

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: Тема работы
Вариант 1

Выполнила:
Данилова Айаана
К314

Проверила:
Артамонова В.Е.

Санкт-Петербург
2024 г.

Содержание отчета

Содержание отчета	2
Задачи по варианту	3
Задача №1. Ввод-вывод	
1. Подзадача $a + b$	
2. Подзадача $a + b^2$	
3. Подзадача $a + b$ с использованием файлов	
4. Подзадача $a + b^2$ с использованием файлов	
Задача №2. Число Фибоначчи	
Задача №3. Еще про числа Фибоначчи	
Задача №4. Тестирование алгоритмов.	8
Вывод	9

Задачи по варианту

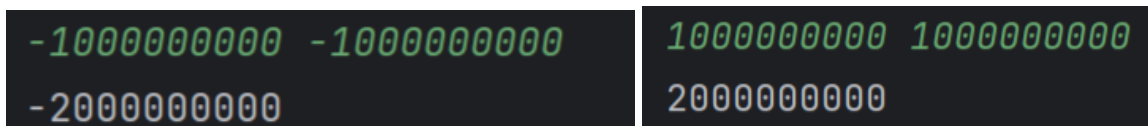
Задача №1. Ввод – вывод

Подзадача 1.

```
def first_1():
    a, b = map(int, input().split())
    proverka = (-10e9 <= a <= 10e10 and -10e9 <= b <= 10e9)
    kolvo = 0
    while proverka == False and kolvo < 2:
        print('Попробуйте еще раз')
        kolvo += 1
        a, b = map(int, input().split())
        proverka = (-10e9 <= a <= 10e10 and -10e9 <= b <= 10e9)
        if kolvo == 3:
            break
    else:
        print(a + b)
)
```

Считываю строку и присваиваю переменным *a* и *b* целочисленное значение с помощью *map*, который преобразовывает вводные данные, разделенные методом *split()*, в *int*. Проверяю удовлетворяют ли числа условиям задачи, дается три попытки на ввод корректных значений. Если все хорошо, то вывожу *a + b*.

Результат работы кода на максимальных и минимальных значениях:



The screenshot shows two lines of green text on a black background. The first line contains the minimum values: `-1000000000 -1000000000`. The second line contains the maximum values: `1000000000 1000000000`. Below these, the program's output is shown: `-2000000000` for the minimum case and `2000000000` for the maximum case.

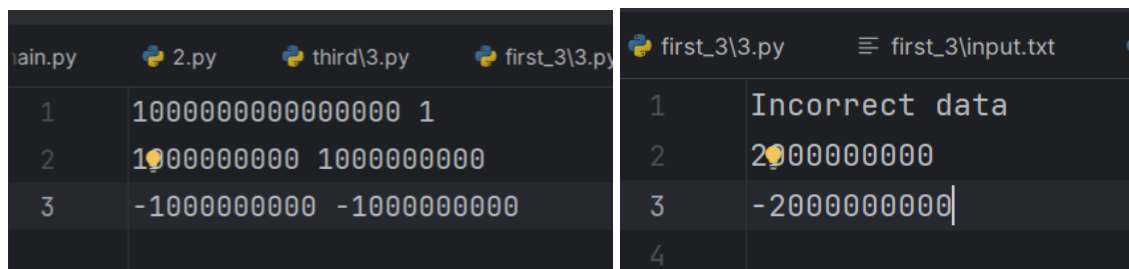
	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	< 1ns	14.7 mb
Верхняя граница диапазона значений входных данных из текста задачи	< 1ns	14.9 mb

Подзадача 2.

```
def first_2():
    a, b = map(int, input().split())
    proverka = (-10e9 <= a <= 10e10 and -10e9 <= b <= 10e9)
    kolvo = 0
    while proverka == False and kolvo < 2:
        print('Попробуйте еще раз')
        kolvo += 1
        a, b = map(int, input().split())
        proverka = (-10e9 <= a <= 10e10 and -10e9 <= b <= 10e9)
        if kolvo == 3:
            break
    else:
        print(a + b ** 2)
```

Считываю строку и присваиваю переменным a и b целочисленное значение с помощью `map`, который преобразовывает вводные данные, разделенные методом `split()`, в `int`. Проверяю удовлетворяют ли числа условиям задачи, дается три попытки на ввод корректных значений. Если все хорошо, то вывожу $a + b^2$.

Результат работы кода на некорректных, максимальных и минимальных значениях:



	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	< 1ns	14.9 mb
Верхняя граница диапазона значений входных данных из текста задачи	< 1ns	15 mb

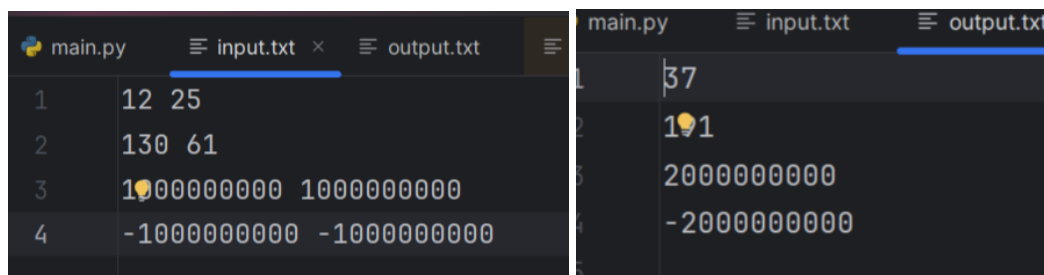
Подзадача 3.

```
with open('input.txt', 'r') as file:
    for line in file:
        a, b = map(int, line.split())
        with open('output.txt', 'a') as ans:
            if abs(a) > 10e9 or abs(b) > 10e9:
                ans.write('Incorrect data' + '\n')
            else:
                ans.write(str(a + b) + '\n')
```

С помощью `with_open()` считываю данные с файла `input.txt`. Получаю текст `file`. Из строки `line` полученного `file` присваиваю числа в переменные `a`, `b`. Получаю текст `file`. Из строки `line` полученного `file` присваиваю числа в переменные `a`, `b`.

Делаю проверку на корректность данных. В случае, когда значение выходит за ограничение условия задачи, дописываю в `output.txt` 'Incorrect data'. Распаковываю `output.txt` как `ans`. В файл с ответом, `ans`, вписываю `a + b`.

Результат работы:



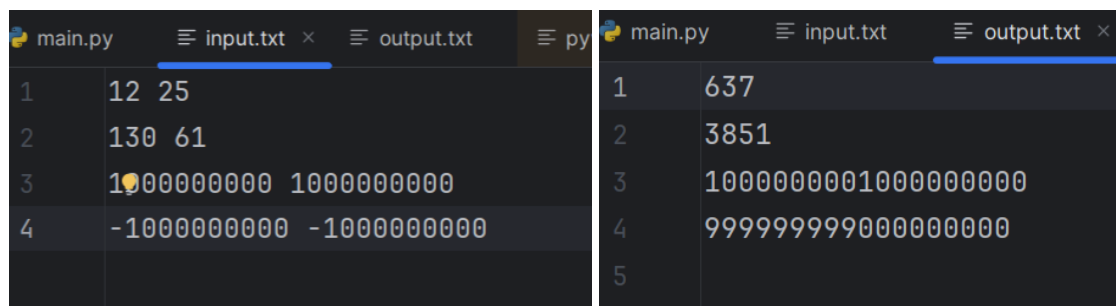
	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.006 s	15.1mb
Пример из задачи	0.006 s	15.4mb
Пример из задачи	0.005 s	15.4 mb
Верхняя граница диапазона значений входных данных из текста задачи	0.007 s	15.5mb

Подзадача 4.

```
with open('input.txt', 'r') as file:
    for line in file:
        a, b = map(int, line.split())
        with open('output.txt', 'a') as ans:
            if abs(a) > 10e9 or abs(b) > 10e9:
                ans.write('Incorrect data' + '\n')
            else:
                ans.write(str(a + b ** 2) + '\n')
```

С помощью `with_open()` считываю данные с файла `input.txt`. Получаю текст `file`. Из строки `line` полученного `file` присваиваю числа в переменные `a`, `b`. Делаю проверку на корректность данных. В случае, когда значение выходит за ограничение условия задачи, дописываю в `output.txt` 'Incorrect data'. Распаковываю `output.txt` как `ans`. В файл с ответом дописываю(тк для меня удобнее все тесты в один файлик записать) $a + b^2$.

Результат работы кода:



main.py	input.txt	output.txt
1	12 25	12 25
2	130 61	130 61
3	1000000000 1000000000	1000000000 1000000000
4	-10000000000 -10000000000	-10000000000 -10000000000

main.py	input.txt	output.txt
1	637	637
2	3851	3851
3	1000000000 1000000000	1000000000 1000000000
4	999999999 0000000000	999999999 0000000000
5		

	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.007s	15.4mb
Пример из задачи	0.006 s	15.4mb
Пример из задачи	0.005 s	15.3 mb
Верхняя граница диапазона значений входных данных из текста задачи	0.006 s	15.5mb

Вывод по задаче: узнала как считывать с файла на питоне.

Задача №2. Число Фибоначчи

```

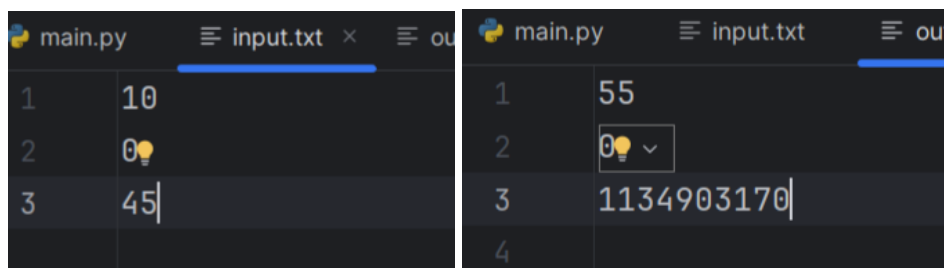
import time

with open('input.txt') as file:
    for line in file:
        n = int(line)
        if not(0 <= n <= 45):
            with open('output.txt', 'a') as ans:
                ans.write('Incorrect data' + '\n')
        else:
            if n == 0:
                with open('output.txt', 'a') as ans:
                    ans.write('0' + '\n')
            else:
                a = [0, 1]
                for i in range(2, n + 1):
                    x = a[i - 1] + a[i - 2]
                    a.append(x)
                with open('output.txt', 'a') as ans:
                    ans.write(str(a[-1]) + '\n')

t_start = time.perf_counter()
print("Время работы %s:" % (time.perf_counter() - t_start))

```

Считываю n . Делаю проверку на корректность данных. В случае, когда значение выходит за ограничение условия задачи, дописываю в output.txt 'Incorrect data'. Если все хорошо, то проверяю равен ли n нулю. При равенстве дописываю 0 в output.txt, иначе создаю список $a = [0, 1]$ и в цикле for, элементам списка с индексами от 2 до n присваиваю $a[i - 1] + a[i - 2]$. Каждый раз новое число Фибоначчи добавляю в конец списка a . Таким образом нахожу F_n и дописываю в файл ответ.



	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.0039 s	15.2mb
Пример из задачи	0.0048 s	15.1mb

Верхняя граница диапазона значений входных данных из текста задачи	0.004 s	15.2mb
---	---------	--------

Вывод по задаче: вспомнила как вычислять числа Фибоначчи.

Задача №3. Еще про число Фибоначчи

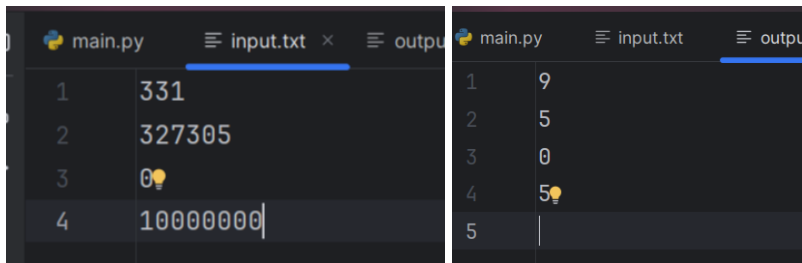
```
import time

with open('input.txt') as file:
    for line in file:
        n = int(line)
        if not(0 <= n <= 10e7):
            with open('output.txt', 'a') as ans:
                ans.write('Incorrect data' + '\n')
        else:
            if n == 0:
                with open('output.txt', 'a') as ans:
                    ans.write('0' + '\n')
            else:
                a = [0, 1]
                for i in range(2, n + 1):
                    x = a[i - 1] + a[i - 2]
                    a.append(x % 10)
                with open('output.txt', 'a') as ans:
                    ans.write(str(a[-1]) + '\n')

t_start = time.perf_counter()
print("Время работы %s:" % (time.perf_counter() - t_start))
```

Считываю n . Делаю проверку на корректность данных. В случае, когда значение выходит за ограничение условия задачи, дописываю в output.txt 'Incorrect data'. Если все хорошо, то проверяю равен ли n нулю. При равенстве дописываю 0 в output.txt, иначе создаю список $a = [0, 1]$ и в цикле for, элементам списка с индексами от 2 до n присваиваю $a[i - 1] + a[i - 2]$. В случае, когда полученный результат больше или равен 10, то уменьшаю его значение на 10. Каждый раз добавляю остаток нового числа Фибоначчи в конец списка a . Таким образом нахожу F_n и дописываю в файл ответ.

Результат работы кода:



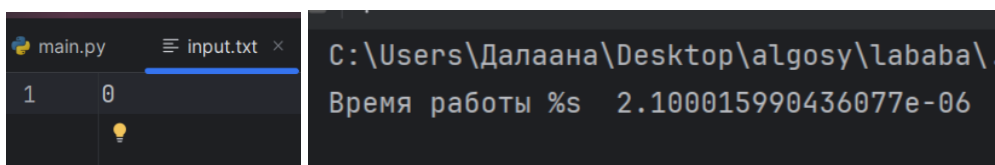
	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.005 s	15.4 mb
Пример из задачи	0.006 s	15.4 mb
Пример из задачи	0.007 s	15.5 mb
Верхняя граница диапазона значений входных данных из текста задачи	0.008 s	15.5 mb

Вывод по задаче: вспомнила остатки и операции над ними

Задача №4. Тестирование алгоритмов

```
def time_test():
    import time
    t_start = time.perf_counter()
    print("Время работы %s ", (time.perf_counter() - t_start))
```

Подключаем библиотеку time. Фиксируем в переменной t_start время начала с помощью time.perf_counter(). При выводе вычитаем из времени окончания работы t_start. Непосредственно функцию вызываем после second() и third().



Вывод

Благодаря вводной лабораторной работе я повторила основу основ питона и узнала, как считывают данные с файла на данном языке программирования, получила опыт пользования функциями библиотеки sys и time в целях определения количества затраченного времени и места памяти на компьютере.