Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы» направление подготовки: 09.03.04 – «Программная инженерия»

**Лабораторная работа №9.**

**«Обработка исключительных ситуаций»**

Выполнил студент гр. РИС-24-2б

Молочко Артём Анатольевич

Проверил:

Доц. Каф. ИТАС

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

г. Пермь, 2025

**Постановка задачи**

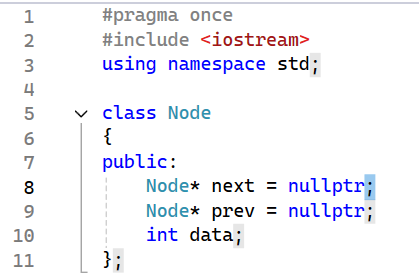
1. Реализовать класс List с базовыми операциями.
2. Отработать механизмы обработки исключений:
   * С использованием стандартного типа данных (например, throw 1).
   * С использованием иерархии пользовательских классов исключений.

**Анализ задачи**

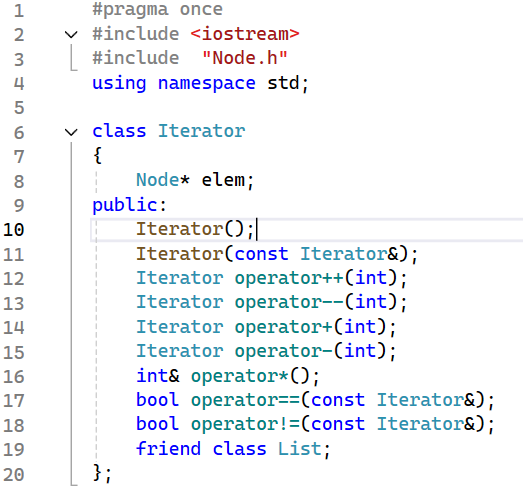
* Реализован двусвязный список с указателями head и tail, а также полем size.
* Есть конструкторы (обычный и копирующий), деструктор, методы добавления (pushback, pushfront) и удаления (popback, popfront).
* Поддерживаются операторы: [] (доступ по индексу), \* (поэлементное умножение списков), << (вывод в поток).
* Реализованы базовые операции с итераторами (++, --, \*, +, -).
* Обработка исключений через throw int и иерархии классов (выход за границы списка, максимальный размер, доступ к итератору и тд)

Код программы

Node.h



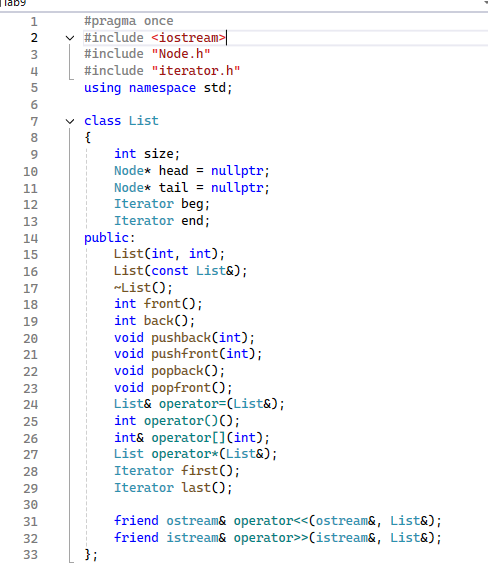
Iterator.h

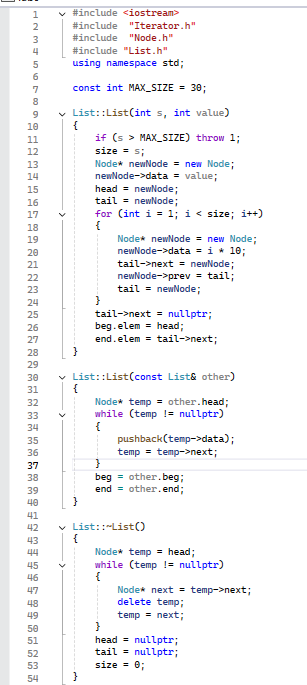


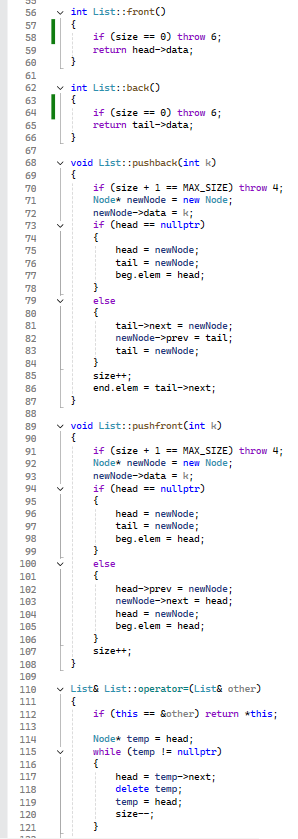
Iterator.cpp

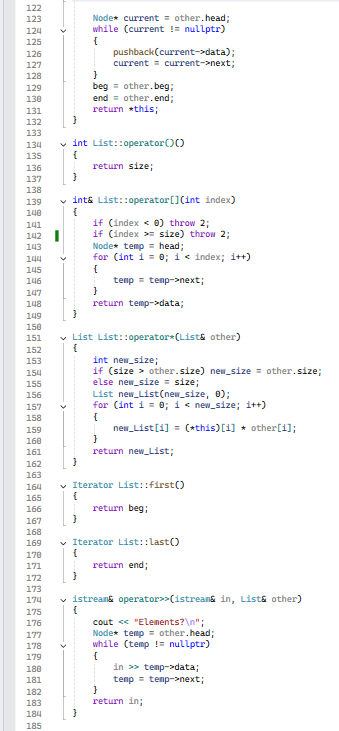


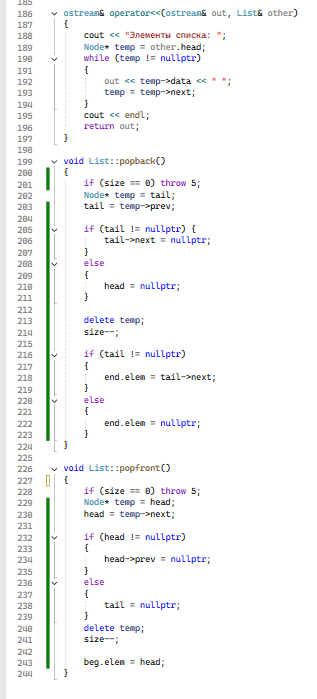
List.h



List.cpp





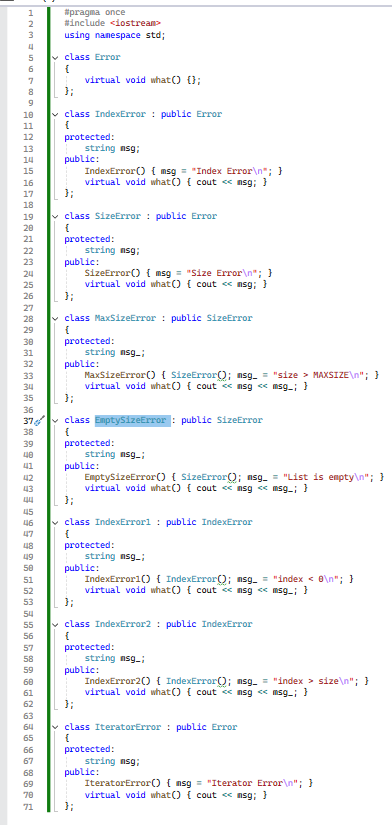


Main.cpp

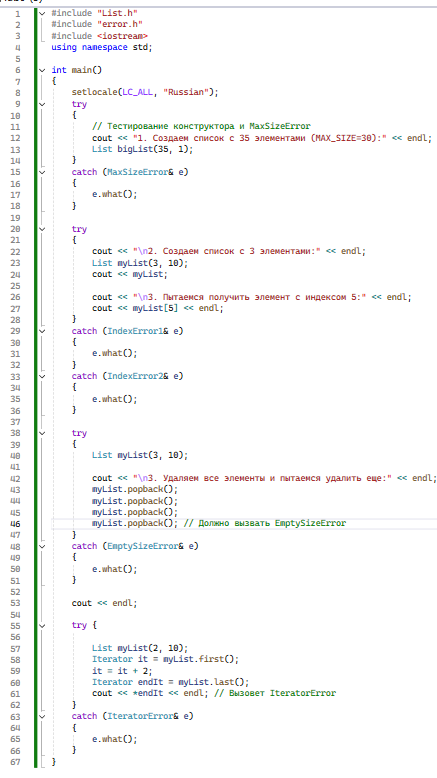


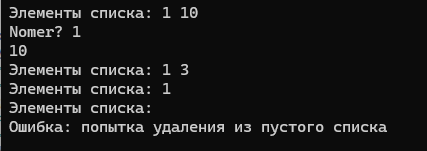
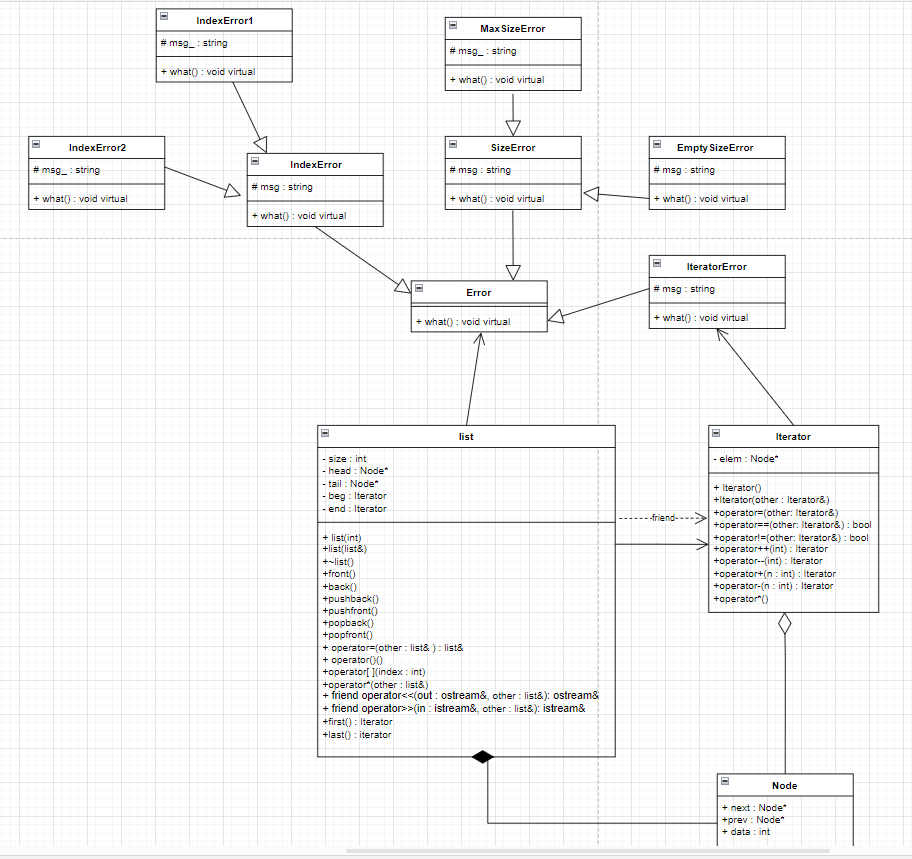
Иерархия классов:

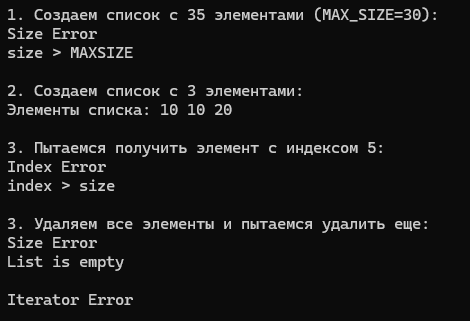
Error.h



Main.cpp







Контрольные вопросы

**1. Что представляет собой исключение в С++?**  
**Ответ:** Исключение в C++ — это механизм обработки ошибок, который позволяет прервать нормальное выполнение программы и передать управление специальному блоку кода (catch).

**2. На какие части исключения позволяют разделить вычислительный процесс? Достоинства такого подхода?**  
**Ответ:**

* **Части:**
  1. Генерация исключения (throw).
  2. Обработка исключения (try-catch).
* **Достоинства:**
  1. Отделение кода обработки ошибок от основной логики.
  2. Возможность централизованной обработки ошибок.

**3. Какой оператор используется для генерации исключительной ситуации?**  
**Ответ:** Оператор throw.

**4. Что представляет собой контролируемый блок? Для чего он нужен?**  
**Ответ:**

* **Контролируемый блок** — это участок кода, заключенный в try { ... }.
* **Назначение:** Позволяет отслеживать и обрабатывать исключения в этом блоке.

**5. Что представляет собой секция-ловушка? Для чего она нужна?**  
**Ответ:**

* **Секция-ловушка** — это блок catch (...), который перехватывает исключения.
* **Назначение:** Обработка возникших исключений.

**6. Какие формы может иметь спецификация исключения в секции ловушке? В каких ситуациях используются эти формы?**  
**Ответ:**

1. catch (тип) — ловит конкретный тип исключения.
2. catch (тип&) — ловит исключение по ссылке (экономит память).
3. catch (...) — ловит все исключения (универсальный обработчик).

**7. Какой стандартный класс можно использовать для создания собственной иерархии исключений?**  
**Ответ:** Базовый класс std::exception из <stdexcept>.

**8. Каким образом можно создать собственную иерархию исключений?**  
**Ответ:** Наследованием от std::exception или другого базового класса исключений.

**9. Если спецификация исключений имеет вид**: void f1() throw(int, double);, **то какие исключения может генерировать функция**f1**?**  
**Ответ:** Только int и double.

**10. Если спецификация исключений имеет вид:**void f1() throw();, **то какие исключения может генерировать функция** f1**?**  
**Ответ:** Никакие (устаревший синтаксис, в C++11+ заменен на noexcept).

**11. В какой части программы может генерироваться исключение?**  
**Ответ:** В любой части программы, где используется throw.

**12. Написать функцию, которая вычисляет площадь треугольника по трем сторонам (формула Герона).**

**1. Без спецификации исключений:**

double triangleArea(double a, double b, double c) {

double p = (a + b + c) / 2;

return sqrt(p \* (p - a) \* (p - b) \* (p - c));

}

#### ****2. Со спецификацией****throw()

double triangleArea(double a, double b, double c) throw() {

double p = (a + b + c) / 2;

return sqrt(p \* (p - a) \* (p - b) \* (p - c));

}

#### ****3. С конкретной спецификацией (стандартное исключение):****

#include <stdexcept>

double triangleArea(double a, double b, double c) throw(std::invalid\_argument) {

if (a <= 0 || b <= 0 || c <= 0 || a + b <= c || a + c <= b || b + c <= a) {

throw std::invalid\_argument("Неверные стороны треугольника");

}

double p = (a + b + c) / 2;

return sqrt(p \* (p - a) \* (p - b) \* (p - c));

}

#### ****4. С собственным исключением:****

class TriangleError : public std::exception {

public:

const char\* what() const noexcept override {

return "Ошибка: недопустимые стороны треугольника";

}

};

double triangleArea(double a, double b, double c) throw(TriangleError) {

if (a <= 0 || b <= 0 || c <= 0 || a + b <= c || a + c <= b || b + c <= a) {

throw TriangleError();

}

double p = (a + b + c) / 2;

return sqrt(p \* (p - a) \* (p - b) \* (p - c));

}