Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Системное программирование

Студент: Лопатнюк П.В.

ФИТ 3 курс 1 группа

Преподаватель: Бернацкий П.В.

Минск 2024

**Лабораторная работа №6**

**Structured Exception Handling**

**Цель работы**: Получение практических навыков в работе с механизмом структурной обработки исключений.

**Постановка задачи:**

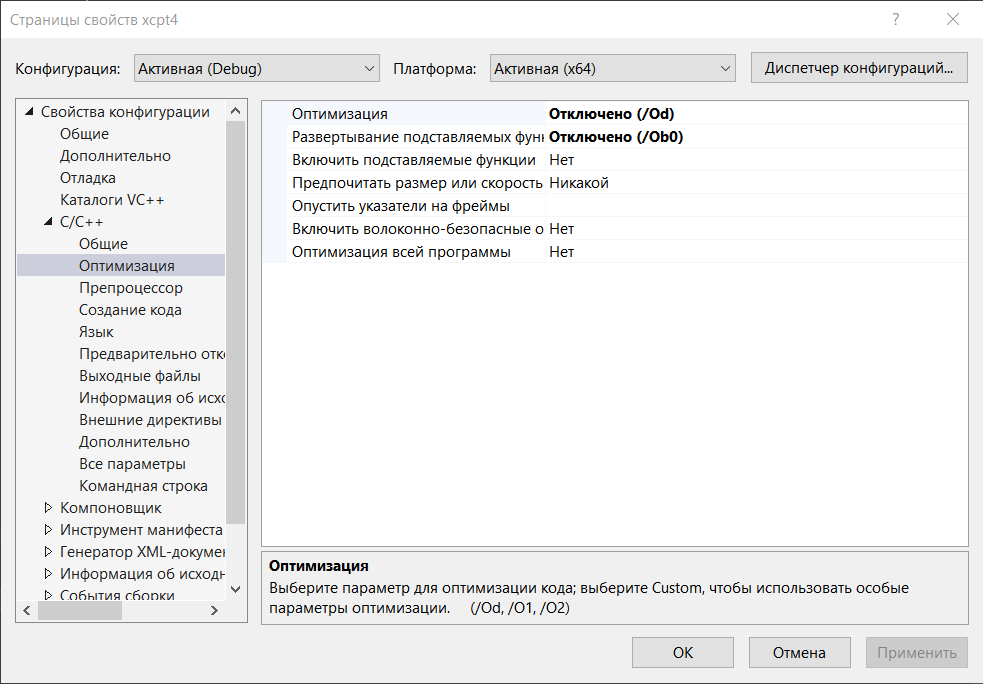
Взять готовый проект с сайта «diskstation.belstu.by:5001» по пути «Для\_студентов\_ФИТ\_БГТУ/ПРЕПОДАВАТЕЛИ/Бернацкий/Системное программирование - 3 курс ИСиТ/Лекции/Примеры для лекций/8/windows\_seh\_tests-main.zip».

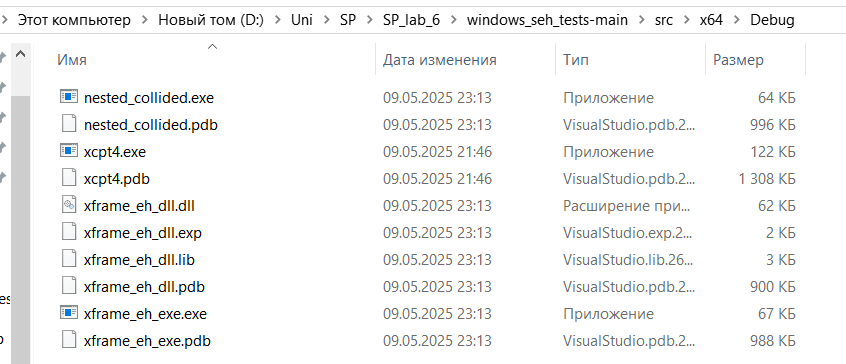
Изучить проект и полностью разобраться со структурой и содержимым подпроектов. Собрать проект согласно README файлу. Собрать проект необходимо как через Visual Studio, так и напрямую в консоли с использованием компилятора MSVC (cl.exe).

Подготовиться к объяснению любого из представленных в xcpt4 тестов.

**Сборка проекта через Visual Studio**

README требует компилировать с опцией /Ob0 (запрет встраивания функций). В Visual Studio это делается через свойства проекта:

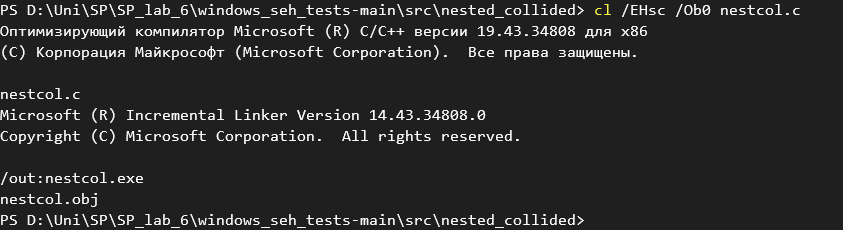


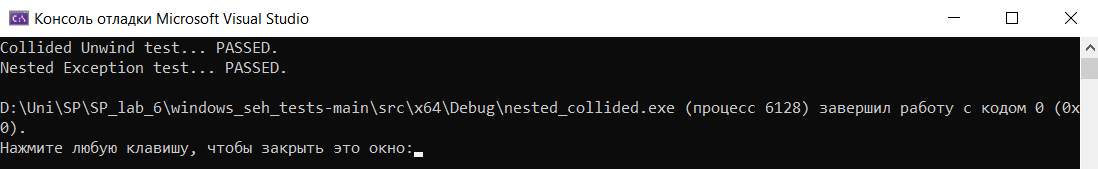


**Сборка проекта через консоль (cl.exe)**

**NESTED\_COLLIDED:**

cl /EHsc /Ob0 nestcol.c



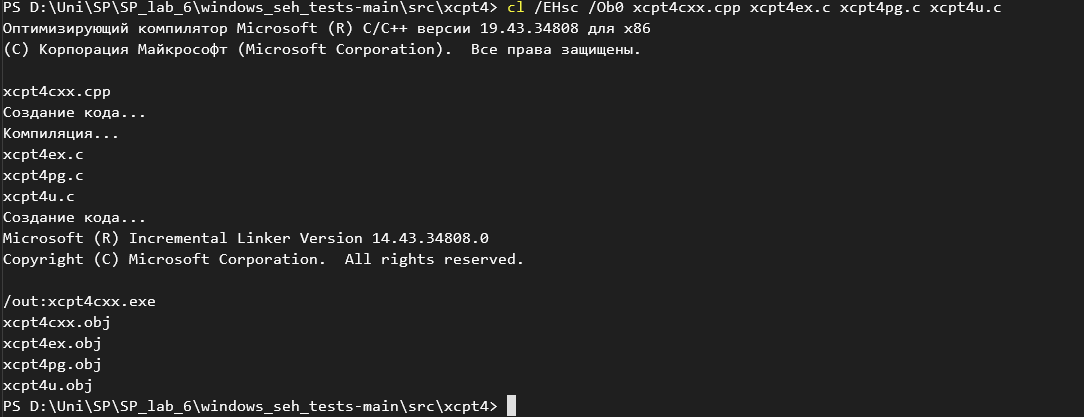


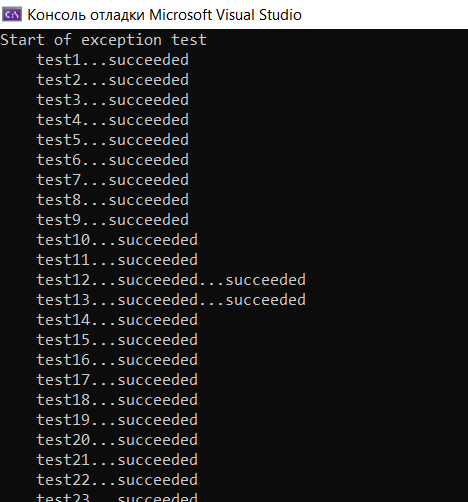
/EHsc: Стандартная обработка исключений (не обязательно для чистого SEH, но хорошая практика для C++ кода).

/Ob0: Отключить любое встраивание функций.

**XCPT4:**

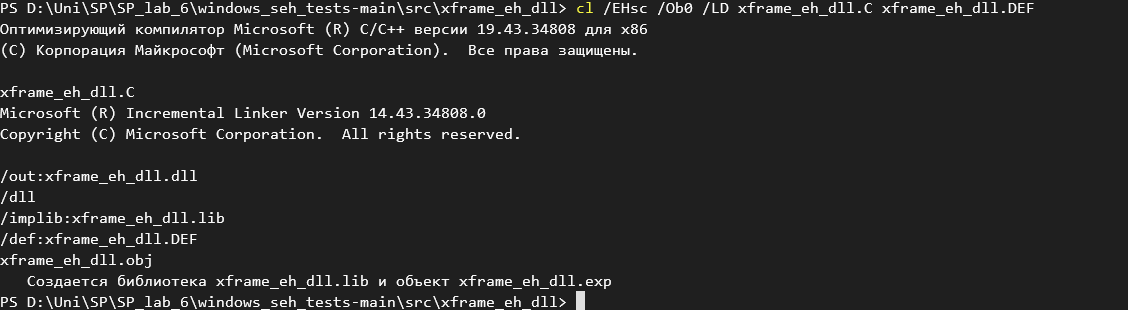
cl /EHsc /Ob0 xcpt4cxx.cpp xcpt4ex.c xcpt4pg.c xcpt4u.c



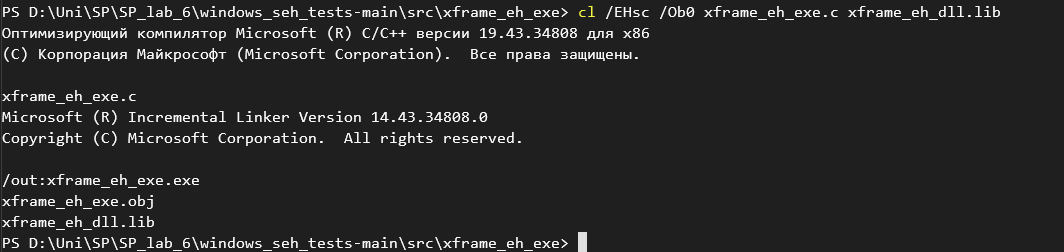


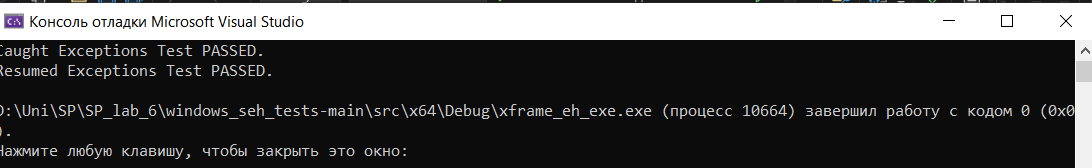
**XFRAME\_TEST (DLL и EXE):**

**Команда для сборки DLL:** cl /EHsc /Ob0 /LD xframe\_eh\_dll.C xframe\_eh\_dll.DEF



**Команда для сборки EXE:** cl /EHsc /Ob0 xframe\_eh\_exe.c xframe\_eh\_dll.lib





**Вопросы для контроля:**

1. **Что такое исключение?**

**Исключение** – это событие, возникающее из-за выполнения определенной команды, которая вызвала ошибку процессора. В результате такого события нормальное выполнение программы становится невозможным. Исключение является синхронным и технически воспроизводимым при тех же условиях. Примеры включают деление на ноль, ошибку доступа к памяти и другие. Исключения бывают аппаратными (возбуждаются процессором) и программными (возбуждаются ОС или приложениями).

1. **Что такое SEH?**

**SEH** (Structured Exception Handling - Структурная обработка исключений) – это механизм операционной системы Windows, который позволяет перехватывать исключения и дает коду возможность их обработать, если это возможно. Он доступен как для кода пользовательского режима, так и для кода режима ядра. Полная поддержка SEH присутствует только в компиляторе MSVC.

1. **Что такое обработка завершений?**

**Обработка завершений** (termination handling) – это одна из двух основных возможностей SEH, предоставляемая ключевым словом \_\_finally. Обработчик завершения (\_\_finally) гарантирует, что блок кода (собственно обработчик) будет выполнен независимо от того, как происходит выход из защищенного блока (\_\_try) – будь то нормальное завершение, преждевременный выход (return, goto и т.п.) или возникновение исключения.

1. **Что такое обработка исключений?**

**Обработка исключений** (exception handling) – это вторая основная возможность SEH, предоставляемая ключевым словом \_\_except с фильтром. Она предназначена для перехвата и реагирования на произошедшие исключения. При возникновении исключения система ищет соответствующий блок \_\_except, проверяет его фильтр и в зависимости от значения фильтра выполняет код обработчика исключений или продолжает поиск.

1. **Что такое защищенный блок кода?**

**Защищенный или охраняемый блок кода** – это блок кода, ограниченный фигурными скобками оператора \_\_try. Предполагается, что в этом блоке может возникнуть исключение, которое следует обработать (с помощью \_\_except) или для которого требуется гарантированное выполнение кода очистки (\_\_finally).

1. **Для чего нужно ключевое слово \_\_leave?**

Ключевое слово \_\_leave используется в блоке \_\_try (в связке с \_\_finally). Оно предоставляет оптимизированный механизм для перехода к блоку \_\_finally из любой точки внутри блока \_\_try. В отличие от операторов return, goto, break, continue, которые вызывают дорогую локальную раскрутку при выходе из \_\_try в \_\_finally, \_\_leave вызывает переход в конец \_\_try естественным образом, минимизируя издержки и позволяя гарантированно выполнить \_\_finally.

1. **Что такое локальная и глобальная раскрутки?**

**Локальная раскрутка** (local unwind) – это процесс, который происходит при преждевременном выходе из блока \_\_try (например, из-за return, goto, break, continue) в блок \_\_finally, следующий за этим \_\_try. Система и компилятор генерируют дополнительный код для обеспечения выполнения \_\_finally и освобождения локальных объектов в стеке текущего блока. Это относительно дорогостоящая операция.

**Глобальная раскрутка** (global unwind) – это процесс, который происходит при обработке исключения (когда фильтр \_\_except возвращает EXCEPTION\_EXECUTE\_HANDLER). Система выполняет глобальную раскрутку стека от места возникновения исключения до места расположения обработчика исключений. В ходе глобальной раскрутки гарантированно выполняются все незавершенные блоки \_\_finally на пути вверх по стеку вызовов.

1. **В чём главное отличие SEH от С++ exception?**

Главное отличие в контексте программирования заключается в их природе и интеграции с языком. SEH – это механизм операционной системы Windows, который может быть использован из разных языков. Исключения C++ – это часть самого языка C++, и их обработка тесно интегрирована с семантикой языка. В частности, при обработке C++ исключений компилятор C++ автоматически генерирует код для вызова деструкторов объектов (механизм RAII - Resource Acquisition Is Initialization) при раскрутке стека, обеспечивая корректную очистку ресурсов. SEH сам по себе не предоставляет такой автоматической очистки для C++ объектов, хотя компилятор MSVC реализует C++ исключения на основе SEH и использует \_\_finally для реализации вызова деструкторов.