

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: Сортировка вставками, выбором, пузырьковая  
Вариант 1

Выполнил:  
Останин А. С.  
К3140

Проверил:  
Афанасьев А. В.

Санкт-Петербург  
2024 г.

## **Содержание отчета**

<b>Содержание отчета</b>	<b>2</b>
<b>Задачи по варианту</b>	<b>3</b>
Задача №1. Сортировка вставкой	
Задача №4. Линейный поиск	
Задача №9. Сложение двоичных чисел	
<b>Вывод</b>	<b>5</b>

## Задачи по варианту

### Задача №1. Сортировка вставкой

Используя код процедуры Insertion-sort, напишите программу и проверьте сортировку массива  $A = \{31, 41, 59, 26, 41, 58\}$ .

```
import time, tracemalloc

def checker(ar):
    for i in ar:
        if abs(i) > 10**9:
            return False
    return True

def insertionsort(file_in, file_out, file_t):
    t_start = time.perf_counter()
    tracemalloc.start()

    file_info = open(file_t, 'w+')
    file_input = open(file_in, 'r')
    file_output = open(file_out, 'w+')

    n = int(file_input.readline())
    s = [int(el) for el in file_input.readline().split()]

    if not (1 <= n <= 10**3):
        file_output.write('Неверное первое число')
        file_output.close()
        exit('Неверное первое число')

    if n != len(s):
        file_output.write('Количество элементов не совпадает с заданным значением')
        file_output.close()
        exit('Количество элементов не совпадает с заданным значением')

    if not checker(s):
        file_output.write('Одно из чисел выходит за ограничение')
        file_output.close()
        exit('Одно из чисел выходит за ограничение')

    for i in range(1, len(s)):
        value = s[i]
        j = i - 1
        while value < s[j] and j >= 0:
            s[j+1] = s[j]
            j -= 1
        s[j+1] = value

    for elem in s:
        file_output.write(f'{elem} ')

    file_info.write(f'время выполнения: {time.perf_counter() - t_start}\nзатраченная память: {tracemalloc.get_traced_memory()[1]/2**20} МБ')

    file_input.close()
    file_output.close()
```

Для начала сделаем проверки чисел на выход за ограничение

```
def checker(ar):
    for i in ar:
        if abs(i) > 10**9:
            return False
    return True

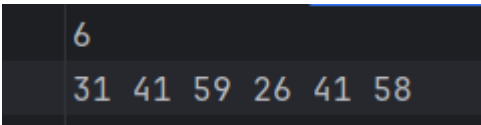
if not (1 <= n <= 10**3):
    file_output.write('Wrong first number')
    file_output.close()
    exit('Wrong first number')

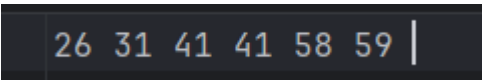
if not checker(s):
    file_output.write('One of the numbers goes out of bounds')
    file_output.close()
    exit('One of the numbers goes out of bounds')
```

Теперь напомним сам алгоритм сортировки

```
for i in range(1, len(s)):
    value = s[i]
    j = i - 1
    while value < s[j] and j >= 0:
        s[j+1] = s[j]
        j -= 1
    s[j+1] = value
```

Будем проходить по длине списка, начиная с первого индекса, далее записываем значение в переменную value в этом индексе и считаем j, который будет правой границей отсортированной части списка. Теперь будем брать записанное в value значение и искать ему место в отсортированной части списка, смещая элементы.

Input: 

Output: 

	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи		
Пример из задачи	0.001395399998727953 с	0.035531044006347656 Мб
Верхняя граница диапазона значений входных данных из текста задачи		

Вывод по задаче: Написан алгоритм сортировки вставкой, который выполняется за  $O(n^2)$

#### Задача №4. Линейный поиск

Рассмотрим задачу поиска.

- **Формат входного файла.** Последовательность из  $n$  чисел  $A = a_1, a_2, \dots, a_n$  в первой строке, числа разделены пробелом, и значение  $V$  во второй строке. Ограничения:  $0 \leq n \leq 10^3$ ,  $-10^3 \leq a_i$ ,  $V \leq 10^3$

```
def checker(ar):
    for i in ar:
        if abs(i) > 10**3:
            return False
    return True

def linearfinder(file_in, file_out):
    file_input = open(file_in, 'r')
    file_output = open(file_out, 'w+')

    s = [int(i) for i in file_input.readline().strip().split()]
    v = int(file_input.readline())

    if not(0 <= len(s) <= 10**3):
        file_output.write('Слишком много элементов')
        file_output.close()
        exit('Слишком много элементов')

    if abs(v) > 3**10:
        file_output.write('Число, которое необходимо найти, выходит за
ограничение')
        file_output.close()
```

```

        exit('Число, которое необходимо найти, выходит за ограничение')

    if not checker(s):
        file_output.write('Одно из чисел выходит за ограничение')
        file_output.close()
        exit('Одно из чисел выходит за ограничение')

    res_i = []
    for i in range(len(s)):
        if s[i] == v:
            res_i.append(i)

    for i in range(len(res_i)):
        if i == len(res_i) - 1:
            file_output.write(str(res_i[i]))
        else:
            file_output.write(str(res_i[i]) + ', ')

    file_input.close()
    file_output.close()

```

Проходим по списку и ищем совпадение с нужным значением, после чего записываем индекс в список с результатом

## Задача №9. Сложение двоичных чисел

```

import time

def list_to_str(ar):
    res = ''
    for i in ar:
        res += str(i)
    return str(int(res))

def str_to_list(s):
    s = str(s)
    res = []
    for i in s:
        res.append(i)
    return res

def bin_sum(A, B):
    C = []
    t = 0
    A.reverse()
    B.reverse()
    for i in range(len(A)):
        a = int(A[i])
        b = int(B[i])
        c = a + b + t
        C.append(c % 2)
        t = c // 2
    C.append(t)
    return C[::-1]

def main binsum(file_in, file_out, file_t):
    t_start = time.perf_counter()

```

```

file_time = open(file_t, 'w+')
file_input = open(file_in, 'r')
file_output = open(file_out, 'w+')

n1, n2 = map(str, file_input.read().split())
if (not (1 <= len(n1) <= 10 ** 3)) or (not (1 <= len(n2) <= 10 ** 3)):
    file_output.write('Одно из чисел выходит за ограничение')
    exit('Одно из чисел выходит за ограничение')

A, B = str_to_list(n1), str_to_list(n2)

C = bin_sum(A, B)
file_output.write(list_to_str(C))

file_time.write(f'время выполнения: {time.perf_counter() - t_start}')

file_input.close()
file_output.close()
file_time.close()

```

```

C = []
t = 0
A.reverse()
B.reverse()
for i in range(len(A)):
    a = int(A[i])
    b = int(B[i])
    c = a + b + t
    C.append(c % 2)
    t = c // 2
C.append(t)
return C[::-1]

```

Так как длина обоих чисел одинакова, значит будем проходить по индексу  $i$  и складывать значения на этих индексах, записывая в список  $C$ , при этом перенося единицу на следующий разряд, если сумма будет равна двум.

## Вывод

При выполнении этой лабораторной, мы научились реализовывать алгоритм сортировки вставкой, линейный поиск и сложение двоичных чисел