**Лабораторная работа №6**

**Тема работы:** генерация и обработка исключений, исключения, методы обработки и генерации исключений, разработка собственных классов обработки исключений, файлы, использование исключительных ситуаций для поддержки инкапсуляции и бизнес-логики, отладка исключений в Visual Studio.

**Цель работы**:

– изучить использование исключительных ситуаций в бизнес-логике,

– поработать с файловым вводом-выводом в С++,

– поработать с исключениями для поддержки инкапсуляции,

– изучить окно Stack Trace в Visual Studio.

**Теоретические сведения** находятся в материалах лекций 14-19.

**Порядок выполнения работы**

1. Изучить краткие теоретические сведения.
2. Ознакомиться с материалами литературных источников.
3. Ответить на контрольные вопросы.
4. Разработать алгоритм программы.
5. Написать, отладить и выполнить программу.

**Варианты заданий**

Задания упорядочены по сложности. Каждое задание выдается одному человеку.

1. Создать текстовый файл с символьным изображением ракеты. Пусть приложение выводит содержимое этого файла бесконечное количество раз с какой-то задержкой (подпрограмма sleep в Windows.h). Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Контакт, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

2. Создать приложение для чтения книги. Книга – файл в текстовом виде. Задача приложения – показывать книгу по страницам с задержкой: показать 20 строчек, подождать несколько секунд (подпрограмма sleep в Windows.h) и показывать следующие 20 строчек и т.д. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Строка, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

3. Имеется некоторый файл с числами, полученными с некоторого устройства. Например: 13 666 33. Задача приложения – рассчитать матожидание по данным из файла. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Студент, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

4. Сформировать арифметическую прогрессию. Задача приложения – записать члены арифметической прогрессии в файл в формате: номер итерации - число. Например:

1 4

2 8

3 16. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Абитуриент, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

5. Разработать приложение, которое сохранит все аргументы командной строки в файл: каждый аргумент с новой строки. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс Рейс с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

6. Разработать приложение, которое сохранит в файл некоторое количество членов ряда Тейлора: каждое число с новой строки. Пусть число членов n и число x приложение спрашивает у пользователя. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Книга, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

7. Разработать утилиту для необратимого удаления файла. Пусть имя файла приходит как аргумент командной строки (или приложение его спрашивает у пользователя). Для уничтожения файлов пусть используется метод Брюса Шнейдера. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Рабочий, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

8. Разработать утилиту для необратимого удаления файла. Пусть имя файла приходит как аргумент командной строки (или приложение его спрашивает у пользователя). Пусть файл удаляется методом Министерства обороны DoD 5220.22-M, вариант E. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Билет. Разработать класс с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

9. Разработать утилиту для шифрования текстового файла методом XOR. Число, с которым нужно xor-ить, имена входного и выходного файла пусть вводятся с клавиатуры. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс Продукт с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

10. Разработать утилиту для копирования файла по блокам. Пусть утилите на вход как аргументы командной строки приходят имена входного, выходного файла, размер буфера, а после копирования утилита выводит время, которое заняло копирование. Определить с помощью утилиты, при каком размере буфера копирование большого файла на флэшку происходит быстрее. Использовать исключения для ошибок в бизнес-логике.

Пример измерения времени:

#include <ctime>

void f() {

using namespace std;

clock\_t begin = clock();

code\_to\_time();

clock\_t end = clock();

double elapsed\_secs = double(end - begin) / CLOCKS\_PER\_SEC;

}

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Пациент, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

11. Разработать утилиту для оценки труда блоггера. Некоторые работодатели оценивают текст по числу строчек. Пусть на вход приходит имя каталога, в котором утилита ищет все текстовые файлы, считает суммарное количество строк текста во всех файлах, умножает полученное число на цену за строку и выводит результат на экран. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Маршрут автобуса, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

12. Разработать утилиту для поиска музыки. Пусть на вход подается имя каталога и имя результирующего файла-отчета, утилита ищет в каталоге и в подкаталогах все музыкальные файлы, список найденных файлов сохраняет в файл-отчет. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Адрес, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

13. Разработать утилиту для сравнения файлов по содержимому. Пусть на вход утилиты приходят как аргументы командной строки имена двух файлов. Пусть приложение выводит на экран, идентичны ли файлы или нет. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Концерт, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

14. Разработать приложение с классом для чтения настроек из .ini файла. Пусть аргументами командной строки будут имя файла с настройками, название настройки. Пусть приложение выведет значение настройки. Например, имеется .ini файл

[General]

ScreenResolution.Width = 400

ScreenResolution.Height = 500

Допустим, нужно прочесть настройку ScreenResolution.Width.

Приложение тогда пусть выведет 400. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Письмо, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

15. Разработать утилиту, которая преобразует текст в текст на азбуке Морзе. Пусть на вход приходит имя входного и выходного файла. Если будет время, можете сделать, чтобы еще и на динамик звуки выводились (beep в Windows.h). Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Запись рейтинговой системы, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

16. Разработать утилиту для шифрования текста. Пусть на вход приходят имена входного и выходного текстовых файлов. Использовать исключения для ошибок в бизнес-логике. Суть шифрования следующая: каждую букву исходного текста менять на следующую за ней, которая идет по алфавиту. А букву я менять на а. Например, «агггггг» зашифруется в «бддддд».

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Занятие в университете. Разработать класс с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

17. Разработать утилиту для необратимого удаления файла. Пусть на вход утилиты приходит имя файла как аргумент командной строки (или приложение его спрашивает). Пусть файл удаляется методом уничтожения файлов Гутмана. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Тренировка, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

18. Разработать утилиту Автоцензор, которая заменяет «нехорошие» слова на иксы в исходном файле и сохраняет результат в итоговый файл. Пусть на вход приходят имена исходного и итогового файлов. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Торрент, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

19. Разработать программу для поиска текста в файле. Пусть утилита выводит, подходит ли файл под критерии поиска или нет. Аргументы командной строки: имя файла, слова для поиска. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Программа, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

20. Разработать утилиту по проверке рассекреченных документов. Имеется каталог с рассекреченными документами. Приложение должно проверить, не встречается ли в каких-либо файлах имя секретного агента. Пусть на вход приложение получает имя каталога, а также слово, которое должно отсутствовать. А на выходе выведет список документов, где это слово встретилось. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Микронутриент. Разработать класс с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

21. Разработать утилиту по конвертации CSV файла в HTML файл. Утилита получает на вход имя входного CSV файла, имя выходного HTML файла. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Покупка, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

22. Разработать класс для чтения CSV файлов и продемонстрировать его использование. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Услуга, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

23. Разработать класс для чтения числовых данных из файла и продемонстрировать его использование. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Товар, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

24. Разработать класс для записи CSV файла и продемонстрировать его использование. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Браузер. Разработать класс с полями, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

25. Разработать класс для записи Html файлов и продемонстрировать его использование. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Ноутбук, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

26. Разработать утилиту для подсчета распределения символов в процентах в заданном текстовом файле. На вход подается файл с текстом классика. Пример вывода:

А – 12,44%

Б – 33,22%

…

Проверить, совпадают ли распределения на книгах разных писателей. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Документ, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

27. Разработать утилиту для проверки XML файла. Пусть на вход приходит имя файла. Пусть приложение выведет, корректный ли это XML файл. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Страна, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

28. Разработать утилиту для шифрования текста с помощью видео-файла. Пусть на вход приходят имена исходного файла, видео-файла, выходного файла, начальная позиция в видео-файле. Суть шифрования: XOR каждого байта входного файла с каждым символом видео-файла (начиная с заданной позиции), затем байт сохранить в выходной файл. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Персонаж, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.

29. Разработать утилиту для шифрования текстового файла шифром Цезаря. Пусть на вход приходят имена входного и выходного файлов, ключ-число. Суть шифрования в том чтобы увеличивать код каждого символа на заданное число. Использовать исключения для ошибок в бизнес-логике.

Продемонстрировать использование окна Stack Trace среды разработки при возникновении исключения.

Разработать класс с полями Файл, добавить геттеры и сеттеры, добавить в класс проверку данных, приходящих извне, через исключения.