Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа №3
по дисциплине "Анализ Алгоритмов"

Тема Алгоритмы сортировки

Студент Ковалец К. Э.

Группа ИУ7-53Б

Преподаватель Волкова Л. Л.

Москва — 2021 г.

Содержание

Введение

В данной лабораторной работе будут рассмотрены алгоритмы сортировки. Сортировкой называют процесс перегруппировки заданной последовательности (кортежа) объектов в некотором определенном порядке. Это одна из главных процудур обработки структурированных данных. Важнейшей характеристикой любого алгоритма сортировки является скорость его работы, время сортировки будет пропорционально количеству сравнений и перестановки элементов данных в процессе их сортировки.

В данной лабораторной работе будут рассмотрены три алгоритма сортировки: пузырёк, вставками и сортировка выбором.

Цель работы: изучение и реализация трёх алгоритмов сортировки, а так же исследование их трудоемкости.

Задачи работы.

1. Изучить три алгоритма сортировки: пузырёк, вставками и выбором.
2. Реализовать эти алгоритмы.
3. Провести сравнительный анализ трудоемкости алгоритмов.
4. Провести сравнительный анализ реализаций алгоритмов по затраченному процессорному времени и памяти.
5. Описать и обосновать полученные результаты в отчете о выполненной лабораторной работе.

1 Аналитическая часть

В данном разделе мы рассмотрим три алгоритма сортировок: пузырёк, вставками и выбором.

1.1 Сортировка пузырьком

Алгоритм сортировки “пузырьком” состоит в повторении проходов по массиву с помощью вложенных циклов. При каждом проходе по массиву сравниваются между собой пары “соседних” элементов. Если элементы какой-то из сравниваемых пар расположены в неправильном порядке – происходит обмен (перезапись) значений ячеек массива. Проходы по массиву повторяются $N - 1$ раз. [1]

1.2 Сортировка выбором

Алгоритм сортировки выбором заключается в поиске на необработанном срезе массива или списка минимального значения и в дальнейшем обмене этого значения с первым элементом необработанного среза. На следующем шаге необработанный срез уменьшается на один элемент. [2]

Идея алгоритма поэтапно:

- Найти наименьшее значение в списке.
- Записать его в начало списка, а первый элемент - на место, где раньше стоял наименьший.
- Снова найти наименьший элемент в списке. При этом в поиске не участвует первый элемент.
- Второй минимум поместить на второе место списка. Второй элемент при этом перемещается на освободившееся место.
- Продолжать выполнять поиск и обмен, пока не будет достигнут конец списка.

1.3 Сортировка вставками

Суть алгоритма заключается в следующем: есть часть массива, которая уже отсортирована, и требуется вставить остальные элементы массива в отсортированную часть, сохранив при этом упорядоченность. Для этого на каждом шаге алгоритма мы выбираем один из элементов входных данных и вставляем его на нужную позицию в уже отсортированной части массива, до тех пор пока весь набор входных данных не будет отсортирован. [3]

1.4 Вывод

В данной лабораторной работе стоит задача реализации трёх алгоритмов сортировки, а именно: пузырьком, вставками и выбором. Такжк требуется оценить теоретическую оценку алгоритмов и проверить ее экспериментально. Все три алгоритма сортировок были описаны в этом разделе.

2 Конструкторская часть

В данном разделе мы рассмотрим схемы алгоритмов сортировок (пузырьком, вставками и выбором), а также найдена их трудоемкость.

2.1 Разработка алгоритмов

На рисунках 2.1, 2.2 и 2.3 представлены схемы алгоритмов сортировки (пузырьком, выбором и вставками).

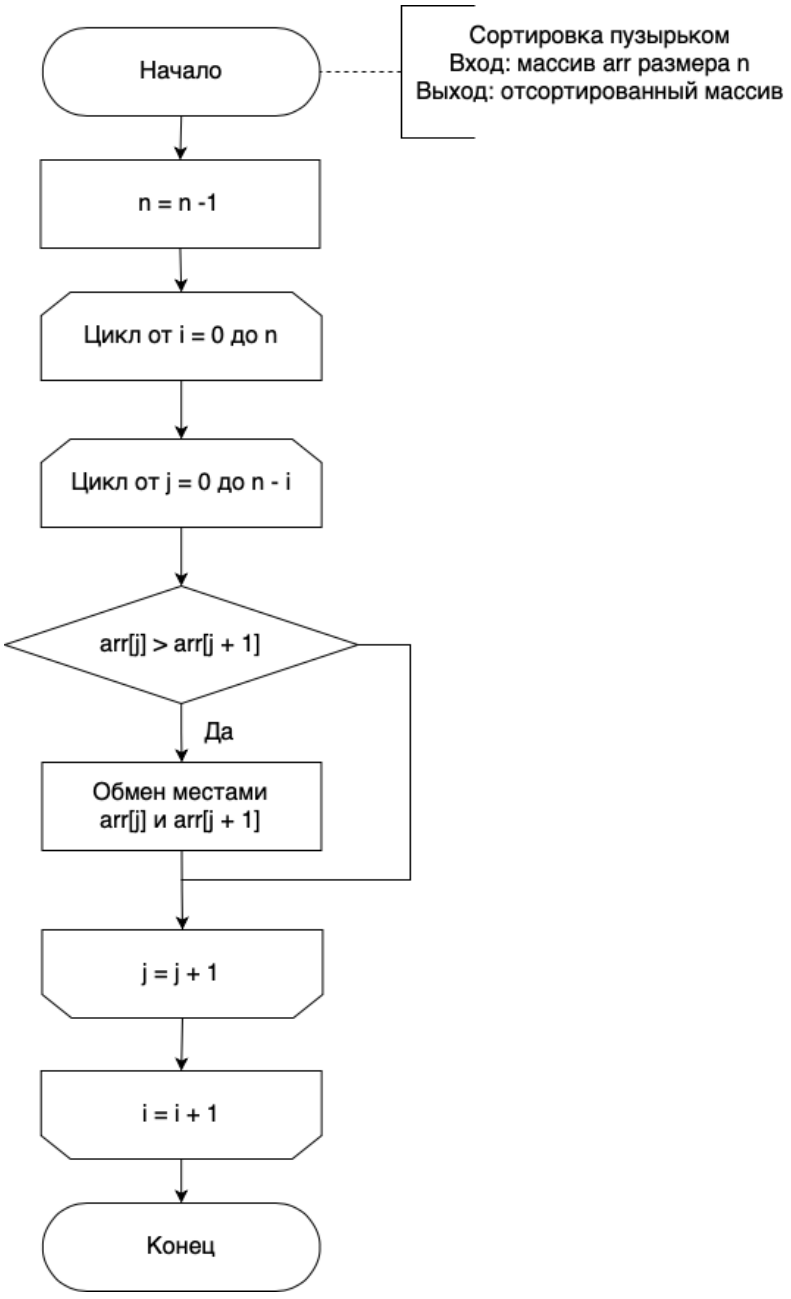

Рисунок 2.1 – Схема алгоритма сортировки пузырьком

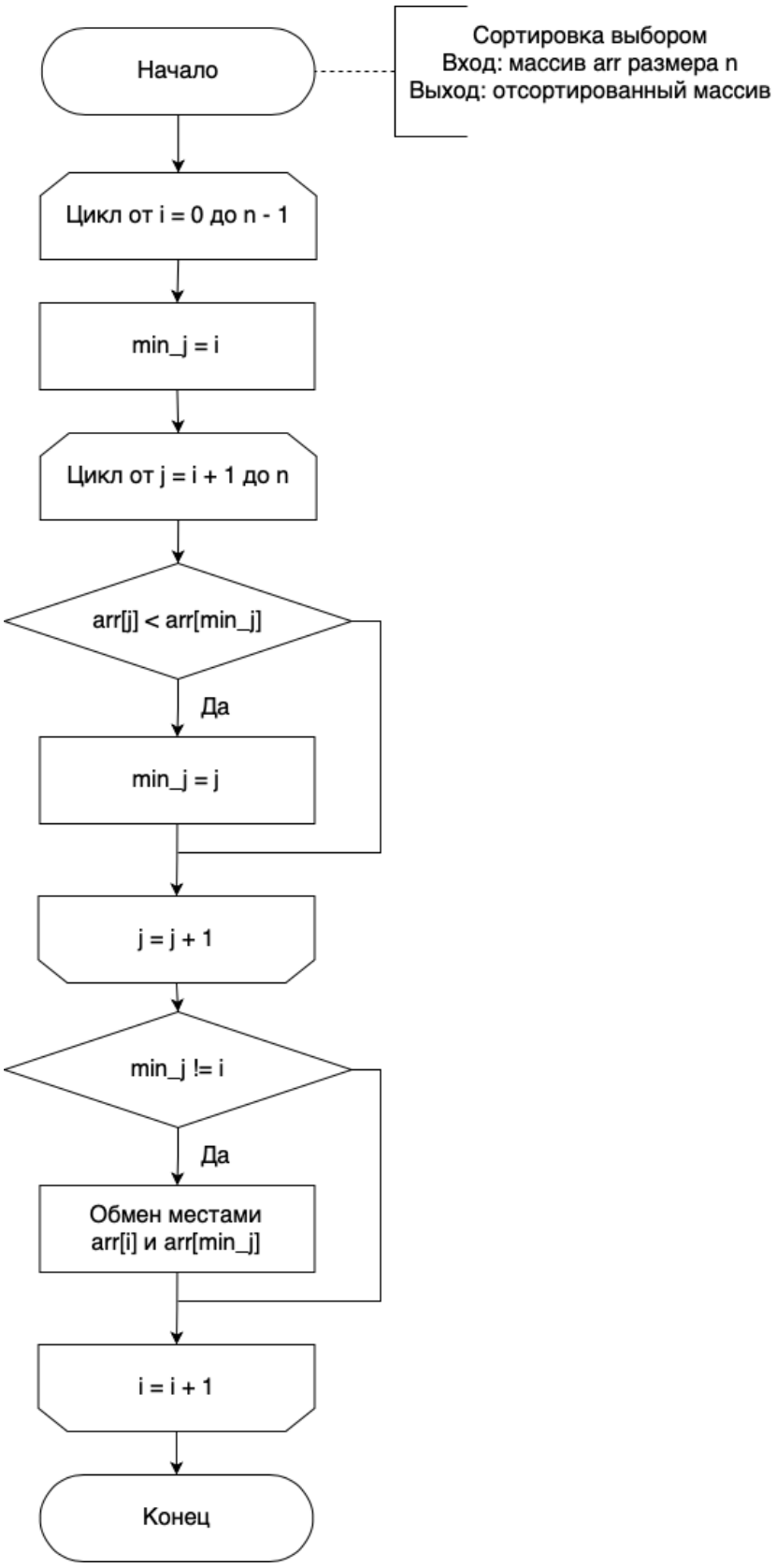

Рисунок 2.2 – Схема алгоритма сортировки выбором

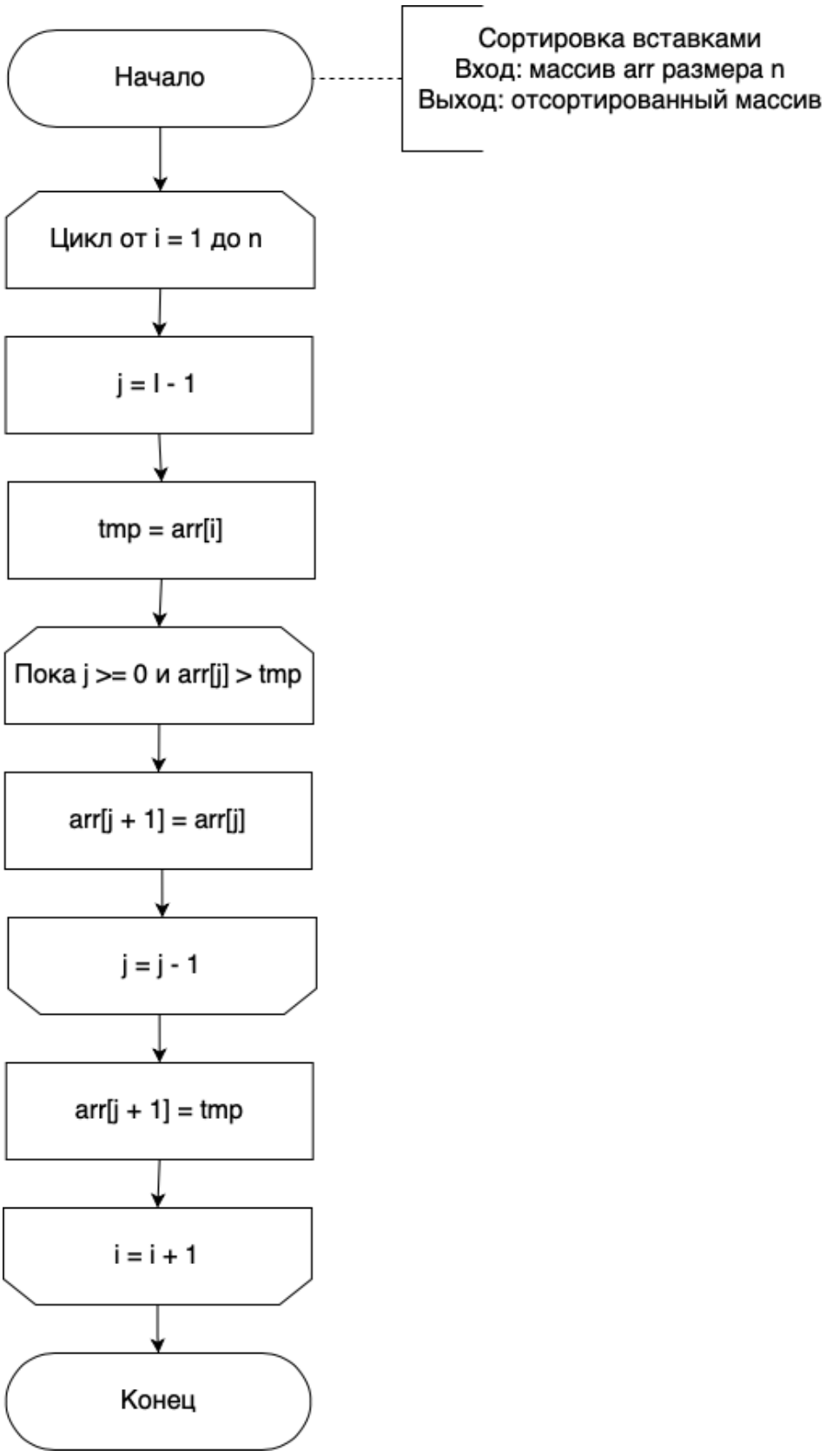

Рисунок 2.3 – Схема алгоритма сортировки вставками

2.2 Модель вычислений для проведения оценки трудоемкости

Введем модель вычислений, которая потребуется для определния трудоемкости каждого отдельно взятого алгоритма сортировки:

1. операции из списка (2.1) имеют трудоемкость равную 1;

$$+, -, /, *, \%, =, +=, -=, *=, /=, \%=, ==, !=, <, >, <=, >=, [], ++, --$$ (2.1)

2. трудоемкость оператора выбора `if условие then A else B` рассчитывается, как (2.2);

$$f_{if} = f_{\text{условия}} + \begin{cases} f_A, & \text{если условие выполняется,} \\ f_B, & \text{иначе.} \end{cases} \quad (2.2)$$

3. трудоемкость цикла рассчитывается, как (2.3);

$$f_{for} = f_{\text{инициализации}} + f_{\text{сравнения}} + N(f_{\text{тела}} + f_{\text{инкремент}} + f_{\text{сравнения}}) \quad (2.3)$$

4. трудоемкость вызова функции равна 0.

N - размер массивов во всех вычислениях. Лучшим случаем сортировки считается тот, когда массив уже отсортирован, худшим - когда отсортирован в обратном порядке.

2.2.1 Алгоритм сортировки пузырьком

Трудоёмкость в лучшем случае (при уже отсортированном массиве) (2.4):

$$f_{best} = 2 + 1 + N \cdot (1 + 1 + 1 + \frac{N-1}{2} \cdot (1 + 1 + 4)) = 3N^2 + 3 = O(N^2). \quad (2.4)$$

Трудоёмкость в худшем случае (при массиве, отсортированном в обратном порядке) (2.5):

$$f_{worst} = 2+1+N\cdot(1+1+1+\frac{N-1}{2}\cdot(1+1+11)) = 6.5N^2-3.5N+3 = O(N^2). \quad (2.5)$$

2.2.2 Алгоритм сортировки вставками

Трудоёмкость в лучшем случае (при уже отсортированном массиве) (2.6):

$$f_{best} = 1+1+(N-1)\cdot(2+2+1+4+2) = 11N-9 = O(N). \quad (2.6)$$

Трудоёмкость в худшем случае (при массиве, отсортированном в обратном порядке) (2.7):

$$f_{worst} = 1+1+(N-1)\cdot(2+2+1+\frac{N}{2}\cdot 9+2) = 4.5N^2+2.5N-5 = O(N^2). \quad (2.7)$$

2.2.3 Алгоритм сортировки выбором

Трудоемкость сортировки выбором:

- В лучшем случае: O(N^2).
- В худшем случае: O(N^2).

2.3 Вывод

В данном разделе были рассмотрены схемы всех трёх алгоритмов сортировки (пузырьком, вставками и выбором). Также для каждого из них были рассчитаны и оценены лучшие и худшие случаи.

3 Технологическая часть

3.1 Требования к программному обеспечению

- Входные данные - массив целых чисел;
- Выходные данные - отсортированный массив в заданном порядке.

3.2 Средства реализации

В данной работе для реализации был выбран язык программирования *Python*. Выбор обсуловлен наличием опыта работы с ним. Время работы было замерено с помощью функции *process_time* из библиотеки *time*.

3.3 Сведения о модулях программы

Программа состоит из следующих модулей:

- *main.py* - файл, содержащий функцию *main*;
- *sorts.py* - файл, содержащий код всех сортировок;
- *test.py* - файл, в котором содержатся функции для замера времени работы сортировок и построения графика;
- *get_array.py* - файл, в котором содержатся функции для генерации массивов разных видов;
- *input_array.py* - файл, в котором содержаться функции для ввода маассива;
- *color.py* - файл, который содержит переменные типа *string* для цветного вывода результата работы программы в консоль.

3.4 Листинги кода

В листингах 3.1, 3.2, 3.3 представлены реализации алгоритмов сортировок (пузырьком, выбором и вставками).

Листинг 3.1 – Алгоритм сортировки пузырьком

```
def bubble_sort(arr, n):
    n -= 1

    for i in range(n):
        for j in range(n - i):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

    return arr
```

Листинг 3.2 – Алгоритм сортировки выбором

```
def select_sort(arr, n):

    for i in range(n - 1):
        min_j = i

        for j in range(i + 1, n):
            if arr[j] < arr[min_j]:
                min_j = j

        if min_j != i:
            arr[i], arr[min_j] = arr[min_j], arr[i]

    return arr
```

Листинг 3.3 – Алгоритм сортировки вставками

```
def insertion_sort(arr, n):

    for i in range(1, n):
        j = i - 1
        tmp = arr[i]

        while j >= 0 and arr[j] > tmp:
            arr[j + 1] = arr[j]
            j -= 1

        arr[j + 1] = tmp

    return arr
```

3.5 Функциональные тесты

В таблице 3.1 приведены тесты для функций, реализующих алгоритмы сортировки. Тесты для всех сортировок пройдены успешно.

Таблица 3.1 – Функциональные тесты

Входной массив	Результат	Правильность
[1 2 3 4 5 6]	[1 2 3 4 5 6]	Верно
[6 5 4 3 2 1]	[1 2 3 4 5 6]	Верно
[7 -3 8 -5 0]	[-5 -3 0 7 8]	Верно
[5 5 5]	[5 5 5]	Верно
[9]	[9]	Верно
[]	[]	Верно
a	Сообщение об ошибке	Верно

3.6 Вывод

В данном разделе были разработаны исходные коды трёх алгоритмов сортировки: пузырьком, выбором и вставками, а также проведено тестирование.

4 Исследовательская часть

4.1 Технические характеристики

Технические характеристики устройства, на котором выполнялось тестирование представлены далее.

- Операционная система: macOS 11.5.2.
- Память: 8 GiB.
- Процессор: 2,3 GHz 4-ядерный процессор Intel Core i5.

При тестировании ноутбук был включен в сеть электропитания. Во время тестирования ноутбук был нагружен только встроенными приложениями окружения, а также системой тестирования.

4.2 Демонстрация работы программы

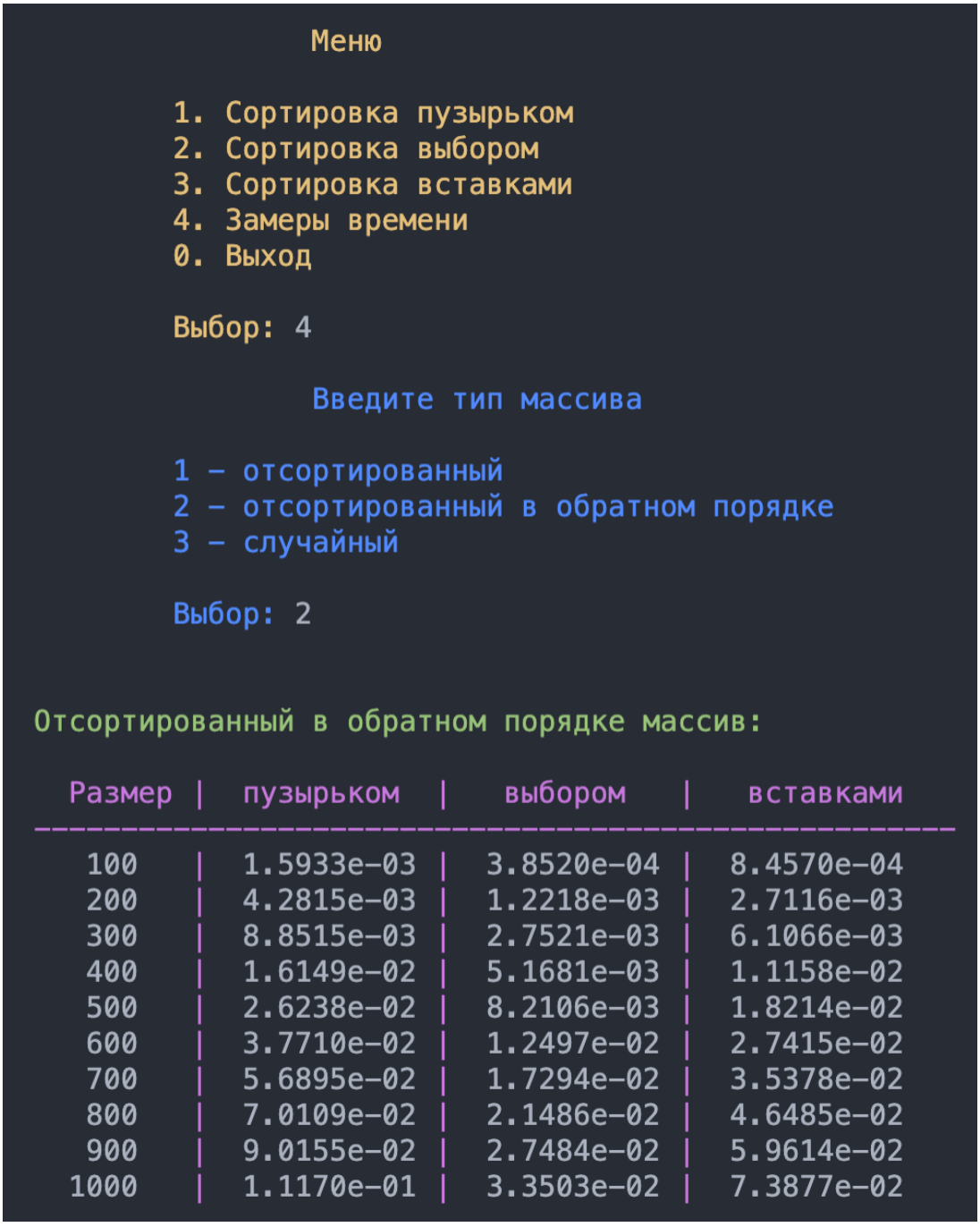

Рисунок 4.1 – Пример работы программы

4.3 Время выполнения алгоритмов

Результаты замеров времени работы алгоритмов сортировок (в мс) приведены в таблицах 4.1, 4.2 и 4.3 Также на рисунках 4.2, 4.3, 4.4 приведены графики зависимостей времени работы алгоритмов сортировки от размеров массивов на отсортированных, обратно отсортированных и случайных данных.

Таблица 4.1 – Отсортированные данные

Размер	Пузырьком	Выбором	Вставками
100	0.673	0.555	0.019
200	1.829	1.412	0.029
300	3.849	2.837	0.042
400	5.965	5.137	0.056
500	9.617	8.237	0.073
600	14.066	11.806	0.092
700	19.506	16.564	0.145
800	25.934	21.103	0.121
900	32.982	27.160	0.148
1000	41.045	33.178	0.152

Таблица 4.2 – Отсортированные в обратном порядке данные

Размер	Пузырьком	Выбором	Вставками
100	1.593	0.385	0.846
200	4.282	1.222	2.712
300	8.852	2.752	6.107
400	16.149	5.168	11.158
500	26.238	8.211	18.214
600	37.710	12.497	27.415
700	56.895	17.294	35.378
800	70.109	21.486	46.485
900	90.155	27.484	59.614
1000	111.702	33.503	73.877

Таблица 4.3 – Случайные данные

Размер	Пузырьком	Выбором	Вставками
100	1.103	0.407	0.480
200	3.189	1.270	1.387
300	6.169	2.709	3.165
400	11.057	5.123	5.892
500	17.704	8.104	8.892
600	26.190	12.312	13.278
700	36.716	16.505	18.171
800	49.156	21.466	23.009
900	66.507	28.209	32.940
1000	79.332	33.690	38.293

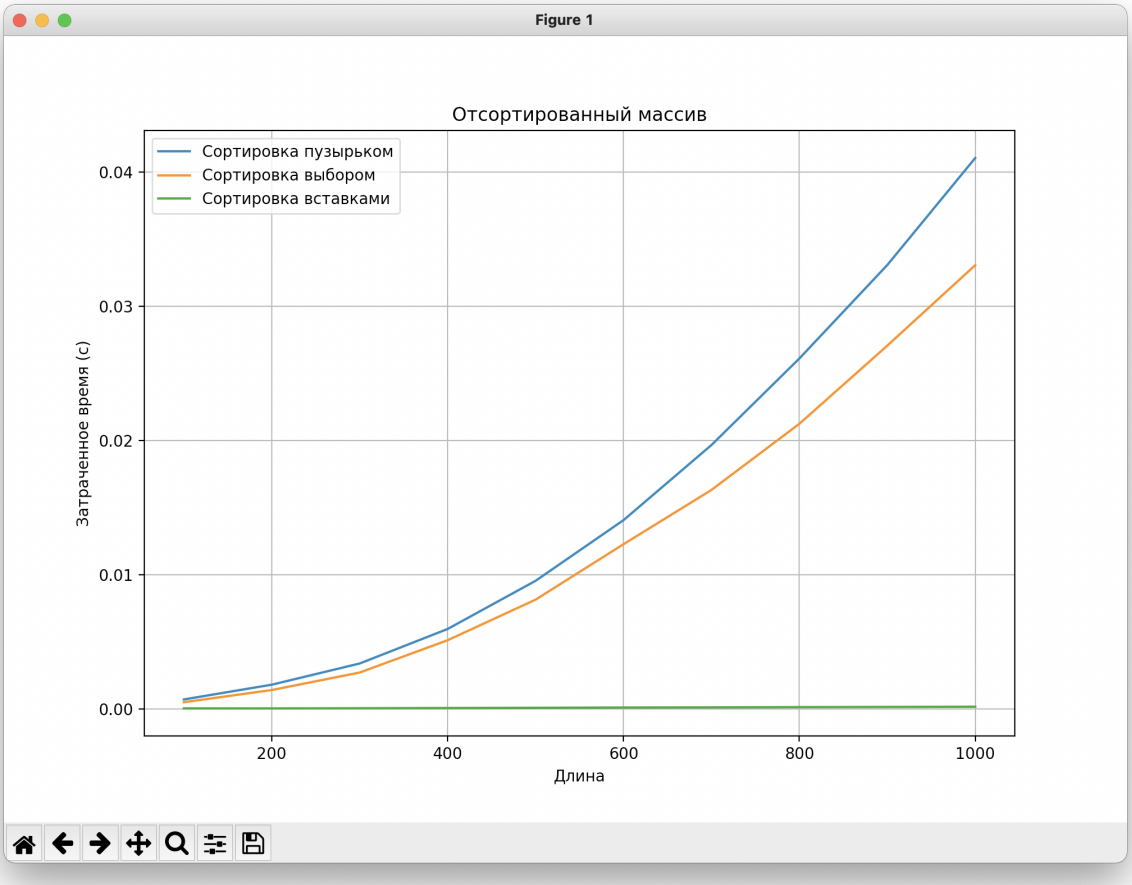

Рисунок 4.2 – Отсортированный массив

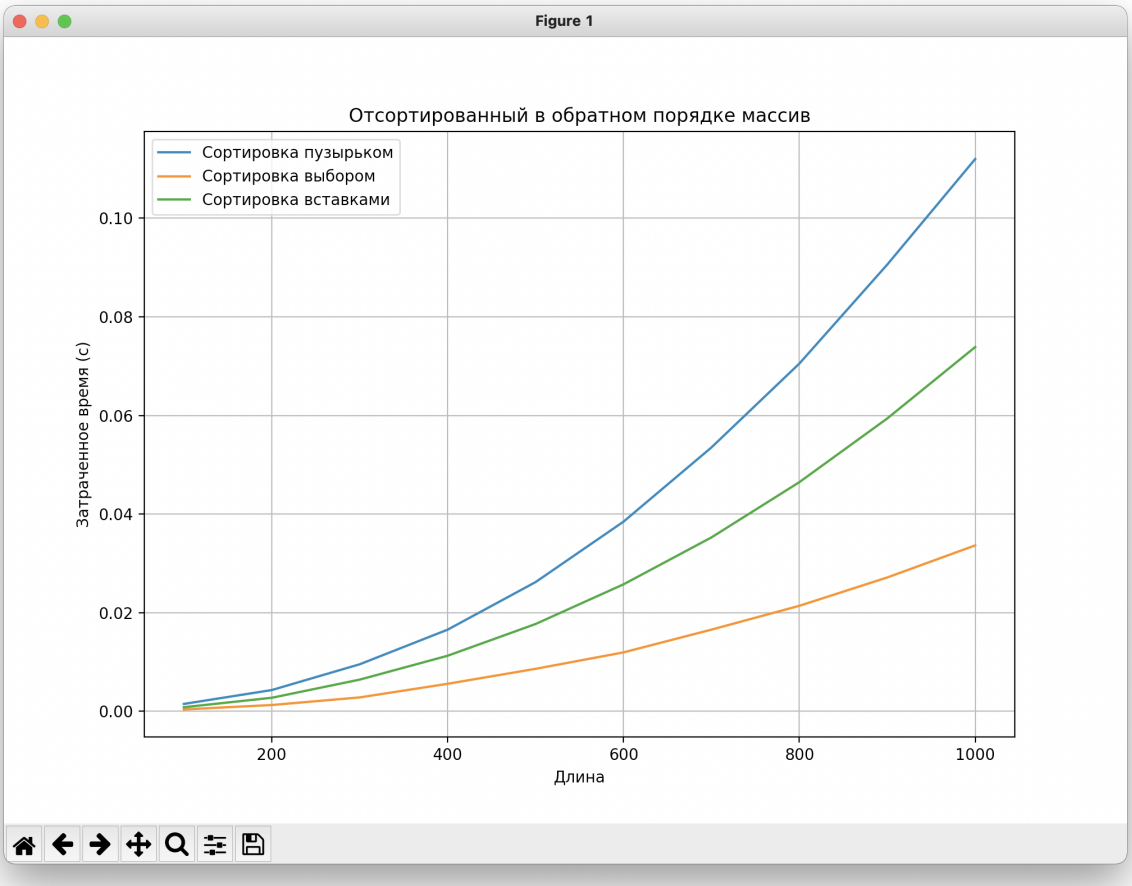

Рисунок 4.3 – Отсортированный в обратном порядке массив

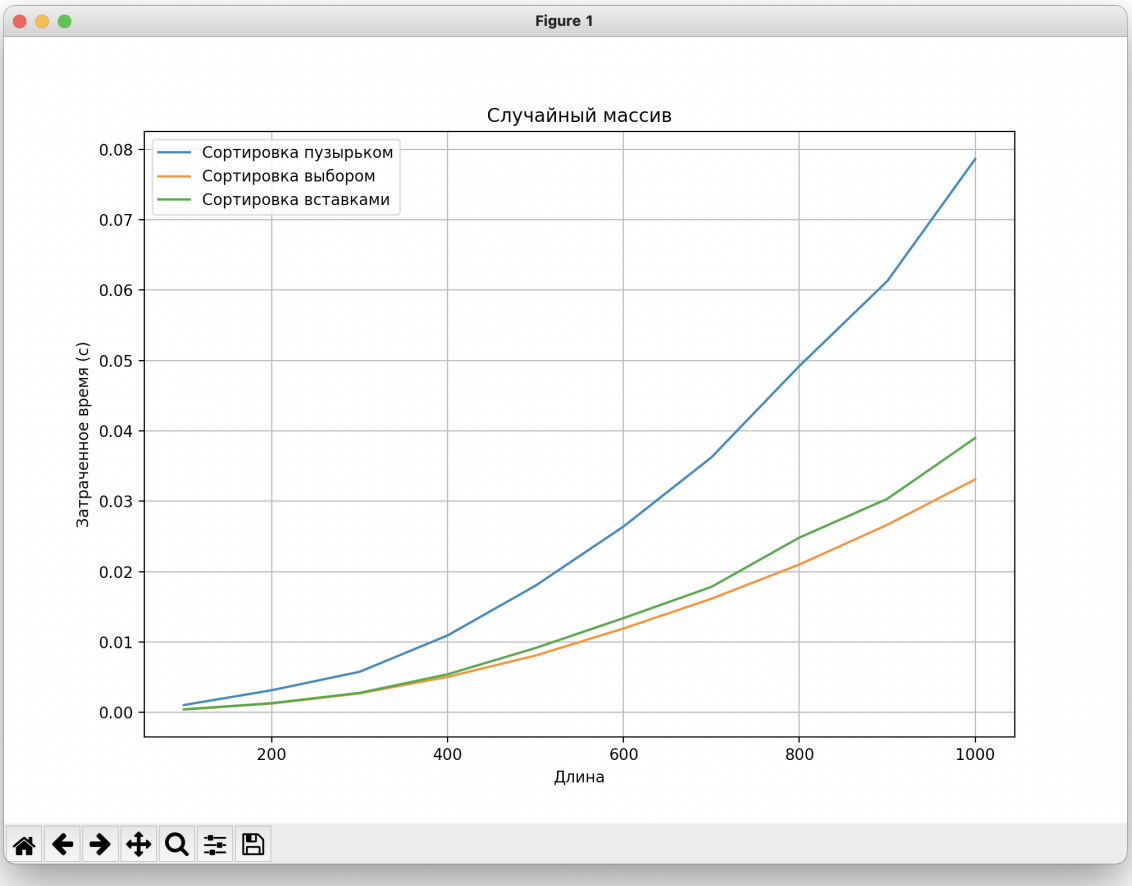

Рисунок 4.4 – Случайный массив

4.4 Вывод

Алгоритм сортировки пузырьком оказался худшим на всех данныых. Сортировка выбором показала себя лучше всех на отсортированных в обратном порядке данных (на 1000 элементах примерно в 2 раза быстрее сортировки вставками и в 3 раза - сортировки пузырьком) и на рандомных данных (более чем в два раза быстрее сортировки пузырьком). При этом сортировка вставками несравнимо быстрее остальных алгоритмов при уже отсортированном массиве, так как в лучшем случае имеет трудоёмкость $Q(N)$.

Заключение

В данной работе было рассмотрено три алгоритма сортировки: пузырёк, выбором и вставками. Был описан и реализован каждый каждый алгоритм. Также были показаны схемы работы алгоритмов и приведены тесты и сравнительный анализ алгоритмов.

- изучены три алгоритма сортировки;
- реализованы изученные алгоритмы;
- произведен сравнительный анализ реализаций алгоритмов сортировки;
- подготовлен отчет по лабораторной работе.

Литература

[1] Сортировка пузырьком [Электронный ресурс]. Режим доступа: `https://kvodo.ru/puzyirkovaya-sortirovka.html` (дата обращения: 14.10.2021)

[2] Сортировка выбором [Электронный ресурс]. Режим доступа: `https://kvodo.ru/sortirovka-vyiborom-2.html` (дата обращения: 14.10.2021)

[3] Сортировка вставками [Электронный ресурс]. Режим доступа: `https://kvodo.ru/sortirovka-vstavkami-2.html` (дата обращения: 14.10.2021)

[4] Кормен Т. Лейзерсон Ч. Риверст Р. Штайн К. Алгоритмы: построение и анализ. – М.: Вильямс, 2013. с. 1296.

[5] Кнут Д. Сортировка и поиск. – М.: Вильямс, 2000. Т. 3 из Искусство программирования. с. 834.