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»

Отчет по лабораторной работе № 3 по курсу "Анализ алгоритмов"

Тема Трудоемкость сортировок

Студент Хамзина Р. Р.

Группа ИУ7-53Б

Оценка (баллы)

Преподаватель Волкова Л. Л.

Москва — 2021 г.

Оглавление

Введение

При решении различных задач встает необходимость работы с упорядоченным набором данных. Например, при поиске элемента в заданном множестве. Для упорядочивания последовательности значений используется сортировка.

Сортировка - процесс перегруппировки заданного множества объектов в некотором определенном порядке [1]. Для реализации этого процесса разрабатываются алгоритмы сортировки. Такие алгоритмы состоят из трех основных шагов:

- сравнение элементов, задающее их порядок;
- обмен элементов в паре;
- сортирующий алгоритм, осуществляющий предыдущие два шага до полного упорядочивания.

Эффективность алгоритма зависит от скорости работы этого алгоритма. Скорость работы алгоритма сортировки определяется функциональной зависимостью среднего времени сортировки последовательностей элементов данных, определенной длины, от этой длины.

Существует большое количество алгоритмов сортировки. Все они решают одну и ту же задачу, причем некоторые алгоритмы имеют преимущества перед другими. Поэтому существует необходимость сравнительного анализа алгоритмов сортировки.

Цель работы - получить навык сравнительного анализа алгоритмов сортировки.

Для решения поставленной цели требуется решить следующие задачи:

- изучить три алгоритма сортировки: выбором, Шелла и гномью;
- разработать и реализовать указанные алгоритмы;
- протестировать реализацию рассматриваемых алгоритмов;
- провести сравнительный анализ реализованных алгоритмов по затраченному процессорному времени, по трудоемкости и по памяти.

1 Аналитическая часть

В данном разделе будут описаны алгоритмы сортировки выбором, Шелла и гномьей сортировки.

1.1 Сортировка выбором

Сортировка выбором[1] состоит из следующих шагов:

1. Выбирается элемент неотсортированной части последовательности с наименьшим значением;
2. Выбранный элемент меняется местами с элементом, стоящим на первой позиции в неотсортированной части. Обмен не нужен, если это и есть минимальный элемент;
3. Повтор шагов 1 и 2 до тех пор, пока не останется только наибольший элемент.

1.2 Сортировка Шелла

Сортировка Шелла[1] является усовершенствованием сортировки вставками. В сортировке вставками на каждом шаге, берут элемент входной последовательности и передают в готовую последовательность, вставляя его на подходящее место. Д. Л. Шелл предложил следующие шаги:

1. Выбирается некоторое расстояние d между элементами последовательности;
2. Сравниваются и сортируются значения, стоящие друг от друга на расстоянии d;
3. Шаг 2 повторяется для меньших значений d, не равных 1;
4. При d, равном 1, элементы упорядочиваются сортировкой вставками.

Приемлема любая последовательность для d, с условием, что последнее значение равно 1.

1.3 Гномья сортировка

Гномья сортировка выполняет следующие действия:

1. Сравниваются текущий и предыдущий элементы последовательности;
2. Если они расположены в необходимом порядке, то осуществляется переход к следующему элементу.
3. Иначе происходит обмен. Если предыдущий элемент не был первым, осуществляется переход на один элемент назад.

Шаги повторяются, пока возможен переход к следующему элементу.

Вывод

Были рассмотрены следующие алгоритмы сортировки: выбором, Шелла и гномья. Для указанных алгоритмов необходимо получить теоретическую оценку и доказать её экспериментально.

2 Конструкторская часть

В данном разделе будут представлены схемы алгоритмов сортировки выбором, Шеллом и гномьей сортировки и вычислены трудоемкости указанных алгоритмов.

2.1 Разработка алгоритмов

На рисунках представлены схемы алгоритмов сортировки выбором, Шеллом и гномьей сортировки.

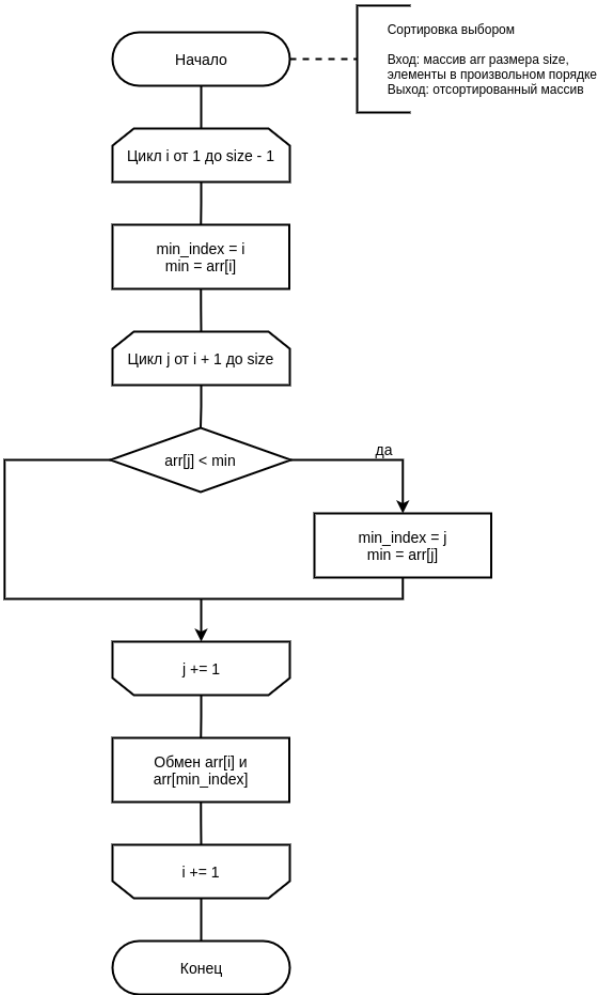

Рисунок 2.1 – Схема алгоритма сортировки выбором

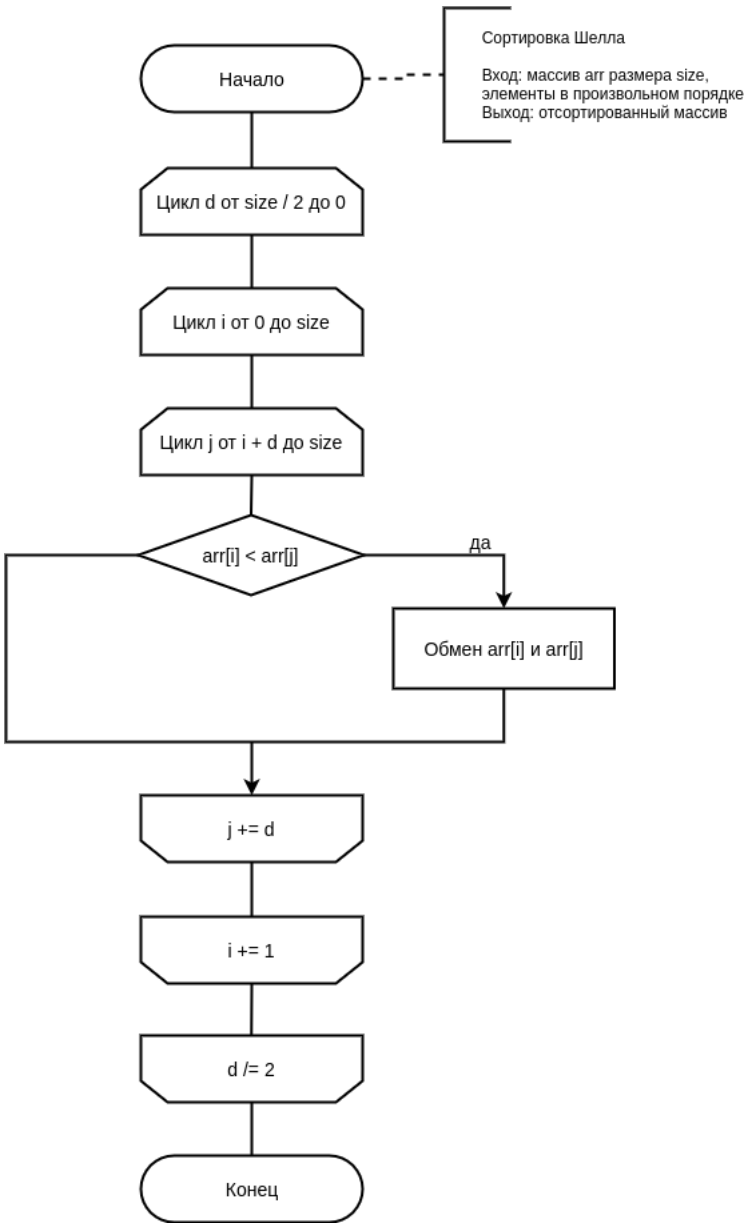

Рисунок 2.2 – Схема алгоритма сортировки Шелла

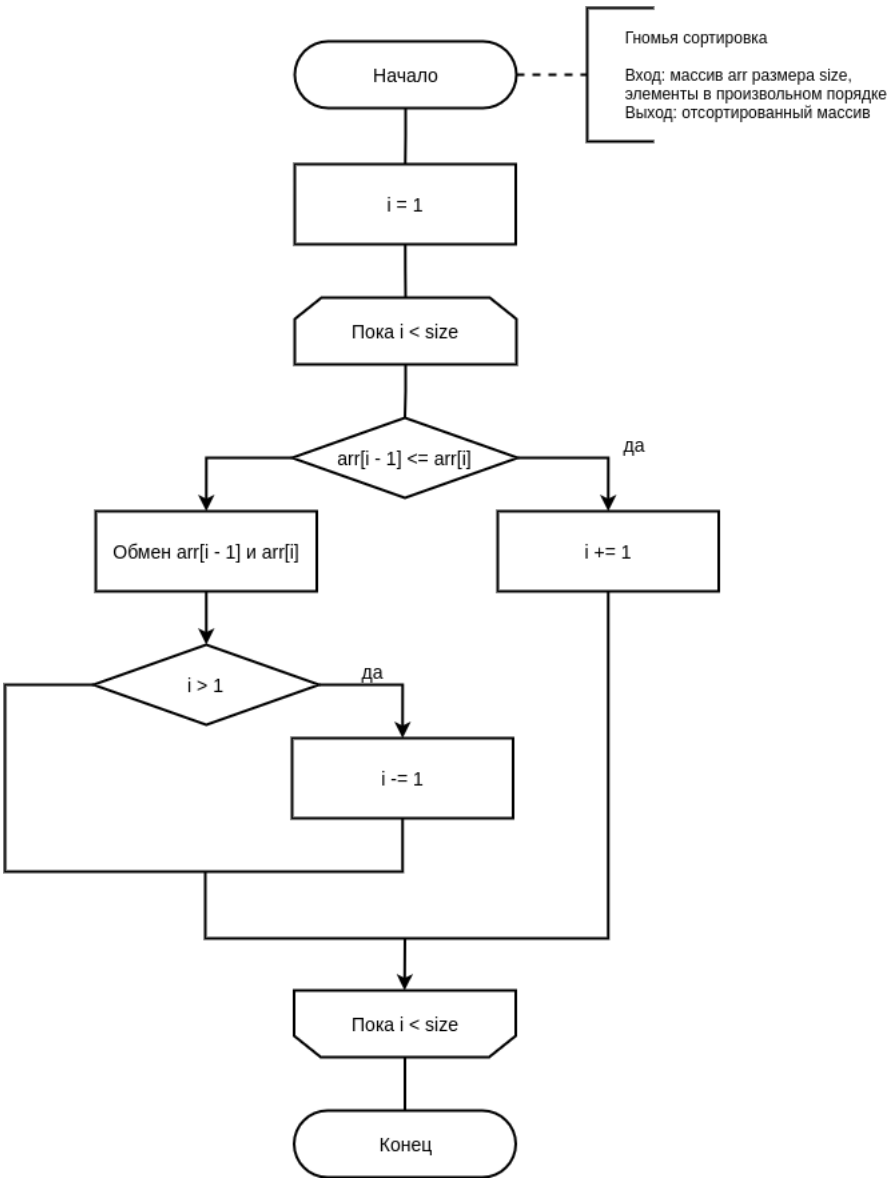

Рисунок 2.3 – Схема алгоритма гномьей сортировки

2.2 Модель вычислений для оценки трудоёмкости алгоритмов

Для определения трудоемкости алгоритмов необходимо ввести модель вычислений:

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);

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

3. трудоемкость цикла рассчитывается, как (2.3);

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

4. трудоемкость вызова функции равна 0.

2.3 Трудоёмкость алгоритмов

Вывод

3 Технологическая часть

В данном разделе будут указаны требования к программному обеспечению и средства реализации, будут представлены листинги кода, а также функциональные тесты.

3.1 Требования к ПО

К программе представлен ряд требований:

- на вход подается массив целых чисел;
- на выходе - отсортированный массив, поданный на вход.

3.2 Средства реализации

Реализация данной лабораторной работы выполнялась при помощи языка программирования Python. Выбор ЯП обусловлен простотой синтаксиса, большим числом библиотек и эффективностью визуализации данных.

Замеры времени проводились при помощи функции process_time из библиотеки time.

3.3 Сведения о модулях программы

Программа состоит из следующих модулей:

- main.py - главный файл программы, предоставляющий пользователю меню для выполнения основных функций;
- sort.py - файл, содержащий функции сортировок массива;
- arr.py - файл, содержащий функции создания массива различного типа и работы с массивом;

- time_test.py - файл, содержащий функции замеров времени работы сортировок;
- graph_result.py - файл, содержащий функции визуализации временных характеристик алгоритмов сортировок.

3.4 Листинги кода

Реализации алгоритмов сортировок выбором, Шелла и гномьей представлены на листингах 3.1, 3.2, 3.3.

Листинг 3.1 – Алгоритм сортировки выбором

```
def selection_sort(arr, size):
    for i in range(size - 1):
        min_element = arr[i]
        min_index = i

        for j in range(i + 1, size):
            if arr[j] < min_element:
                min_element = arr[j]
                min_index = j

        arr[i], arr[min_index] = arr[min_index], arr[i]

    return arr
```

Листинг 3.2 – Алгоритм сортировки Шелла

```
def shell_sort(arr, size):
    d = size // 2

    while d > 0:
        for i in range(0, size):
            for j in range(i + d, size, d):
                if arr[i] > arr[j]:
                    arr[i], arr[j] = arr[j], arr[i]
        d //= 2

    return arr
```

Листинг 3.3 – Алгоритм гномьей сортировки

```
def gnome_sort(arr, size):
    i = 1

    while i < size:
        if arr[i - 1] <= arr[i]:
            i += 1
        else:
            arr[i - 1], arr[i] = arr[i], arr[i - 1]

            if i > 1:
                i -= 1

    return arr
```

3.5 Функциональные тесты

В таблице 3.1 приведены функциональные тесты, для функций, реализующих алгоритмы сортировок. Все тесты пройдены успешно.

Таблица 3.1 – Функциональные тесты

Входной массив	Ожидаемый результат	Результат
$[1, 2, 3, 4, 5, 6]$	$[1, 2, 3, 4, 5, 6]$	$[1, 2, 3, 4, 5, 6]$
$[5, 4, 3, 2, 1]$	$[1, 2, 3, 4, 5]$	$[1, 2, 3, 4, 5]$
$[0, -5, 3, 7, -8]$	$[-8, -5, 0, 3, 7]$	$[-8, -5, 0, 3, 7]$
$[17]$	$[17]$	$[17]$
$[]$	$[]$	$[]$

Вывод

Были реализованы функции алгоритмов сортировки выбором, Шелла и гномьей. Было проведено функциональное тестирование указанных функций.

4 Исследовательская часть

Заключение

Литература

[1] Н.Вирт Алгоритмы и структуры данных. 1989.