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:
Янин Е. В.
К3141

Проверил:
Афанасьев А. В.

Санкт-Петербург
2024 г.

Содержание отчета

Оглавление

Задачи по варианту

Задача №1. Ввод-вывод

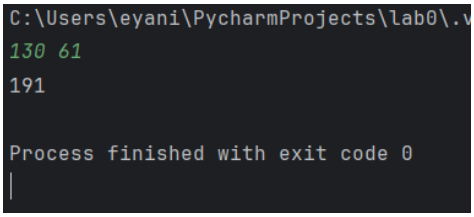
1. В данной задаче требуется вычислить сумму двух заданных чисел. Вход: одна строка, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-10^9 \le a, b \le 10^9$. Выход: единственное целое число — результат сложения a + b

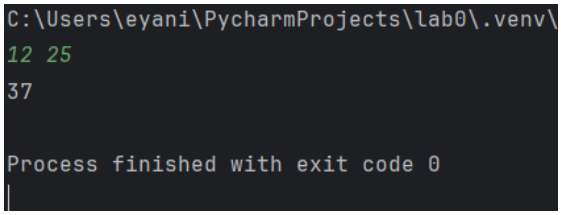
```
a, b = map(int, input().split())
while not (-10**9 <= a <= 10**9 and -10**9 <= b <= 10**9):
    print('Неверные данные. Введите ещё раз')
    a, b = map(int, input().split())
print(a + b)
```

Объяснение решения:

1) На вход поступает 2 числа через пробел
2) Числа проверяются на выполнение условий задачи
3) Если числа соответствуют условиям, выводится их сумма

Результат работы кода:





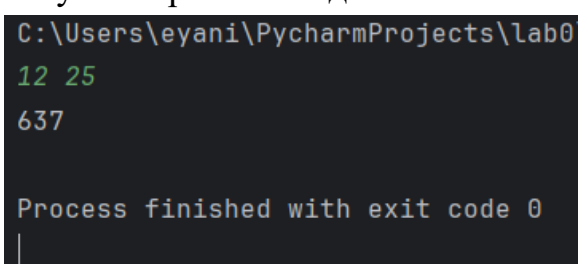
2. Задача $a + b^2$. В данной задаче требуется вычислить значение $a + b^2$. Вход: одна строка, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-10^9 \le a, b \le 10^9$. Выход: единственное целое число — результат сложения $a + b^2$.

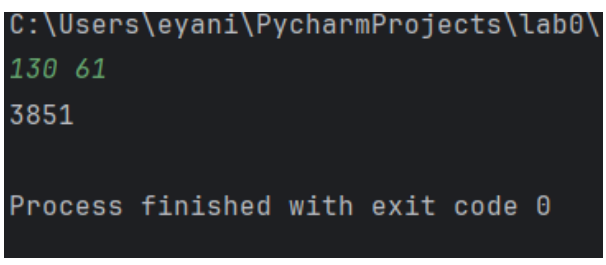
```
a, b = map(int, input().split())
while not (-10**9 <= a <= 10**9 and -10**9 <= b <= 10**9):
    print('Неверные данные. Введите ещё раз')
    a, b = map(int, input().split())
print(a + b**2)
```

Объяснение решения:

1) На вход поступает 2 числа через пробел
2) Числа проверяются на выполнение условий задачи
3) Если числа соответствуют условиям, выводится сумма первого числа и квадрата второго

Результат работы кода:





3. Выполните задачу a + b с использованием файлов.

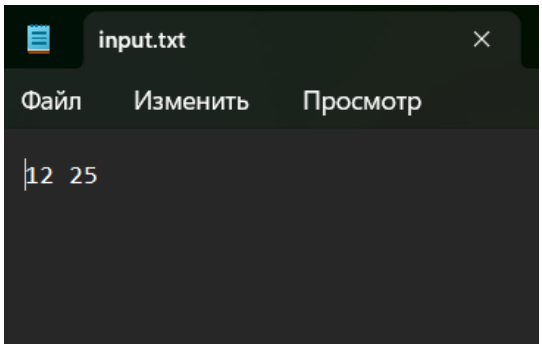
- Имя входного файла: input.txt
- Имя выходного файла: output.txt
- Формат входного файла. Входной файл состоит из одной строки, которая содержит два целых числа a и b. Для этих чисел выполняются условия $-10^9 \le a, b \le 10^9$.
- Формат выходного файла. Выходной файл единственное целое число — результат сложения a + b.

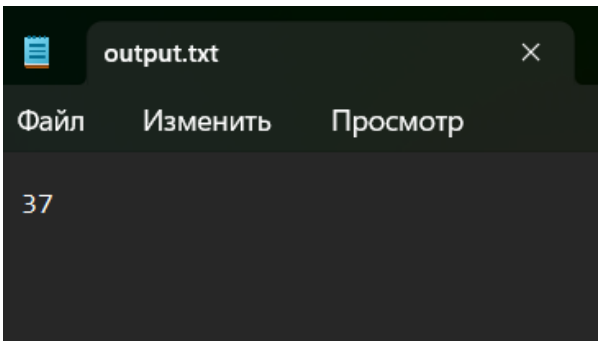
```
fin = open('input.txt')
a, b = map(int, fin.read().split())
fin.close()
if -10**9 <= a <= 10**9 and -10**9 <= b <= 10**9:
    fout = open('output.txt', '+w')
    fout.write(str(a + b))
    fout.close()
else:
    print('Неверные входные данные')
```

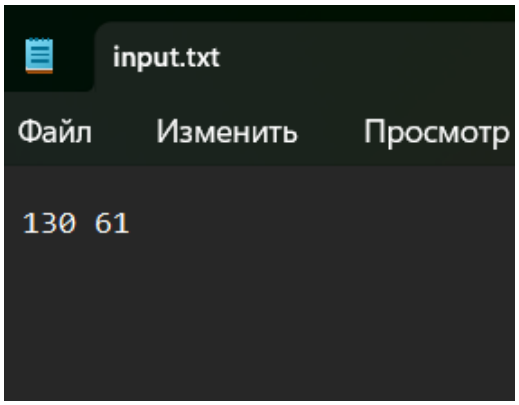
Объяснение решения:

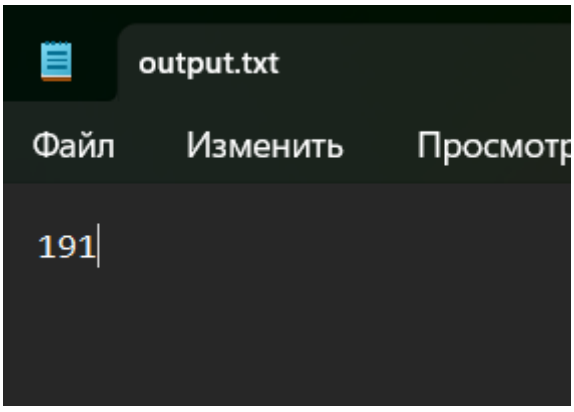
1) Открывается файл input.txt, оттуда считывается первая строчка с двумя числами, записанными через пробел, затем файл закрывается
2) Числа проверяются на выполнение условий задачи
3) Если числа соответствуют условиям, в только что открытый файл output.txt записывается их сумма, затем файл закрывается.

Пример работы кода:









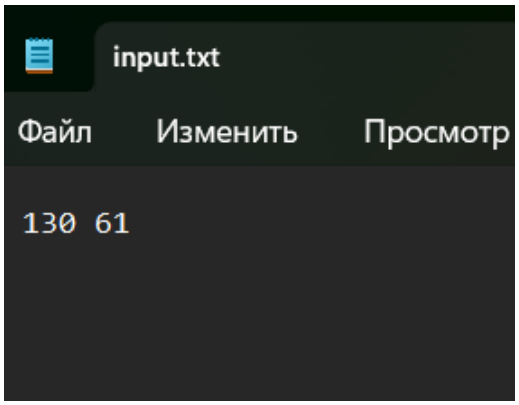
4. Выполните задачу a+b 2 с использованием файлов аналогично предыдущему пункту.

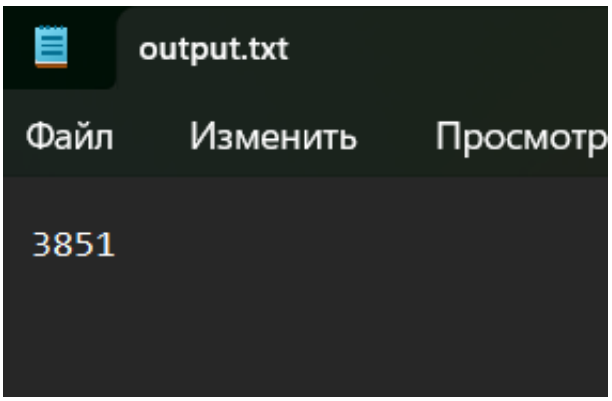
```
fin = open('input.txt')
s = fin.readline()
fin.close()
a, b = map(int, s.split())
if -10**9 <= a <= 10**9 and -10**9 <= b <= 10**9:
    fout = open('output.txt', '+w')
    fout.write(str(a + b**2))
    fout.close()
else:
    print('Неверные входные данные')
```

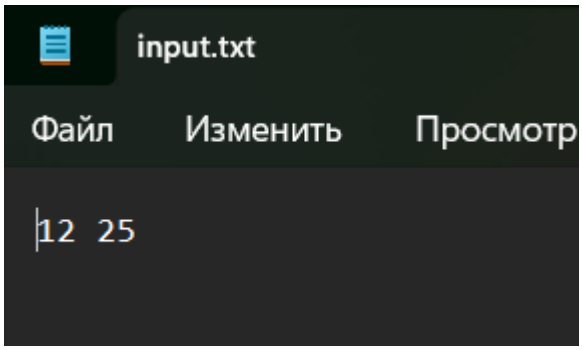
Объяснение решения:

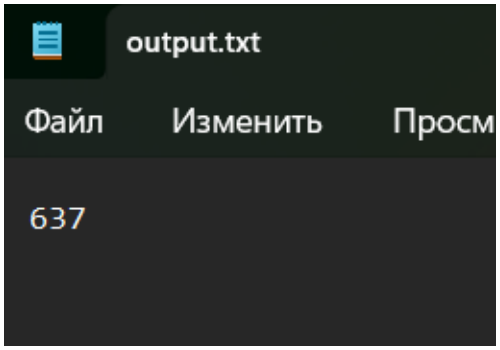
1) Открывается файл input.txt, оттуда считывается первая строчка с двумя числами, записанными через пробел, затем файл закрывается
2) Числа проверяются на выполнение условий задачи
3) Если числа соответствуют условиям, в только что открытый файл output.txt записывается сумма первого числа и квадрата второго, затем файл закрывается.

Пример работы кода:









Вывод: при решении этих задач были использованы переменные, условные конструкции, чтение информации из фалов и её запись в них. Алгоритмы удовлетворяют требованиям задачи и успешно справляются с поставленными целями.

Задача №2. Число Фибоначчи

Определение последовательности Фибоначчи: $F0 = 0$, $F1 = 1$, $F_i = F_{i-1} + F_{i-2}$ 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
return calc_fib(n - 1) + calc_fib(n - 2)
n = int(input())
print(calc_fib(n))
```

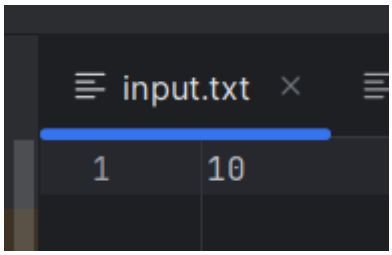
- Имя входного файла: input.txt
- Имя выходного файла: output.txt
- Формат входного файла. Целое число n. $0 \leq n \leq 45$.
- Формат выходного файла. Число F_n.

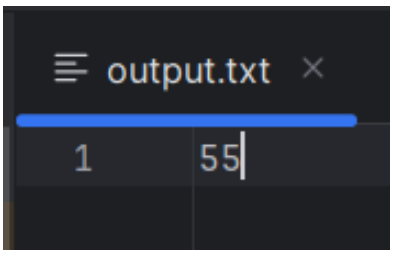
```
fin = open('input.txt')
n = int(fin.readline())
fin.close()
fib = [0, 1]
if 0 <= n <= 45:
    for i in range(2, n + 1):
        fib.append(fib[-1] + fib[-2])
    fout = open('output.txt', '+w')
    fout.write(str(fib[n]))
    fout.close()
else:
    print('Неверные входные данные')
```

Объяснение решения:

1) Открывается файл input.txt, оттуда считывается первая строчка с числом n, затем файл закрывается.
2) Создаётся массив fib с первыми двумя числами Фибоначчи.
3) Число проверяется на выполнение условий задачи.
4) Если число соответствует условиям, в массив fib записываются все числа Фибоначчи вплоть до F_n.
5) В только что открытый файл output.txt записывается число F_n, затем файл закрывается.

Пример работы кода:





Вывод: на заданном диапазоне алгоритм справляется с поставленной задачей.

Задача №3. Ещё 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 mod 10.

- Имя входного файла: input.txt
- Имя выходного файла: output.txt
- Формат входного файла. Целое число n. $0 \le n \le 10^7$
- 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.

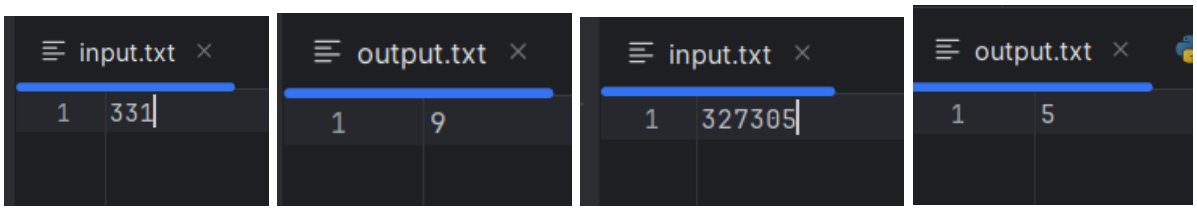
```
def main():
    fin = open('input.txt')
    n = int(fin.readline())
    fin.close()
    fib = [0, 1]
    if 0 <= n <= 10**7:
        for i in range(2, n + 1):
            fib.append((fib[-1] + fib[-2]) % 10)
        fout = open('output.txt', '+w')
        fout.write(str(fib[n]))
        fout.close()
    else:
        print('Неверные входные данные')

main()
```

Объяснение решения:

1) Открывается файл input.txt, оттуда считывается первая строчка с числом n, затем файл закрывается.
2) Создаётся массив fib с первыми двумя числами Фибоначчи.
3) Число проверяется на выполнение условий задачи.
4) Если число соответствует условиям, в массив fib записываются все последние цифры чисел Фибоначчи вплоть до F_n.

5) В только что открытый файл output.txt записывается последняя цифра числа F_n, затем файл закрывается.

Пример работы кода:



Вывод: алгоритм справляется с вычислением последней цифры числа Фибоначчи за небольшие промежутки времени.

Задача №4. Тестирование ваших алгоритмов

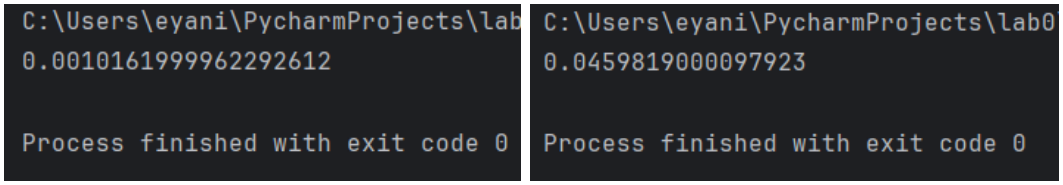
Задача: вам необходимо протестировать время выполнения вашего алгоритма в Задании 2 и Задании 3. Дополнительно: вы можете протестировать объем используемой памяти при выполнении вашего алгоритма.

```
import time
from t3 import main


start = time.perf_counter()
main()
print(time.perf_counter() - start)
```

Объяснение решения:

1) Импортируем библиотеку time для работы с временем выполнения алгоритма и функцию main из предыдущей задачи.
2) Записываем в переменную start время начала выполнения алгоритма.
3) Выполняем алгоритм предыдущей задачи.
4) Выводим разность времени завершения выполнения программы и переменной start.

Пример работы кода:



Время работы алгоритма для задачи №3 при n = 331 и n = 327305 соответственно.

Вывод: увеличение значения числа n линейно увеличивает время работы алгоритма.

Вывод

В данной лабораторной работе я попрактиковался в решении задач с числами Фибоначчи, работе с информацией из файлов, а также познакомился с библиотекой time и способами измерить время работы алгоритма.