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

Выполнил:

Авдиенко Данила Андреевич

Группа К3140

Проверил:

Афанасьев А. В.

Санкт-Петербург

2024 г.

# Содержание отчета

**Оглавление**

Оглавление

[Содержание отчета 2](#_Toc179820933)

[Задачи по варианту 3](#_Toc179820934)

[Задание № 1. Сортировка вставкой 3](#_Toc179820935)

[Задание №3. Обратная сортировка вставкой 4](#_Toc179820936)

[Задание №4. Линейный поиск 5](#_Toc179820937)

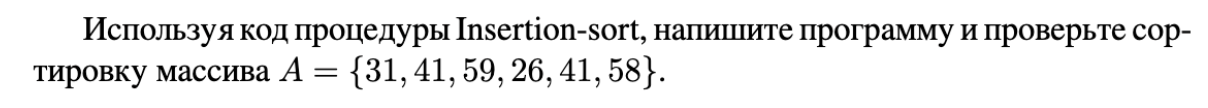
[Задание №6. Пузырьковая сортировка 7](#_Toc179820938)

[Задание №8. Секретарь Своп. 8](#_Toc179820939)

# Задачи по варианту

## Задание № 1. Сортировка вставкой

Текст задачи.



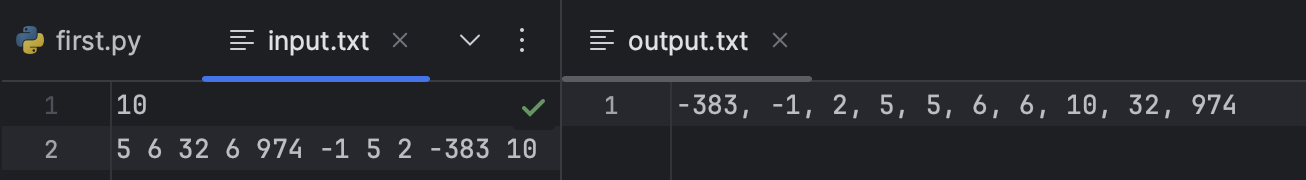
Код:

def insertion\_sort(list\_length, arr):  
 for i in range(1, list\_length):  
 key = arr[i]  
 j = i - 1  
 while j >= 0 and key < arr[j]:  
 arr[j + 1] = arr[j]  
 j -= 1  
 arr[j + 1] = key  
 return arr  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 with open("input.txt", "r") as inp:  
 list\_length = int(inp.readline())  
 arr = list(map(int, inp.readline().split()))  
 inp.close()  
 answ = ', '.join(str(i) for i in insertion\_sort(list\_length, arr))  
 with open("output.txt", "w+") as out:  
 out.write(answ)  
 out.close()

Текстовое объяснение функции insertion\_sort.

При помощи цикла for программа проходит весь массив, начиная с первого индекса, заканчивая последним. На каждой итерации берем новый элемент и вставляем его в нужное место в уже отсортированной части

Примеры работы кода:



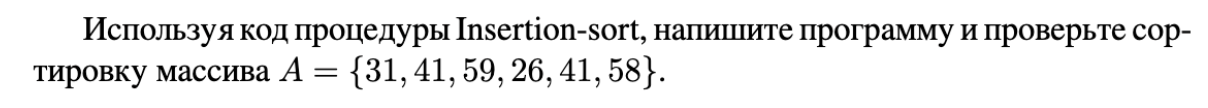
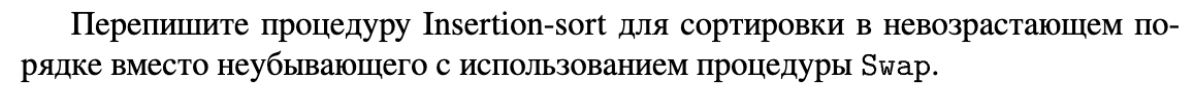
Тесты:

import unittest  
from ..task1.first import insertion\_sort  
  
  
class TestFirst(unittest.TestCase):  
  
 def test\_insertion\_sort0(self):  
 arr = [234, 325, 5, 1, -6, 0]  
 list\_length = len(arr)  
 result = insertion\_sort(list\_length, arr)  
 self.assertEqual(result, [-6, 0, 1, 5, 234, 325])  
  
 def test\_insertion\_sort1(self):  
 arr = [-1, -5, 5, 5, 3, -3]  
 list\_length = len(arr)  
 result = insertion\_sort(list\_length, arr)  
 self.assertEqual(result, [-5, -3, -1, 3, 5, 5])

Вывод: для решения задачи была реализована функция insertion\_sort, функция была протестирована при помощи unittest.

## Задание №3. Обратная сортировка вставкой

Текст задачи.



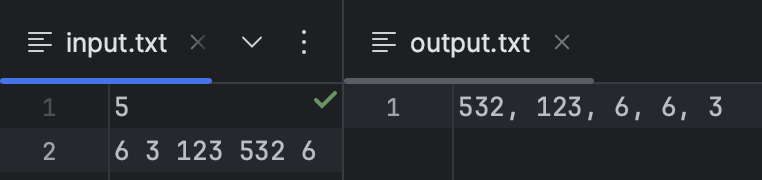
Код:

def reversed\_insertion\_sort(list\_length, arr):  
 for i in range(1, list\_length):  
 key = arr[i]  
 j = i - 1  
 while j >= 0 and key > arr[j]:  
 arr[j + 1] = arr[j]  
 j -= 1  
 arr[j + 1] = key  
 return arr  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 with open("input.txt", "r") as inp:  
 list\_length = int(inp.readline())  
 arr = list(map(int, inp.readline().split()))  
 inp.close()  
 answ = ', '.join(str(i) for i in reversed\_insertion\_sort(list\_length, arr))  
 with open("output.txt", "w+") as out:  
 out.write(answ)  
 out.close()

Текстовое объяснение функции insertion\_sort.

При помощи цикла for программа проходит весь массив, начиная с первого индекса, заканчивая последним. На каждой итерации берем новый элемент и вставляем его в нужное место в уже отсортированной части

Примеры работы кода:



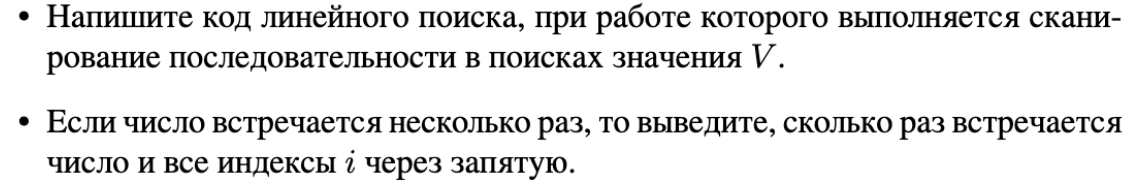
Тесты:

import unittest  
from ..task3.third import reversed\_insertion\_sort  
  
  
class TestTask3(unittest.TestCase):  
  
 def test\_reverse\_insertion\_sort0(self):  
 arr = [4, 1, 24, 512, 5, 3]  
 list\_length = len(arr)  
 result = reversed\_insertion\_sort(list\_length, arr)  
 self.assertEquals(result, [512, 24, 5, 4, 3, 1])  
  
 def test\_reverse\_insertion\_sort1(self):  
 arr = [-5, -10, -12034, -431, -4]  
 list\_length = len(arr)  
 result = reversed\_insertion\_sort(list\_length, arr)  
 self.assertEquals(result, [-4, -5, -10, -431, -12034])

Вывод: для решения задачи была реализована функция reverse\_insertion\_sort, функция была протестирована при помощи unittest.

## Задание №4. Линейный поиск

Текст задачи:



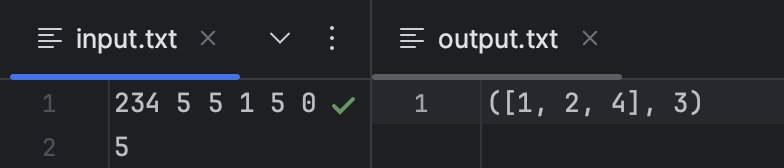
Код:

def linear\_search(v: int, arr: list):  
 cnt = 0  
 idx = []  
 for i in arr:  
 if i == v:  
 cnt += 1  
 idx.append(arr.index(i))  
 arr[arr.index(i)] = v+1  
 if cnt == 1:  
 return idx[0]  
 elif cnt == 0:  
 return -1  
 else:  
 return idx, cnt  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 with open("input.txt", "r") as inp:  
 arr = list(map(int, inp.readline().split()))  
 v = int(inp.readline())  
 inp.close()  
  
 with open("output.txt", "w") as out:  
 answ = str(linear\_search(v, arr))  
 out.write(answ)

Текстовое объяснение решения.

Для выполнения задания была реализована функция linear\_search, обработаны случаи, когда в списке только один подходящий элемент, несколько или такового нет вовсе. При помощи цикла for, каждый элемент списка сравнивается с искомым и в случае нахождения заменяется на другой, чтобы обрабатывать индексы при нескольких подходящих элементах.

Пример работы кода:



Тесты:

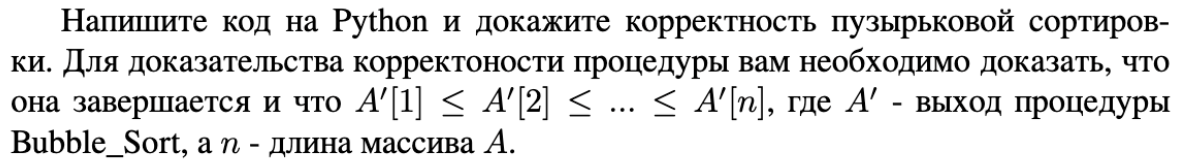
import unittest  
from ..task4.four import linear\_search  
  
  
class TestFirst(unittest.TestCase):  
  
 def test\_linear\_search0(self):  
 arr = [234, 325, 5, 1, -6, 0]  
 v = 5  
 result = linear\_search(v, arr)  
 self.assertEqual(result, 2)  
  
 def test\_linear\_search1(self):  
 arr = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]  
 v = 7  
 result = linear\_search(v, arr)  
 self.assertEqual(result, 6)  
  
 def test\_linear\_search(self):  
 arr = [1, 2, 3, 4, 5, 6, 7, 7, 7, 10, 11, 12, 13, 14, 15, 16]  
 v = 7  
 result = linear\_search(v, arr)  
 self.assertEqual(result, ([6, 7, 8], 3))

Вывод по задаче:

Для решения задачи была реализована функция linear\_search, функция была протестирована при помощи unittest.

## Задание №6. Пузырьковая сортировка

Текст задачи.



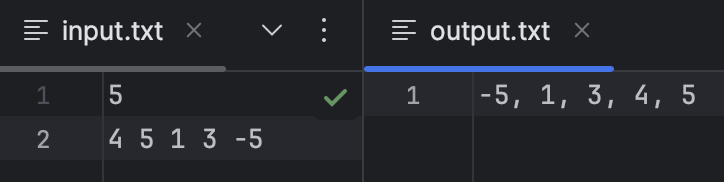
Код:

def bubble\_sort(list\_length, arr):  
 for i in range(list\_length):  
 for j in range(list\_length - 1):  
 if arr[j] > arr[j + 1]:  
 arr[j], arr[j + 1] = arr[j + 1], arr[j]  
 return arr  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 with open("input.txt", "r") as inp:  
 list\_length = int(inp.readline())  
 arr = list(map(int, inp.readline().split()))  
 inp.close()  
 answ = ', '.join(str(i) for i in bubble\_sort(list\_length, arr))  
 with open("output.txt", "w+") as out:  
 out.write(answ)  
 out.close()

Текстовое объяснение решения.

  Для выполнения задания была реализована функция bubble\_sort, в ней, при помощи двух циклов сравниваются соседние элементы в сортируемом массиве и выстраиваются в нужном порядке.

Пример работы кода:



Тесты:

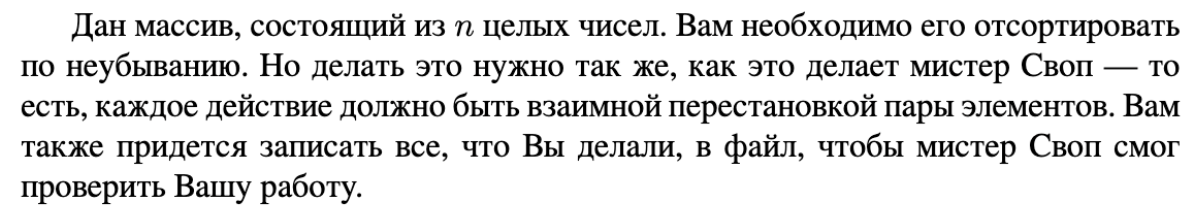
import unittest  
from ..taks6.six import bubble\_sort  
  
  
class TestFirst(unittest.TestCase):  
  
 def test\_bubble\_sort0(self):  
 arr = [234, 325, 5, 1, -6, 0]  
 list\_length = len(arr)  
 result = bubble\_sort(list\_length, arr)  
 self.assertEqual(result, [-6, 0, 1, 5, 234, 325])  
  
 def test\_bubble\_sort1(self):  
 arr = [-1, -5, 5, 5, 3, -3]  
 list\_length = len(arr)  
 result = bubble\_sort(list\_length, arr)  
 self.assertEqual(result, [-5, -3, -1, 3, 5, 5])

Вывод по задаче:

Для решения задачи была реализована функция bubble\_sort, функция была протестирована при помощи unittest, доказательство корректности прослеживается в тестах кода.

## Задание №8. Секретарь Своп.

Текст задачи:



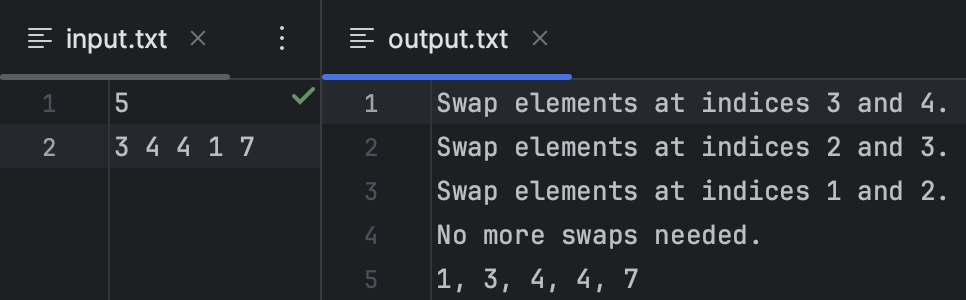
Код:

def secretar\_swap(list\_length: int, arr: list, out):  
 for j in range(list\_length):  
 for i in range(list\_length - 1):  
 if arr[i] > arr[i + 1]:  
 arr[i], arr[i + 1] = arr[i + 1], arr[i]  
 out.write(f"Swap elements at indices {i+1} and {i + 2}.\n")  
 out.write("No more swaps needed.\n")  
 return arr  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 with open("input.txt", "r") as inp:  
 list\_length = int(inp.readline())  
 arr = list(map(int, inp.readline().split()))  
 inp.close()  
  
 with open("output.txt", "w") as out:  
 answ = ', '.join(str(i) for i in secretar\_swap(list\_length, arr, out))  
 out.write(answ)

Текстовое объяснение решения.

Для выполнения задания была реализована функция secretar\_swap, которая при помощи двух циклов проходится по массиву и сортирует элементы взаимными перестановками, записывая свои шаги в выходной файл. Тестов для этого кода нет.

Пример работы кода:



**Вывод:**

В лабораторной работе я использовал алгоритмы сортировки вставкой, пузырьковой сортировки и сортировки выбором, код каждой функции, кроме 8-ой был протестирован при помощи unittest.