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 № 3

Выполнил студент группы КС-36 (Потапов Никита Александрович)

Ссылка на репозиторий: (Ссылка на лабораторную в репозитории)

Приняли: Пысин Максим Дмитриевич

Краснов Дмитрий Олегович

Лобанов Алексей Владимирович

Крашенинников Роман Сергеевич

Дата сдачи: (10.03.2025)

Оглавление

[Описание задачи. 2](#_Toc63548272)

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

[Выполнение задачи. 2](#_Toc63548274)

[Заключение. 2](#_Toc63548275)

# Описание задачи.

Задание на лабораторную #3.

1. Написать свою реализацию двусвязного списка

- Добавление элемента в начало, в конец, в произвольное место

- Удаление элемента по из списка

В рамках лабораторной работы необходимо изучить и реализовать одну из трёх структур(двухсвязный список, стек, очередь), в соответствии со своим вариантом, при этом, все структуры должны:

- Использовать шаблонный подход, обеспечивая работу контейнера с произвольными данными.

- Реализовывать свой итератор предоставляющий стандартный для языка механизм работы с ним(для С++ это операции ++ и операция !=, )

- Обеспечивать работу стандартных библиотек и конструкции for each если она есть в языке, если их нет, то реализовать собственную функцию использующую итератор.

- Проверку на пустоту и подсчет количества элементов.

Для демонстрации работы структуры необходимо создать набор тестов (под тестом понимается функция, которая создаёт структуру, проводит операцию или операции над структурой и удаляет структуру):

- заполнение контейнера 1000 целыми числами в диапазоне от -1000 до 1000 и подсчет их суммы, среднего, минимального и максимального.

- Провести проверку работы операций вставки и изъятия элементов на коллекции из 10 строковых элементов.

- заполнение контейнера 100 структур содержащих фамилию, имя, отчество и дату рождения (от 01.01.1980 до 01.01.2020) значения каждого поля генерируются случайно из набора заранее заданных. После заполнение необходимо найти всех людей младше 20 лет и старше 30 и создать новые структуры, содержащие результат фильтрации, проверить выполнение на правильность подсчётом кол-ва элементов, не подходящих под условие в новых структурах.

Тесты по вариантам:

1. Список

- Перемешать все элементы

- Выполнить серию тестирования сортировки из первой лабораторной работы на реализованном списке и сравнить производительность с полученной на массиве.

# Описание метода/модели.

Список это структура данных, в которой каждый элемент связан с одним из своих соседних элементов, для двунаправленного списка это следующий и предыдущий элемент:



Структура одного элемента списка:

Двунаправленный – поле указателя на предыдущий, поле значения, поле указателя на последующий

Имея такую структуру список позволяет легко проводить операции вставки и удаления, так же в зависимости от реализации списки могут поддерживать обращения по индексу, хотя в базовой модели такого не предполагается.

*Вставка*

Вне зависимости о того, является ли список двунаправленным или однонаправленным, для вставки элемента алгоритм будет выглядеть так:

1. Получить номер (или, чаще всего, итератор) элемента после которого производиться вставка.
2. Получить вставляемое значение
3. Создать новый объект элемента списка с полем значения нужного типа и полем указателя на следующий элемент.
4. Записать в поле значения нового объекта вставляемое значение
5. Записать в поле указателя на следующий элемент поставить значение аналогичного поля после которого производиться вставка
6. У элемента, после которого производиться вставка, поменять значения поля указателя на следующий элемент на указатель на новосозданный элемент списка

Для двунаправленного списка добавляются аналогичные действия с указателем на предыдущий элемент



*Удаление*

Вне зависимости о того, является ли список двунаправленным или однонаправленным, для удаления элемента алгоритм будет выглядеть так:

1. Получить номер (или, чаще всего, итератор) элемента который подлежит удалению из списка.
2. Найти элемент предшествующий удаляемому элементу
3. Сохранить значение поля указателя на последующий элемент удаляемого элемента
4. Вставить в поле указателя на последующий элемент предшествующего удаляемому элементу сохраненное значение
5. Удалить элемент

Для двунаправленного списка добавляются аналогичные действия с указателем на предыдущий элемент.



**Преимущества**:

* Невозможность переполнения такой структуры
* Быстрота и простота операций удаления и вставки по сравнению с массивом (и простота реализации по сравнению с другими структурами)
* Быстрота манипуляции списком при работе с комплексными большими объектами, так как минимизируются операции копирования самих элементов, работа видеться с указателями.

**Недостатки**:

* Использование дополнительного места для хранения указателей.
* Отсутствие эффективного способа обращения к произвольному элементу.
* Слабая локализация данных в памяти, что делает невозможным применение процессорных векторных оптимизаций, т.е. не дает процессору сразу манипулировать группами элементов.

# Выполнение задачи.

Используемый язык и среда разработки

- Язык программирования: C++.

- Среда разработки: Visual Studio

Организация программы

Программа состоит из следующих компонентов:

1. Структура Node<T>:

- Представляет узел двусвязного списка.

- Содержит данные (data), указатель на предыдущий узел (prev) и указатель на следующий узел (next).

2. Класс DoublyLinkedList<T>:

- Реализует двусвязный список.

- Содержит методы для работы со списком:

- push\_front, push\_back — добавление элементов в начало и конец.

- insert, erase — вставка и удаление элементов по итератору.

- begin, end — получение итераторов на начало и конец списка.

- empty, count — проверка на пустоту и получение размера списка.

3. Класс Iterator:

- Реализует итератор для двусвязного списка.

- Поддерживает операции:

- operator\* — доступ к данным текущего узла.

- operator++ — переход к следующему узлу.

- operator==, operator!= — сравнение итераторов.

4. Тестовые функции:

- test1: Проверка работы с целыми числами (заполнение, подсчет суммы, среднего, минимума и максимума).

- test2: Проверка операций вставки и удаления строковых элементов.

- test3: Работа со структурами (фильтрация по возрасту).

- test4: Реализация сортировки вставками

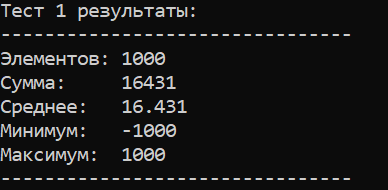
Проведенные тесты

1. Тест 1: Работа с целыми числами:

- Заполнение списка 1000 случайными числами в диапазоне [-1000, 1000].

- Подсчет суммы, среднего значения, минимального и максимального элементов.

- Цель: Проверить корректность работы с числовыми данными и производительность при большом объеме данных.



2. Тест 2: Работа со строками:

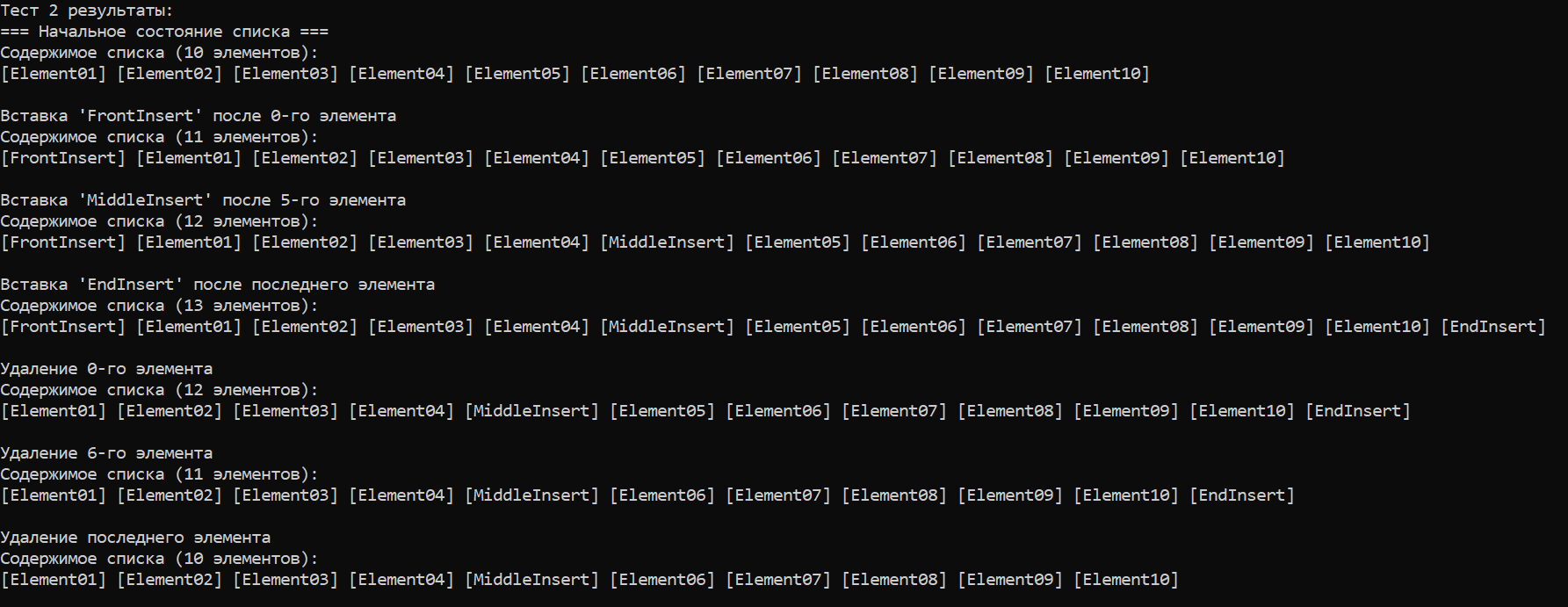
- Заполнение списка 10 строковыми элементами.

- Проверка операций вставки и удаления:

- Вставка в начало, середину и конец.

- Удаление из начала, середины и конца.

- Цель: Проверить корректность работы со строками



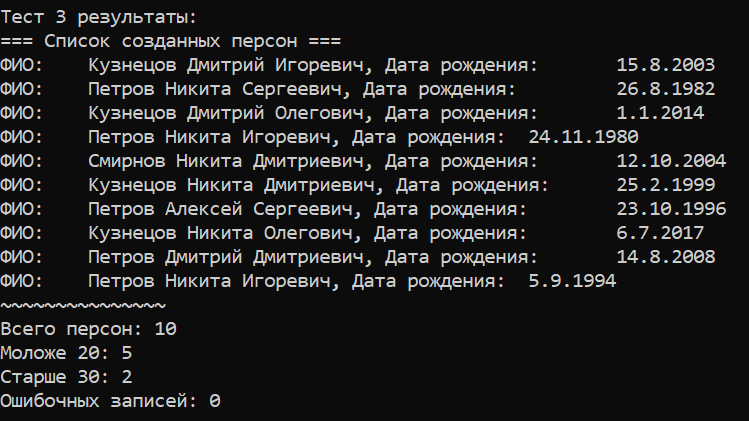
3. Тест 3: Работа со структурами:

- Заполнение списка 100 структурами, содержащими ФИО и дату рождения.

- Фильтрация по возрасту (младше 20 и старше 30 лет).

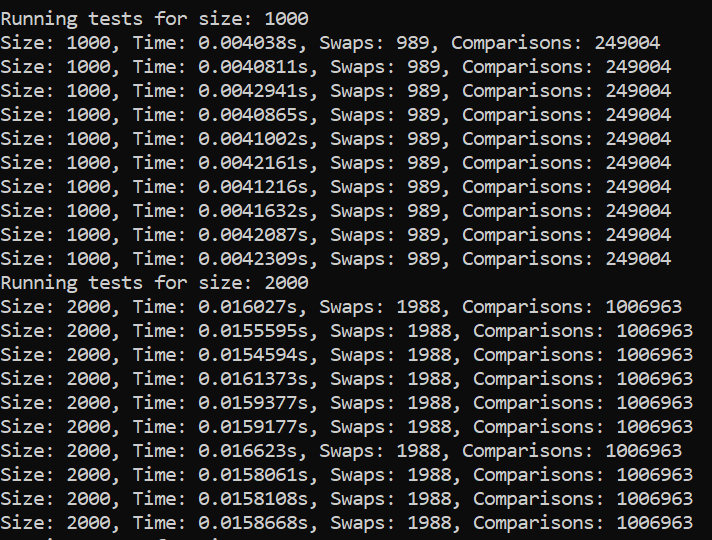
- Проверка корректности фильтрации.

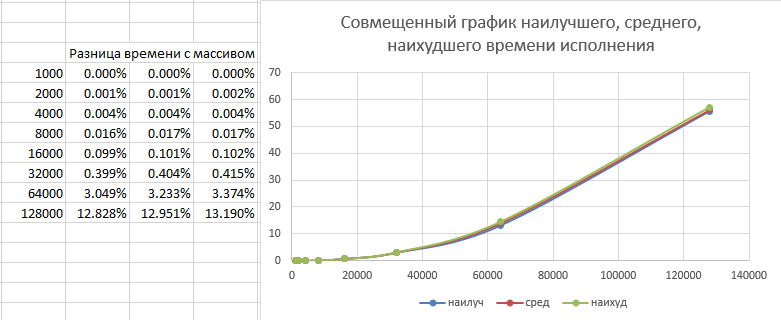
- Цель: Проверить работу со сложными типами данных.



4. Тест 4: Реализация метода вставками

- Цель: Выполнить сортировку вставками на списке и сравнить результаты с массивом





# Заключение.

Двусвязный список — это гибкая и мощная структура данных, которая хорошо подходит для задач с динамическим изменением данных и двунаправленным обходом. Однако его реализация требует внимательного подхода к управлению памятью и работе с указателями. В ходе выполнения лабораторной работы были успешно реализованы и протестированы основные операции, что подтвердило корректность и эффективность структуры данных. Личный опыт реализации показал, что двусвязный список — это ценный инструмент, но его использование должно быть обосновано требованиями задачи.