Національний університет «Львівська Політехніка»

Інститут комп’ютерних технологій, автоматики та метрології

Кафедра електронних обчислювальних машин



Звіт

Про виконання лабораторної роботи №9

### З дисципліни «Кросплатформлені засоби програмування»

**Виконав:**

студент групи КІ-305

Рудий В.В.

**Перевірив:**

Доцент кафедри ЕОМ

Іванов Ю. С.

Львів – 2023

**Тема:** Основи об’єктно-орієнтованого програмування у Python.

**Мета:** 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.

**Завдання згідно варіанту:**

***Варіант: 23 Завдання:***

***Базовий клас:*** Плитка для приготування їжі

***Похідні класи:*** Газова плитка

**Виконання:**

**Код програми:**

***main.py***

from GasStove import GasStove  
  
# основна програма  
if \_\_name\_\_ == "\_\_main\_\_":  
 # об'єкт на основі класу газова плита  
 stove = GasStove()  
  
 # використання методів класу за допомогою створеного об'єкта  
 stove.repaintStoveForCooking("Red")  
 stove.getInfoLook()  
 stove.getBurnerArea()  
 stove.powerUpButtonResource()  
 stove.powerDownButtonResource()  
 for i in range(5):  
 stove.volumeUp()  
 for i in range(3):  
 stove.volumeDown()  
 stove.volumeMax()  
 stove.volumeMin()  
 stove.stoveHeating()  
 stove.addGas(110)  
 stove.stoveWorking()  
 stove.getGas()

***StoveForCooking.py***

from Type import Type  
from Burner import Burner  
from Button import Button  
  
# клас аудіоплеєр  
class StoveForCooking:  
 # конструктор  
 def \_\_init\_\_(self):  
 self.\_\_typeOfStoveForCooking = Type()  
 self.\_\_burnerOfStoveForCooking = Burner()  
 self.\_\_volumeUpButton = Button()  
 self.\_\_volumeDownButton = Button()  
  
 # метод змінює колір корпусу плити  
 def repaintStoveForCooking(self, uColor):  
 self.\_\_typeOfStoveForCooking.setColor(uColor)  
 print("New color of stove for cooking is " + uColor + ".")  
  
 # метод змінює колір корпусу плити  
 def resizeStoveForCooking(self, uLength, uWidth, uHeight):  
 self.\_\_typeOfStoveForCooking.setSize(uLength, uWidth, uHeight)  
 print("New size of stove for cooking is " + uLength + "x" + uWidth + "x" + uHeight + " .")  
  
  
 # метод повертає повну інформацію про зовнішній вигляд плити  
 def getInfoLook(self):  
 print("Size of stove for cooking: " + str(self.\_\_typeOfStoveForCooking.getSize()) + " m^2.")  
 print("Weight of stove for cooking: " + str(self.\_\_typeOfStoveForCooking.getWeight()) + " kg.")  
 print("Color of stove for cooking: " + str(self.\_\_typeOfStoveForCooking.getColor()) + ".")  
  
 # метод повертає площу, яку займають комфорка на плиті  
 def getBurnerArea(self):  
 print("Burner occupy " + str(self.\_\_burnerOfStoveForCooking.getArea() \* 2) + " cm^2 of size.")  
  
 # метод виводить ресурс кнопки збільшення потужності  
 def powerUpButtonResource(self):  
 print("You can tap the power up button " + str(self.\_\_volumeUpButton.getResource()) + " more times.")  
  
 # метод виводить ресурс кнопки зменшення потужності  
 def powerDownButtonResource(self):  
 print("You can tap the power down button " + str(self.\_\_volumeDownButton.getResource()) + " more times.")  
  
 # метод симулює натискання на кнопку збільшення потужності  
 def volumeUp(self):  
 self.\_\_volumeUpButton.clickButton(True)  
 print("You tap the power up button.")  
 if self.\_\_burnerOfStoveForCooking.getPower() > 100:  
 self.\_\_burnerOfStoveForCooking.getPower(100)  
  
 # метод симулює натискання на кнопку зменшення потужності  
 def volumeDown(self):  
 self.\_\_volumeDownButton.clickButton(False)  
 print("You tap the power down button.")  
 if self.\_\_burnerOfStoveForCooking.getPower() < 0:  
 self.\_\_burnerOfStoveForCooking.getPower(0)  
  
 # метод симулює затискання кнопки зменшення потужності  
 def volumeMin(self):  
 self.\_\_volumeDownButton.clickButton(False)  
 print("You press the power down button.")  
 self.\_\_burnerOfStoveForCooking.setPower(0)  
  
 # метод симулює затискання кнопки збільшення потужності  
 def volumeMax(self):  
 self.\_\_volumeUpButton.clickButton(True)  
 print("You press the power up button.")  
 self.\_\_burnerOfStoveForCooking.setPower(100)

***GasStove.py***

from StoveForCooking import StoveForCooking  
  
# клас диктофон  
class GasStove(StoveForCooking):  
 # конструктор  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_()  
 self.\_\_connectingTheCylinder = True  
 self.\_\_amountOfGas = 30  
  
 # метод симулює розігрів плити  
 def stoveHeating(self):  
 print("The stove is heating up.")  
  
 # метод додає газу у балон  
 def addGas(self, additionalGas):  
 self.\_\_amountOfGas += additionalGas  
  
 # метод симулює процес роботи плити  
 def stoveWorking(self):  
 if self.\_\_connectingTheCylinder:  
 print("The stove is working.")  
 else:  
 print("The stove is NOT working.")  
 self.\_\_amountOfGas -= 5  
  
 # метод повертає кількість газу в балоні  
 def getGas(self):  
 print("Amount of gas in the cylinder: " + str(self.\_\_amountOfGas) + " L.")

***Type.py***

# клас корпус  
class Type:  
 #конструктор  
 def \_\_init\_\_(self):  
 self.\_\_color = "White"  
 self.\_\_length = 0.5  
 self.\_\_width = 0.6  
 self.\_\_height = 0.8  
   
 # метод встановлює колір корпусу  
 def setColor(self, newColor):  
 self.\_\_color = newColor  
  
 # метод повертає колір корпусу  
 def getColor(self):  
 return self.\_\_color  
  
 # метод встановлює розмір корпусу  
 def setSize(self, newLength, newWidth, newHeight):  
 self.\_\_length = newLength  
 self.\_\_width = newWidth  
 self.\_\_height = newHeight  
  
 # метод повертає розмір плити  
 def getSize(self):  
 return self.\_\_length \* self.\_\_width  
  
 # метод повертає вагу плити  
 def getWeight(self):  
 return self.\_\_height \* self.\_\_length \* self.\_\_width \* 166.6

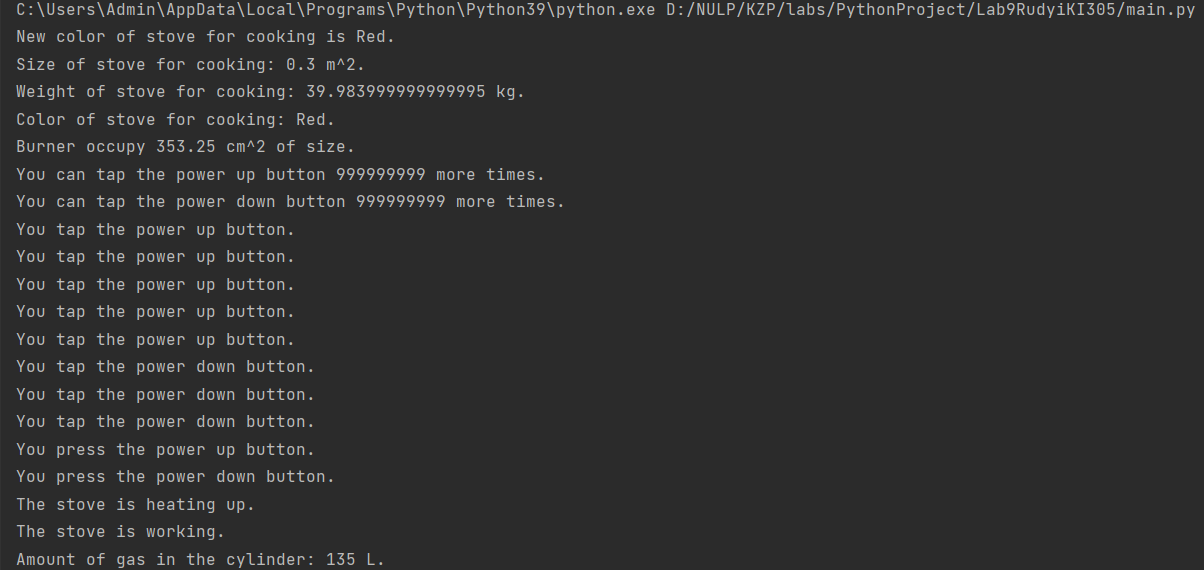
***Burner.py***

# клас комфорка  
class Burner:  
 # конструктор  
 def \_\_init\_\_(self):  
 self.\_\_diametr = 15  
 self.\_\_power = 50  
  
 # метод повертає площу комфорки  
 def getArea(self):  
 return 3.14 \* (self.\_\_diametr / 2) \* (self.\_\_diametr / 2)  
  
 # метод повертає потужність плити  
 def getPower(self):  
 return self.\_\_power  
  
 # метод втановлює потужності плити  
 def setPower(self, newPower):  
 if newPower < 0 or newPower > 100:  
 return "Incorrectly specified power!"  
 else:  
 self.\_\_power = newPower

***Button.py***

# клас кнопка  
class Button:  
 # конструктор  
 def \_\_init\_\_(self):  
 self.\_\_btnResource = 999999999  
 self.\_\_count = 0  
  
 # метод симулює натискання на кнопку  
 def clickButton(self, type):  
 if type:  
 self.\_\_btnResource -= 1  
 self.\_\_count += 1  
 else:  
 self.\_\_btnResource -= 1  
 self.\_\_count -= 1  
  
 # метод повертає ресурс кнопки  
 def getResource(self):  
 return self.\_\_btnResource  
  
 # метод повертає кількість натисків на кнопку за раз  
 def getCountClicks(self):  
 return self.\_\_count

**Результат роботи програми у консолі:**



**Висновок:** на цій лабораторній роботі я оволодів навиками реалізації парадигм об’єктно-орієнтованого програмування використовуючи засоби мови Python. Створив клас аудіоплеєр, який наслідує деякі властивості класів корпус, динамік та кнопка. Також створив клас диктофон, на основі класу аудіоплеєр і розробив програму-драйвер, у якій створив об’єкт на основі цього класу та протестував його методи.