МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное образовательное учреждение высшего образования «Санкт-петербургский государственный морской технический университет»

ФАКУЛЬТЕТ ЦИФРОВЫХ ПРОМЫШЛЕННЫХ ТЕХНОЛОГИЙ

Кафедра Киберфизических систем

ЛАБОРАТОРНАЯ РАБОТА №1

По дисциплине «Программирование»

Выполнил: Зорин П. С.

Проверил:

Санкт-Петербург

2024

* **Цели и формулировка задачи**
* Работа с файлами,
* Работа с несколькими функциями, а не только с main,
* Реализации рекурсивной функции.
* **Результаты работы**

Создает файл output.txt, в который записывает ответ

* **Ход работы**

Программа предназначена для поиска выражения, составленного списка чисел(X1...Xn), которое в результате вычисления дает ответ(target). Начинается выполнение с чтения входных данных:N - кол-во чисел, X1-Xn - сами числа, S - число, которое долнжо получиться. Входные данные преобразуются в список, который затем передается в рекурсивную функцию для поиска решения. Функция find\_expression исследует все возможные комбинации чисел, используя операции сложения и вычитания. Она принимает параметры: текущий индекс, строку выражения и текущее значение, которое обновляется при каждом вызове функции. Как только рекурсия достигает конца списка чисел, программа проверяет, совпадает ли текущее значение с S. Если найдено подходящее выражение, оно возвращается.

* **Демонстрация работы программы**

Ввод данных:



Результат:



* **Листинг кода**

def find\_expression(numbers, target, index=1, current\_expression=None, current\_value=None):  
 if current\_expression is None:  
 current\_expression = str(numbers[0])  
 if current\_value is None:  
 current\_value = numbers[0]  
  
 # Если дошли до конца списка  
 if index == len(numbers):  
 if current\_value == target:  
 return current\_expression + "=" + str(target)  
 return None  
  
 # Номер с +  
 result = find\_expression(numbers, target, index + 1,  
 current\_expression + "+" + str(numbers[index]),  
 current\_value + numbers[index])  
 if result:  
 return result  
  
 # Номер с -  
 result = find\_expression(numbers, target, index + 1,  
 current\_expression + "-" + str(numbers[index]),  
 current\_value - numbers[index])  
 if result:  
 return result  
  
 return None  
  
# Чтение входных данных  
data = input("N, X1...Xn, S: ").strip().split()  
N = int(data[0])  
numbers = list(map(int, data[1:N+1]))  
S = int(data[N+1])  
  
# Начальная точка рекурсии  
result = find\_expression(numbers, S)  
  
# Запись результата в файл  
with open('output.txt', 'w') as f:  
 if result:  
 f.write(result + '\n')  
 else:  
 f.write("no solution\n")

**ЗАКЛЮЧЕНИЕ**

**Список использованных источников**