Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский химико-технологический университет имени Д.И. Менделеева»

Кафедра информационных компьютерных технологий

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

Выполнил студент группы КС-36 Меркушов Даниил Игоревич

Ссылка на репозиторий: https://github.com/VooDooCry/Algorithms\_KS36.git

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Лобанов Алексей Владимирович

Крашенинников Роман Сергеевич

Дата сдачи: 17.02.2025

Оглавление

[Описание задачи. 2](#_Toc63548272)

[Описание метода/модели. 2](#_Toc63548273)

[Выполнение задачи. 2](#_Toc63548274)

[Заключение. 2](#_Toc63548275)

# Описание задачи.

Необходимо реализовать алгоритм сортировки пузырьком, который упорядочивает элементы массива целых чисел в порядке возрастания.

# Описание метода/модели.

Сортировка **пузырьком** – это простой алгоритм, который многократно проходится по массиву, сравнивает соседние элементы и меняет их местами, если они находятся в неправильном порядке. Этот процесс повторяется до тех пор, пока массив не будет отсортирован.

# Выполнение задачи.

Язык программирования: **Golang**

Входные данные / тесты: (1000, 2000, 4000, 8000, 16000, 32000, 64000, 128000)

1. Импортирование пакетов: код импортирует необходимые пакеты: fmt для форматированного вывода, math/rand для генерации случайных чисел и time для работы с временными измерениями.
2. Функция **bubbleSort**: принимает массив чисел с плавающей запятой и сортирует его методом пузырьковой сортировки. Подсчитывает количество проходов и обменов (swap), выполняемых в процессе сортировки, и возвращает их вместе с отсортированным массивом.
3. Функция **generateRandomArray**: создает и возвращает массив случайных чисел с плавающей запятой заданного размера. Числа в диапазоне от -1 до 1.
4. Функция **runSeries**: выполняет серию сортировок для массива заданного размера и фиксированного количества итераций (***seriesLength***). Генерирует случайный массив, сортирует его с помощью **bubbleSort**, измеряет время сортировки и выводит результаты на экран.
5. Функция **main**: инициализирует генератор случайных чисел и задает размеры массивов, по которым будет проведена работа. Запускает серию сортировок для каждого размера массива, собирает и сохраняет результаты времени выполнения в карту ***allTimes***. В конце вычисляет и выводит среднее время сортировки для каждого размера массива.

# Заключение.

Основные выводы:

1. Простейший для понимания и реализации по сравнению с другими алгоритмами.
2. Эффективен для массивов небольшого размера.
3. Сложность алгоритма: O(n²). Это делает его очень медленным для больших наборов данных.

**Графики и результаты**:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **1000(s)** | **2000(s)** | **4000(s)** | **8000(s)** | **16000(s)** | **32000(s)** | **64000(s)** | **128000(s)** |
| Среднее значение | 0,001 | 0,00354 | 0,01285 | 0,0575 | 0,3011 | 1,73 | 7,67 | 33,3 |
| Наилучшее | 0,00052 | 0,00267 | 0,01132 | 0,05571 | 0,2959 | 1,72 | 7,62 | 31,4 |
| Наихудшее | 0,0021 | 0,00475 | 0,01485 | 0,06114 | 0,308 | 1,76 | 7,79 | 35,3 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **1000** | **2000** | **4000** | **8000** | **16000** | **32000** | **64000** | **128000** |
| Среднее значение | 972 | 1965 | 3944 | 7891 | 15673 | 31788 | 63635 | 127743 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **N** | **1000** | **2000** | **4000** | **8000** | **16000** | **32000** | **64000** | **128000** |
| Среднее значение | 250882 | 1005099 | 3980944 | 15955084 | 63969055 | 255330811 | 1023462884 | 4094346558 |