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 "

15 вариант

Выполнил студент гр. РИС-24-2б

Иванова Елена Олеговна

Проверил:

Доц. Каф. ИТАС

Ольга Андреевна Полякова

(оценка) (подпись)

(дата)

г. Пермь, 2025

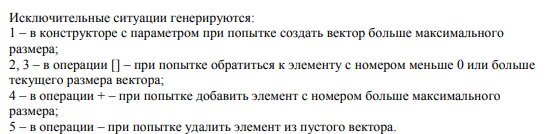
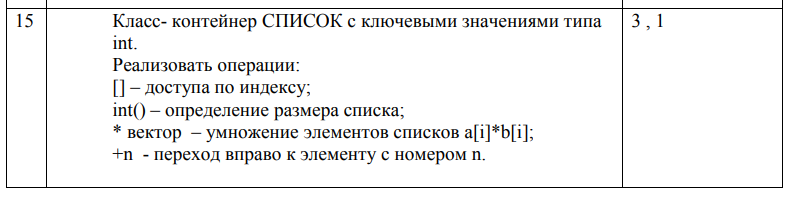
Цель задания:

1. Создание консольного приложения, состоящего из нескольких файлов в системе программирования Visual Studio.
2. Разработка программы, обрабатывающей исключительные ситуации.

Постановка задачи:

1. Реализовать класс, перегрузить для него операции, указанные в варианте.
2. Определить исключительные ситуации.
3. Предусмотреть генерацию исключительных ситуаций.

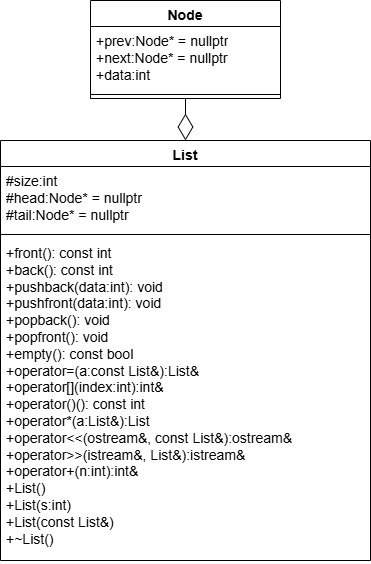
Задача:



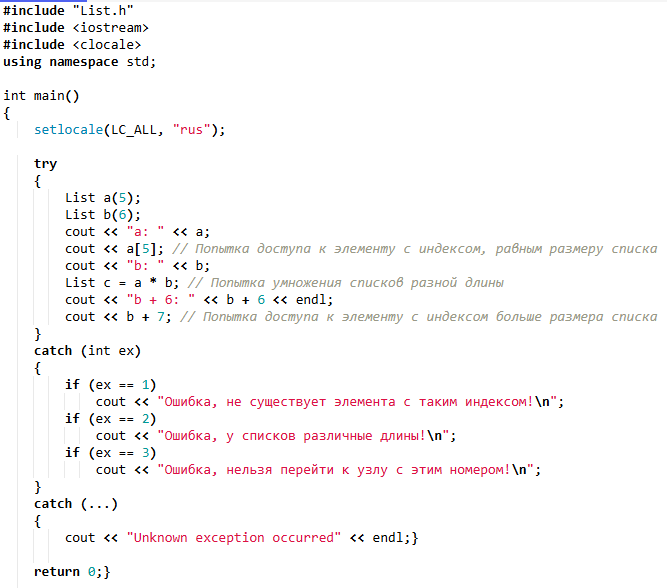
Вариант реализации 1

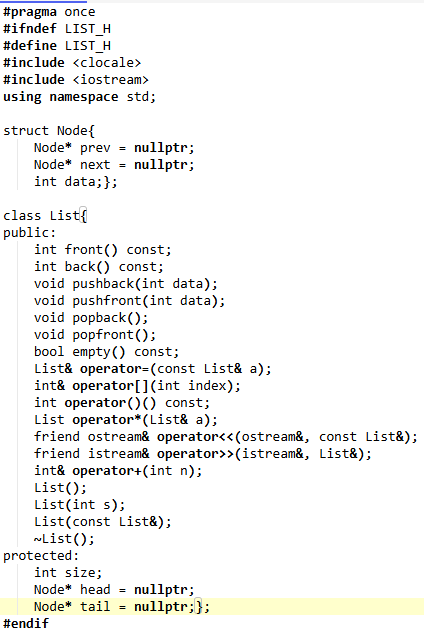
Информация об исключительных ситуациях передается с помощью стандартного типа данных.

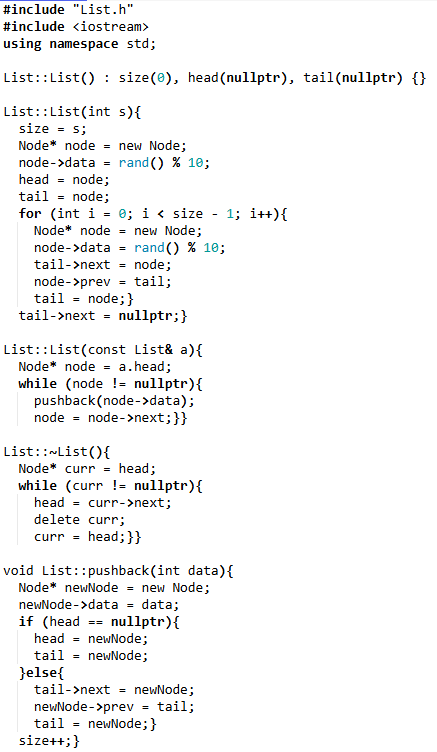
Диаграмма.

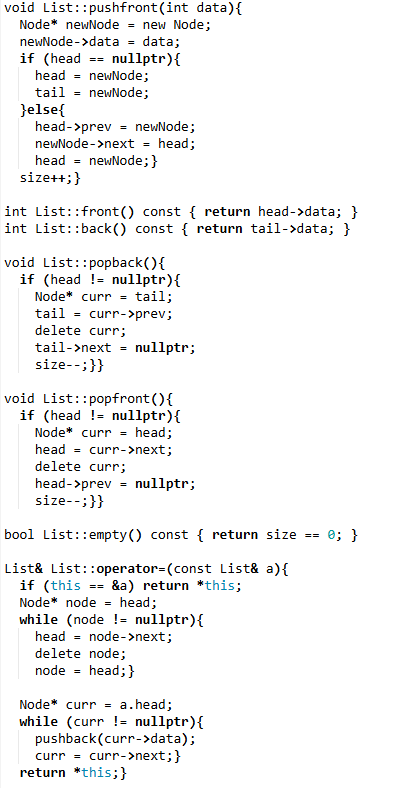


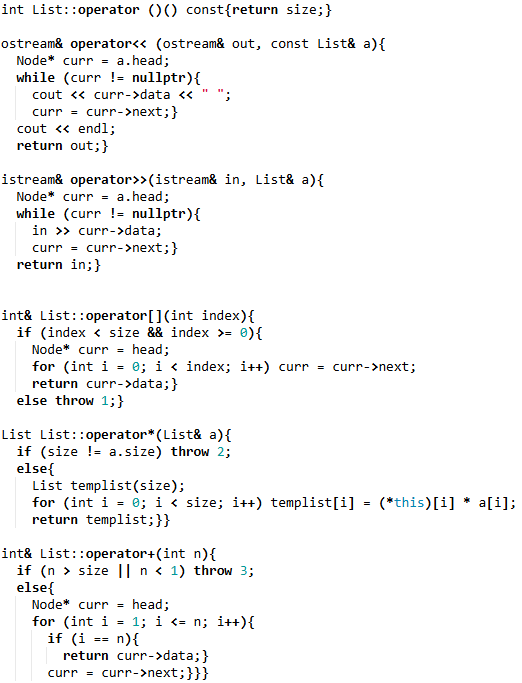
Код.







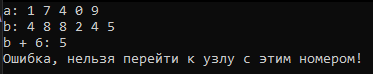




Вывод.



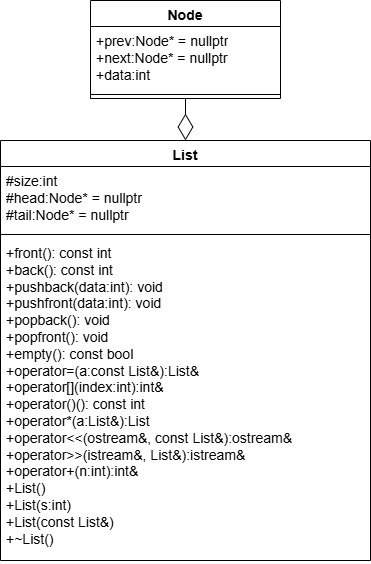


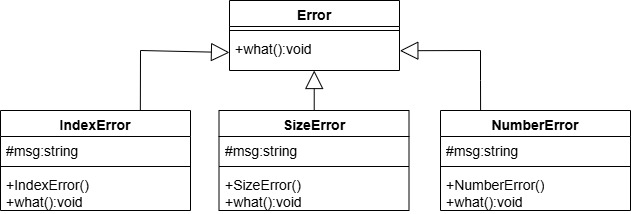


Вариант реализации 3

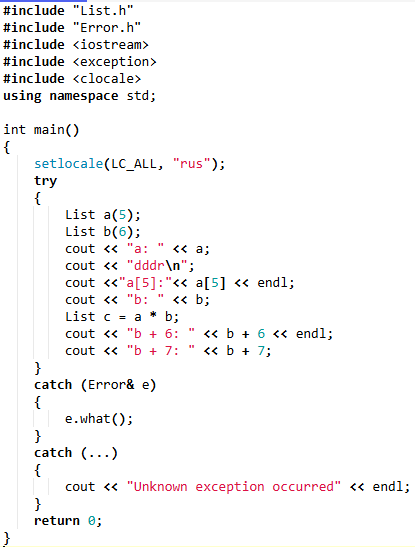
Информация об исключительных ситуациях передается с помощью иерархии пользовательских классов.

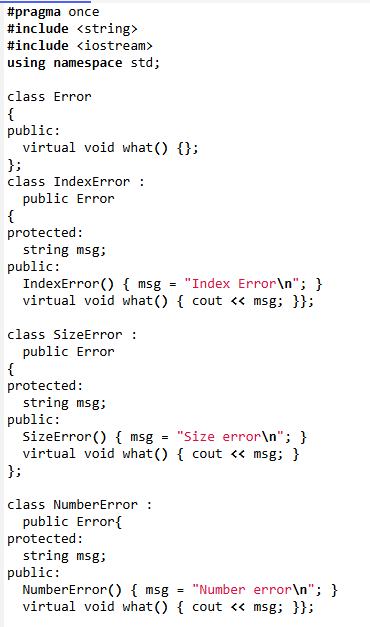
Диаграмма.

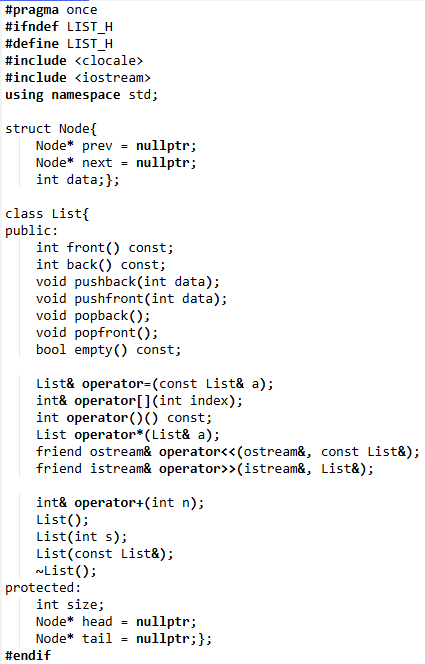


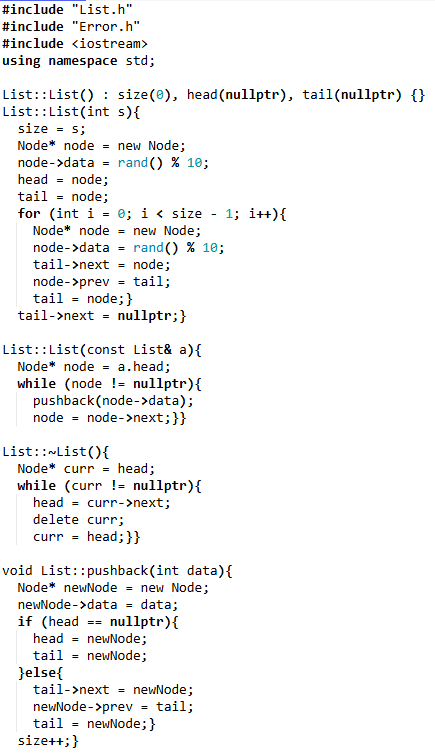


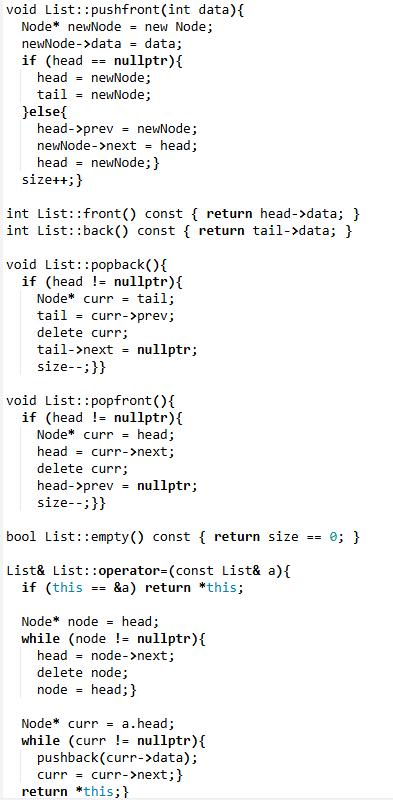
Код.

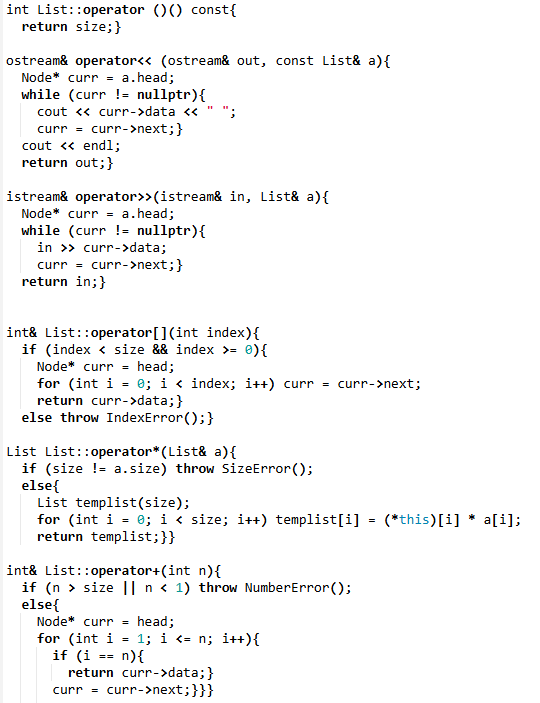








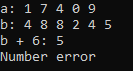




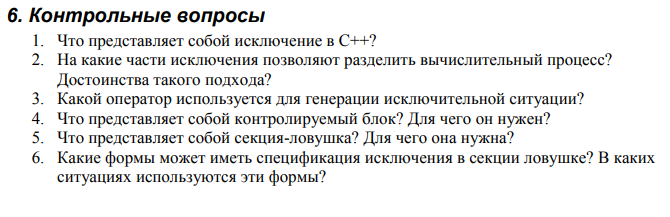
Вывод.

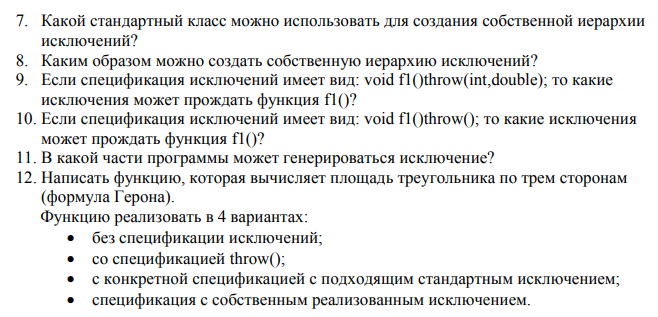






Контрольные вопросы.





1. В С++ исключение – это объект, который система должна генерировать при возникновении исключительной ситуации. Генерация такого объекта и создает исключительную ситуацию.
2. Исключения позволяют разделить вычислительный процесс на 2 части:

⦁ Обнаружение аварийной ситуации (неизвестно как обрабатывать);

⦁ Обработка аварийной ситуации (неизвестно, где она возникла).

Достоинства такого подхода:

⦁ Удобно использовать в программе, которая состоит из нескольких модулей;

⦁ Не требуется возвращать значение в вызывающую функцию.

1. Исключение генерируется оператором throw <выражение>, где <выражение> -

⦁ либо константа,

⦁ либо переменная некоторого типа,

⦁ либо выражение некоторого типа.

1. Проверка возникновения исключения делается с помощью оператора try, с которым неразрывно связаны одна или несколько блоков обработки исключений — catch. Оператор try объявляет в любом месте программы контролируемый блок. Контролируемый блок, помимо функции контроля, обладает функциями обычного блока: все переменные, объявленные внутри него, являются локальными в этом блоке и не видны вне его.
2. После блока try обязательно прописывается один или несколько блоков catch. Блок кода после catch предложения является обработчиком исключений. Это обработчик, который перехватывает исключение, которое возникает, если типы в throw выражениях и catch совместимы.
3. Спецификация исключения может иметь три формы:

1) (тип имя)

2) (тип)

3) (…)

Тип – это встроенный тип или тип, определенный программистом.

Формы 1 и 2 обрабатывают конкретные исключения, а форма 3 перехватывает все исключения, такую ловушку надо помещать последней, тогда она будет обрабатывать все исключения, которые еще не были обработаны.

Форма 1 означает, что объект передается в блок обработки, чтобы его каким-то образом там использовать, например, для вывода информации в сообщении об ошибке. Примеры:

⦁ catch( exception e) // по значению

⦁ catch( exception &e) // по ссылке

⦁ catch( const exception &e) // по константной ссылке

⦁ catch( exception \*e) //по указателю

1. Можно определять собственные исключения, унаследовав их от класса exception.
2. Для создания собственной иерархии исключений надо объявить свой базовый класс-исключение, например:

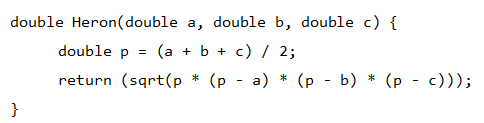
class BaseException{};

Остальные классы будут наследниками этого класса, аналогично тому, как это сделано в иерархии стандартных исключений:

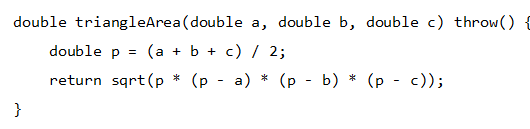
class Child\_Exception1:public BaseException{};

class Child\_Exception2:public BaseException{};

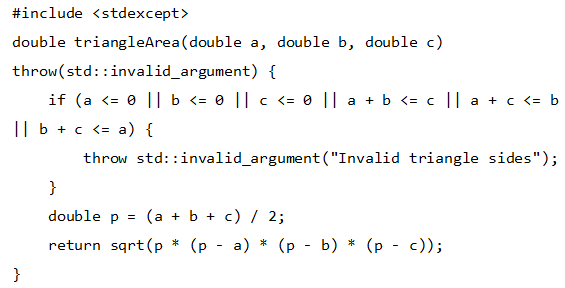
1. Функция f1() может генерировать исключения типа int и double.
2. Если спецификация имеет вид такой вид, то считается, что функция исключений не генерирует.
3. Исключение могут генерируется внутри оператора try { }.
4. ⦁ без спецификации исключений



⦁ со спецификацией throw()



⦁ с конкретной спецификацией с подходящим стандартным исключением



⦁ спецификация с собственным реализованным исключением

