**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНОМУ УНІВЕРСИТЕТІ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

**Кафедра систем штучного інтелекту**

**Лабораторна робота №4**

з дисципліни

«Системний аналіз»

**Варіант №25**

**Виконав:**

студент групи КН-207

Шиманський П.С.

**Викладач:**

Бойко Н.І.

Львів – 2019

**Шаблон Одинак**

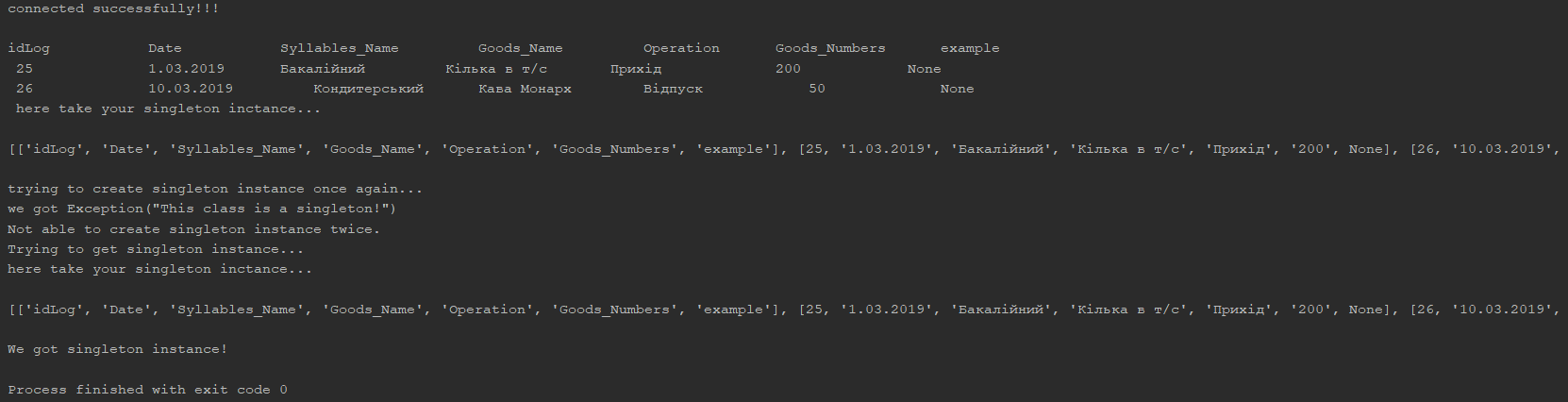
**Опис використання в програмі**

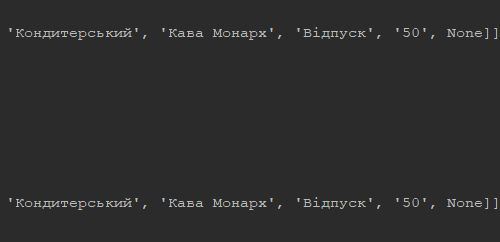
Створення шаблону Одинак(Singleton) для створення одного виключного каналу доступу до таблиці log\_book.

**Код:**

class operand():  
 def get\_worker():  
 connection = connect()  
 try:  
 with connection.cursor() as cursor:  
 sql = "select \* from syllables\_list"  
 cursor.execute(sql)  
 print()  
 ret=[]  
 for row in cursor:  
 if (print\_db\_row==True):  
 print(row)  
 ret.append(row)  
 finally:  
 connection.close()  
 return ret  
  
 def get\_log\_book():  
 connection = connect()  
 try:  
 with connection.cursor() as cursor:  
 sql = "select \* from log\_book"  
 cursor.execute(sql)  
 print()  
 ret=[]  
 for row in cursor:  
 if (print\_db\_row==True):  
 print(row)  
 ret.append(row)  
 finally:  
 connection.close()  
 return ret  
  
  
class printer():  
 def show\_beautifully(array):  
 for x in array:  
 for y in x:  
 if(type(y)==type("test") and len(y)<9):  
 print(y, "\t\t\t", end=" ")  
 elif(type(y)==type("test") and len(y)<17):  
 print(y, "\t\t", end=" ")  
 elif(type(y)==type("test") and len(y)>17):  
 print(y, "\t", end=" ")  
 elif(type(y)==type(type\_test)):  
 print(y, "\t\t", end=" ")  
 else:  
 print(y, "\t\t\t", end=" ")  
 print("\n", end=" ")  
  
 def adapter(dbarray):  
 output=[]  
 a=[]  
 for b in dbarray[0]:  
 a.append(b)  
 output.append(a)  
 for row in dbarray:  
 elements=[]  
 for element in row:  
 elements.append(row[element])  
 output.append(elements)  
 return output  
  
   
  
class Singleton:  
 \_\_instance = None  
 @staticmethod  
   
 def getInstance():  
 *""" Static access method. """* if Singleton.\_\_instance == None:  
 Singleton()  
 print("here take your singleton inctance...\n")  
 return Singleton.\_\_instance  
  
   
 def \_\_init\_\_(self):  
 *""" Virtually private constructor. """* if Singleton.\_\_instance != None:  
 raise Exception("This class is a singleton!")  
  
 else:  
 Singleton.\_\_instance = printer.adapter(operand.get\_log\_book())  
 printer.show\_beautifully(Singleton.\_\_instance)  
  
o = Singleton()  
  
s = Singleton.getInstance()  
print (s, "\n")  
  
def try\_to\_create():  
 try:  
 print("trying to create singleton instance once again...")  
 t=Singleton()  
 except Exception:  
 print('we got Exception("This class is a singleton!")')  
 print("Not able to create singleton instance twice. ")  
 print("Trying to get singleton instance...")  
 t=Singleton.getInstance()  
 print(t, "\n")  
 print("We got singleton instance!")  
  
try\_to\_create()

**Результат виконання:**





**Висновок:**

На цій лабораторній роботі, я ознайомився з шаблоном Одинак, а також програмно його застосував.