length):  
 super().\_\_init\_\_(name, material, damage, blade\_length)  
  
 def assassinate(self):  
 print("Assassinating with the kindjal...") # Assassinate method  
  
  
class Shuriken(ThrowingWeapon):  
 def \_\_init\_\_(self, name, material, damage, quantity):  
 super().\_\_init\_\_(name, material, damage, quantity)  
  
 def throw(self):  
 print("Throwing the shuriken...") # Throw method  
  
  
class Spear(PoleWeapon):  
 def \_\_init\_\_(self, name, material, damage, length):  
 super().\_\_init\_\_(name, material, damage, length)  
  
 def thrust(self):  
 print("Thrusting with the spear...") # Thrust method  
  
  
# Demonstrating objects  
sword = Sword("Longsword", "Steel", 10, 36)  
print()  
sword.display\_info()  
sword.swing()  
  
dagger = Dagger("Dagger", "Iron", 5, 12)  
print()  
dagger.display\_info()  
dagger.stab()  
  
spear = Spear("Spear", "Wood", 8, 72)  
print()  
spear.display\_info()  
spear.thrust()  
  
rapier = Rapier("Rapier", "Steel", 7, 42, "High")  
print()  
rapier.display\_info()  
rapier.parry()  
  
kindjal = Kindjal("Kindjal", "Steel", 6, 14)  
print()  
kindjal.display\_info()  
kindjal.assassinate()  
  
shuriken = Shuriken("Shuriken", "Metal", 3, 5)  
print()  
shuriken.display\_info()  
shuriken.throw()

**Приклад вигляду ієрархії класів до коду вище:**

Weapon

+name

+material

+damage

+display\_info()

Sword

+blade\_length

+swing()

Dagger

+stab()

Spear

+thrust()

+blade\_length

+lenght

Rapier

+parry()

+blade\_lenght

+flexibility

Kindjal

+assassinate()

+blade\_lenght

Shuriken

+throw()

+quantity

BladeWeapon

+blade\_lenght

ThrowingWeapon

+quantity

PoleWeapon

+length

**Результати:**

Зображення, що містить текст, знімок екрана, Шрифт

Автоматично згенерований опис

Рисунок 1 − Результати виконання першого завдання

Зображення, що містить текст, знімок екрана, програмне забезпечення

Автоматично згенерований опис

Рисунок 2 − Результат виконання другого завдання

**Висновок:** У цій практичній роботі я вивчив та засвоїв базові навички роботи з класами та їх спадкуванням у Python. Ознайомився з поняттями конструктор, деструктор, метод класу, рівні доступу, перевизначення методу, ієрархія класів, доповнення методу та отримав практичні навички їх застосування.