**Харківський національний економічний університет**

**імені Семена Кузнеця**

**ЗВІТ**

**З ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 15**

**за дисципліною: *“*Технології програмування**”

**на тему: “: Ітератори, генератори, декоратори”**

**Варіант: 4**

**Виконав: студент факультету Інформаційних технологій**

**2 курсу, спец. Кібербезпека,**

**групи 6.04.125.010.21.2**

**Бойко Вадим Віталійович**

**Перевірив:**

**Венгріна Олена Сергіївна**

**ХНЕУ ім. С. Кузнеця**

**2023**

**Мета:** Метою роботи є набуття навичок технологій розробки програмного коду із використання механізмів ітераторів, генераторів та декораторів.

**Завданя:** Відповідно до варіанту завдання розробити клас ітератор, функцію генератор, декоратор для логування часу початку та закінчення подій й файл логів.

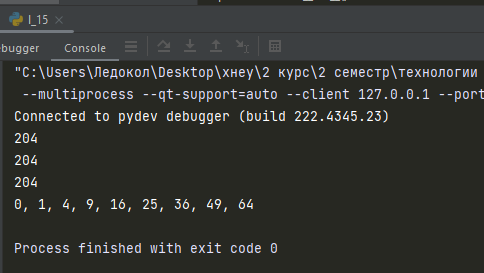
Варінт 4

Квадратична послідовність. Надається розмір послідовності. Визначається елементи послідовності. Визначити суму перших 10, 100, 1000 елементів.

Я написав наступне:

*from* datetime *import* datetime  
  
  
*def* logger(*method*):  
 *def* wrapper(*ref*, *\*args*, *\*\*kwargs*):  
 logs = open('logs.log', 'a+')  
 logs.write(f'function {*method*.\_\_name\_\_} starts at {datetime.now().isoformat()}\n')  
 result = *method*(*ref*, \**args*, \*\**kwargs*)  
 logs.write(f'function {*method*.\_\_name\_\_} ends at {datetime.now().isoformat()} with result {result}\n')  
 logs.close()  
 *return* result  
 *return* wrapper  
  
*class* SquareQueue:  
 *def \_\_init\_\_*(self, *length*: int = 0):  
 self.\_\_length = *length* self.\_\_generate\_queue()  
  
  
 @logger  
 *def* \_\_generate\_queue(self):  
 self.\_\_square\_queue = [x \* x *for* x *in* range(self.\_\_length)]  
  
 @logger  
 *def* sum\_of\_elements(self, *count*: int = 10):  
 *return* sum(self.\_\_square\_queue[:*count*])  
  
 @logger  
 *def \_\_str\_\_*(self):  
 *return* ', '.join(map(str, q.\_\_square\_queue))  
  
*if* \_\_name\_\_ == '\_\_main\_\_':  
 q = SquareQueue(9)  
 print(q.sum\_of\_elements())  
 print(q.sum\_of\_elements(100))  
 print(q.sum\_of\_elements(1000))  
 print(q)

як результат в консолі я бачу наступне



Файл з логами

function \_\_generate\_queue starts at 2023-05-14T18:02:37.305669  
function \_\_generate\_queue ends at 2023-05-14T18:02:37.305669 with result None  
function sum\_of\_elements starts at 2023-05-14T18:02:37.312684  
function sum\_of\_elements ends at 2023-05-14T18:02:37.312684 with result 204  
function sum\_of\_elements starts at 2023-05-14T18:02:37.318667  
function sum\_of\_elements ends at 2023-05-14T18:02:37.318667 with result 204  
function sum\_of\_elements starts at 2023-05-14T18:02:37.325668  
function sum\_of\_elements ends at 2023-05-14T18:02:37.325668 with result 204  
function \_\_generate\_queue starts at 2023-05-14T18:02:51.986231  
function \_\_generate\_queue ends at 2023-05-14T18:02:51.987230 with result None  
function sum\_of\_elements starts at 2023-05-14T18:02:51.995229  
function sum\_of\_elements ends at 2023-05-14T18:02:51.995229 with result 204  
function sum\_of\_elements starts at 2023-05-14T18:02:52.002230  
function sum\_of\_elements ends at 2023-05-14T18:02:52.002230 with result 204  
function sum\_of\_elements starts at 2023-05-14T18:02:52.010229  
function sum\_of\_elements ends at 2023-05-14T18:02:52.010229 with result 204  
function \_\_str\_\_ starts at 2023-05-14T18:02:52.018226  
function \_\_str\_\_ ends at 2023-05-14T18:02:52.018226 with result 0, 1, 4, 9, 16, 25, 36, 49, 64