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:

Аксянова А.Р

К3140

Проверил:

Афанасьев А.В.

Санкт-Петербург

2024 г.

# Содержание отчета

[Содержание отчета 2](#_Toc178798681)

[Задачи 3](#_Toc178798682)

[Задача №1. Сортировка вставкой 3](#_Toc178798683)

[Задача №2. Сортировка вставкой + 4](#_Toc178798684)

[Задача №4. Линейный поиск 5](#_Toc178798685)

[Вывод 7](#_Toc178798686)

# Задачи

## Задача №1. Сортировка вставкой

1. Текст задачи

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

* **Формат входного файла (input.txt).** В первой строке входного файла содержится число *n* (1 ≤ *n* ≤ 103) — число элементов в массиве. Во второй строке находятся *n* различных целых чисел, по модулю не превосходящих

109.

* **Формат выходного файла (output.txt).** Одна строка выходного файла с отсортированным массивом. Между любыми двумя числами должен стоять ровно один пробел.
* Ограничение по времени. 2сек.
* Ограничение по памяти. 256 мб.

Выберите любой набор данных, подходящих по формату, и протестируйте алгоритм.

1. Листинг кода

import time  
start = time.perf\_counter()  
  
k = open("../tests/input.txt")  
numb = int(k.readline())  
b = k.readline().split(" ")  
s = [int(l) for l in b ]  
  
for i in range(1, numb):  
 elem = s[i]  
 j = i  
 while j >= 1 and s[j - 1] > elem:  
 s[j] = s[j - 1]  
 j -= 1  
 s[j] = elem  
 print(j)  
otv = " ".join(str(t) for t in s)  
print('Отсортированный список:', otv)  
  
y = open("../tests/output.txt", "w")  
y.write(otv)  
  
stop = time.perf\_counter()  
print("time: %s ms" % (stop - start))

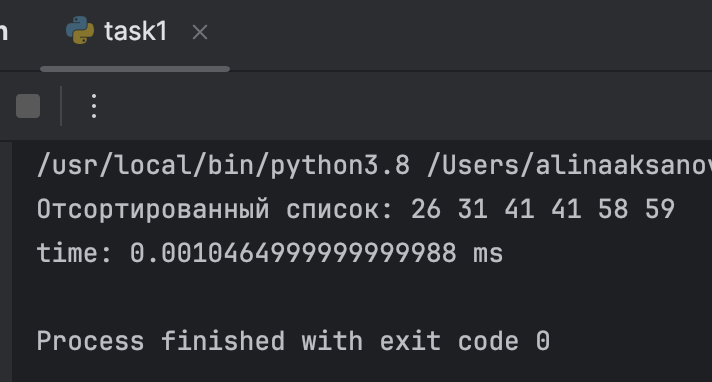
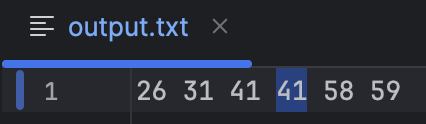
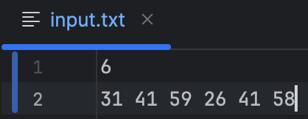
1. Текстовое объяснение решения.

Открываю для считывания файл input с числами, в переменную numb вношу число элементов, далее создаю список, в который заношу числа из второй строки файла. Приступаю к сортировке вставкой: беру первый элемент из неотсортированной части списка, далее, пока элемент слева существует и больше текущего элемента, смещаю j-й элемент отсортированной части в право, идем влево и ищем место для текущего элемента, когда место найдено, вставляем его на индекс j в отсортированной части.

Открываю для записи файл output, записываю туда результат работы.

Также я замерила время выполнения кода.

1. Результат работы кода на примерах:



1. Вывод по задаче:

В результате выполнения задачи я написала алгоритм, который с помощью сортировки вставкой, сортирует массив и выводит его в порядке возрастания

## Задача №2. Сортировка вставкой +

1. Текст задачи

Измените процедуру Insertion-sort для сортировки таким образом, чтобы в выходном файле отображалось в первой строке n чисел, которые обозначают новый индекс элемента массива после обработки.

* **Формат выходного файла (input.txt).**В первой строке выходного файла выведите *n* чисел. При этом *i*-ое число равно индексу, на который, в момент обработки его сортировкой вставками, был перемещен *i*-ый элемент исходного массива. Индексы нумеруются, начиная с единицы. Между любыми двумя числами должен стоять ровно один пробел.

Пример.

|  |  |
| --- | --- |
| input.txt | output.txt |
| 10 | 1 2 2 2 3 5 5 6 9 1 |
| 1 8 4 2 3 7 5 6 9 0 | 0 1 2 3 4 5 6 7 8 9 |

1. Листинг кода

import time  
start = time.perf\_counter()  
  
k = open("../tests/input.txt")  
numb = int(k.readline())  
b = k.readline().split(" ")  
s = [int(l) for l in b ]  
index = [1]  
for i in range(1, numb):  
 elem = s[i]  
 j = i  
 while j >= 1 and s[j - 1] > elem:  
 s[j] = s[j - 1]  
 j -= 1  
 index.append(j+1)  
 s[j] = elem  
  
otv = " ".join(str(t) for t in s)  
list\_ind = " ".join(str(o) for o in index)  
  
print("Новые индексы: ",list\_ind)  
print('Отсортированный список:', otv)  
  
y = open("../tests/output.txt", "w")  
y.write(list\_ind+"\n")  
y.write(otv)  
  
stop = time.perf\_counter()  
print("time: %s ms" % (stop - start))

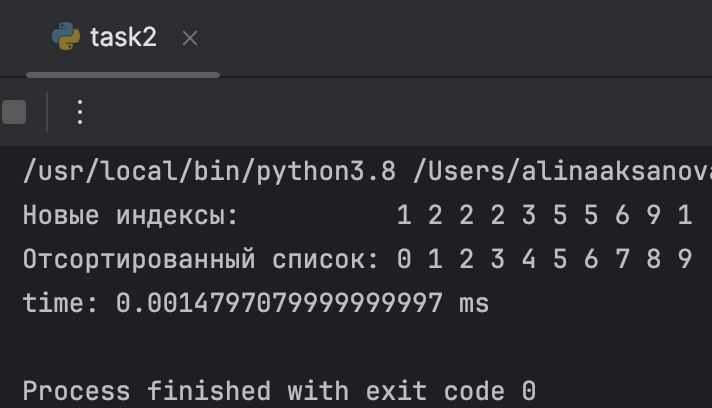
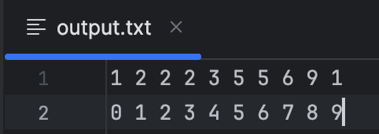
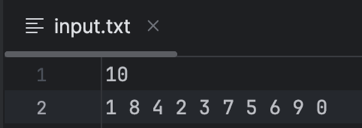
1. Текстовое объяснение решения.

Открываю для считывания файл с числами, в переменную numb вношу число элементов, далее создаю список, в который заношу числа из второй строки файла. Приступаю к сортировке вставкой аналогично задаче 1. Чтобы запоминать новые индексы элементов, создаю список ,уже содержащий новый индекс первого элемента в сортировке.

Открываю для записи файл output, записываю туда результат работы.

Также я замерила время выполнения кода.

1. Результат работы кода на примерах из текста задачи:



1. Вывод по задаче:

Аналогично задаче 1 в результате выполнения задачи я написала алгоритм, который с помощью сортировки вставкой, сортирует массив и выводит его в порядке возрастания, а так же выводит новый индекс элемента массива после обработки.

## Задача №4. Линейный поиск

1. Текст задачи

Рассмотрим задачу поиска.

1. **Формат входного файла**. Последовательность из *n* чисел *A* = *a*1*,a*2*,. . . ,an* в первой строке, числа разделены пробелом, и значение *V* во второй строке. Ограничения: 0 ≤ *n* ≤ 103, −103 ≤ *ai,V* ≤ 103
2. **Формат выходного файла**. Одно число - индекс *i*, такой, что *V* = *A*[*i*], или значение −1, если *V* в отсутствует.
3. Напишите код линейного поиска, при работе которого выполняется сканирование последовательности в поисках значения *V* .
4. Если число встречается несколько раз, то выведите, сколько раз встречается число и все индексы *i* через запятую.
5. Дополнительно: попробуйте найти свинью, как в лекции. Используйте во входном файле последовательность слов из лекции, и найдите соответствующий индекс.
6. Листинг кода

import time  
start = time.perf\_counter()  
  
k = open("../tests/input.txt")  
b = k.readline().split(" ")  
s = [int(l) for l in b ]  
numb = int(k.readline())  
indexes = []  
ind = 0  
for i in range(len(s)):  
 if s[i]==numb:  
 indexes.append(i)  
 ind = i  
if ind == 0:  
 ind = -1  
y = open("../tests/output.txt", "w")  
if len(indexes)>1:  
 print("Сколько раз встречается число:",len(indexes))  
 ind1 = ", ".join(str(o) for o in indexes)  
 print("Индексы:",ind1)  
 y.write(ind1)  
else:  
 print(ind)  
 y.write(str(ind))  
  
stop = time.perf\_counter()  
print("time: %s ms" % (stop - start))

1. Текстовое объяснение решения.

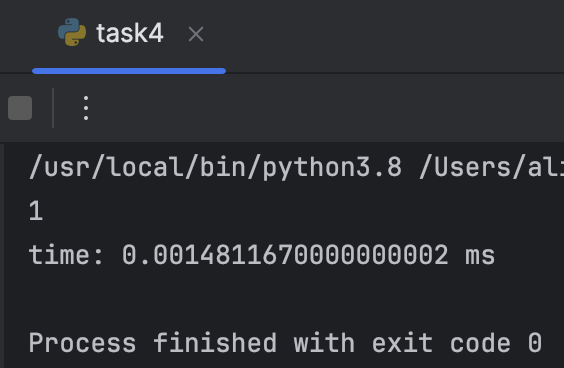
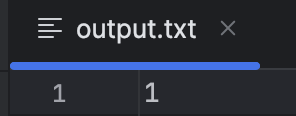
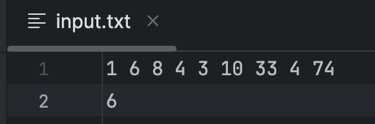
Открываю для считывания файл с числами, в переменную numb вношу элемент, который мы будем искать, далее создаю список, в который заношу числа из первой строки файла. С помощью цикла for прохожусь по каждому из чисел и проверяю, совпадает ли оно с нужным значением. Чтобы запоминать индексы элементов, создаю список , так же завожу переменную куда также буду сохранять новый индекс. Если количество значений в списке равных исходному V больше 1, вывожу список с индексами и количество совпадающих значений. Если оно равно 1, то вывожу только переменную , содержащую этот индекс. Если же таких значений нет, вывожу -1.

Открываю для записи файл output, записываю туда результат работы.

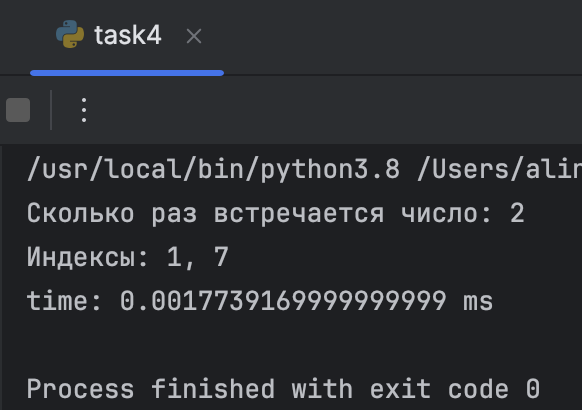
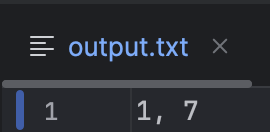
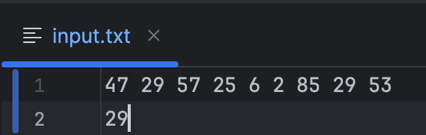
Также я замерила время выполнения кода.

1. Результат работы кода на примерах из текста задачи:

Пример 1:



Пример 2:



1. Вывод по задаче:

С помощью цикла For я провела линейный поиск по строке и вывела, элементы под какими индексами совпадают с числом, данным в условии.

# Вывод

В результате лабораторной я написала несколько алгоритмов сортировки вставками и алгоритм линейной сортировки, поработала с файлами и считыванием/записью данных в них.