№ 6 Исключения

# Задание

Дополнить предыдущую лабораторную работу № 5. 1) Создайте иерархию классов исключений (собственных) – 3 типа и более. Сделайте наследование пользовательских типов исключений от стандартных классов .Net (например, Exception, IndexOutofRange).

<https://learn.microsoft.com/en-us/dotnet/api/system.exception?view=net-6.0>

1. Смоделируйте и обработайте как минимум пять различных исключительных ситуаций на основе своих и стандартных исключений. Например, не позволять при инициализации объектов передавать неверные данные, обрабатывать ошибки при работе с памятью и ошибки работы с файлами, деление на ноль, неверный индекс, нулевой указатель и т. д.
2. В конце поставьте универсальный обработчик catch.
3. Используйте классический вид **try-catch-finally**.
4. Продемонстрируйте возможность многоразовой обработки одного исключения и проброс его выше по стеку вызовов.
5. Обработку исключений вынести в main. При обработке выводить специфическую информацию о месте, диагностику и причине исключения. Последним должен быть блок, который отлавливает все исключения (finally).
6. Добавьте код в одной из функций макрос Assert. Объясните что он проверяет, как будет выполняться программа в случае не выполнения условия. Объясните назначение Assert.
7. Ознакомьтесь с классами Debug и Debugger:

<https://learn.microsoft.com/en-us/dotnet/api/system.diagnostics.debugger?view=net-6.0>

[https://docs.microsoft.com/en-us/dotnet/api/system.diagnostics.debug?view=net-6.0](https://docs.microsoft.com/en-us/dotnet/api/system.diagnostics.debug?view=netframework-4.8)

Пример:

int[] aa= null;

Debug.Assert(aa !=null, "Values array cannot be null");

Не забудьте подключить using System.Diagnostics;

# Дополнительное задание

1. Создайте класс **Logger**, который будет заниматься логгированием различных событий и исключений. Логгер должен уметь логгировать ошибки/исключения, предупреждения и просто какую-то информацию.
2. Логгер должен записывать лог в виде: время, тип\_записи\_лога:

дополнительное сообщение. 27.10.2019 02:36, INFO: Test log message.

1. Создайте 2 реализации логгера: **FileLogger и ConsoleLogger**. **FileLogger** будет записывать сообщения лога в файл, добавляя записи к уже существующим. **ConsoleLogger** – выводить сообщения на консоль.
2. Добавьте в классы из л.р. 6 логгер так, чтобы его возможно было быстро заменить во время выполнения другим и вместо простого вывода на консоль сообщения об ошибке, используйте свой логгер.

# Вопросы

1. Расскажите как генерируется исключение.
2. Расскажите методику обработки исключений.
3. Какое ключевое слово служит для обозначения блока кода, в котором можно генерировать исключение?
4. Какие ключевые слова используются для обработки и генерации исключений? Расскажите об механизме обработке исключения?
5. Что будет, если в программе нет предложения catch, способного обработать исключение?
6. Что такое фильтры исключения? Приведите пример
7. Могут ли исключения быть вложенными?
8. Какой синтаксис нужно использовать в C# для отлова любого возможного исключения?
9. Чем следует руководствоваться при размещении обработчиков исключения?
10. Что будет выведено на консоль в результате выполнения фрагмента листинга?

static void Main(string[] args)

{

string[] str = new string[5];

try { str[4] = "anything";

Console.WriteLine("It's OK");

}

catch (IndexOutOfRangeException e)

{

Console.WriteLine("IndexOutOfRangeException");

}

catch (Exception e)

{

Console.WriteLine("Exception");

}

}

**}**

1. Как повторно сгенерировать то же самое исключение в блоке обработчике catch?
2. .Какие методы содержаться в классе Exception? Где и как их можно использовать?