ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОМСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра «Информатика и вычислительная техника»

Отчёт по лабораторной работе № 3

по дисциплине

«Проектирование и тестирование программного обеспечения»

Выполнил:

Студент гр. ПИН-222

Лубенский И. В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подп., дата)

Проверил:

Старший преподаватель каф. ИВТ

Блохин А. В. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подп., дата)

Омск, 2024

**ОПИСАНИЕ ЗАДАЧИ “ПОПАРНО”**

Любой набор из n целых чисел образует n(n - 1)/2 сумм если сложить все возможные пары. Ваша задача состоит в том, чтобы найти n целых чисел по заданному набору сумм.

**Входные данные**

Каждая строка входных данных содержит n, за которым следуют n(n- 1)/2 целых чисел, разделенных пробелами, причем 2 < n < 10.

**Выходные данные**

Для каждой строки входных данных выведите одну строку, содержащую n целых чисел в неубывающем порядке таких, что входные числа - это попарные суммы этих n чисел. Если существует более одного решения, то подойдет любое. Если решения не существует, выведите "Impossibleeee".

**Пример входных данных**

3 1269 1160 1663

3 1 1 1

5 226 223 225 224 227 229 228 226 225 227

5 216 210 204 212 220 214 222 208 216 210

5 -1 0 -1 -2 1 0 -1 1 0 -1

5 79950 79936 79942 79962 79954 79972 79960 79968 79924 79932

**Соответствующие выходные данные**

383 777 886

Impossibleeee

111 112 113 114 115

101 103 107 109 113

-1 -1 0 0 1

39953 39971 39979 39983 39989

**АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ “ ПОПАРНО”**

Схема алгоритма решения поставленной задачи показана на рисунке 1.

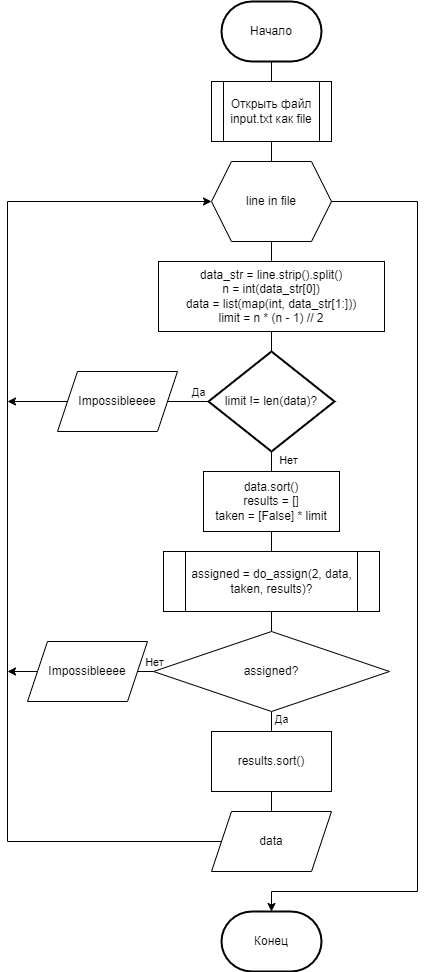


Рисунок 1 — Схема составленного алгоритма

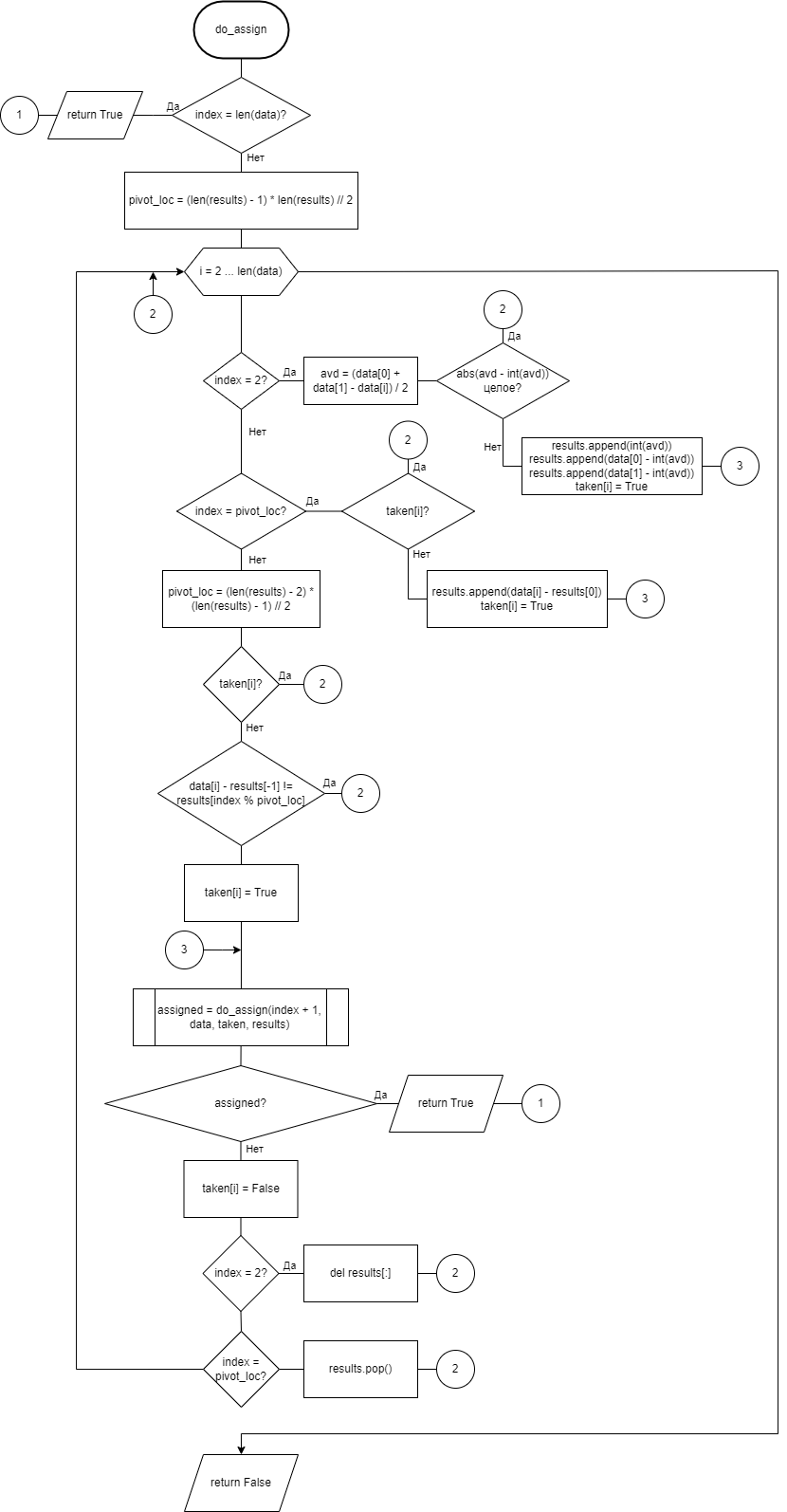


Рисунок 1.2 — Схема оставленного алгоритма

Исходный код программы показан на рисунке 2.

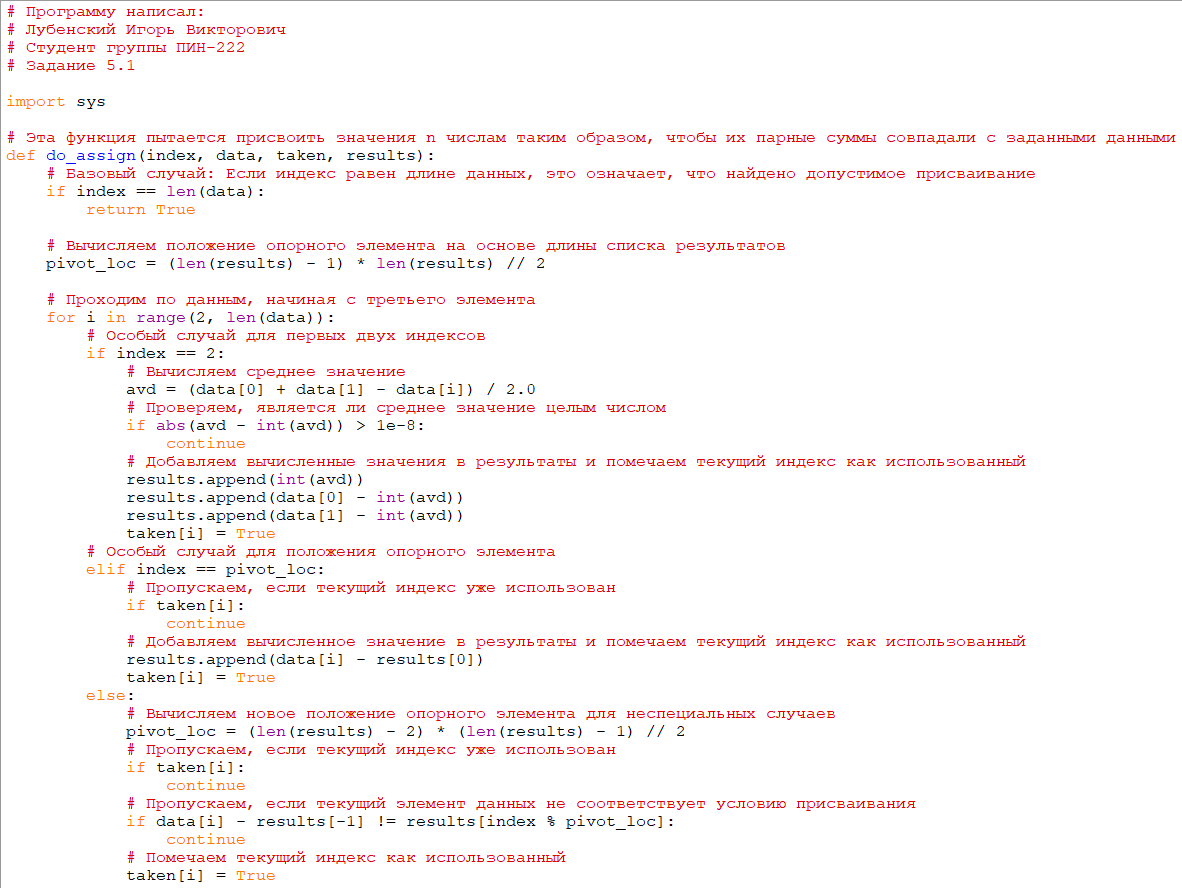


Рисунок 2 — Исходный код для решения задачи

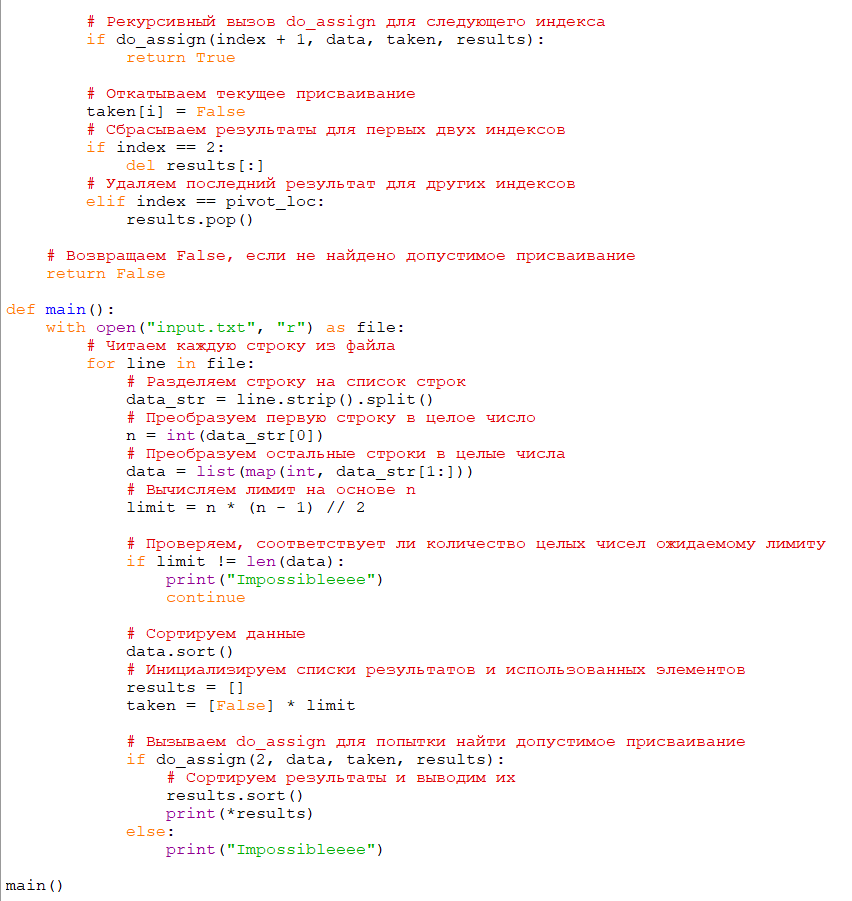


Рисунок 2.2 — Исходный код для решения задачи

Данный код решает задачу поиска всех уникальных комбинаций чисел в массиве, сумма которых равна заданному числу. Он использует рекурсивный подход для обхода всех возможных комбинаций чисел, проверяя каждую комбинацию на соответствие заданному условию.

В начале работы программы, в функции main, происходит чтение входных данных из файла. Для каждой строки файла выполняются следующие шаги:

- строка разбивается на список строк, где первая строка преобразуется в целое число n, а остальные строки преобразуются в целые числа и сохраняются в списке data;

- вычисляется лимит на основе n, который определяет количество чисел в комбинации;

- проверяется, соответствует ли количество чисел в data ожидаемому лимиту. Если нет, выводится сообщение "Impossibleeee", и программа переходит к следующей строке файла;

- eсли количество чисел соответствует лимиту, выполняется сортировка массива data;

- инициализируются списки results для хранения найденных комбинаций и taken для отслеживания использованных чисел;

- вызывается функция do\_assign для поиска комбинаций, сумма которых равна заданному числу.

Функция do\_assign реализует рекурсивный алгоритм для поиска комбинаций. Она принимает текущий индекс, массив данных, список использованных чисел и список результатов. В начале работы функции проверяется базовый случай: если индекс равен длине данных, это означает, что найдена допустимая комбинация, и функция возвращает True.

В процессе работы функции происходит обход данных, начиная с третьего элемента. Для каждого элемента проверяются условия, определяющие, как его можно использовать в комбинации. Если условия не выполняются, элемент пропускается.

Если текущая комбинация не приводит к решению, функция откатывается, отменяя последнее присваивание и изменяя индекс для поиска других возможных комбинаций.

В случае успешного нахождения решения, функция возвращает True, и результаты выводятся в неубывающем порядке. Если решение не найдено, выводится сообщение "Impossibleeee".

**ТЕСТОВЫЕ ДАННЫЕ И РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

На рисунке 3 показаны входные данные, находящиеся в файле.

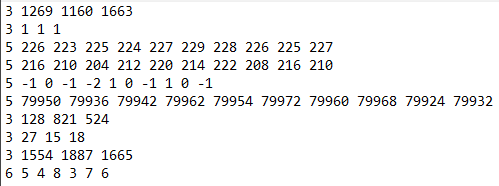


Рисунок 3 — Входные данные

Некоторые входные данные были взяты из методических указаний. Также тестовые данные были пополнены другими тестами, которые составлялись в обратном от работы программы порядке — были выбраны числа, которые в дальнейшем были попарно просуммированы и занесены в тестовые данные.

Результат обработки указанных входных данных показан на рисунке 4.

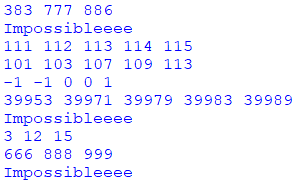


Рисунок 4 — Результат работы программы

Результаты обработки тестов, взятых из методических указаний, сходятся с выводом, указанном там же, что свидетельствует о правильности работы программы. Тесты, добавленные во входной файл самостоятельно, также были обработаны программой верно.

**ОПИСАНИЕ ЗАДАЧИ “СКОЛЬКО ЧИСЕЛ?”**

Напомним определение чисел Фибоначчи:

f1 = 1

f2 = 2

fn = fn-1 + fn-2 (n ≥ 3).

Для двух заданных чисел а и b определите, сколько чисел Фибоначчи лежит в диапазоне [а, b].

**Входные данные**

Входные данные содержат несколько тестовых блоков. Каждый тестовый блок состоит из двух неотрицательных целых чисел а и b. Входные данные завершаются блоком a = b = 0. В других случаях а ≤ b ≤ 10100 .

**Выходные данные**

Для каждого тестового блока выведите одну строку, содержащую количество чисел Фибоначчи fn, таких, что a ≤ fn ≤ b.

**Пример входных данных**

10 100

1234567890 9876543210

0 0

**Соответствующие выходные данные**

5

4

**АЛГОРИТМ РЕШЕНИЯ ЗАДАЧИ “ СКОЛЬКО ЧИСЕЛ?”**

Схема алгоритма решения поставленной задачи показана на рисунке 5.

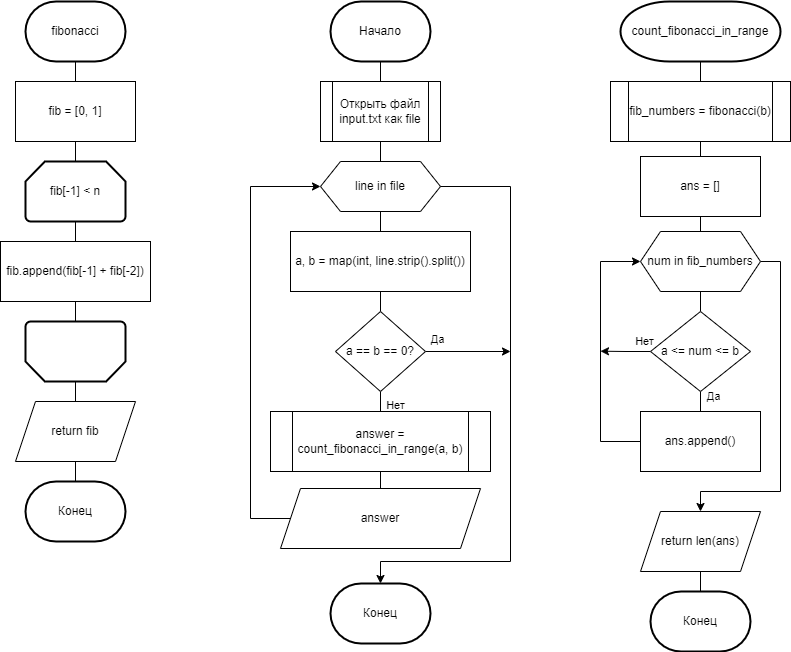


Рисунок 5 — Схема алгоритма решения задачи

Исходный код программы показан на рисунке 6.

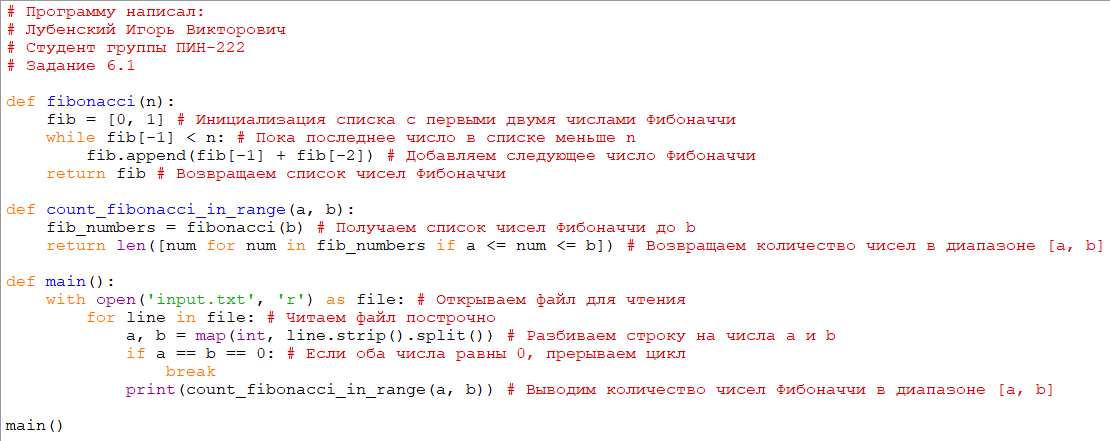


Рисунок 6 — Исходный код решения задачи

В программе заведено две функции для работы с числами Фибоначчи. Первая находит последовательность чисел Фибоначчи до указанного числа n, вторая находит числа Фибоначчи в указанном диапазоне от а до b.

В главной функции программы открывается входной файл и построчно считывается. В каждой строчке находятся числа a и b, разделённые пробелом. Для каждого диапазона производится проверка на равенство a и b нулю, что свидетельствует о конце входного файла.

**ТЕСТОВЫЕ ДАННЫЕ И РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ**

На рисунке 7 показаны входные данные, находящиеся в файле.

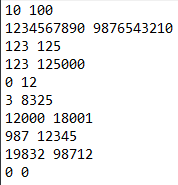


Рисунок 7 — Входные данные для задачи

Первые два теста (не считая строчки 0 0, означающей конец файла) были скопированы из методических указаний.

Результат обработки входных данных показаны на рисунке 8.



Рисунок 8 — Результат работы программы

В результате работы программы первые два теста сошлись с методическими указаниями. Остальные тестовые данные и результаты их обработки были проверены с помощью внутренней функции count\_fibonacci\_in\_range, которая показала, какие именно числа Фибоначчи лежат в указанном диапазоне, после чего вывела правильное их количество.