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

Выполнил:

Юшков А.М.

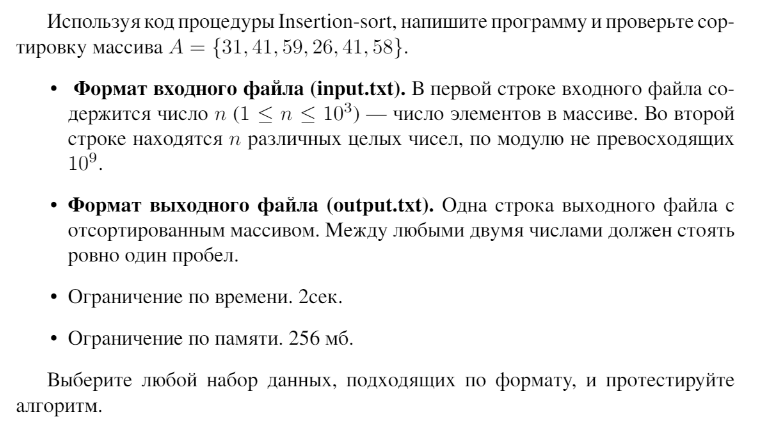
К3139

Проверил:

Санкт-Петербург

2024 г.

**Задача №1. Сортировка вставкой**



Код:

import time, psutil  
t\_start = time.perf\_counter()  
with open("input.txt", "r") as file:  
 n = int(file.readline())  
 a = [int(i) for i in file.readline().split()]  
for i in range(1, n):  
 x = a[i]  
 j = i  
 while j > 0 and a[j - 1] > x:  
 a[j] = a[j - 1]  
 j -= 1  
 a[j] = x  
ans = str(a[0])  
for i in range (n-1):  
 ans = ans + ' ' + str(a[i+1])  
with open("output.txt", "w") as file:  
 file.write(ans)  
  
print("Время работы: %s секунд " % (time.perf\_counter() - t\_start))  
print(psutil.Process().memory\_info().rss/1024\*\*2)

1. Открываем файл и считываем данные
2. Cчитываем значения из файла *input.txt* и записываем в *n* и *а*
3. Вывполняем сортировку

- Основной цикл начинается со второго элемента. Для каждого элемента определяется его позиция относительно уже отсортированной части массива слева от него путем перемещения больших значений вправо до нахождения подходящей позиции для текущего значения (х).

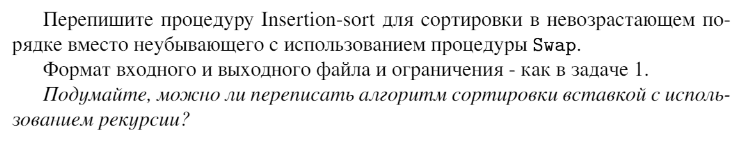
- Далее открываем файл *output.txt*  и записываем значения

1. После программа выводит время работы программы и затраченный обьем озу

Вывод по задаче: В ходе выполнения задания был написан алгоритм, выполняющий сортировку вставками в заданном диапазоне значений, алгоритм выводит время работы и затраченный обьем озу

**Задача №3. Сортировка вставкой по убыванию**



import time, psutil  
t\_start = time.perf\_counter()  
with open("input.txt", "r") as file:  
 n = int(file.readline())  
 a = [int(i) for i in file.readline().split()]  
def swap(list, i, j):  
 list[i], list[j] = list[j], list[i]  
 return list  
for i in range(1, n):  
 x = a[i]  
 j = i  
 while j > 0 and a[j - 1] < x:  
 a[j] = a[j - 1]  
 j -= 1  
 a[j] = x  
ans = str(a[0])  
for i in range (n-1):  
 ans = ans + ' ' + str(a[i+1])  
with open("output.txt", "w") as file:  
 file.write(ans)  
print("Время работы: %s секунд " % (time.perf\_counter() - t\_start))  
print(psutil.Process().memory\_info().rss/1024\*\*2)

* Ход работы алгоритма аналогичен первому заданию, за исключением наличия фуккции Swap и того, что алгоритм сортируется по невозрастанию

Результат:

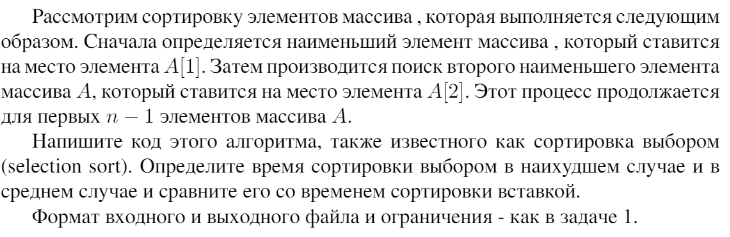






Вывод по задаче: В ходе выполнения задания был модифицирован код к задаче 1, задача соответствует условиям

**Задача №5. Сортировка выбором**



import time, psutil  
t\_start = time.perf\_counter()  
with open("input.txt", "r") as file:  
 n = int(file.readline())  
 a = [int(i) for i in file.readline().split()]  
def selection\_sort(n, a):  
 for i in range(n):  
 min\_index = i  
 for j in range(i + 1, n):  
 if a[j] < a[min\_index]:  
 min\_index = j  
 a[i], a[min\_index] = a[min\_index], a[i]  
 ans = str(a[0])  
 for i in range(n - 1):  
 ans = ans + ' ' + str(a[i + 1])  
 with open("output.txt", "w") as file:  
 file.write(ans)  
selection\_sort(n, a)  
print("Время работы: %s секунд " % (time.perf\_counter() - t\_start))  
print(psutil.Process().memory\_info().rss/1024\*\*2)

1. Из файла input достаем количество чилел и сами числа, записываем их в массив
2. *selection\_sort:*

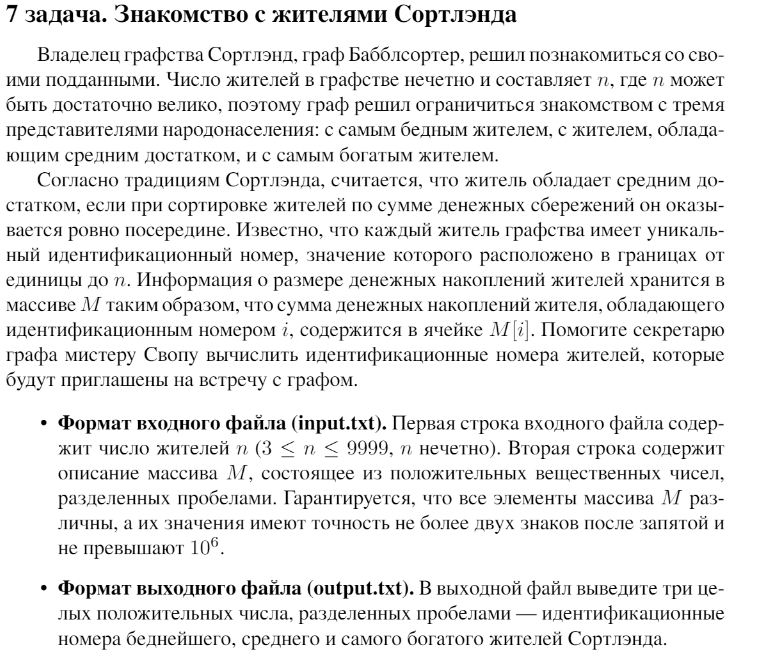
* функция на вход принимает два элемента, после чего выполняет следующий алгоритм: ищет минимальный элемент и ставит его в начало массива, остальные элементы сдвигает
  1. Далее открываем файл *output.txt*  и записываем в него получившиеся значения

Результат:





Вывод по задаче: в ходе работы над задачей я разобрал принцип сортировки выбором и реализовал его



Код

import time, psutil  
t\_start = time.perf\_counter()  
with open("input.txt", "r") as file:  
 n = int(file.readline())  
 a = [int(i) for i in file.readline().split()]  
def bubble\_sort(n, a):  
 for i in range(n):  
 for j in range(0, n - i - 1):  
 if a[j] > a[j + 1]:  
 a[j], a[j + 1] = a[j + 1], a[j]  
 ans = str(a[0]) + ' ' + str(a[n // 2]) + ' ' + str(a[-1])  
 with open("output.txt", "w") as file:  
 file.write(ans)  
  
bubble\_sort(n, a)  
print("Время работы: %s секунд " % (time.perf\_counter() - t\_start))  
print(psutil.Process().memory\_info().rss/1024\*\*2)

1 Из файла input достаем количество чилел и сами числа, записываем их в массив

*2 bubblesort:*

функция на вход принимает два элемента, после чего выполняет следующий алгоритм: сравнивает два соседних числа, если условия выполняются то числа меняются местами

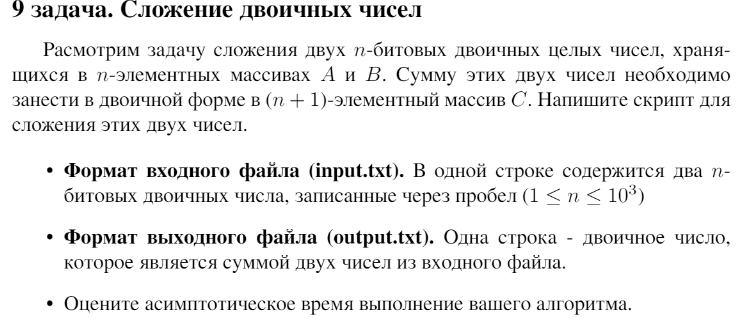
3 Далее открываем файл *output.txt*  и записываем в него первое число, число из середины, и последнее число

Вывод





Вывод: В ходе работы над данной задачей я познакомился с сортировкой пузырьком



Код

import time, psutil  
t\_start = time.perf\_counter()  
with open('input.txt', 'r') as file:  
 input\_string = file.readline().strip()  
a, b = map(str, input\_string.split())  
def sum(a, b):  
 n = len(a)  
 c = [0] \* (n + 1)  
 flag = 0  
 for i in range(n - 1, -1, -1):  
 s = int(a[i]) + int(b[i]) + flag  
 c[i + 1] = s % 2  
 flag = s // 2  
 c[0] = flag  
 ans = str(c[0])  
 for i in range(n):  
 ans = ans + str(c[i + 1])  
 with open("output.txt", "w") as file:  
 file.write(ans)  
sum(a, b)  
print("Время работы: %s секунд " % (time.perf\_counter() - t\_start))  
print(psutil.Process().memory\_info().rss/1024\*\*2)

1 Из файла input достаем два числа

*2 sum:*

функция на вход принимает два элемента, запускает цикл по длинне чисел, после чего выполняет следующий алгоритм: складывает значения из I – ой ячейка двух чисел и остаток от деления на 2 записывается в новое число, в флаг записывается деление нацело на 2

3 Далее открываем файл *output.txt*  и записываем в него полученное число

Пример:





**Вывод**

 В ходе выполнения лабораторной работы были изучены методы сортировки вставкой, сортировкой и пузырьком, а так же метод сложения двоичных чисел.