Институт информационных технологий

Кафедра: Математическое и программное обеспечение ЭВМ

Дисциплина: Объектно-ориентированное программирование

ЛАБОРАТОРНАЯ РАБОТА № 8

Тема: Программирование обработки исключительных ситуаций

Выполнил:

студент гр. 1ПИб-02-2оп-23

Кринкин Олег Алексеевич

Проверил:

ассистент Матевосян Ремик Артурович

ЗАДАНИЕ

1. Ознакомьтесь с УМП по ООП часть 2 раздел X.
2. Запустите несколько исключений в одном из АТД, при нарушении логики программы. Например, передача некорректных данных, отсутствие данные, несоветские данных определенному типу и т.д.
3. Правильно обработайте все возможные исключительные ситуации.
4. Проверьте работоспособность АТД на тестовом наборе данных.

**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Дайте определение исключения.
2. В каком блоке описывается список обработчиков для возбужденного исключения?
3. Опишите синтаксис обработчика исключения.
4. Что такое спецификация исключения?
5. Для каких целей используется функция unexpected?

ХОД РАБОТЫ

1. Тип поля clients в классе Switch был изменён с std::array на std::vector, чтобы в объекты Switch можно было вносить произвольное количество объектов.
2. В конструкторах класса Switch установлены проверки на количество введённых подключаемых объектов и если оно превышает допустимое – будет вызвано исключение:

if (clients.size() > cli\_cap) throw std::overflow\_error("clients vector is too big");

1. В контейнерном классе в методе геттере, возвращающем объекты в виде массива установлена проверка на ненулевой размер:

if (size == 0) throw std::length\_error("Pointers array size is zero");

1. В тестировании создан новый тест – Exceptions. В нем производится проверка описанных выше исключений.
2. В первом тесте создаётся объект Switch и в его конструктор передаётся вектор клиентов размером 40, хотя максимально возможный размер установлен как 32.

Assert::ExpectException<std::overflow\_error>([] {

std::vector<int> clients;

clients.resize(40);

Switch sw = Switch(clients);

});

1. Во втором тесте создаётся объект контейнерного класса, после чего тут же вызывается геттер для получения массива указателей на объекты, хотя ни одного элемента ещё добавлено не было.

Assert::ExpectException<std::length\_error>([] {

ServerRoom<Router> sr\_r = ServerRoom<Router>();

const Router\*\* arr = sr\_r.get\_array();

});

1. Assert::ExpectException представляет собой слой абстракции для блока try catch. Тест можно было реализовать и классическим образом и выглядело бы это следующим образом:

try {

ServerRoom<Router> sr\_r = ServerRoom<Router>();

const Router\*\* arr = sr\_r.get\_array();

}

catch (std::length\_error) {

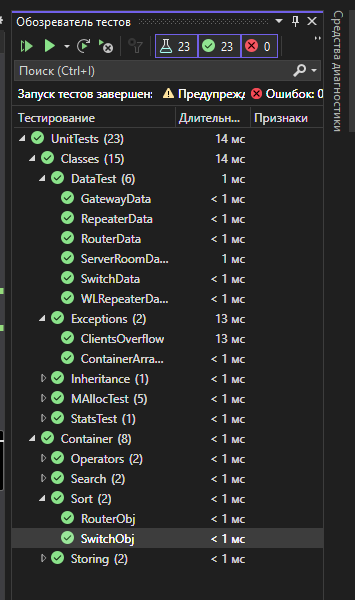
return;

}

Assert::Fail();

РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ РАБОТЫ

Результат исполнения тестирования



Как можно заметить, тест сортировки после смены типа поля clients стал выполняться успешно. Это произошло потому, что array::size() возвращал 32 – весь размер массива, а vector::size() из-за своего динамического размера возвращает фактическое количество элементов в нём.

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были освоены навыки создания иерархии классов с использованием множественного наследования.

Ответы на контрольные вопросы:

1. Дайте определение исключения.

Исключение – события, которые возникают во время выполнения программы и прерывают нормальный ход ее выполнения.

1. В каком блоке описывается список обработчиков для возбужденного исключения?

catch(исключение) {обработчики}

1. Опишите синтаксис обработчика исключения.

try {код} catch(исключение) {обработка}

1. Что такое спецификация исключения?

Спецификация исключения – указывает может ли функция вызывать исключения. Работу с функцией с таким блоком в заголовке необходимо обернуть в блок try catch, обрабатывающий известное исключение.

1. Для каких целей используется функция unexpected?

std::unexpected используется для обработки неожиданных исключений