Міністерство освіти і науки України

Державний університет ,,Житомирська політехніка”

Кафедра КІ та КБ

Група: КБ-21-1

Програмування мовою Python

Лабораторна робота №10

«unit-тестування в мові Python»

Виконала: Поліщук К. Р.

Прийняв: Морозов Д. С.

**Мета роботи:** : ознайомитися з фреймворками для unit-тестування в мові Python

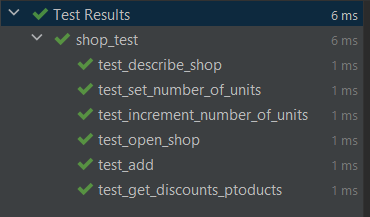
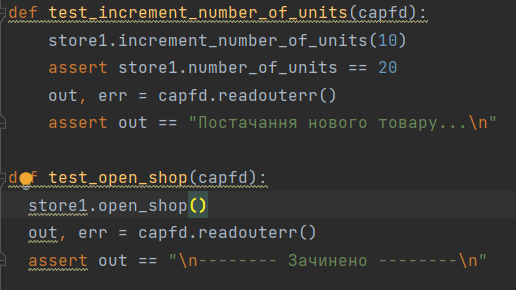
**Хід роботи**

Використовуючи фреймворки для unit-тестування unittest або pytest напишіть тести для класів створених під час виконання завданнь 8 і 9 в лабораторній роботі №8. За потреби модифікуйте код класів для виправлення можливих помилок, що будуть знайдені під час покриття класів тестами.

Лістинг тестування класу Shop:

from classLAB import Shop, Discount  
import pytest  
  
store1 = Shop("Products", "магазин фермерської продукції")  
store\_discount = Discount()  
  
def test\_describe\_shop(capfd):  
 store1.describe\_shop()  
 out, err = capfd.readouterr()  
 assert out == "\nМагазин 'Products' - це магазин фермерської продукції\n"  
  
def test\_set\_number\_of\_units(capfd):  
 if store1.set\_number\_of\_units(10):  
 assert store1.number\_of\_units == 10  
 else:  
 store1.set\_number\_of\_units(-1)  
 out, err = capfd.readouterr()  
 assert out == "Недопустиме значення\n"  
  
def test\_increment\_number\_of\_units(capfd):  
 store1.increment\_number\_of\_units(10)  
 assert store1.number\_of\_units == 20  
 out, err = capfd.readouterr()  
 assert out == "Постачання нового товару...\n"  
  
def test\_open\_shop(capfd):  
 store1.open\_shop()  
 out, err = capfd.readouterr()  
 assert out == "\n-------- Зачинено --------\n"  
  
def test\_add(capfd):  
 store\_discount.add("Сукня", "Джинси")  
 out, err = capfd.readouterr()  
 assert out == "Додаємо акціні товари...\n"  
  
def test\_get\_discounts\_ptoducts(capfd):  
 store\_discount.get\_discounts\_ptoducts()  
 out, err = capfd.readouterr()  
 assert out == "Акційні товари: Сукня Джинси\n"

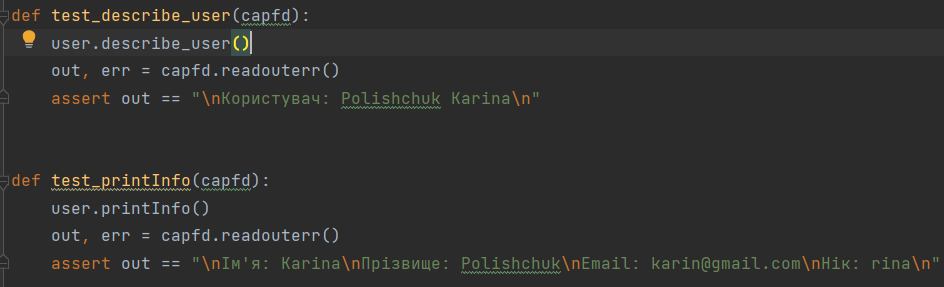
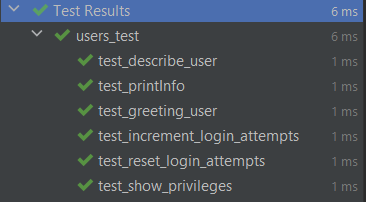
Результат виконання:

Лістинг тестування класу User:

from classLAB import User, Privileges, Admin  
import pytest  
  
user = User("Karina", "Polishchuk", "karin@gmail.com", "rina")  
admin = Admin("Admin", "", "admin@gmail.com", "Adm", True)  
  
  
def test\_describe\_user(capfd):  
 user.describe\_user()  
 out, err = capfd.readouterr()  
 assert out == "\nКористувач: Polishchuk Karina\n"  
  
  
def test\_printInfo(capfd):  
 user.printInfo()  
 out, err = capfd.readouterr()  
 assert out == "\nІм'я: Karina\nПрізвище: Polishchuk\nEmail: karin@gmail.com\nНік: rina\n"  
  
  
def test\_greeting\_user(capfd):  
 user.greeting\_user()  
 out, err = capfd.readouterr()  
 assert out == "Вітаємо, Karina Polishchuk\n"  
  
  
def test\_increment\_login\_attempts(capfd):  
 user.increment\_login\_attempts()  
 out, err = capfd.readouterr()  
 assert out == "Здійснюється вхід в аккаунт...\n"  
 assert User.login\_attempts == 1  
  
  
def test\_reset\_login\_attempts(capfd):  
 user.reset\_login\_attempts()  
 out, err = capfd.readouterr()  
 assert out == "Обнуляємо спроби входу...\n"  
 assert User.login\_attempts == 0  
  
  
def test\_show\_privileges(capfd):  
 admin.priv.add("Allowed to add message", "Allowed to delete users", "Allowed to ban users")  
 admin.priv.show\_privileges()  
 out, err = capfd.readouterr()  
 assert out == "Привілеї: Allowed to add message,\nAllowed to delete users,\nAllowed to ban users\n"

Результат виконання:



Лістинг класу classLAB:

import random  
import datetime  
  
  
class Shop:  
 def \_\_init\_\_(self, shop\_name, store\_type, number\_of\_units=0):  
 self.shop\_name = shop\_name  
 self.store\_type = store\_type  
 self.number\_of\_units = number\_of\_units  
  
 def describe\_shop(self):  
 print(f"\nМагазин '{self.shop\_name}' - це {self.store\_type}")  
  
 def set\_number\_of\_units(self, number\_of\_units):  
 if number\_of\_units > 0:  
 self.number\_of\_units = number\_of\_units  
 else:  
 print("Недопустиме значення")  
  
 def increment\_number\_of\_units(self, amount):  
 print("Постачання нового товару...")  
 self.number\_of\_units += amount  
  
 def open\_shop(self):  
 def time\_in\_range(start, end, current):  
 return start <= current <= end  
  
 start = datetime.time(8, 0, 0)  
 end = datetime.time(20, 0, 0)  
 current = datetime.datetime.now().time()  
 if (time\_in\_range(start, end, current) == True):  
 print("\n-------- Відчинено --------")  
 else:  
 print("\n-------- Зачинено --------")  
  
  
class Discount(Shop):  
 def \_\_init\_\_(self):  
 self.discount\_products = []  
  
 def add(self, \*a):  
 print("Додаємо акціні товари...")  
 self.discount\_products.extend(a)  
  
 def get\_discounts\_ptoducts(self):  
 print(f"Акційні товари:", \*(self.discount\_products))  
 return self.discount\_products  
  
  
# ------------------------------------------------------------- task9 -----------------------------------------------------------------  
  
class User:  
 login\_attempts = 0  
  
 def \_\_init\_\_(self, first\_name="", last\_name="", email="", nickname="", agreement=True):  
 self.first\_name = first\_name  
 self.last\_name = last\_name  
 self.email = email  
 self.nickname = nickname  
 self.agreement = agreement  
  
 def describe\_user(self):  
 print("\nКористувач:", self.last\_name, self.first\_name)  
  
 def printInfo(self):  
 print(f"\nІм'я: {self.first\_name}\nПрізвище: {self.last\_name}\nEmail: {self.email}\nНік: {self.nickname}")  
  
 def greeting\_user(self):  
 print("Вітаємо,", self.first\_name, self.last\_name)  
  
 @staticmethod  
 def increment\_login\_attempts():  
 print("Здійснюється вхід в аккаунт...")  
 User.login\_attempts += 1  
  
 @staticmethod  
 def reset\_login\_attempts():  
 print("Обнуляємо спроби входу...")  
 User.login\_attempts = 0  
  
  
class Privileges():  
 privileges = []  
  
 def \_\_init\_\_(self, admin):  
 self.privileges = list(admin.privileges)  
  
 def add(self, \*a):  
 self.privileges.extend(a)  
  
 def show\_privileges(self):  
 print(f"Привілеї:", ',\n'.join(self.privileges))  
  
  
class Admin(User):  
 privileges = []  
  
 def \_\_init\_\_(self, first\_name="", last\_name="", email="", nickname="", agreement=True):  
 self.priv = Privileges(self)  
 self.first\_name = first\_name  
 self.last\_name = last\_name  
 self.email = email  
 self.nickname = nickname  
 self.agreement = agreement  
  
  
def add(firstNumber, secondNumber):  
 return firstNumber + secondNumber  
  
  
def subtract(firstNumber, secondNumber):  
 return firstNumber - secondNumber

**Висновок:** в ході виконання лабораторної роботи було вивчено як працювати з фреймворками для unit-тестування в мові Python