Лабораторная работа 1 - Сортировки и линейный поиск

Цель

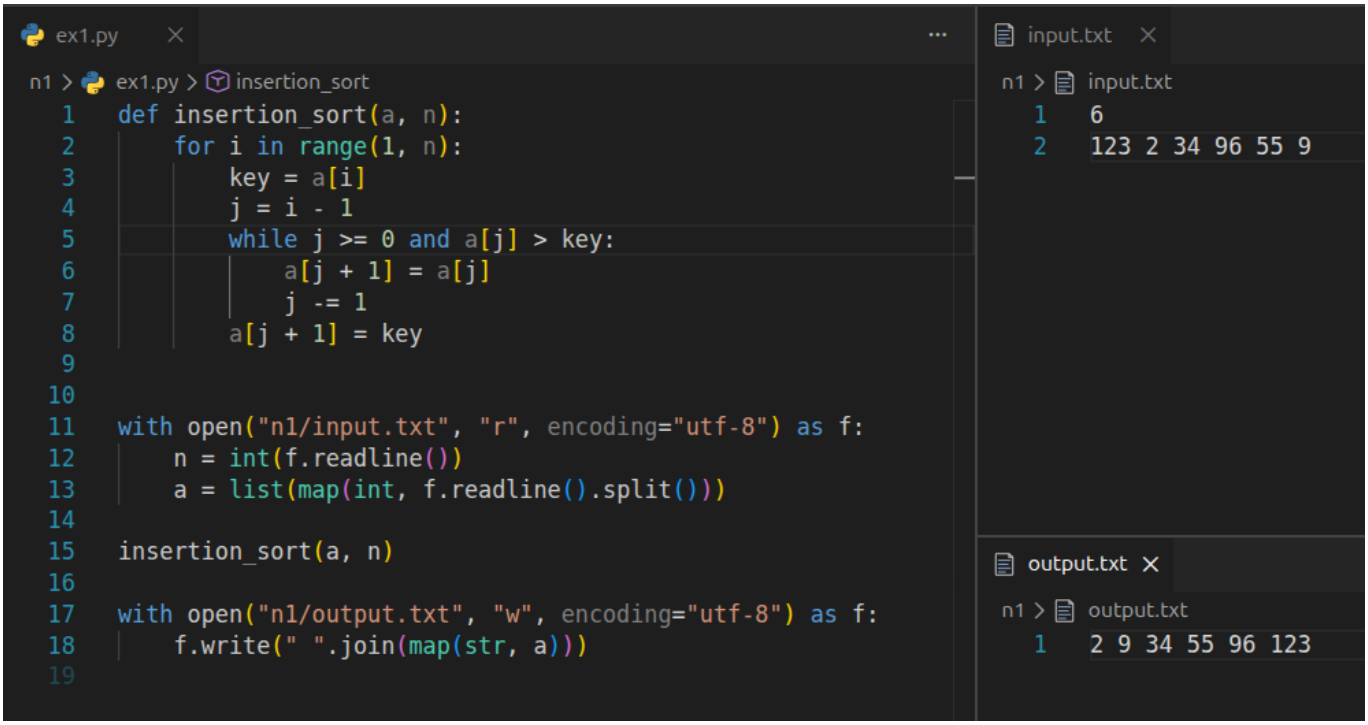
Реализовать базовые алгоритмы сортировки и поиска, а также проверить соответствие ограничениям по времени (1 сек) и памяти (256 МБ).

Задания и решения

Задание 1 - Сортировка вставкой (Insertion sort)

Реализована классическая сортировка вставками: для каждого элемента `a[i]` выполняется сдвиг больших элементов влево и вставка `key` на нужную позицию.

- Время: O(n^2)
- Память: O(1)
- Ввод: `n`, затем `n` чисел
- Вывод: отсортированный массив через пробел



Задание 4 - Линейный поиск (Linear search)

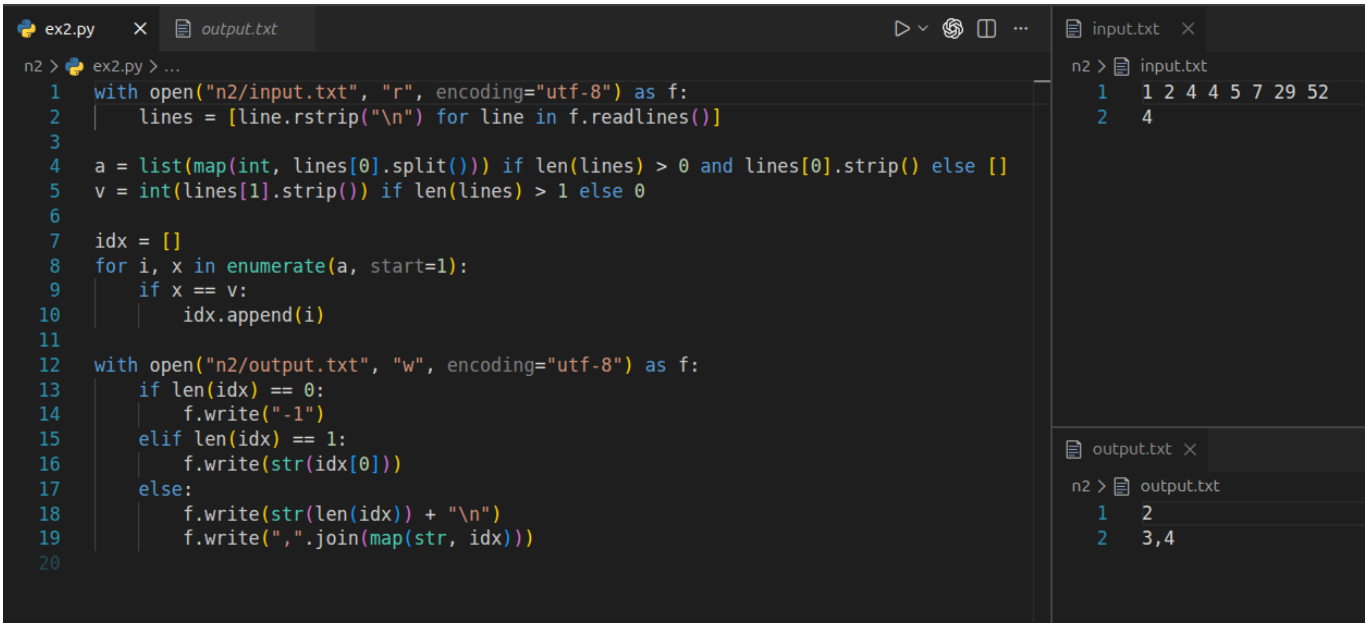
Выполняется сканирование массива слева направо и сбор индексов всех совпадений `V`.

Формат вывода объединённый:

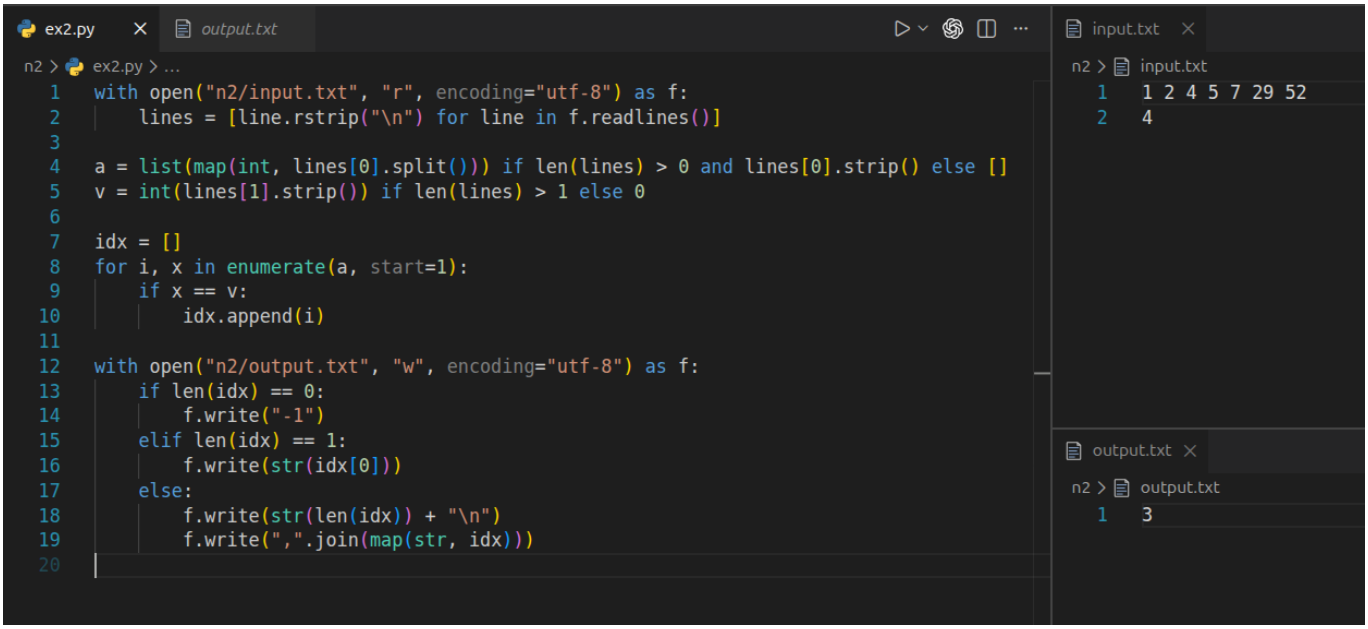
- если совпадений нет - `-1`
- если ровно одно совпадение - печатается только индекс
- если совпадений 2 и более - печатается `k`, затем на новой строке индексы через запятую

Индексация: с 1.

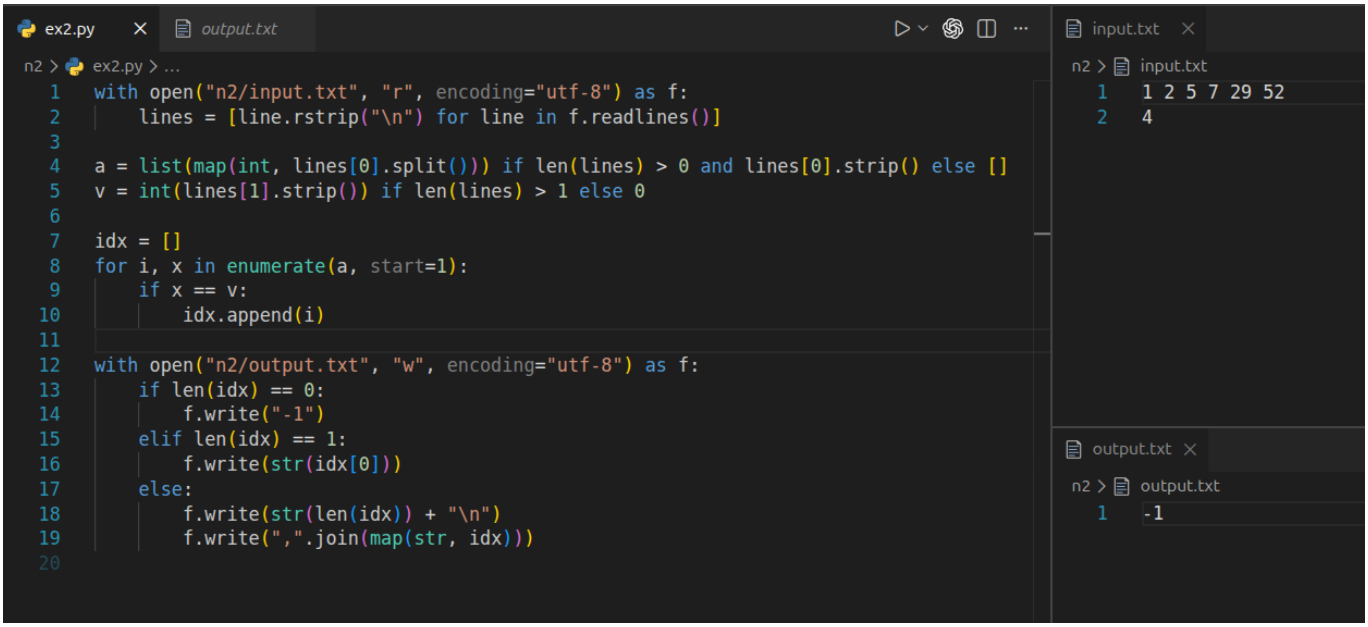
Пример (2 совпадения):



Пример (1 совпадение):



Пример (0 совпадений):



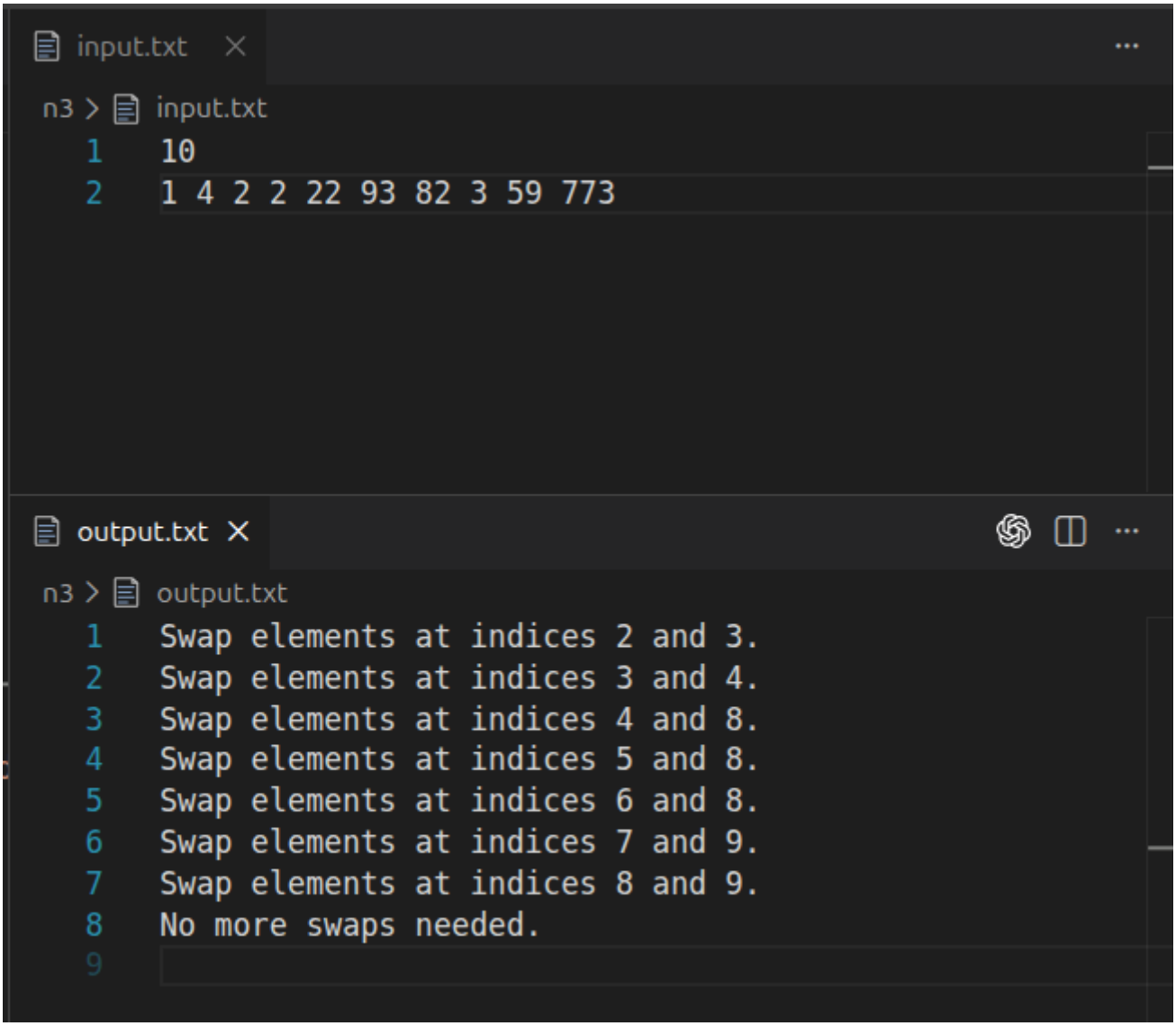
Задание 8 - Секретарь Своп

Нужно отсортировать массив по неубыванию и вывести все произведённые перестановки в формате: `Swap elements at indices X and Y.`
После всех перестановок: `No more swaps needed.`

Чтобы не генерировать слишком длинный лог, используется подход с выбором минимального элемента на каждом шаге (по сути selection-sort):

- максимум перестановок: `n - 1`
- лог пишется сразу в файл (без накопления строк в памяти)

Пример входа и лога перестановок:



Код

```
import time
import tracemalloc


def secretary_swap_sort_with_log(a, n, out):
    for i in range(n):
        min_idx = i
        for j in range(i + 1, n):
            if a[j] < a[min_idx]:
                min_idx = j

        if min_idx != i:
            x, y = i + 1, min_idx + 1
            if x > y:
                x, y = y, x
            out.write(f"Swap elements at indices {x} and {y}.\n")
            a[i], a[min_idx] = a[min_idx], a[i]
```

```
        out.write("No more swaps needed.\n")


    def bytes_to_mb(b):
        return b / (1024 * 1024)


    if __name__ == "__main__":
        with open("n3/input.txt", "r", encoding="utf-8") as f:
            n = int(f.readline())
            a = list(map(int, f.readline().split()))

        tracemalloc.start()
        t0 = time.perf_counter()

        with open("n3/output.txt", "w", encoding="utf-8") as out:
            secretary_swap_sort_with_log(a, n, out)

            t1 = time.perf_counter()
            current, peak = tracemalloc.get_traced_memory()
            tracemalloc.stop()

            elapsed = t1 - t0

            print(f"Time: {elapsed:.6f} sec\n")
            print(f"Peak memory: {peak} bytes ({bytes_to_mb(peak):.3f} MB)\n")

            print("Limits: 1 sec, 256 MB\n")
```

Время и память для кода выше

```
chikirao@chikirao-MCLG-XX:~/main/itmo/algo/lab1$
go/lab1/n3/ex3.py
Time: 0.000790 sec

Peak memory: 5941 bytes (0.006 MB)

Limits: 1 sec, 256 MB
```