Министерство образования и науки Российской Федерации

Федеральное государственное образовательное автономное учреждение высшего образования

"Пермский национальный исследовательский политехнический университет"

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №9

Дисциплина: Основы алгоритмизации и программирования

Тема: Обработка исключительных ситуаций

Вариант 15

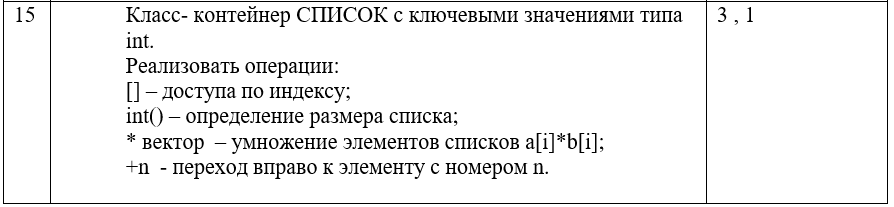
|  |  |
| --- | --- |
| Выполнил работу | |
| Студент группы РИС-22-1б | |
| Мерзляков В.Н. | |
|  | |
| Проверил работу | |
| Доцент кафедры ИТАС | |
| Полякова О.А. | |
|  | |

Пермь – 2023

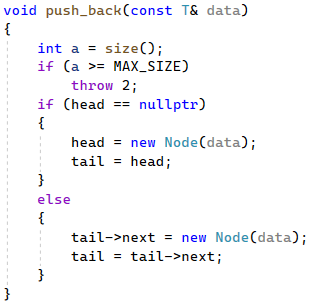
**Постановка задачи**

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

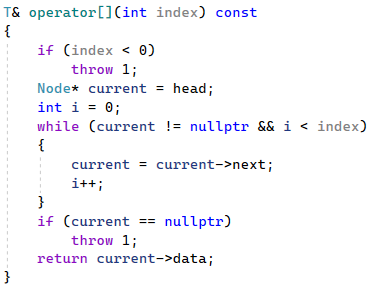
Вариант 15:



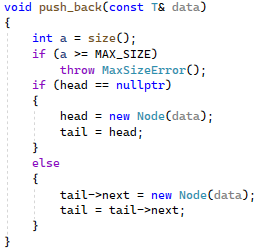
# Описание методов, вызывающие исключения



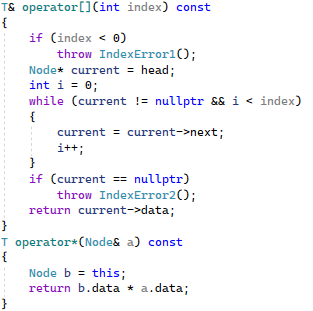
*Приложение 1 – Описание методов, вызывающие исключения (Вариант 1)*

**

*Приложение 2 – Описание методов, вызывающие исключения (Вариант 1)*

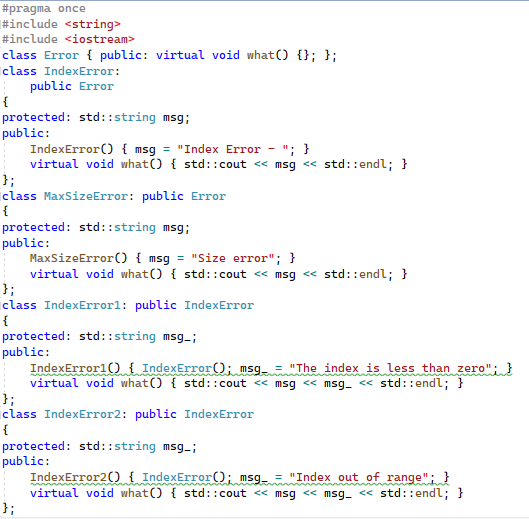


*Приложение 3 – Описание методов, вызывающие исключения (Вариант 3)*



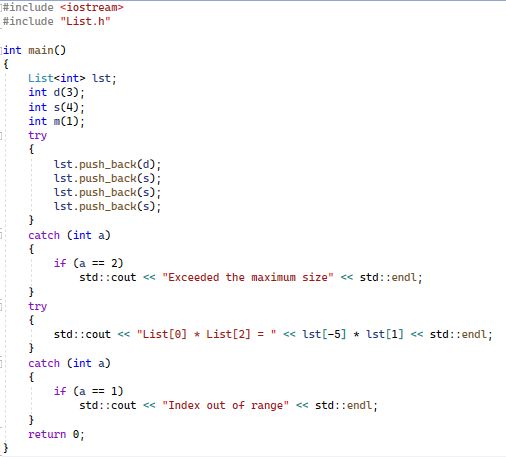
*Приложение 4 – Описание методов, вызывающие исключения (Вариант 3)*

# Описание класса исключений

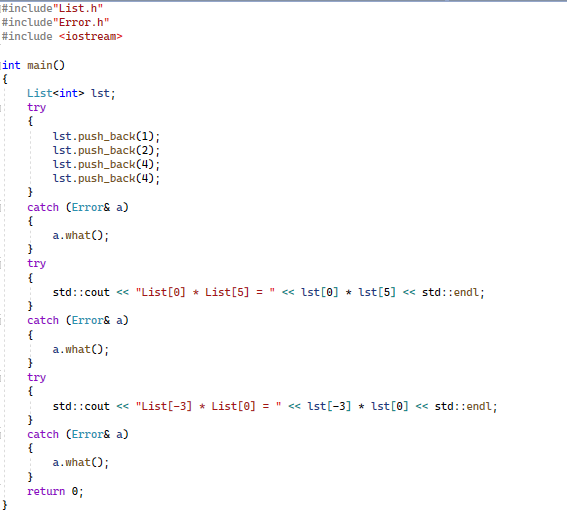
****

*Приложение 5 – Описание класса исключений*

**Функция main()**

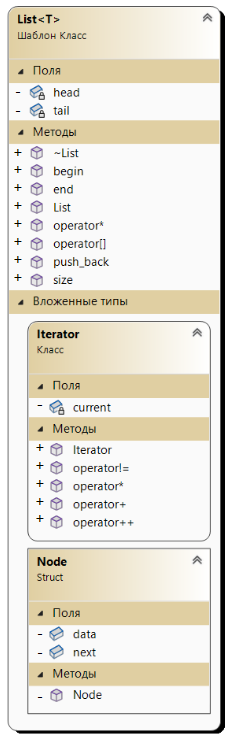
****

*Приложение 6 – Главная функция (Вариант реализации 1)*

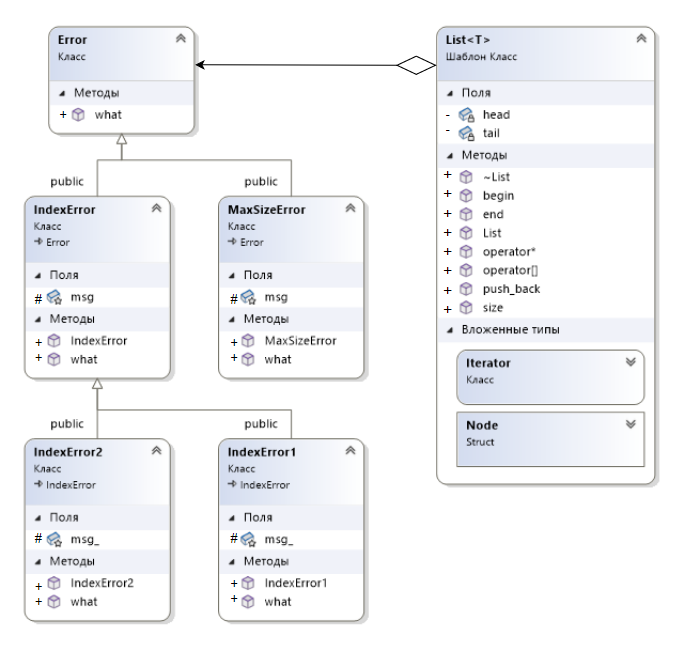
****

*Приложение 7 – Главная функция (Вариант реализации 3)*

**UML-Диаграмма**



*Приложение 8 – UML-Диаграмма (Вариант 1)*

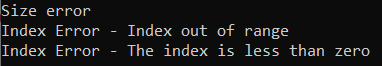


*Приложение 8 – UML-Диаграмма (Вариант 3)*

**Работа программы**



*Приложение 9 – Работа программы (Вариант реализации 1)*

**

*Приложение 10 – Работа программы (Вариант реализации 3)*

**Ответы на контрольные вопросы**

1. Исключение в С++ представляет собой специальный объект, который генерируется в процессе выполнения программы при возникновении ошибки или иной исключительной ситуации.

2. Исключения позволяют разделить вычислительный процесс на две части: блок кода, который может вызвать исключение, и блок кода, который обрабатывает это исключение. Достоинством такого подхода является возможность более точной обработки ошибок и упрощение кода за счет выноса обработки ошибок в отдельные блоки.

3. Для генерации исключительной ситуации используется оператор throw.

4. Контролируемый блок представляет собой участок кода, в котором может произойти исключение. Для того чтобы контролировать и обрабатывать это исключение, необходимо обернуть этот блок кода в try-блок.

5. Секция-ловушка представляет собой блок кода, который обрабатывает исключение, сгенерированное в try-блоке. Она нужна для того, чтобы определить, какие действия необходимо выполнить при возникновении исключения.

6. Спецификация исключения может иметь форму типа данных или множества типов данных, которые могут быть сгенерированы в try-блоке. В ситуациях, когда необходимо обрабатывать несколько типов исключений, используется множественная форма спецификации. В ситуациях, когда необходимо обработать любое исключение, используется форма без указания типа данных.

7. Стандартный класс, который можно использовать для создания собственной иерархии исключений, называется std::exception.

8. Для создания собственной иерархии исключений необходимо создать класс, наследующийся от класса std::exception или его производных классов, и определить в нем необходимые методы и поля.

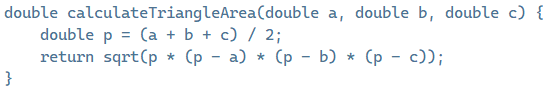
9. Функция f1() может принимать исключения типа int и double.

10. Функция f1() не ожидает никаких исключений.

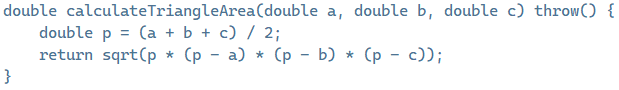
11. Исключение может генерироваться в любой части программы, но обычно это происходит в тех участках кода, где есть потенциальная возможность возникновения ошибок или исключительных ситуаций. Например, это могут быть операции с файлами, работа с сетью, выделение памяти и т.д.

12.

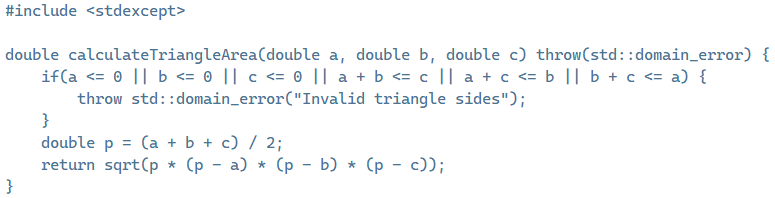
Без спецификации исключений:



Со спецификацией throw():



С конкретной спецификацией с подходящим стандартным исключением:



Спецификация с собственным реализованным исключением:

