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

Выполнила:
Пантюхова В.А.
К3126

Отчет по лабораторной работе № 1
по курсу «Алгоритмы и структуры данных»
Тема: Сортировка вставками, выбором, пузырьковая.
Вариант 18

Проверила:
Артамонова В.Е.

Санкт-Петербург
2022 г.

Содержание отчета

Задачи по варианту

Задача №7. Знакомство с жителями Сортлэнда

Владелец графства Сортлэнд, граф Бабблсортер, решил познакомиться со своими подданными. Число жителей в графстве нечетно и составляет n, где n может быть достаточно велико, поэтому граф решил ограничиться знакомством с тремя представителями народонаселения: с самым бедным жителем, с жителем, обладающим средним достатком, и с самым богатым жителем. Согласно традициям Сортлэнда, считается, что житель обладает средним достатком, если при сортировке жителей по сумме денежных сбережений он оказывается ровно посередине. Известно, что каждый житель графства имеет уникальный идентификационный номер, значение которого расположено в границах от единицы до n. Информация о размере денежных накоплений жителей хранится в массиве M таким образом, что сумма денежных накоплений жителя, обладающего идентификационным номером i, содержится в ячейке M[i]. Помогите секретарю графа мистеру Свопу вычислить идентификационные номера жителей, которые будут приглашены на встречу с графом.

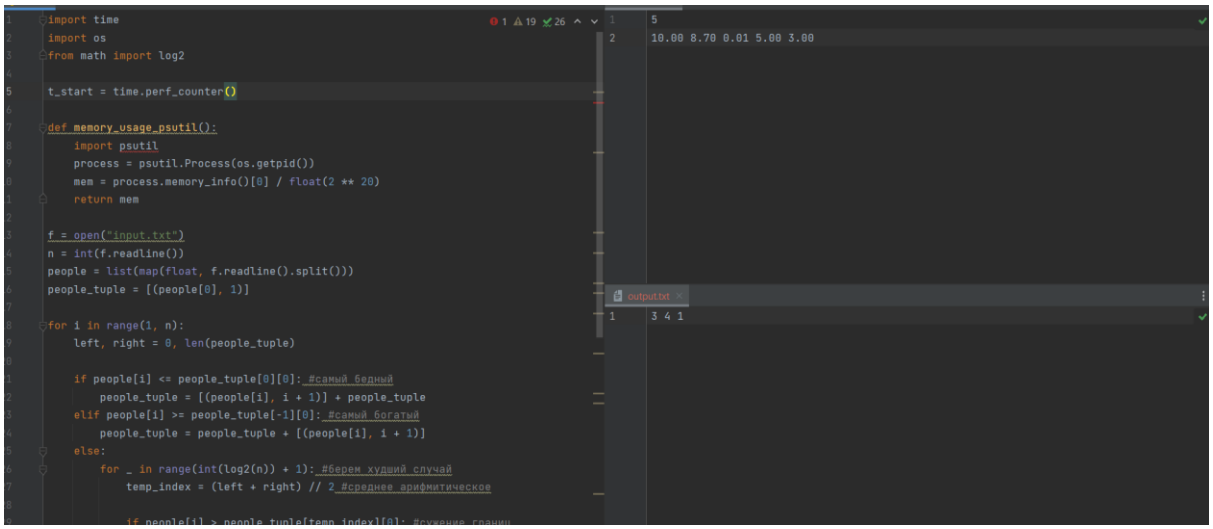
Листинг кода.

```
import time
import os
from math import log2

t_start = time.perf_counter()

def memory_usage_psutil():
    import psutil
    process = psutil.Process(os.getpid())
    mem = process.memory_info()[0] / float(2 ** 20)
    return mem

f = open("input.txt")
n = int(f.readline())
people = list(map(float, f.readline().split()))
people_tuple = [(people[0], 1)]

for i in range(1, n):
    left, right = 0, len(people_tuple)

    if people[i] <= people_tuple[0][0]: #самый бедный
        people_tuple = [(people[i], i + 1)] + people_tuple
    elif people[i] >= people_tuple[-1][0]: #самый богатый
        people_tuple = people_tuple + [(people[i], i + 1)]
    else:
```

```
        for _ in range(int(log2(n)) + 1): #берем худший случай
            temp_index = (left + right) // 2 #среднее арифмитическое

            if people[i] > people_tuple[temp_index][0]: #сужение границ
                left = temp_index
            elif people[i] < people_tuple[temp_index][0]:
                right = temp_index
            else: #если число = числу
                left, right = temp_index, temp_index - 1
                break

        insert_index = (left + right + 1) // 2 #место куда надо вставить
число
        people_tuple.insert(insert_index, (people[i], i + 1)) #впихиваем
элемент и двигаем остальные вправо

mf = open("output.txt", "w+")
mf.write(str(people_tuple[0][1]) + ' ' + str(people_tuple[n // 2][1]) + ' '
+ str(people_tuple[-1][1]))
mf.close()

print("Время:", time.perf_counter() - t_start)
print("Память:", memory_usage_psutil(), "мб")

f.close()
```

Текстовое объяснение решения.
Это алгоритм для упорядочивания элементов в массиве

Результат работы кода:(скрины input, output файлов).



	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из		

текста задачи		
Пример из задачи	0.004664699999999994	
Пример из задачи		
Верхняя граница диапазона значений входных данных из текста задачи		

Вывод по задаче: научились сортировке.

Задача №8. Секретарь Своп

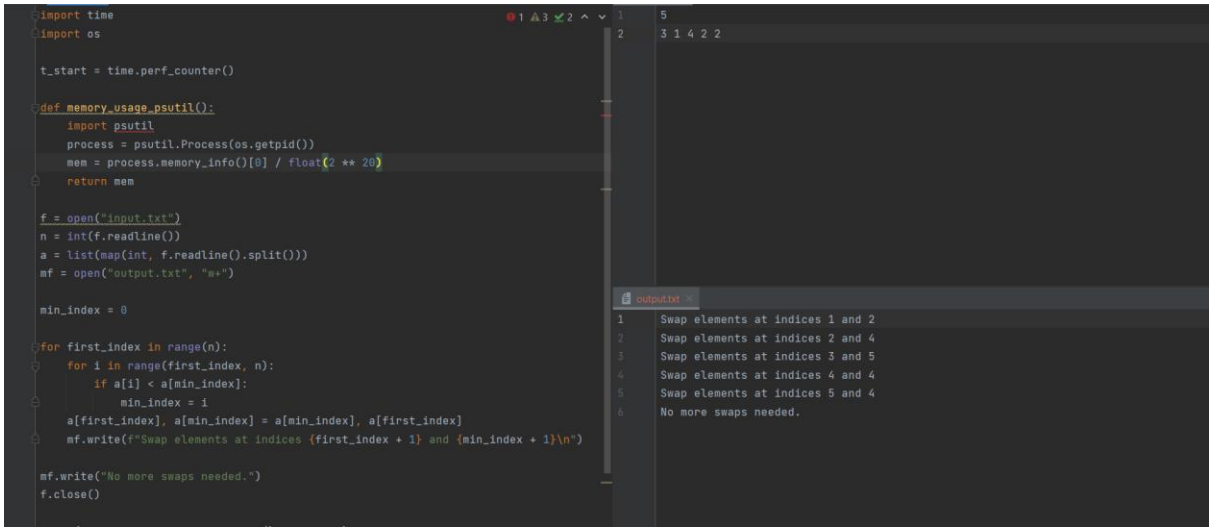
Дан массив, состоящий из n целых чисел. Вам необходимо его отсортировать по неубыванию. Но делать это нужно так же, как это делает мистер Своп — то есть, каждое действие должно быть взаимной перестановкой пары элементов. Вам также придется записать все, что Вы делали, в файл, чтобы мистер Своп смог проверить Вашу работу.

Листинг кода.

```
import time
import os

t_start = time.perf_counter()

def memory_usage_psutil():
    import psutil
    process = psutil.Process(os.getpid())
    mem = process.memory_info()[0] / float(2 ** 20)
    return mem

f = open("input.txt")
n = int(f.readline())
a = list(map(int, f.readline().split()))
mf = open("output.txt", "w+")

min_index = 0

for first_index in range(n):
    for i in range(first_index, n):
        if a[i] < a[min_index]:
            min_index = i
    a[first_index], a[min_index] = a[min_index], a[first_index]
    mf.write(f"Swap elements at indices {first_index + 1} and {min_index +
1}\n")

mf.write("No more swaps needed.")
f.close()

print("Время:", time.perf_counter() - t_start)
print("Память:", memory_usage_psutil(), "мб")

mf.close()
```

Текстовое объяснение решения.
Мы выбираем элемент из определённого количества и помещаем его на нужную позицию массива.

Результат работы кода на примерах из текста задачи:(скрины input output файлов)



	Время выполнения	Затраты памяти
Нижняя граница диапазона значений входных данных из текста задачи		
Пример из задачи	0.0060751	
Пример из задачи		
Верхняя граница диапазона значений входных данных из текста задачи		

Вывод по задаче: здесь мы изучили .

Вывод:

В этой лабораторной работе я рассмотрела и узнала много нового и познавательного.