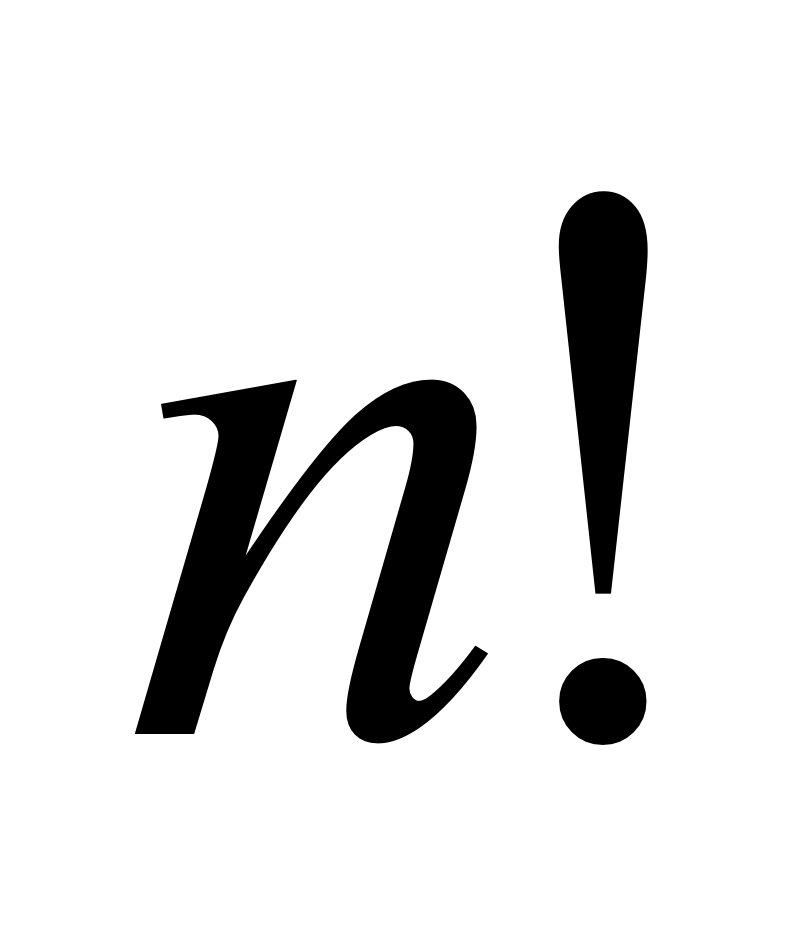
**Практичне заняття 5-6**

**Кобець Анастасія 331 ФКНФМ**

**Задача 3 1. Факторіал**

***Задача 1.*** *Знайти число нулів, якими закінчується число у десятковій системі числення.*

# Введення числа

num = int(input("Введіть число: "))

# Ініціалізація лічильника нулів

count\_zeros = 0

# Ініціалізація факторіалу

factorial\_result = 1

# Розкладання числа на прості множники

i = 5

while num // i > 0:

count\_zeros += num // i

i \*= 5

# Обчислення факторіалу та виведення результату

for j in range(1, num + 1):

factorial\_result \*= j

# Виведення результату

print(f"{num}! = {factorial\_result}")

print(f"Кількість нулів в десятковій системі числення на кінці {num}! = {count\_zeros}")

*Результатом виконання буде:*

Введіть число: 123

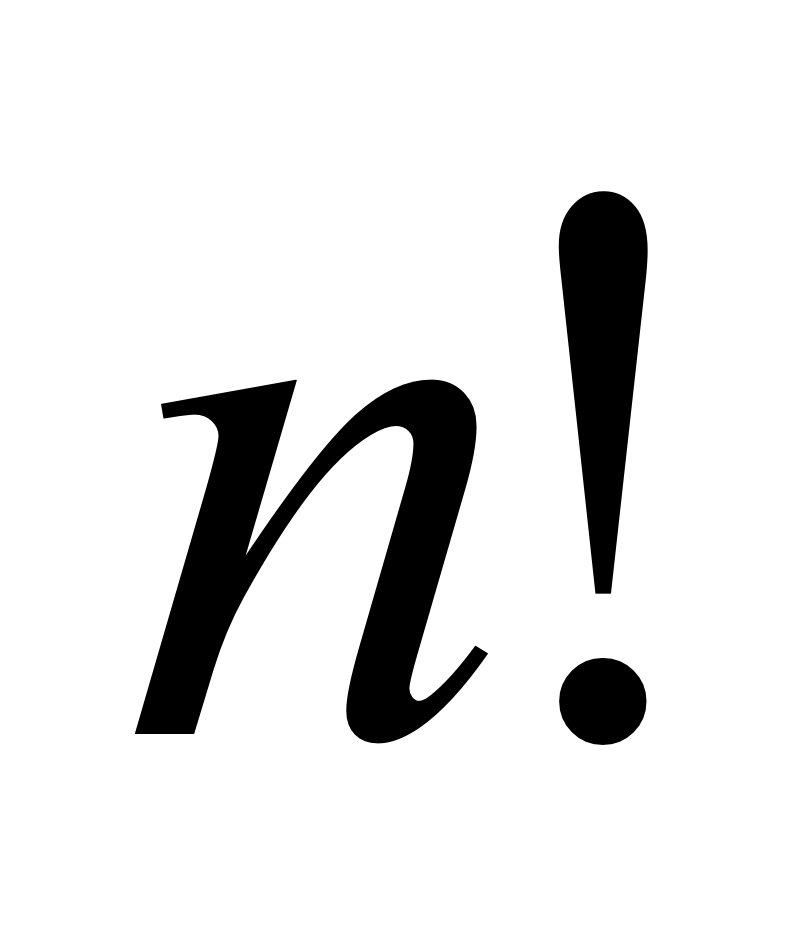
123! = 12146304367025329675766243241881295855454217088483382315328918161829235892362167668831156960612640202170735835221294047782591091570411651472186029519906261646730733907419814952960000000000000000000000000000

Кількість нулів в десятковій системі числення на кінці 123! = 28

Введіть число: 6

6! = 720

Кількість нулів в десятковій системі числення на кінці 6! = 1

***Задача 2.*** *Знайти молодшу не рівну нулю цифру числа  у десятковій системі числення.*

# Введення числа

num = int(input("Введіть число: "))

# Ініціалізація факторіалу та молодшої ненульової цифри

factorial\_result = 1 # Початкове значення факторіалу

last\_nonzero = None # Ініціалізація молодшої ненульової цифри

# Розкладання числа на прості множники для обчислення факторіалу

for i in range(1, num + 1):

factorial\_result \*= i

# Знаходження молодшої ненульової цифри

temp\_result = factorial\_result # тимчасова змінна для збереження значення факторіалу

while temp\_result > 0:

digit = temp\_result % 10 # Отримання молодшої цифри

if digit != 0:

last\_nonzero = digit # Збереження молодшої ненульової цифри

break

temp\_result //= 10 # Перехід до наступного розряду

# Виведення самого факторіалу та результату

print(f"{num}! = {factorial\_result}")

print(f"Молодша ненульова цифра у десятковій системі числення {num}! = {last\_nonzero}")

*Результатом виконання буде:*

Введіть число: 123

123! = 12146304367025329675766243241881295855454217088483382315328918161829235892362167668831156960612640202170735835221294047782591091570411651472186029519906261646730733907419814952960000000000000000000000000000

Молодша ненульова цифра у десятковій системі числення 123! = 6

Введіть число: 6

6! = 720

Молодша ненульова цифра у десятковій системі числення 6! = 2

**Задача № 4.1. Діагональ прямокутника**

def gcd(a, b):

# Знаходження найбільшого спільного дільника

while b:

a, b = b, a % b

return a

def count\_cells\_cut\_by\_diagonal(M, N):

# Знаходження найбільшого спільного дільника для розмірів прямокутника

greatest\_common\_divisor = gcd(M, N)

# Кількість клітинок, які розрізає діагональ, визначається за формулою

cells\_cut\_by\_diagonal = M + N - greatest\_common\_divisor

return cells\_cut\_by\_diagonal

# Введення розмірів прямокутника

M = int(input("Введіть довжину прямокутника M: "))

N = int(input("Введіть ширину прямокутника N: "))

# Перевірка введених значень

if M <= 0 or N <= 0:

print("Невірні вхідні дані. Введіть натуральні числа більше 0.")

else:

# Отримання кількості клітинок, які розрізає діагональ

result = count\_cells\_cut\_by\_diagonal(M, N)

# Виведення результату

print(f"Діагональ розрізає {result} клітинок.")

*Результатом виконання буде:*

Введіть довжину прямокутника M: 4

Введіть ширину прямокутника N: 6

Діагональ розрізає 8 клітинок.

**Задача № 4.2 Діагональ паралелепіпеда**

def gcd(a, b):

# Знаходження найбільшого спільного дільника

while b:

a, b = b, a % b

return a

def count\_cubes\_cut\_by\_diagonal(K, M, N):

# Знаходження найбільшого спільного дільника для розмірів переллелепіпеда

gcd\_k\_m = gcd(K, M)

gcd\_k\_n = gcd(K, N)

gcd\_n\_m = gcd(N, M)

gcd\_k\_m\_n = gcd(gcd\_k\_m, N)

# Кількість кубіків, які розрізає діагональ, визначається за формулою включень-исключень

cubes\_cut\_by\_diagonal = K + M + N - (gcd\_k\_m + gcd\_k\_n + gcd\_n\_m) + gcd\_k\_m\_n

return cubes\_cut\_by\_diagonal

# Введення розмірів переллелепіпеда

K = int(input("Введіть довжину переллелепіпеда K: "))

M = int(input("Введіть ширину переллелепіпеда M: "))

N = int(input("Введіть висоту переллелепіпеда N: "))

# Перевірка введених значень

if K <= 0 or M <= 0 or N <= 0:

print("Невірні вхідні дані. Введіть натуральні числа більше 0.")

else:

# Отримання кількості кубіків, які розрізає діагональ

result = count\_cubes\_cut\_by\_diagonal(K, M, N)

# Виведення результату

print(f"Діагональ розрізає {result} кубіків.»)

*Результатом виконання буде:*

Введіть довжину переллелепіпеда K: 4

Введіть ширину переллелепіпеда M: 6

Введіть висоту переллелепіпеда N: 4

Діагональ розрізає 8 кубіків

**Задача Сума чисел.** Дано набір натуральних чисел x1, x2, … xn та натуральне число А. Знайти серед них такі два числа xi xj такі, що xi + xj = A чи показати що такої пари не існує

def find\_all\_pairs\_with\_sum(numbers, sum):

pairs = []

for i in range(len(numbers)):

for j in range(i + 1, len(numbers)):

if numbers[i] + numbers[j] == sum:

pairs.append((numbers[i], numbers[j]))

return pairs

# Введення набору чисел та цільової суми

numbers\_str = input("Введіть натуральні числа через пробіл: ")

numbers = list(map(int, numbers\_str.split()))

sum = int(input("Введіть цільову суму A: "))

# Знаходження всіх пар чисел

result\_pairs = find\_all\_pairs\_with\_sum(numbers, sum)

# Виведення результату

if result\_pairs:

print(f"Знайдені пари чисел, які утворюють суму {sum}:")

for pair in result\_pairs:

print(f"{pair[0]} та {pair[1]}")

else:

print(f"Таких пар чисел, які утворюють суму {sum}, не існує.")

*Результатом виконання буде:*

Введіть натуральні числа через пробіл: 3 4 5 6 7 7 7 8 9 9 7 6 65 5 4

Введіть цільову суму A: 15

Знайдені пари чисел, які утворюють суму 15:

6 та 9

6 та 9

7 та 8

7 та 8

7 та 8

8 та 7

9 та 6

9 та 6