**Практичне заняття 1,2**

**Кобець Анастасія 331 ФКНФМ**

**Задача 1**. Числа Фібоначчі

# Ініціалізуємо перші два числа Фібоначчі

a, b = 1, 1

index = 1

# Починаємо цикл, який триває, доки не знайдемо потрібне число

while True:

# Обчислюємо наступне число Фібоначчі

fib = a + b

# Оновлюємо значення a та b для наступної ітерації

a, b = b, fib

# Збільшуємо лічильник індексу

index += 1

# Перевіряємо, чи число Фібоначчі закінчується на "0000"

if fib % 10000 == 0:

# Якщо так, виходимо з циклу

break

# Виводимо номер числа Фібоначчі та саме число

print(f"The Fibonacci number that ends with 0000 is {fib}, found at index {index}")

Результатом виконання буде:

The Fibonacci number that ends with 0000 is 11423965231520587047220488928656904198487186633317560797959030595738263643588305263964321080516991429937628886229555340146644442744473185460778302934743807002248109695741208782411159189994651520930091202035101269350523609417276542209682261168150544790025062794209091503702088574338650460569295592498666443239807989522593072562158640947468656887645879356201301594841872491497556389555817277508349058330498007583814270123329724353233156029127910968370052734811192660492733375394472692191584489489590970254440914222778382439339334175624660291588778456250479185237898309112318829984358216337347549014336517486496643224502773380042071174360597192343056318489287038447004730922073980870072990706067508624038407888471294048912294153491398930715643640170172837379127969101176561450586945715460276780809807889664272818316865711724985646554559305334340318994612185260719042008960311269000122672589731283419608098303367260382379660402261886574952211783683104453334281684425994447306306414660032519055079504313562694958935754118796157632978970220780288168992181699708922971417067735144929461193639081445200786881549331150381216073705417531166786634690469206418611524663013854198045284806720735273715046888704916821855277543026346215355286395854263168251068150374988851620501196943905031285049077628443804052134507022504682483293396215268186620124762379744668092166035314553541731537245946256422861852573006230492322259630342294350827184840607509969289328320360093204783447860955806396350723341261564285649453007949089154165288839814442677339344794691881510389855765582716774490000,

found at index **7499**

**Задача 2**. Доміно і ейлерові шляхи у графах

# Запитуємо користувача про кількість костяшок доміно

n = int(input("Введіть кількість костяшок доміно: "))

# Ініціалізація масиву ступенів D

D = [0] \* n

# Введіть ступені вершин костяшок доміно

for i in range(n):

domino\_degrees = input(f"Введіть ступені вершин {i + 1}-ої костяшки (через пробіл): ").split()

D[i] = sum(map(int, domino\_degrees))

OddDeg = 0 # Кількість вершин з непарними ступенями

# Аналіз масиву ступенів

for j in range(n):

if D[j] % 2 == 1:

OddDeg += 1

# За теоремою Ейлера, можливість утворення цепочки визначається за кількістю вершин

# з непарними ступенями - це повинно бути 0 або 2.

if OddDeg == 2:

print("Можливо утворити цепочку з костяшок доміно.")

else:

print("Неможливо утворити цепочку з цього числа костяшок доміно.")

Результатом виконання буде:

Введіть кількість костяшок доміно:

5

Введіть ступені вершин 1-ої костяшки (через пробіл):

3

Введіть ступені вершин 2-ої костяшки (через пробіл):

3

Введіть ступені вершин 3-ої костяшки (через пробіл):

2

Введіть ступені вершин 4-ої костяшки (через пробіл):

2

Введіть ступені вершин 5-ої костяшки (через пробіл):

2

*Можливо утворити цепочку з костяшок доміно.*

**Задача 3.** Задача про голандський національний прапор

# Вхідний масив з кульками

colors = [2, 0, 2, 1, 1, 0]

# Ініціалізуємо вказівники для кольорів

blue, white, red = 0, 0, len(colors) - 1

while white <= red:

if colors[white] == 0:

# Якщо зустріли синій кольор, міняємо його зі значенням на позиції blue

# та збільшуємо вказівник для синього кольору.

colors[blue], colors[white] = colors[white], colors[blue]

blue += 1

white += 1

elif colors[white] == 1:

# Якщо зустріли білий кольор, просто збільшуємо вказівник для білого кольору.

white += 1

else:

# Якщо зустріли червоний кольор, міняємо його зі значенням на позиції red

# та зменшуємо вказівник для червоного кольору.

colors[white], colors[red] = colors[red], colors[white]

red -= 1

# Результат виводимо після сортування

print(colors)

Результатом виконання буде: [0, 0, 1, 1, 2, 2]