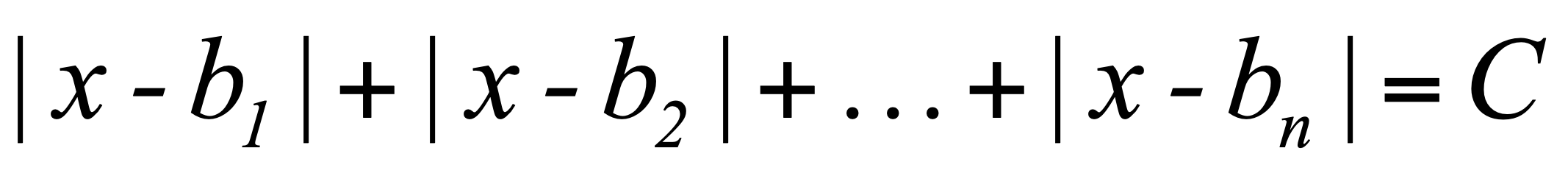
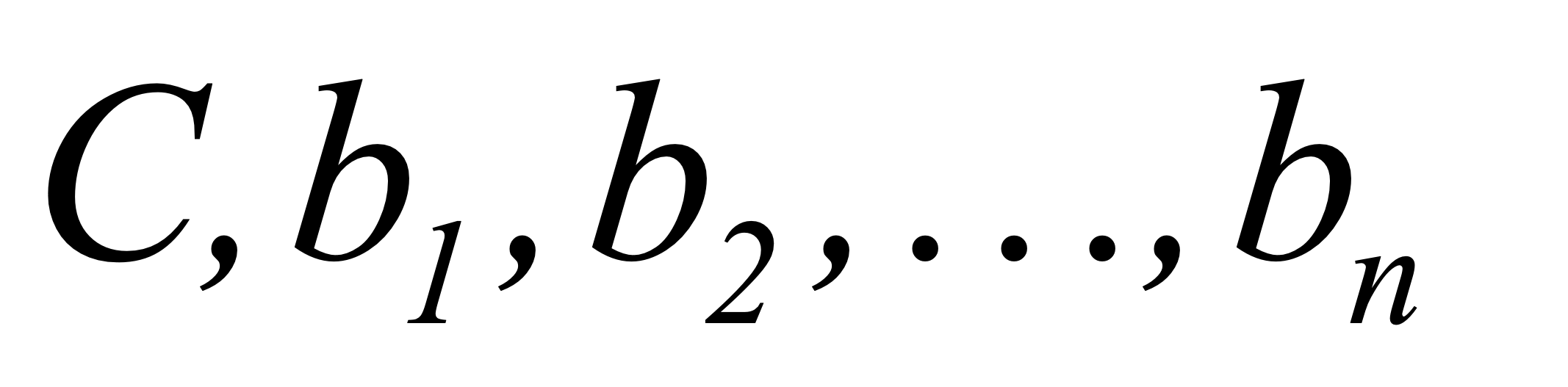
**Практичне заняття 9-10**

**Кобець Анастасія 331 ФКНФМ**

**Завдання 11.1. Лінійне рівняння з модулями**

*Написати програму, яка вирішує рівняння виду****,*** *де* ***-*** *цілі числа.*

coefficients = [1, 5, 2, 3]

target\_sum = 12

coefficients = sorted(coefficients)

n = len(coefficients)

result = []

# Виключення значень за межами інтервала

result.append((sum(coefficients) - target\_sum) / n)

if result[0] > coefficients[0]:

result.pop()

result.append((sum(coefficients) + target\_sum) / n)

if result[-1] < coefficients[-1]:

result.pop(-1)

# Пошук рішень в інтервалах, що залишились

for i in range(n - 1):

d = n - 2 \* (i + 1)

if d != 0:

solution = (sum(coefficients[i + 1:]) - sum(coefficients[:i + 1]) - target\_sum) / d

if coefficients[i] <= solution <= coefficients[i + 1]:

result.append(solution)

if result:

print(f'Solutions: {result}')

else:

print('No solution')

*Результатом виконання буде:*

Solutions: [-0.25, 5.75]

**Завдання 15. Правильно побудовані вирази.**

*Арифметичний вираз складено з однолітерних змінних знаків операцій +, -, \*, / і круглих дужок. Складіть програму, яка перевіряє дотримання синтаксису у цьому виразі.*

def expression():

global i, sym

term()

while sym in {'+', '-'}:

next\_symbol()

term()

def term():

global i, sym

factor()

while sym in {'\*', '/'}:

next\_symbol()

factor()

def factor():

global i, sym

if sym.isalpha():

next\_symbol()

elif sym == '(':

next\_symbol()

expression()

if sym == ')':

next\_symbol()

else:

error()

else:

error()

def error():

print('Terrible Error')

exit()

def next\_symbol():

global i, sym

i += 1

sym = A[i]

print(sym, end=' ')

A = input('Input expression: ')

i = 0

sym = A[i]

expression()

if sym == '.':

print('OK')

else:

print('Uncorrect End of Expression’)

*Результатом виконання буде:*

Input expression: a+b\*c-(d/e).

a + b \* c - ( d / e ) . OK

**Задача Алгебра множин.** *Нехай L1 та L2 – послідовності, елементи яких впорядковані за зростанням. Розглядаючи ці послідовності як множини, знайти:*

*Перетин L1 \* L2 = L.*

*Об’єднання L1 + L2 = L*

*Різницю L1 - L2 = L*

*де L впорядкована за зростанням.*

def intersection(L1, L2):

result = []

i = 0

j = 0

while i < len(L1) and j < len(L2):

if L1[i] == L2[j]:

result.append(L1[i])

i += 1

j += 1

elif L1[i] < L2[j]:

i += 1

else:

j += 1

return result

def union(L1, L2):

result = []

i = 0

j = 0

while i < len(L1) and j < len(L2):

if L1[i] == L2[j]:

result.append(L1[i])

i += 1

j += 1

elif L1[i] < L2[j]:

result.append(L1[i])

i += 1

else:

result.append(L2[j])

j += 1

while i < len(L1):

result.append(L1[i])

i += 1

while j < len(L2):

result.append(L2[j])

j += 1

return result

def difference(L1, L2):

result = []

i = 0

j = 0

while i < len(L1) and j < len(L2):

if L1[i] == L2[j]:

i += 1

j += 1

elif L1[i] < L2[j]:

result.append(L1[i])

i += 1

else:

j += 1

while i < len(L1):

result.append(L1[i])

i += 1

return result

# Приклад використання:

L1 = [1, 3, 5, 7, 9]

L2 = [3, 6, 7, 8, 9]

intersection\_result = intersection(L1, L2)

union\_result = union(L1, L2)

difference\_result = difference(L1, L2)

print("Перетин:", intersection\_result)

print("Об'єднання:", union\_result)

print("Різниця:", difference\_result)

*Результатом виконання буде:*

Перетин: [3, 7, 9]

Об'єднання: [1, 3, 5, 6, 7, 8, 9]

Різниця: [1, 5]