**Обработка исключений**

**1. Пример работы с файлами с помощью базового класса exception**

**- что такое Обработчка исключений?**

**1.2 какие типы мы можем бросать с помощью throw**

**1.3 Обработка с помощью специализированного дочернего от**

**exception класса ifstream::failure для получения более детальной инфы об проблеме**

**1.4 когда используется Несколько блоков catch?**

**- что значит catch (…) ?**

**2. Свой класс exception c++**

**- что такое what()? Где и для чего применяется.**

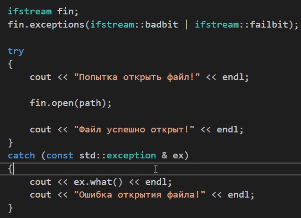
**- в чем суть полиморфной обработки исключений.**

**- Что будет Если catch-тчи равноценны(принимают один и тот же тип данных )?**

**Каков их порядок?**

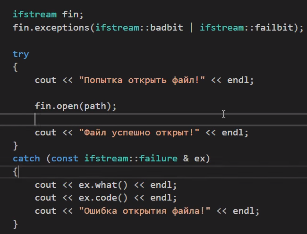
**Обработчка исключений** – обработка исключительных ситуаций

**1.2 Пример работы с файлами с помощью базового класса exception**

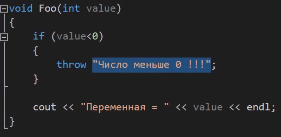


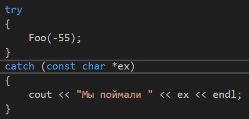
 - вкл. Обработку искл для работы с файлом

**1.3 работка с помощью специализированного дочернего от *exception* класса *ifstream::failure* для получения более детальной инфы об проблеме**

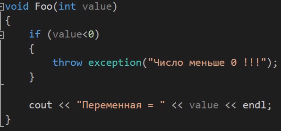


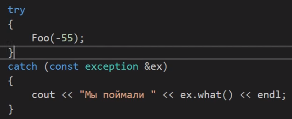
В **throw** мы можем передать любой тип



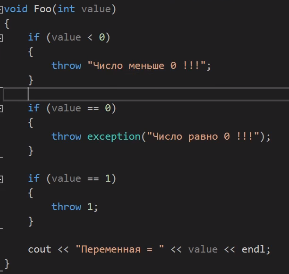


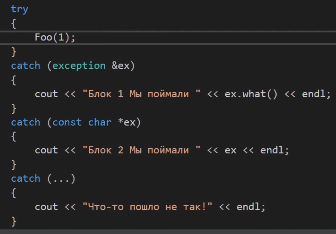
Но обычно в таких ситуациях бросают class exeption() т. К. обработка искл. Будет более универсальной





**1.4 Несколько блоков catch**





catch (…) – всегда в конце иначе будет error

# 2 Свой класс exception c++

# У стандартного *exeption* есть только метод *what()* поэтому класс ограничен

# Даже в такой виде класс можно ловить и кидать

# 

# 

# ---

# Т.к. класс унаследован, то *catch* с типом *exeption* обработает искл.

# 

# 

# ---

# 

# 

# Main()

# 

# Если *catch*-тчи равноценны, то есть по сути принимают один и тот же тип данных то отработает тот *catch* который будет идти первым, поэтому узко специализированные *catch*-тчи ставим впереди.