**ЛАБОРАТОРНА РОБОТА № 5**

**ФУНКЦІЇ**

***Мета***: ознайомитися основами функціонального програмування і використання користувацьких функцій в мові Python

**Хід роботи**

**Завдання 1:** Користувач вводить дві сторони трьох прямокутників. Вивести їх площі.

Лістинг програми

a = 0  
b = 0  
for i in range(3):  
 a = float(input('Введіть довжишу прямокутника: '))  
 while a <= 0:  
 print('Довжина прямокутника <=0!')  
 a = float(input('Введіть довжишу прямокутника: '))  
 b = float(input('Введіть ширину прямокутника: '))  
 while b <= 0:  
 print('Ширина прямокутника <=0!')  
 b = float(input('Введіть ширину прямокутника: '))  
 res = a \* b  
 print("Площа прямокутника: ", res)

Результат виконання програми:

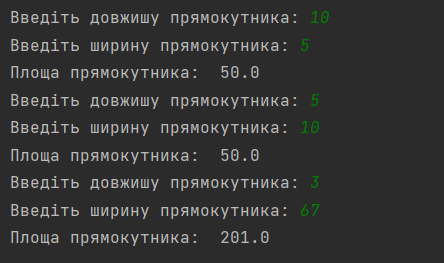


Рис. 1. Результат виконання програми

**Завдання 2:** Дано катети двох прямокутних трикутників. Написати функцію обчислення довжини гіпотенузи цих трикутників. Порівняти і вивести яка з гіпотенуз більше, а яка менше.

Лістинг програми:

import math  
  
a = 0  
b = 0  
c = 0  
minimum = 1000000  
maximum = 0  
for i in range(2):  
 a = float(input('Введіть довжину катету(1): '))  
 while a <= 0:  
 print('Довжина катету <=0!')  
 a = float(input('Введіть довжину катету(1): '))  
 b = float(input('Введіть довжину катету(2):'))  
 while b <= 0:  
 print('Довжина катету <=0!')  
 b = float(input('Введіть довжину катету(2): '))  
 rez = math.pow(a, 2) + math.pow(b, 2)  
 c = round(math.sqrt(rez))  
 if c < minimum:  
 minimum = c  
 if c > maximum:  
 maximum = c  
 print("Гіпотенуза : ", c)  
print("Найбільша гіпотенуза: {0}\nНайменша гіпотенуза: {1}".format(maximum, minimum))

Результат виконання програми:

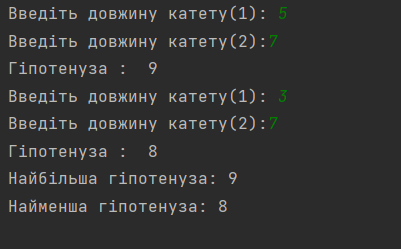


Рис. 2. Результат виконання програми

**Завдання 3:** Задано коло (x-a)2 + (y-b)2 = R2 і точки Р (р1, р2), F (f1, f1), L (l1, l2). З'ясувати і вивести на екран, скільки точок лежить всередині кола. Перевірку, чи лежить точка всередині кола, оформити у вигляді функції.

Лістинг програми

import math  
  
import time  
  
  
def timing(f):  
 def wrap(\*args):  
 time1 = time.time()  
 ret = f(\*args)  
 time2 = time.time()  
 print('%s функція зайняла %0.2f секунд' % (f.\_\_name\_\_, (time2 - time1) \* 100.0))  
 return ret  
  
 return wrap  
  
  
@timing  
def circle01(x, y, a, b, r):  
 one\_part = math.pow((float(x) - float(a)), 2)  
 two\_part = math.pow((float(y) - float(b)), 2)  
 third\_part = one\_part + two\_part  
 powradius = math.pow(r, 2)  
 if third\_part <= powradius:  
 info = 'Точка лежить у колі'  
 elif third\_part > powradius:  
 info = 'Точка не лежить у колі'  
 return info  
  
  
centerX = float(input('Введіть центральні точки\nВведіть точку x: '))  
centerY = float(input('Введіть точку y: '))  
R = float(input('Введіть радіус кола: '))  
tochka\_one = float(input('Точка (1): '))  
tochka\_two = float(input('Точка (2): '))  
rez = circle01(centerX, centerY, tochka\_one, tochka\_two, R)  
print(rez)

Результат виконання програми:

*Точка не лежить у колі:*

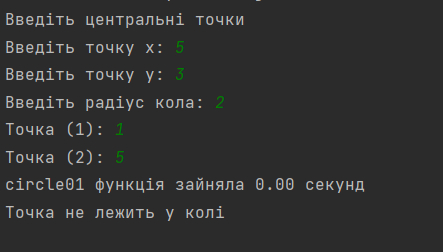


Рис. 3.1 Результат виконання програми

*Точка на колі:*

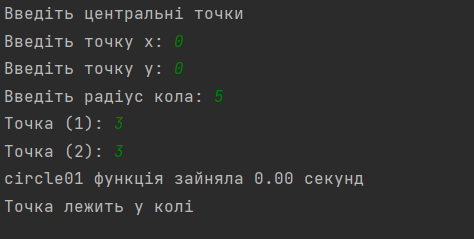


Рис. 3.2 Результат виконання програми

*Точка на колі:*

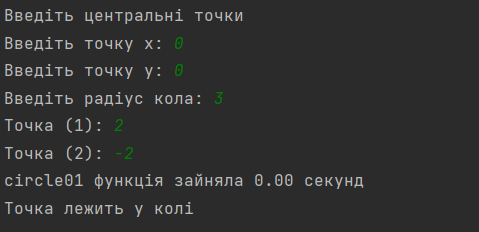


Рис. 3.3 Результат виконання програми

**Завдання 4:** Дано числа X, Y, Z, Т - довжини сторін чотирикутника. Обчислити його площу, якщо кут між сторонами довжиною X і Y - прямий. Використовувати дві підпрограми для обчислення площ: прямокутного трикутника і прямокутника.

Лістинг програми

import math  
  
  
def smallestTriangle(x, y): # s прямок.тр. і діагоналі  
 return round((x \* y) / 2, 2), math.sqrt(math.pow(x, 2) + math.pow(y, 2))  
  
  
def sTriangle(pivperiment, a1, b1, diagonal): # s трикутника  
 return math.sqrt((pivperiment - a1) \* (pivperiment - b1) \* (pivperiment - diagonal) \* pivperiment)  
  
  
A = 0  
B = 0  
C = 0  
D = 0  
A = float(input("Введіть сторону:"))  
B = float(input("Введіть сторону:"))  
C = float(input("Введіть сторону:"))  
D = float(input("Введіть сторону:"))  
while A <= 0:  
 print('Сторона <=0')  
 A = float(input('Введіть сторону: '))  
while B <= 0:  
 print('Сторона <=0')  
 B = float(input('Введіть сторону: '))  
while C <= 0:  
 print('Сторона <=0')  
 C = float(input('Введіть сторону: '))  
while D <= 0:  
 print('Сторона <=0')  
 D = float(input('Введіть сторону: '))  
s1Triangle, diagonal = smallestTriangle(B, A)  
print("Діагональ= ", round(diagonal, 2))  
pivperum = round((C + D + diagonal) / 2)  
print("Півпериметр= ", round(pivperum, 2))  
print("Площа меншого трикутника: ", round(s1Triangle, 2))  
SSStriangle = sTriangle(pivperum, C, D, diagonal)  
print("Площа іншого трикутника: ", round(SSStriangle, 2))  
rezzz = round(SSStriangle + s1Triangle, 2)  
print("Площа чотирикутника: ", rezzz)  
# 3 4 5 6 Вводити

Результат виконання програми:

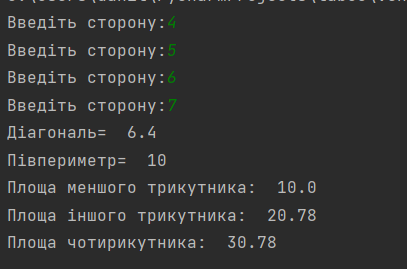


Рис. 4. Результат виконання програми

**Завдання 5:** Знайти всі натуральні числа, що не перевищують заданого n, які діляться на кожне із заданих користувачем чисел.

Лістинг програми

tmp = 1  
number = int(input('Введіть число : '))  
print('Для зупинки програми введіть "0" ')  
while tmp != 0:  
 dil = int(input('Введіть дільник числа: '))  
 if dil == 0:  
 break  
 if number % dil == 0:  
 print("Число є дільником числа")  
 else:  
 print("Число не є дільником числа")

Результат виконання програми:

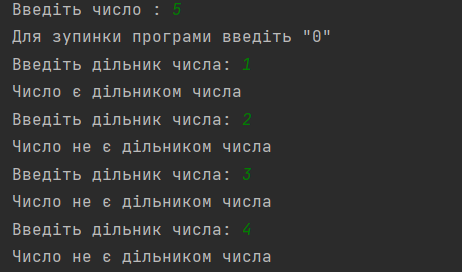


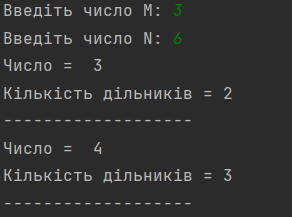
Рис. 5. Результат виконання програми

**Завдання 6:** Скласти програму для знаходження чисел з інтервалу [М, N], що мають найбільшу кількість дільників.

Лістинг програми

max\_NUM = 0  
tmp = 0  
M = int(input('Введіть число M: '))  
N = int(input('Введіть число N: '))  
for i in range(M, N + 1, 1):  
 tmp1 = 0  
 print('Число = ', i)  
 for j in range(1, i + 1, 1):  
 if int(i) % int(j) == 0:  
 tmp1 += 1  
 print('Кількість дільників = {0}\n-------------------'.format(tmp1))  
 if tmp1 > max\_NUM:  
 max\_NUM = tmp1  
 tmp = i  
print("Найбільше дільників у числа: {0}, їх кількість {1}".format(tmp, max\_NUM))

Результат виконання програми:



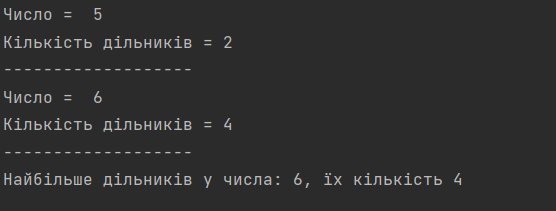


Рис. 6. Результат виконання програми

**Завдання 7:** Написати функцію для пошуку всіх простих чисел від 0 до N з можливістю вибору формату представлення результату (списком; рядками в стовпчик; просто вивести кількість простих чисел.

Лістинг програми:

import time  
  
  
def timing(f):  
 def wrap(\*args):  
 time1 = time.time()  
 ret = f(\*args)  
 time2 = time.time()  
 print('%s функція зайняла %0.2f секунд' % (f.\_\_name\_\_, (time2 - time1) \* 100.0))  
 return ret  
  
 return wrap  
  
  
def checkforproste(chuslo):  
 i = 2  
 while i < chuslo:  
 if chuslo % i == 0:  
 return False  
 i += 1  
 return True  
  
  
@timing  
def show\_info(number, element\_menu):  
 array = []  
 for i in range(2, number + 1):  
 if checkforproste(i):  
 array.append(i)  
 if element\_menu == 1:  
 print("Список: ")  
 print(array)  
 if element\_menu == 2:  
 print("Рядками у стовпчик: ")  
 for i in range(len(array)):  
 tmp = array[i]  
 print(tmp)  
 if element\_menu == 3:  
 print("Кількість простих чисел: ", len(array))  
  
  
number = int(input("Введіть границю N: "))  
rez = int(input("Введіть елемент меню в якому буде представлено результат:\n1. Список\n"  
 "2. Рядками у стовпчик:\n3. Кількість простих чисел\n"))  
show\_info(number, rez)

Результат виконання програми:

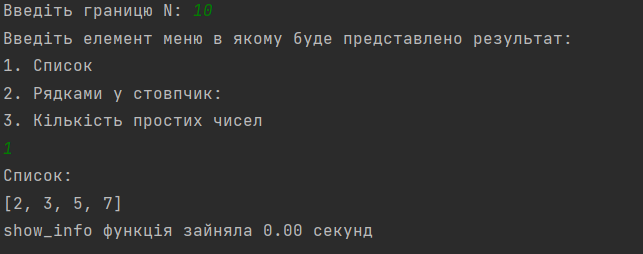


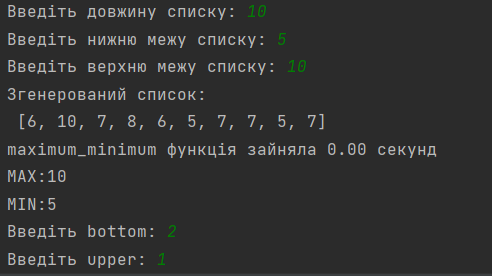
Рис. 7. Результат виконання програми

**Завдання 8:** Дано список з випадкових натуральних чисел довільної довжини. Написати програму, що формуватиме з заданого другий список, що міститиме тільки значення від MIN+bottom до MAX-upper. Де MIN і MAX – відповідно найменше і найбільше число в списку, а botton і upper – нижня і верхня межа значень вибірки нового списку. Програма має містити обробку винятків на випадок введення символів невірного типу, дробових чисел, вихід за межі мінімального і максимального значення.

Лістинг програми

import time  
import random  
  
  
def timing(f):  
 def wrap(\*args):  
 time1 = time.time()  
 ret = f(\*args)  
 time2 = time.time()  
 print('%s функція зайняла %0.2f секунд' % (f.\_\_name\_\_, (time2 - time1) \* 100.0))  
 return ret  
  
 return wrap  
  
  
@timing  
def maximum\_minimum(list\_randomly\_num):  
 return max(list\_randomly\_num), min(list\_randomly\_num)  
  
  
a1 = 0  
b1 = 0  
  
leanth\_list = int(input('Введіть довжину списку: '))  
while leanth\_list <= 0:  
 print("Введіть додатнє значення!")  
 leanth\_list = int(input('Введіть довжину списку: '))  
a1 = int(input("Введіть нижню межу списку: "))  
b1 = int(input("Введіть верхню межу списку: "))  
while a1 >= b1:  
 print("Верхня межа повинна бути більшою за нижню межу!")  
 a1 = int(input("Введіть нижню межу списку: "))  
 b1 = int(input("Введіть верхню межу списку: "))  
list = [random.randint(a1, b1) for i in range(leanth\_list)]  
print("Згенерований список:\n", list)  
max\_list, min\_list = maximum\_minimum(list)  
print('MAX:{0}\nMIN:{1}'.format(max\_list, min\_list))  
bottom = int(input("Введіть bottom: "))  
upper = int(input("Введіть upper: "))  
while min\_list + bottom >= max\_list - upper:  
 print('Ви вийшли за межі рандомізації! Введіть bottom та upper ще раз!')  
 bottom = int(input("Введіть bottom: "))  
 upper = int(input("Введіть upper: "))  
list2 = [random.randint(min\_list + bottom, max\_list - upper) for i in range(leanth\_list)]  
print("Згенерований новий список:\n", list2)  
max\_list2, min\_list2 = maximum\_minimum(list2)  
print('MAX (новий список):{0}\nMIN (новий список):{1}'.format(max\_list2, min\_list2))

Результат виконання програми:



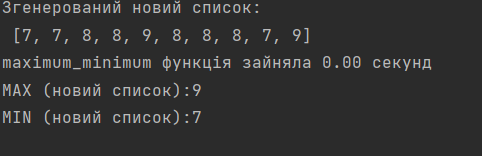


Рис. 8. Результат виконання програми

**Завдання 9:** Для завдань 6 – 8 написати декоратор, що дозволить визначати час виконання програми. Виконати перевірку часу виконання написаних функцій для 10\*\*n елементів при n <=6 з кроком в n. Тобто визначити час виконання функцій для десятків, сотень, тисяч і так до мільйону елементів.

Лістинг програми

def timing(f):  
 def wrap(\*args):  
 time1 = time.time()  
 ret = f(\*args)  
 time2 = time.time()  
 print('%s функція зайняла %0.2f секунд' % (f.\_\_name\_\_, (time2 - time1) \* 100.0))  
 return ret  
  
 return wrap

***Висновки:*** ознайомилися основами функціонального програмування і використання користувацьких функцій в мові Python