

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

Выполнил:
Блинова П. В.
К3139

Проверил:
Афанасьев А. В.

Санкт-Петербург
2024 г.

Содержание отчета

Содержание отчета	2
Задачи по варианту	3
Задача №1. Ввод-вывод [N баллов]	3
Задача №2. Число Фибоначчи [N баллов]	7
Задача №3. Ещё про числа Фибоначчи [N баллов]	9
Задача №4. Тестирование алгоритмов [N баллов]	10
Вывод	14

Задачи по варианту

Задача №1. Ввод-вывод [N баллов]

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 $a + b$.

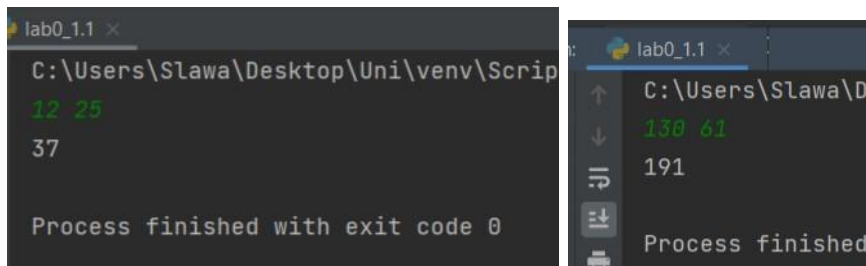
Листинг кода:

```
a, b = map(int, list(input().split(' ')))
if (-10) ** 9 <= a <= 10 ** 9 and (-10) ** 9 <= b <= 10
** 9:
    print(a + b)
else:
    print('Введите другое число')
```

Текстовое объяснение решения:

Ввожу две переменные a и b , которые принимают значения из `input()`. Применяю `split()`: делит строку на несколько частей, - `map()`: функция, которая применяется к итерируемому объекту. С помощью `map(int, <..>)` делаю список чисел. Проверяю условие и вывожу в `print()` результат суммы чисел a и b .

Результат работы кода на примерах из текста задачи:



Результат работы кода на максимальных и минимальных значениях:

```
lab0_1.1 x C:\Users\Slawa\Desktop\Uni\venv\Scr
1000000000 1000000000 -1000000000 -1000000000
2000000000 -2000000000
Process finished with exit code 0 Process finished with exit code 0
```

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 $a + b^2$.

Листинг кода:

```
a, b = map(int, list(input().split(' ')))
if (-10) ** 9 <= a <= 10 ** 9 and (-10) ** 9 <= b <= 10
** 9:
    print(a + b ** 2)
else:
    print('Введите другое число')
```

Текстовое объяснение решения:

Ввожу две переменные a и b , которые принимают значения из `input()`. Применяю `split()`: делит строку на несколько частей, - `map()`: функция, которая применяется к итерируемому объекту. С помощью `map(int, <..>)` делаю список чисел. Проверяю условие и вывожу в `print()` результат суммы чисел a и $b ** 2$.

Результат работы кода на примерах из текста задачи:

```
lab0_1.1 x lab0_1.2 x lab0_1.1 x lab0_1.2 x
C:\Users\Slawa\De C:\Users\Slawa\De
12 25 130 61
637 3851
Process finished Process finished wi
```

Результат работы кода на максимальных и минимальных значениях:

```
C:\Users\Slawa\
1000000000 1
1000000001
-1000000000 -1000000000
-2000000000
Process finished with exit code 0
```

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 $-10^9 \leq a, b \leq 10^9$.
- Формат выходного файла. Выходной файл — единственное целое число — результат сложения $a + b$.

Листинг кода:

```
file_output = open('output.txt', 'w')

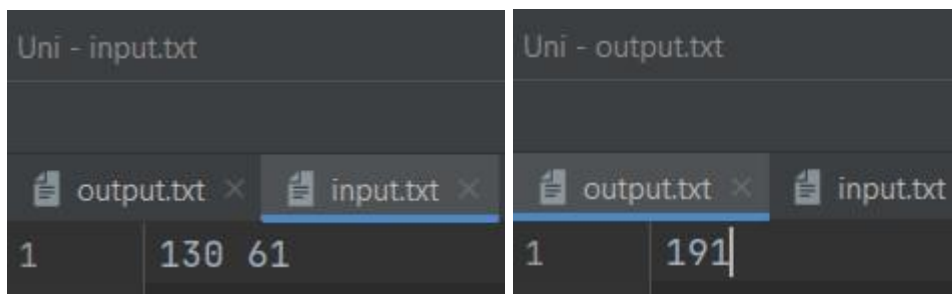
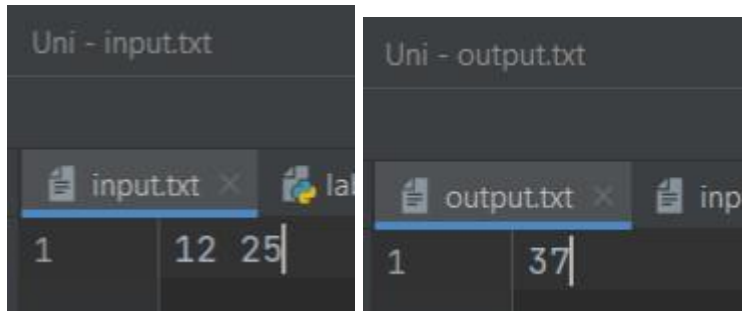
with open('input.txt', 'r') as f:
    file = f.readline()
    a, b = map(int, list(file.split(' ')))
    if (-10) ** 9 <= a <= 10 ** 9 and (-10) ** 9 <= b <=
10 ** 9:
        file_output.write(str(a + b))
    else:
        print('Введите другое число')

file_output.close()
```

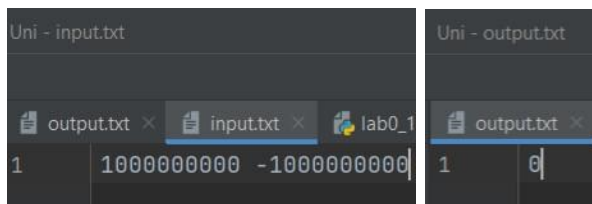
Текстовое объяснение решения:

Открываю файл output.txt в режиме записи с помощью open(). С помощью оператора with открываю файл input.txt в режиме чтения. Записываю в переменную file первую строку из файла с помощью readline() и аналогично первому заданию нахожу сумму. Записываю результат в файл output.txt и закрываю файл.

Результат работы кода на примерах из текста задачи:



Результат работы кода на максимальных и минимальных значениях:



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

Листинг кода:

```
file_output = open('output.txt', 'w')

with open('input.txt', 'r') as f:
    file = f.readline()
```

```

a, b = map(int, list(file.split(' ')))
if (-10) ** 9 <= a <= 10 ** 9 and (-10) ** 9 <= b <= 10 ** 9:
    file_output.write(str(a + b ** 2))
else:
    print('Введите другое число')

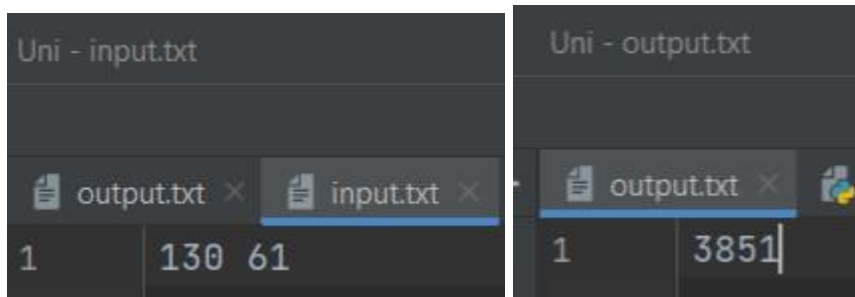
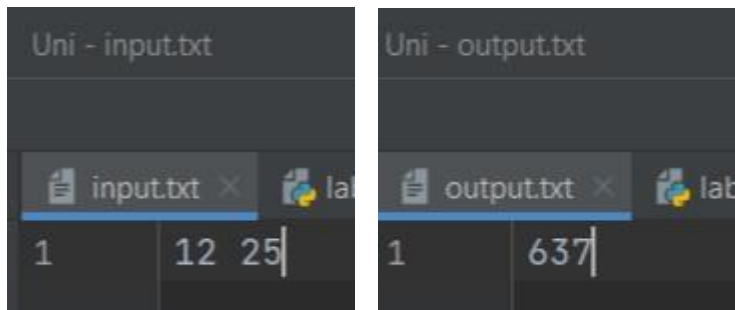
file_output.close()

```

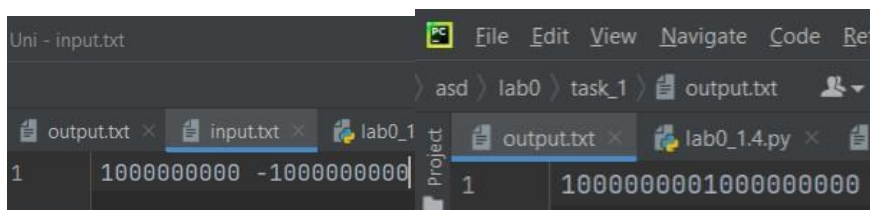
Текстовое объяснение решения:

Открываю файл output.txt в режиме записи с помощью open(). С помощью оператора with открываю файл input.txt в режиме чтения. Записываю в переменную file первую строку из файла с помощью readline() и аналогично второму заданию нахожу необходимый результат. Записываю результат суммы чисел a и b ** 2 в файл output.txt и закрываю файл.

Результат работы кода на примерах из текста задачи:



Результат работы кода на максимальных и минимальных значениях:



Задача №2. Число Фибоначчи [N баллов]

Ваша цель – разработать эффективный алгоритм для подсчета чисел Фибоначчи. Вам предлагается начальный код на Python, который содержит наивный рекурсивный алгоритм.

- Имя входного файла: input.txt
- Имя выходного файла: output.txt
- Формат входного файла. Целое число n . $0 \leq n \leq 45$.
- Формат выходного файла. Число F_n .

Листинг кода:

```
import time

start = time.perf_counter()

def main():
    file_output = open('output.txt', 'w')

    def calc_fib(n):
        if n <= 1:
            return n

        return calc_fib(n - 1) + calc_fib(n - 2)

    with open('input.txt', 'r') as file:
        f = file.readline()
        n = int(f)
        if 0 <= n <= 45:
            res = calc_fib(n)
            file_output.write(str(res))
        else:
            print('Введите другое число')

    file_output.close()

if __name__ == '__main__':
    main()

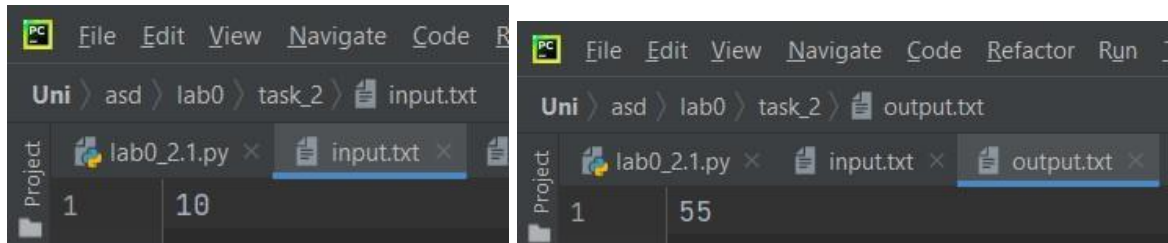
finish = time.perf_counter()
print('Время работы: ' + str(finish - start))
```

Текстовое объяснение:

Открываю файл output.txt в режиме записи с помощью open(). С помощью оператора with открываю файл input.txt в режиме чтения. Записываю в

переменную `file` первую строку из файла с помощью `readline()`. С помощью функции `calc_fib` из условия задачи находим число Фибоначчи, проверяя условие. Записываю результат в файл `output.txt` и закрываю файл.

Результат работы кода на примерах из текста задачи:



Задача №3. Ещё про числа Фибоначчи [N баллов]

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

$$F_{331} = 668996615388005031531000081241745415306766517246774551964595292186469.$$

- Пример 2.

input.txt	327305
output.txt	5

Это число не влезет в страницу, но оканчивается действительно на 5.

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

Листинг кода:

```

import time

start = time.perf_counter()

def main():
    file_output = open('output.txt', 'w')

    def calc_fib_last_digit(n):
        if n <= 1:
            return n

        num1, num2 = 0, 1
        for i in range(2, n + 1):
            num1, num2 = num2, (num1 + num2) % 10
        return num2

    with open('input.txt', 'r') as file:
        f = file.readline()
        n = int(f)
        if 0 <= n <= 10 ** 7:
            res = calc_fib_last_digit(n)
            file_output.write(str(res))
        else:
            print('Введите другое число')

    file_output.close()

if __name__ == '__main__':
    main()

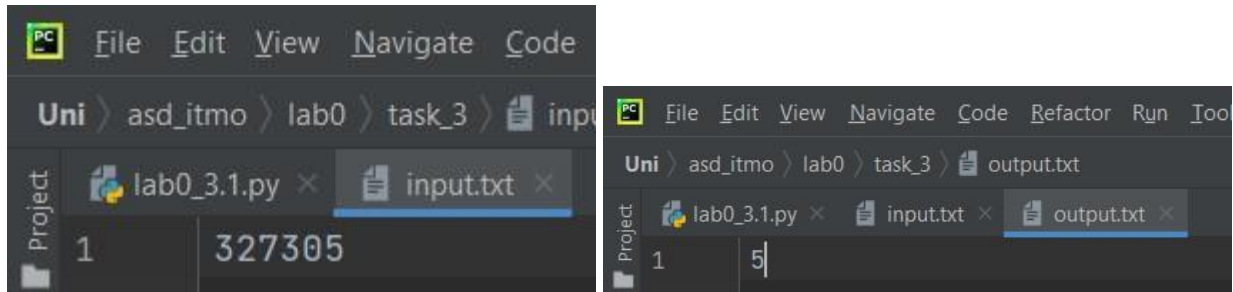
finish = time.perf_counter()
print('Время работы: ' + str(finish - start))

```

Текстовое объяснение:

Открываю файл output.txt в режиме записи с помощью open(). С помощью оператора with открываю файл input.txt в режиме чтения. Записываю в переменную file первую строку из файла с помощью readline(). С помощью функции calc_fib_last_digit находим число Фибоначчи, проверяя условие. Задаю переменным num1 и num2 значения 0 и 1 и запускаю цикл for. На каждой итерации перезаписываю числа и запоминаю последнюю цифру числа, чтобы оптимизировать алгоритм. Записываю результат в файл output.txt и закрываю файл.

Результат работы кода на примерах из текста задачи:



Задача №4. Тестирование алгоритмов [N баллов]

Задача: вам необходимо протестировать время выполнения вашего алгоритма в Задании 2 и Задании 3.

Листинг кода:

```
import time

start = time.perf_counter()

def main():
    file_output = open('output.txt', 'w')

    def calc_fib(n):
        if n <= 1:
            return n

        return calc_fib(n - 1) + calc_fib(n - 2)

    with open('input.txt', 'r') as file:
        f = file.readline()
        n = int(f)
        if 0 <= n <= 45:
            res = calc_fib(n)
            file_output.write(str(res))
        else:
            print('Введите другое число')

    file_output.close()

if __name__ == '__main__':
    main()

finish = time.perf_counter()
print('Время работы: ' + str(finish - start))
```

```

import time

start = time.perf_counter()

def main():
    file_output = open('output.txt', 'w')

    def calc_fib_last_digit(n):
        if n <= 1:
            return n

        num1, num2 = 0, 1
        for i in range(2, n + 1):
            num1, num2 = num2, (num1 + num2) % 10
        return num2

    with open('input.txt', 'r') as file:
        f = file.readline()
        n = int(f)
        if 0 <= n <= 10 ** 7:
            res = calc_fib_last_digit(n)
            file_output.write(str(res))
        else:
            print('Введите другое число')

    file_output.close()

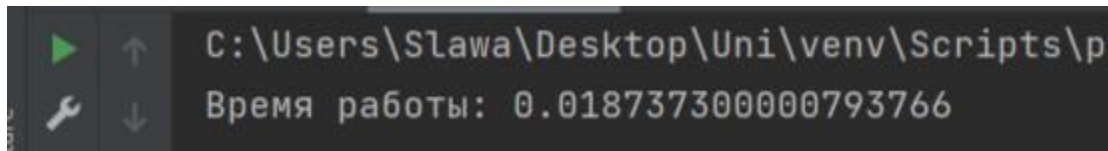
if __name__ == '__main__':
    main()

finish = time.perf_counter()
print('Время работы: ' + str(finish - start))

```

Текстовое объяснение:

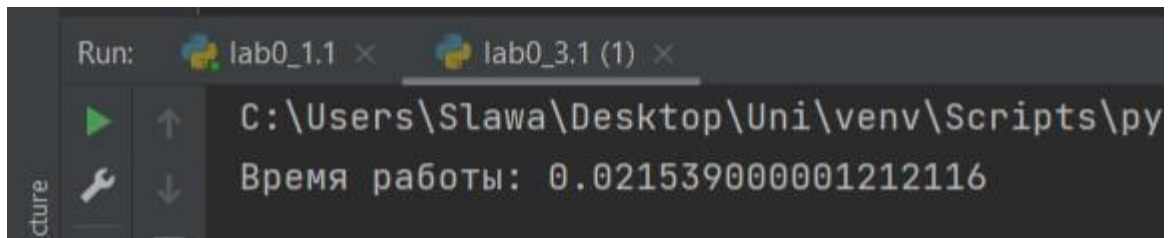
И в задаче 2, и в задаче 3 импортируем модуль time, засекает начальное время и вычитаем из конечного времени начальное. Выводим результат в print().



```

C:\Users\Slawa\Desktop\Uni\venv\Scripts\p
Время работы: 0.018737300000793766

```



```
Run: lab0_1.1 × lab0_3.1 (1) ×  
C:\Users\Slawa\Desktop\Uni\venv\Scripts\py  
Время работы: 0.021539000001212116
```

Вывод

- 1) В ходе лабораторной работы был изучен способ применения файлов в решении задач. Было изучено, как читать входные файлы.
- 2) Был изучен способ измерения времени работы программы с помощью модуля time
- 3) Был изучен способ получения числа Фибоначчи