

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ  
УНИВЕРСИТЕТ**

**ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ**

**ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ**

**Отчет по лабораторной работе №0  
по курсу «Алгоритмы и структуры данных»**

**Тема: Введение**

**Выполнила:**

**Анисимова В. А.**

**K3162**

**Проверил:**

**Царьков Г. И.**

**Санкт-Петербург**

**2025 г.**

## **Содержание отчета:**

Содержание отчета

Задача №1. Ввод-Вывод ( $a+b$ )

Задача №1. Ввод-Вывод ( $a+b^2$ )

Задача №2. Числа Фибоначчи

Задача №3. Еще числа Фибоначчи

Вывод

## Задачи:

### Задание 1. Ввод-Вывод

1. Задача  $a + b$ . В данной задаче требуется вычислить сумму двух заданных чисел. Вход: одна строка, которая содержит два целых числа  $a$  и  $b$ . Для этих чисел выполняются условия  $-10^9 \leq a, b \leq 10^9$ . Выход: единственное целое число — результат сложения  $a + b$ .

```
a, b = map(int, input().split())
if -10**9 <= a <= 10**9 and -10**9 <= b <= 10**9:
    result = a + b
    print(result)
```

#### Объяснение задача:

Цель решения задачи заключается в том, что нужно сложить два целых числа  $a$  и  $b$

Каждое число должно находиться в диапазоне от  $-10^9$  до  $10^9$  включительно, в итоге программа должна выводить результат сложения

#### Алгоритм:

- Чтение входных данных: программе нужно получить строку с двумя числами от пользователя
- Разделение и преобразование: разделить строку на отдельные элементы и преобразовать их в целые числа
- Проверка: убедиться, что оба числа удовлетворяют ограничениям по диапазону
- Вычисление: выполнить арифметическую операцию сложения чисел
- Вывод результата: отобразить полученную сумму

#### Вывод:

Данная программа работает правильно, решая задачу. Два числа проверяются, в нужном ли они диапазоне, складывает их и показывает результат. При этом код простой и понятный, использовался в Python.

2. Задача  $a + b^{**2}$ . В данной задаче требуется вычислить значение  $a + b^{**2}$ . Вход: одна строка, которая содержит два целых числа  $a$  и  $b$ . Для этих чисел выполняются условия  $-10^9 \leq a, b \leq 10^9$ . Выход: единственное целое число — результат сложения  $a + b^{**2}$ .

```
a, b = map(int, input().split())
if -10**9 <= a <= 10**9 and -10**9 <= b <= 10**9:
    result = a + b**2
print(result)
```

#### Объяснение:

Цель решения задачи заключается в том, что нужно вычислить два числа по формуле  $a+b^2$

Каждое число должно находиться в диапазоне от  $-10^9$  до  $10^9$  включительно, в итоге программа должна выводить результат

#### Алгоритм:

- Чтение входных данных: программе нужно получить строку с двумя числами от пользователя
- Проверить, что оба числа в диапазоне
- Вычислить результат
- Показать ответ с помощью вывода результата

#### Вывод:

Программа корректно вычисляет сумму первого числа и квадрата второго числа. Код простой и выполняет поставленную задачу, также аналогичен первому заданию.

3. Выполните задачу  $a + b$  с использованием файлов. • Имя входного файла: input.txt • Имя выходного файла: output.txt • Формат входного файла. Входной файл состоит из одной строки, которая содержит два целых числа  $a$  и  $b$ . Для этих чисел выполняются условия  $-109 \leq a, b \leq 109$ . • Формат выходного файла. Выходной файл единственное целое число — результат сложения  $a + b$ .

```
with open('input.txt', 'r') as f_input:
    line = f_input.readline()
    numbers = line.split()
    a = int(numbers[0])
    b = int(numbers[1])
    result = a + b
with open('output.txt', 'w') as f_output:
    f_output.write(str(result))
```

#### Объяснение:

Цель решения задачи заключается в том, что программа должна прочитать из файла последовательность чисел, где их складывает и записывает результат в другой файл

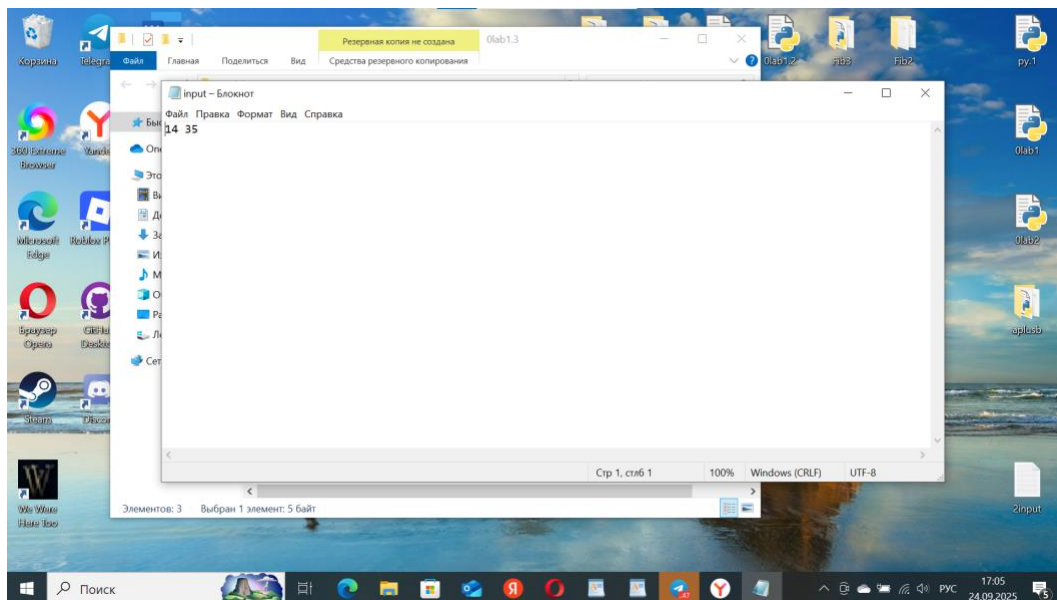
### Алгоритм:

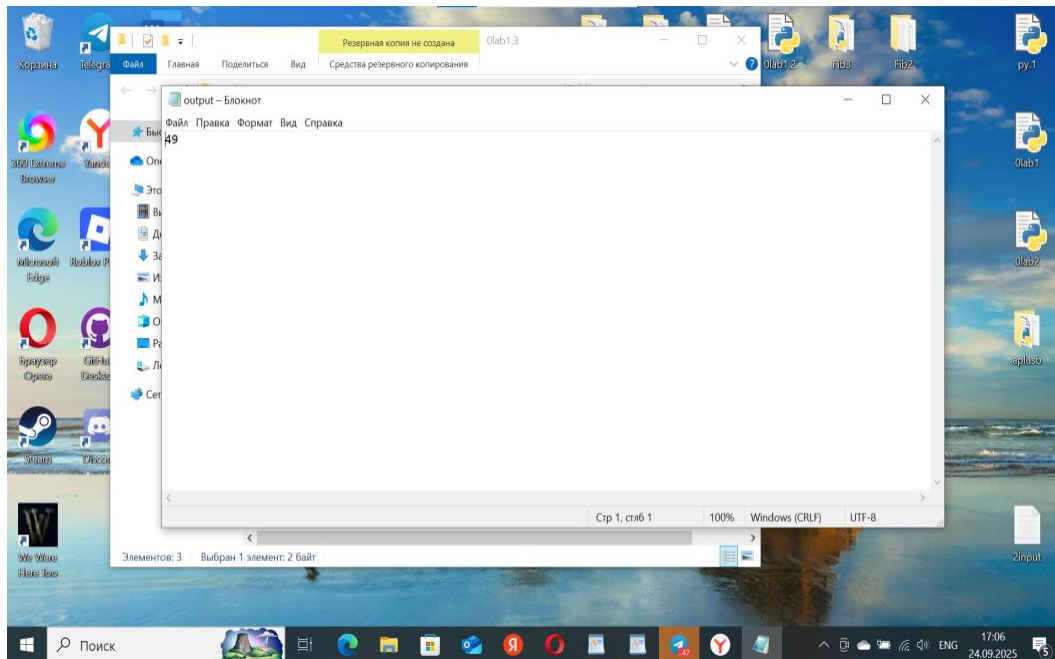
- Программа должно открыть файл input.txt и прочитать числа из него
- Разбить содержимое на отдельные числа
- Обработать числа по парам, то есть к примеру сложить первое число со вторым и тд.
- Сохранить все результаты сложения в список
- Записать полученный результат в файл output.txt

### Вывод:

Программа отлично обрабатывает числовые данные из файла, по парам складывая числа и сохраняет результаты. Код корректно работает с файлами.

Я считаю, что данный способ обработки числовых данных очень быстрый и удобный вариант для автоматической обработки пар чисел.





4. Выполните задачу  $a+b \cdot 2$  с использованием файлов аналогично предыдущему пункту

```
with open('input.txt', 'r') as f_input:
    line = f_input.readline()
    numbers = line.split()
    a = int(numbers[0])
    b = int(numbers[1])
    result = a + b**2
    with open('output.txt', 'w') as f_output:
        f_output.write(str(result))
```

#### Объяснение:

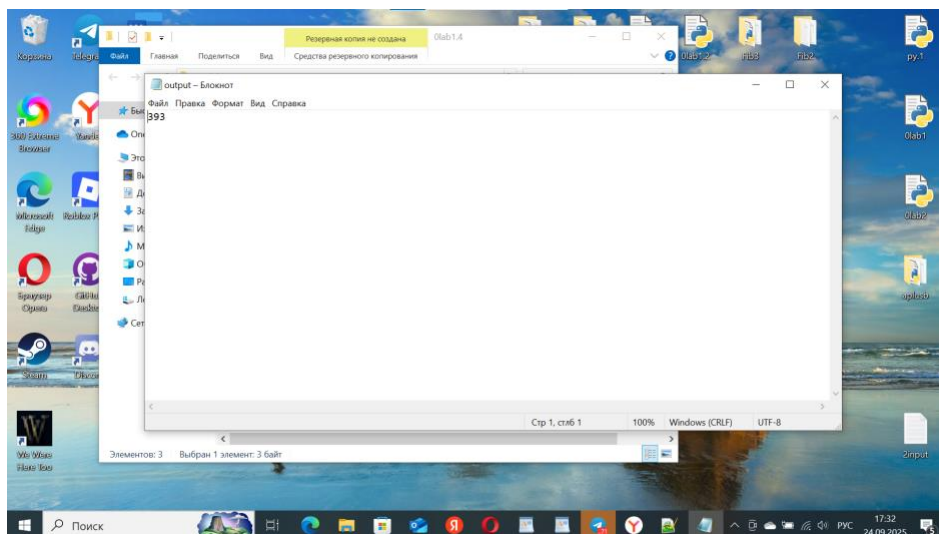
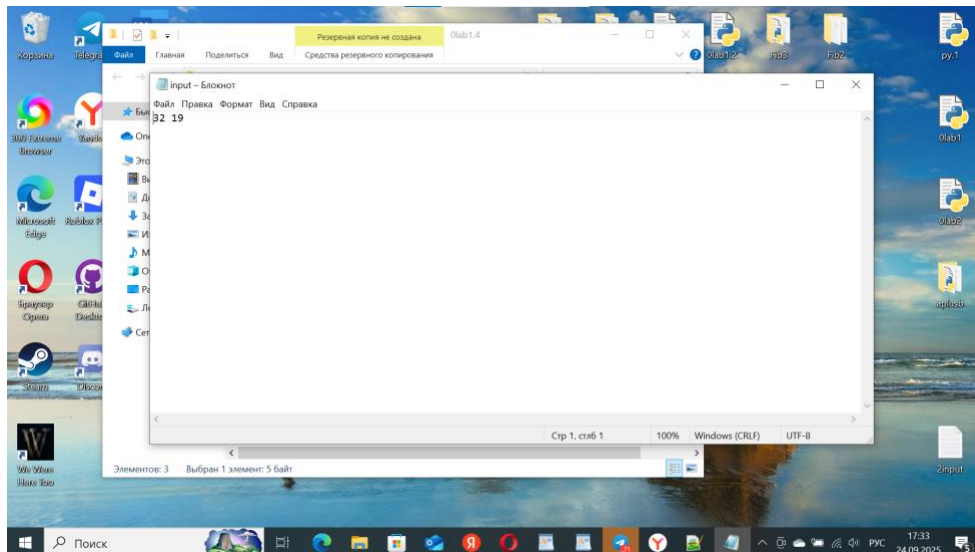
Цель решения задачи заключается в том, чтобы программа могла прочитать из файла input.txt последовательность чисел, сложила их и записала в файле output.txt

#### Алгоритм:

- Чтение данных из файла, то есть открыть input.txt и прочитать строку и разбить ее на отдельные числа
- Попарная обработка чисел, каждую пару преобразовать в целые числа в строки
- И записать результаты в файл output.txt

#### Вывод:

Программа выполняет попарное сложение чисел, корректно читает и сохраняет результат из файла в файл.



## Задание 2. Число Фибоначчи

Определение последовательности Фибоначчи:  $F_0 = 0$  (1)  $F_1 = 1$   $F_i = F_{i-1} + F_{i-2}$  для  $i \geq 2$ . Таким образом, каждое число Фибоначчи представляет собой сумму двух предыдущих, что дает последовательность 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ... Ваша цель – разработать эффективный алгоритм для подсчета чисел Фибоначчи. Вам предлагается начальный код на Python, который содержит наивный рекурсивный алгоритм

```
def calc_fib(n):
    if n <= 1:
        return n
    a, b = 0, 1
    for i in range(2, n + 1):
        a, b = b, a + b
    return b
with open('fib.input.txt', 'r') as input_file:
    n = int(input_file.read().strip())
    result = calc_fib(n)
    with open('fib.output.txt', 'w') as output_file:
        output_file.write(str(result))
```

### Объяснение:

Программа должна вычислить n-ное число Фибоначчи и сохраняет результат в файл.

Последовательность Фибоначчи: 0, 1, 2, 3, 5, 8, 13, 21

### Алгоритм:

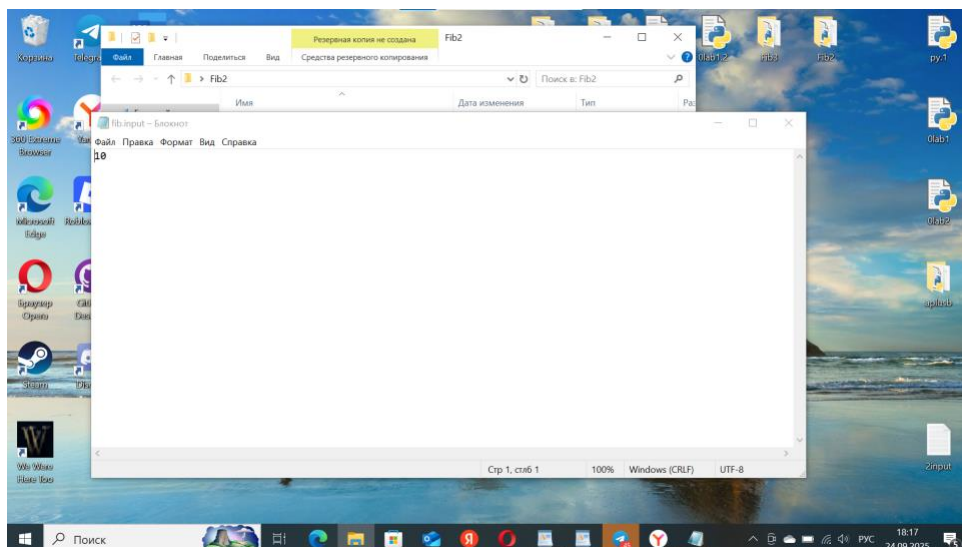
-Для начала нужно прочитать число n из файла fib.input.txt

-Проверка граничных случаев

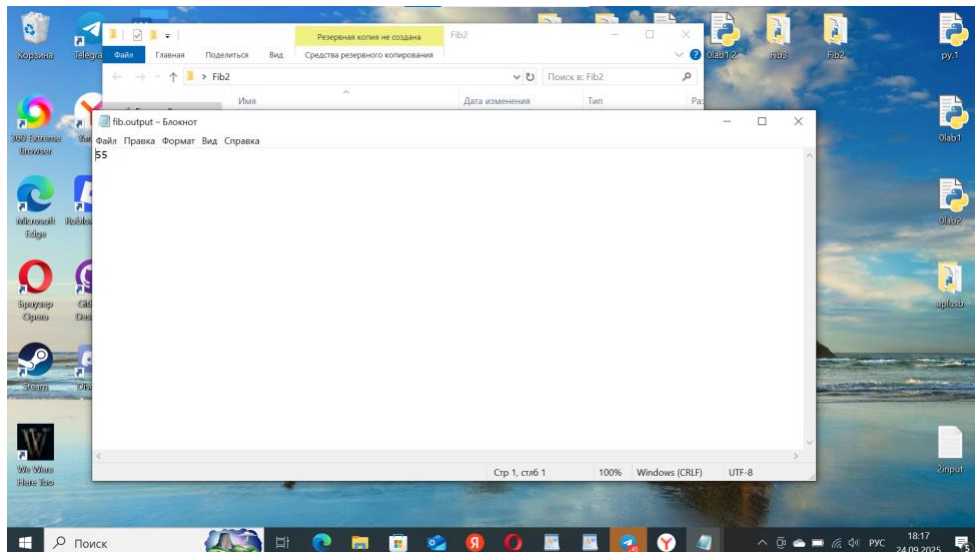
-Выводим результат в файл fib.output.txt

### Вывод:

Программа отлично реализует алгоритм для вычисления n-ного числа Фибоначчи. Код работает эффективно вводит и выводит решения в файлах







### Задание 3. Еще про числа Фибоначчи

Определение последней цифры большого числа Фибоначчи. Числа Фибоначчи растут экспоненциально. Например,  $F_{200} = 280571172992510140037611932413038677189525$ . Хранить такие суммы в массиве, и при этом подсчитывать сумму, будет достаточно долго. Найти последнюю цифру любого числа достаточно просто:  $F \bmod 10$ .

- Имя входного файла: input.txt
- Имя выходного файла: output.txt
- Формат входного файла. Целое число  $n$ .  $0 \leq n \leq 10^{**7}$
- Формат выходного файла. Одна последняя цифра числа  $F_n$ .

Пример 1. input.txt  
331 output.txt 9  $F_{331} = 668996615388005031531000081241745415306766517246774551964595292186469$ . Это число не влезет в страницу, но оканчивается действительно на 5.

Ограничение по времени: 5сек. • Ограничение по памяти: 512 мб.

```
def calc_fib_last_digit(n):
    if n <= 1:
        return n
    a, b = 0, 1
    for i in range(2, n + 1):
        a, b = b, (a + b) % 10
    return b
with open('input.txt', 'r') as input_file:
    n = int(input_file.read().strip())
    result = calc_fib_last_digit(n)
with open('output.txt', 'w') as output_file:
    output_file.write(str(result))
```

#### Объяснение:

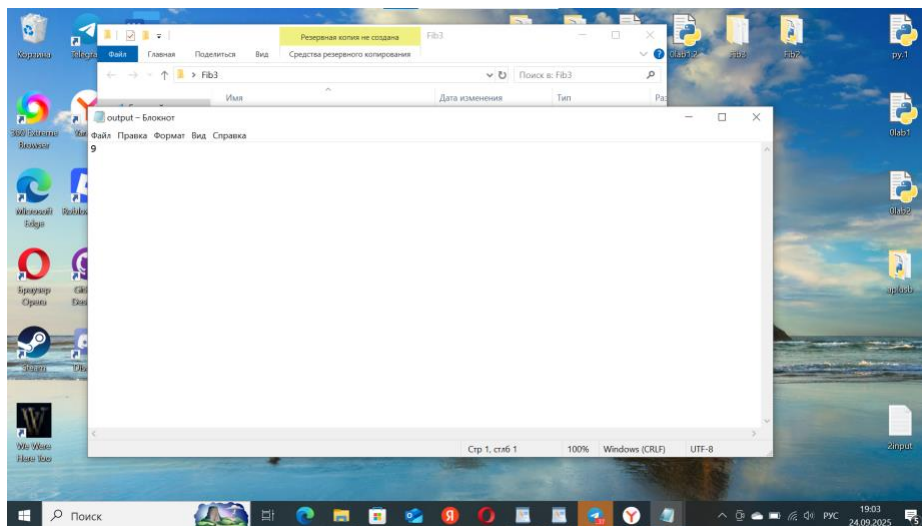
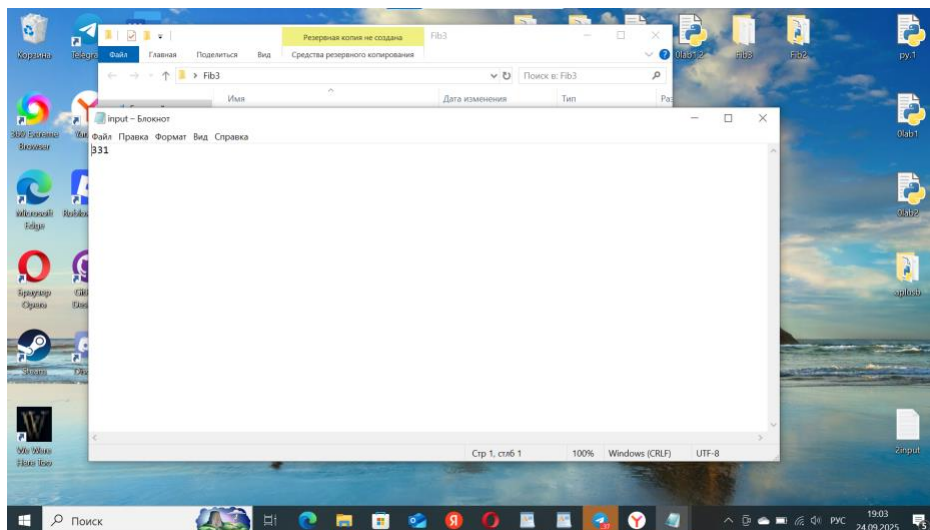
Цель решения задачи заключается в том, что нужно найти только последнюю цифру  $n$ -ного числа Фибоначчи, не вычисляя само огромное число.

#### Алгоритм:

- Выполняется вход данных, то есть нужно прочитать число n из файла input.txt
- Проверка граничных случаев
- Вычисление
- Запись результата в файл output.txt

### Вывод:

Алгоритм решает задачу нахождения последней цифры числа Фибоначчи. Использование итеративного подхода с сохранением только последних чисел позволяет быстро работать с очень большими числами n.



### Задание 4. Тестирование ваших алгоритмов.

Тест задания №2:

	Время выполнения
--	------------------

Нижняя граница диапазона значений входных данных из текста задачи: $F(0)=0$	0.000004
Пример из задачи: $F(30)=832040$	0.000007
Пример из задачи: $F(5)=5$	0.000005
Верхняя граница диапазона значений входных данных из текста задачи: $F(10)=55$	0.000003

Тест задания №3:

	Время выполнения
Нижняя граница диапазона значений входных данных из текста задачи: $N=0; 0$	0.0002
Пример из задачи: $n=331; 9$	0.0004
Пример из задачи: $n=327305; 5$	0.0297
Верхняя граница диапазона значений входных данных из текста задачи: $N=10000000; 5$	0.4412

### Вывод по всей лабораторной работе:

Данные задачи позволили мне сравнить разные алгоритмы вычисления, с помощью ввода и вывода, работа с файлами и с числами Фибоначчи.

Для каждого решения задачи важен правильный выбор алгоритма.