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

По дисциплине «Программирование»

Выполнил:

Кузьмин Кирилл Алексеевич гр. 20121

Проверил:

Санкт-Петербург

2024

Оглавление

[1. Цели и формулировка задачи 3](#_Toc187676633)

[2. Результаты работы 4](#_Toc187676634)

[2.1. Реализация программы с использованием функционального программирования языка Python 4](#_Toc187676635)

[2.1.1. Ход работы 4](#_Toc187676636)

[2.1.2. Демонстрация работы программы 4](#_Toc187676637)

[2.1.3. Листинг кода 5](#_Toc187676638)

[ЗАКЛЮЧЕНИЕ 6](#_Toc187676639)

# Цели и формулировка задачи

При работе над данной лабораторной работой будут затронуты следующие вопросы:

1. Работа с файлами,
2. Работа с несколькими функциями, а не только с main,
3. Реализации рекурсивной функции.

# Результаты работы

## **Реализация программы с использованием функционального программирования языка Python**

## Ход работы

1. Считываю данные из файла в список data
2. Вхожу в рекурсию, принимающую параметры: numbers, target, index, current\_expression и current\_sum .

numbers – набор чисел  
target – искомая сумма  
index – индекс числа, которому мы будем менять знак сurrent\_sum – сумма на текущем уровне рекурсии

current\_expression – строка с выражением на текущем уровне рекурсии

1. Точка остановки рекурсии – index == len(numbers), если сurrent\_sum == target, то возвращает строку, изначальных чисел с изменёнными знаками, которые дают в результате суммирования искомое значение.

Иначе рекурсия закончилась, не найдя ответ. Возвращает “no solution” – решений нет

1. Тело рекурсии – Беру число по index. Пробую добавить + или – и перехожу на следующий уровень рекурсии. Если новая сумма не соответствует искомому числу, то возвращаюсь на предыдущий уровень рекурсии пока не проверятся все варианты.
2. По окончании работы рекурсии переменная result содержит в себе строку “no solution”, либо арифметическое выражение.
3. Записываем строку result в файл output.txt.

## Демонстрация работы программы

## 

## Пример входных данных:

8 2 3 4 6 4 1 6 7 1

## Вывод данных в файл:

2+3+4-6-4+1-6+7=1

## Пример входных данных:

8 2 3 4 6 4 1 6 7 100

## Вывод данных в файл:

no solution

## Листинг кода

from typing import List, Optional

#Рекурсивно находит выражение, используя + или -,равное требуемому значению.

def find\_expression(numbers: List[int], target: int, index: int = 0, current\_expression: str = "", current\_sum: int = 0) -> Optional[str]:

    # Базовый случай: если мы достигли конца списка

    # Возращает строку выражения, если она найдена, в противном случае - None.

    if index == len(numbers):

        if current\_sum == target:

            return current\_expression

        else:

            return None

    # Текущее число

    number: int = numbers[index]

    # Пробуем добавить знак "+" перед текущим числом

    result\_with\_plus = find\_expression(numbers, target, index + 1,

                                       current\_expression + (f"+{number}" if current\_expression else f"{number}"),

                                       current\_sum + number)

    if result\_with\_plus is not None:

        return result\_with\_plus

    # Пробуем добавить знак "-" перед текущим числом

    result\_with\_minus = find\_expression(numbers, target, index + 1,

                                        current\_expression + f"-{number}",

                                        current\_sum - number)

    if result\_with\_minus is not None:

        return result\_with\_minus

    return None

# Чтение входных данных из файла; N-количество чисел, numbers-список чисел, S-цель

with open("input.txt", "r") as f:

    data = list(map(int, f.read().strip().split()))

N: int = data[0]

numbers: List[int] = data[1:N + 1]

S: int = data[N + 1]

# Поиск выражения с помощью рекурсивной функции find\_expression

result: Optional[str] = find\_expression(numbers, S)

# Запись результата в файл

with open("output.txt", "w") as f:

    if result is not None:

        f.write(f"{result}={S}\n")

    else:

        f.write("no solution\n")

# ЗАКЛЮЧЕНИЕ

Во время работы я получил базовые навыки в работе с рекурсией. Научился считывать из файла и записывать в файл.