

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ФАКУЛЬТЕТ ИНФОКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ

Отчет по лабораторной работе №1
по курсу «Алгоритмы и структуры данных»
Тема: Введение

Выполнил:
Волжева М.И.
К3141

Проверила:
Артамонова В.Е.

Санкт-Петербург
2022 г.

Содержание отчета

Содержание отчета	2
Задачи по варианту	3
Задача №1. Ввод-вывод	3
Задача №2. Число Фибоначчи	6
Задача №3. Ещё про числа Фибоначчи	9
Вывод	12

Задачи по варианту

Задача №1. Ввод-вывод

Текст задачи:

1. 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 $a + b$.
- Пример:

Input.txt	12 25	130 61
Output.txt	37	191

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

- Пример:

Input.txt	12 25	130 61
Output.txt	637	3851

Листинг кода:

```
1. f = open("1_1_input.txt")
   m = open("1_1_output.txt", "w")
   num = f.read()
   a, b = num.split()
   m.write(str(int(a)+int(b)))

   f.close()
   m.close()





2. f = open("1_2_input.txt")
   m = open("1_2_output.txt", "w")
   num = f.read()
   a, b = num.split()
   m.write(str(int(a)+int(b)**2))

   f.close()
   m.close()
```





Текстовое объяснение решения:

Я прочла числа из файла, выполнила необходимые операции и вывела результат в файл.

Результат работы кода на примерах из текста задачи:
1.





 1_1_input.txt – Блокнот Файл Правка Формат Вид Справка 12 25	 1_1_input.txt – Блокнот Файл Правка Формат Вид Справка 130 61
 1_1_output.txt – Блокнот Файл Правка Формат Вид Справка 37	 1_1_output.txt – Блокнот Файл Правка Формат Вид Справка 191

2.





 1_2_input.txt – Блокнот Файл Правка Формат Вид Справка 12 25	 1_2_input.txt – Блокнот Файл Правка Формат Вид Справка 130 61
 1_2_output.txt – Блокнот Файл Правка Формат Вид Справка 637	 1_2_output.txt – Блокнот Файл Правка Формат Вид Справка 3851

Результат работы кода на максимальных и минимальных значениях:

1.

 1_1_input.txt – Блокнот Файл Правка Формат Вид Справка -1000000000 -1000000000	 1_1_input.txt – Блокнот Файл Правка Формат Вид Справка 1000000000 1000000000
 1_1_output.txt – Блокнот Файл Правка Формат Вид Справка -2000000000	 1_1_output.txt – Блокнот Файл Правка Формат Вид Справка 2000000000

2.

 1_2_input.txt – Блокнот Файл Правка Формат Вид Справка -1000000000 -1000000000	 1_2_input.txt – Блокнот Файл Правка Формат Вид Справка 1000000000 1000000000
 1_2_output.txt – Блокнот Файл Правка Формат Вид Справка 999999999000000000	 1_2_output.txt – Блокнот Файл Правка Формат Вид Справка 1000000001000000000

Вывод по задаче: Мы научились читать данные из файла и записывать их в файл. Выполнять базовые математические процедуры.

Задача №2. Число Фибоначчи

Текст задачи:

1. Каждое число Фибоначчи представляет собой сумму двух предыдущих, что дает последовательность 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...
2. Ваша цель – разработать эффективный алгоритм для подсчета чисел Фибоначчи.
 - 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
 - Пример:

Input.txt	2	6
Output.txt	1	8

•

Листинг кода:

```
import time
import os, psutil

t_start = time.perf_counter()
process = psutil.Process(os.getpid())

results = [0]*10000
def calc_fib(n):
    if results[n] == 0:
        if n == 1 or n == 2:
            results[n] = 1
        if n > 2:
            results[n] = calc_fib(n - 1) + calc_fib(n - 2)
    return int(results[n])





f = open("2_input.txt")
m = open("2_output.txt", "w")
num = f.read()
m.write(str(calc_fib(int(num))))
f.close()
m.close()
print("Time of working: %s second" % (time.perf_counter() - t_start))
print((process.memory_info().rss)/(1024*1024))
```

Текстовое объяснение решения:





Результаты вычисления чисел в последовательности Фибоначчи сохраняются в массив, тем самым при подсчете следующего числа Фибоначчи

происходит только сложение двух предыдущих, а не вычисление предыдущих чисел с нуля.

Результат работы кода на примерах из текста задачи:

 2_input.txt – Блокнот Файл Правка Формат Вид Справка 2  2_output.txt – Блокнот Файл Правка Формат Вид Справка 1	 2_input.txt – Блокнот Файл Правка Формат Вид Справка 6  2_output.txt – Блокнот Файл Правка Формат Вид Справка 8
---	---

Результат работы кода на максимальных и минимальных значениях:

 2_input.txt – Блокнот Файл Правка Формат Вид Справка 0  2_output.txt – Блокнот Файл Правка Формат Вид Справка 0	 2_input.txt – Блокнот Файл Правка Формат Вид Справка 45  2_output.txt – Блокнот Файл Правка Формат Вид Справка 1134903170
---	---

Тестирование алгоритма:

	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.015148700000000015 second	13.37890625 mb3
Пример из задачи	0.003082899999999995 second	13.37890625 mb
Пример из задачи	0.0037092999999999987 second	13.37890625 mb
Верхняя граница диапазона значений входных данных из текста задачи	0.004147799999999993 second	13.3828125 mb

Вывод по задаче: Я изучила алгоритм поиска числа Фибоначчи по порядковому номеру и протестировала реализацию алгоритма на затрачиваемую память и время.

Задача №3. Ещё про числа Фибоначчи

Текст задачи:

1. Определение последней цифры большого числа Фибоначчи. Числа Фибоначчи растут экспоненциально. Например, $F_{200} = 280571172992510140037611932413038677189525$
2. Хранить такие суммы в массиве, и при этом подсчитывать сумму, будет до статочно 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
- Ограничение по времени: 5сек.
- Ограничение по памяти: 512 мб
- Пример:

Input.txt	331	327305
Output.txt	9	5

Листинг кода

```
import time
import os, psutil

t_start = time.perf_counter()
process = psutil.Process(os.getpid())

results = [0]*10000
def calc_fib(n):
    if results[n] == 0:
        if n == 1 or n == 2:
            results[n] = 1
        if n > 2:
            results[n] = calc_fib(n - 1) + calc_fib(n - 2)
    return int(results[n])



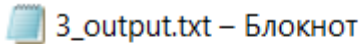
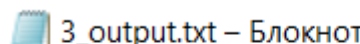
f = open("2_input.txt")
m = open("2_output.txt", "w")
num = f.read()
m.write(str(calc_fib(int(num))))
f.close()
m.close()
print("Time of working: %s second" % (time.perf_counter() - t_start))
print((process.memory_info().rss)/(1024*1024))
```

Текстовое объяснение решения:



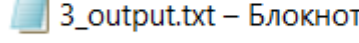
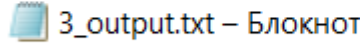
Результаты вычисление чисел в последовательности Фибоначчи сохраняется в массив, тем самым при подсчете следующего числа Фибоначчи

происходит только сложение двух предыдущих, а не вычисление предыдущих чисел с нуля.

Результат работы кода на примерах из текста задачи:

 331	 327305
 9	 5

Результат работы кода на максимальных и минимальных значениях:

 0	 10000000
 0	 5

Тестирование алгоритма:

	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи	0.001947999999999914 second	13.31640625 mb
Пример из задачи	0.003307999999999915 second	13.3203125 mb
Пример из задачи	0.0315709 second	13.32421875 mb
Верхняя граница диапазона значений входных данных из текста задачи	0.5660073999999999 second	13.31640625 mb

Вывод по задаче: Я изучила алгоритм поиска последней цифры в конкретном числе Фибоначчи и протестировала реализацию алгоритма на затрачиваемую память и время.

Вывод

Я узнала, как работают некоторые алгоритмы, научилась писать отчет по лабораторной работе и тестировать свои реализации алгоритмов на затрачиваемое время и память.