|  |  |
| --- | --- |
| Picture 1 | **МИНОБРНАУКИ РОССИИ**  **федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  «Санкт-Петербургский государственный морской технический университет» (СПбГМТУ) |

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Факультет цифровых промышленных технологий

Направление подготовки 09.03.01.03 "Интеллектуальные технологии киберфизических систем"

**«Лабораторная работа №»**

Студент 1 курса группы 20121

Очного отделения

Руденко В. С.

Проверил:

Поделенюк П. П.

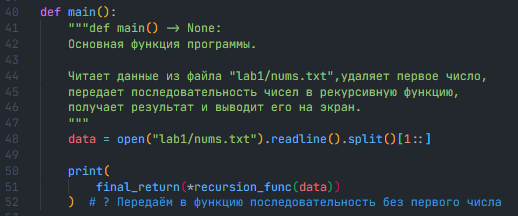
2023

# Цель работы

1. Научиться работать с файлами.
2. Научиться работать с несколькими функциями, а не только с main.
3. Реализовать рекурсивную функцию.

# Ход работы

1. В функции main мы считываем данные из файла. Затем убираем первое число из файла, потому что оно нам не нужно:

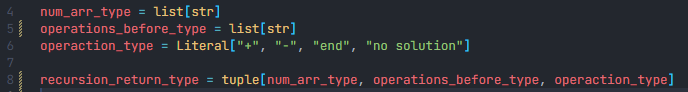


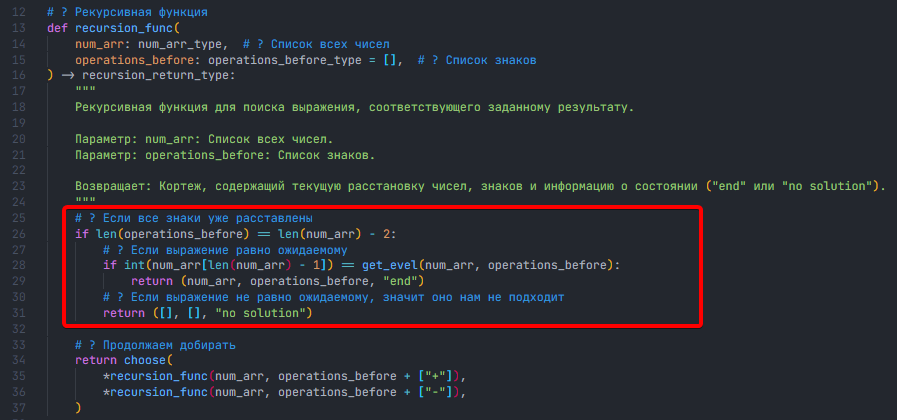
1. Создаём функцию “recursion\_func”

1 параметр функции: num\_arr, который имеет тип “ list[str]”.

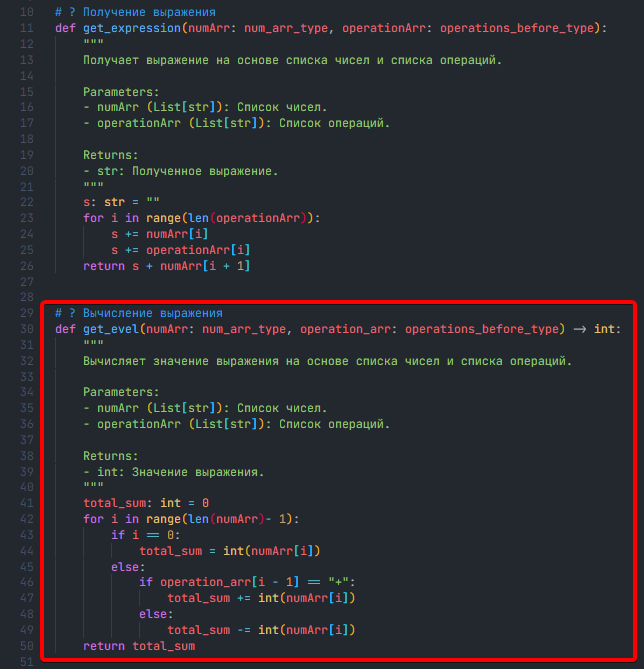
2 параметр функции: operations\_before, который по стандарту пустой. Он имеет тип list[str].

Функция возвращает: num\_arr, operations\_before, operaction c типом “recursion\_return\_type”.

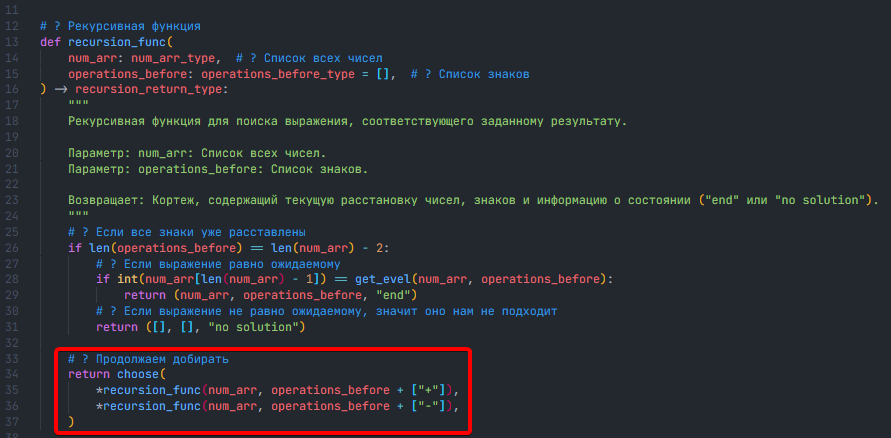


1. В функции сначала делаем проверку набралось ли у нас уже достаточно операций для подстановки их в выражение. Если да, то сравниваем значения выражения с накопленными операторами и числа, которое должно у нас получиться. Если они равны, то возвращаем набор чисел, все операторы и “end”, чтоб указать, что решение нашлось. Если они не равны, то решения нет, значит возвращаем пустой список чисел, пустой список операций и “no solution”, дабы обозначить, что решения нет. 

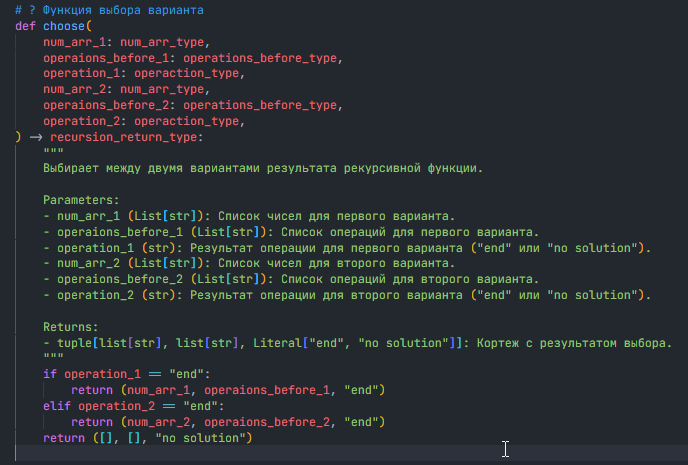
Функция get\_evel считает значение выражения.



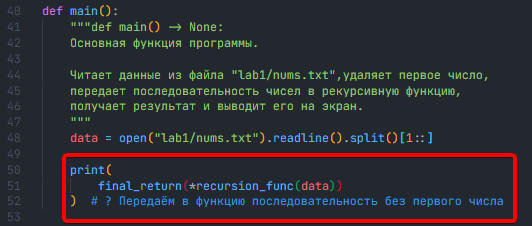
1. Если знаков не хватает, то продолжаем добирать знаки в массив вторым параметром.

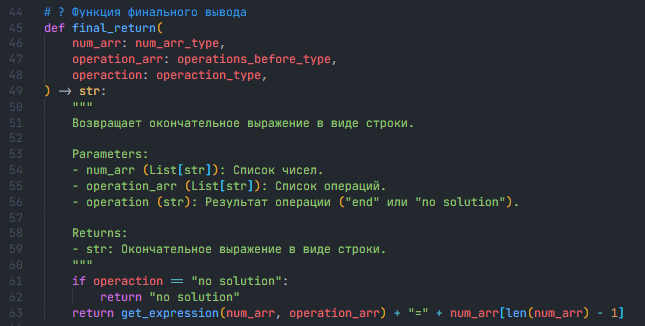


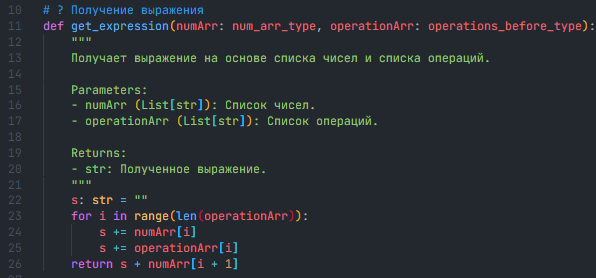
Функция choose выбирает вариант, который подходит из двух. Если ни одиин вариант не подходит, то возвращает “no solution”.



1. В конце “recursion\_func” возвращает: набор чисел, набор операций и “end” если решение нашлось и “no solution” если решение не нашлось.
2. Данные из “recursion\_func” попадают в функцию “final\_return”, которая возвращает “no solution” либо выражение в консоль





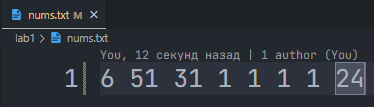


# Результаты работы

## Набор данных №1

Файл nums.txt:

“6 51 31 1 1 1 1 24”



Вывод программы:

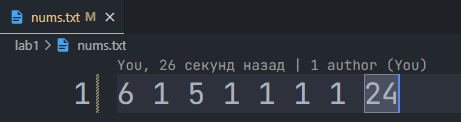
“51-31+1+1+1+1=24”



## Набор данных №2

Файл nums.txt:

“6 1 5 1 1 1 1 24”



Вывод программы:

“no solution”



# Вывод

В ходе выполнения данной лабораторной работы мы успешно решали несколько ключевых задач, связанных с программированием на языке Python. В частности, были затронуты следующие важные аспекты:

1. Работа с файлами: Мы ознакомились с основными операциями по работе с файлами в Python, включая открытие, чтение, запись и закрытие файлов. Этот опыт позволяет нам более эффективно управлять вводом и выводом данных в наших программах.
2. Работа с несколькими функциями: в процессе написания кода мы уделяли внимание структурированию программы с использованием нескольких функций, а не ограничивались только функцией main. Это позволяет создавать более модульные и легко поддерживаемые программы.
3. Реализация рекурсивной функции: Мы успешно разработали рекурсивную функцию, что дает нам возможность элегантно решать задачи, которые могут быть разбиты на более мелкие подзадачи. Рекурсия предоставляет нам инструмент для более гибкого и эффективного программирования.

Эта лабораторная работа предоставила нам не только теоретические знания, но и практические навыки по применению вышеописанных концепций в реальных программных проектах. Полученные знания о работе с файлами, использовании нескольких функций и реализации рекурсивных алгоритмов являются важным шагом в повышении наших навыков программирования на языке Python.