|  |
| --- |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МИРЭА – Российский технологический университет"**  **РТУ МИРЭА** |

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-14 «Цифровые технологии обработки данных»

**ОТЧЕТ   
о выполнении лабораторной работы №1**

**«Реализация сортировки линейных структур данных»**

**по дисциплине   
«Алгоритмы и структуры данных»**

**Вариант № {{ task\_num }}**

Выполнил: студент 2 курса

группы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

шифр \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(фио студента)*

Проверил: *\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Москва 2022 г.

**Задание на лабораторную работу № 1.**

В рамках лабораторной работы №1 требуется программно реализовать (с помощью указателей (однонаправленных/двунаправленный динамический линейный связанный список, массива или используя стандартный контейнер библиотеки STL “stack” или «queue» - по варианту) абстрактный тип данных (АТД) в соответствии с заданием (стек, дек, очередь с одной головой, очередь с головой и хвостом).

Абстрактный тип данных должен позволять осуществлять только операции, присущие типу линейного связанного списка:

* получить значение первого элемента (на выходе),
* добавить элемент (на вход),
* удалить элемент из списка (на выходе),
* проверить – список пуст,
* обнулить (проинициализировать) список (конструктор, при необходимости).
* деструктор (при необходимости)

Используя разработанный АТД и указанный набор операций, необходимо реализовать заданный алгоритм сортировки последовательности элементов заданного типа, при этом следует учитывать, что разрешен доступ (чтение/извлечение) только к элементу на выходе.

На основе исходного текста программы получить аналитическую оценку трудоемкости работы алгоритма сортировки, используя О-символику для каждого реализованного метода АТД и сортировки в целом.

**Вариант № {{ task\_num }}.**

**Реализация связи элементов линейного списка:** **{{ link\_type }}**

**Способ организации линейного связанный список: {{ collection\_type }}**

**Алгоритм сортировки**: **{{ sort\_name }}**

**Теория о сортировках.**

{{p sort*\_*theory}}

**Листинг программы с расчетами.**

{{ task\_code }}

F(n) = {{ sort\_asymptotic }}

O(F(n)) = {{ sort\_bigO\_asymptotic }}

**Таблица результата экспериментов и графики зависимостей**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **n** | **F(n)** | **O(F(n))** | **T(n) (сек)** | **N\_OP** |
| **{%tr for row in tests\_results %}** | | | | |
| {{ row.n }} | {{ row.f\_n }} | {{ row.o\_f\_n }} | {{ row.time }} | {{ row.n\_op }} |
| **{%tr endfor %}** | | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **C1=F(n)/T(n)** | **C2=O(F(n))/T(n)** | **C3=F(n)/N\_OP** | **C4=F(n)/N\_OP** |
| **{%tr for row in tests\_results %}** | | | |
| {{ row.c1 }} | {{ row.c2 }} | {{ row.c3 }} | {{ row.c4 }} |
| **{%tr endfor %}** | | | |

{{ c1 }}

{{ c2}}

{{ c3 }}

{{ c4 }}

**Скриншот работы программы:**

{{ task\_out\_example }}

Рисунок 1. Результат работы программы в режиме example

{{ task\_out\_tests }}

Рисунок 2. Результат работы программы в режиме test

**Выводы.**

В результате данной работы был реализован класс {{ collection\_name }} реализующий {{ collection\_type.lower() }} с типом связи – {{ link\_type.lower() }}. Также были реализованы базовые операции над структурой и рассчитана их алгоритмическая сложность.

По результатам экспериментов было установлено, что алгоритм сортировки «{{ sort\_name }}» для этой структуры имеет {{ sort\_bigO\_asymptotic}} зависимость от числа элементов.

**Литература:**

1. Алгоритмы: построение и анализ. Кормен Т.Х., Лейзерсон Ч.И.
2. The Art of Computer Programming. Volume 3. Sorting and Searching / под ред. В. Т. Тертышного (гл. 5) и И. В. Красикова (гл. 6).