**Лабораторная работа 2**

(3 занятия)

**Цели**

1. Познакомиться с классами потоков (streams) и вспомогательными классами для чтения из потоков и записи в них.

2. Реализовать класс или классы потоков или вспомогательные классы в соответствии с заданием.

**Ход работы**

1. Продумать необходимые типы и их интерфейсы.

2. Составить UML диаграмму этих типов. Включить в нее те классы из .NET Framework, от которых наследуют пользовательские типы.

3. Реализовать необходимые типы.

4. Составить набор тестов для реализованных классов. Модульные тесты приветствуются.

5. Подготовить отчет о лабораторной работе (электронный вариант).

В отчет включить:

* цели;
* UML диаграмму с пояснениями (в особенности, по интерфейсам и использованию базовых типов);
* код реализованных типов.

**Задания**

*1. BackupStream*

Содержит внутри два потока (streams) - главный и резервный (main & backup). Оба потока файловые (FileStream). Запись и перемещение (seeking) по обоим потокам осуществляются синхронно. Чтение - только из главного потока, резервный только синхронизируется с помощью перемещения. Имя резервного файла формировать из имени главного добавлением к расширению ".bak". Один из перегруженных конструкторов должен принимать в качестве параметра директорию резервного файла. Для простоты ограничиться работой только с новыми файлами, т.е. не открывать существующие.

\* Реализовать переключение операций на резервный поток при отказе главного.

\*\* Вместо реализации обертки на двумя файловыми потоками унаследоваться от FileStream и обернуть только один поток - резервный.

*2. SpyStream<T>*

Наследует от MemoryStream, содержит в себе поток указанного типа T. Копирует в себя все, что записывается в оборачиваемый поток. Синхронизация при чтении и перемещении. Для простоты ограничиться работой только с новыми потоками.

Должен быть метод получения потока с копией данных, например GetSpyStream().

*3. PagingStream*

Наследует от Stream. Гарантирует, что его экземпляр никогда не будет помещен в Large Object Heap.

*4. BeepingWriter и BeepingReader*

а) BeepingWriter должен иметь метод WriteBeep(Beep object), записывающий в файловый поток переданный объект и производящий с помощью Console.Beep(...) соответствующий звуковой сигнал.

б) BeepingReader должен иметь метод Beep ReadBeep(), считывающий из файла (файлового потока) объект типа Beep и при этом производящий с помощью Console.Beep(...) соответствующий звуковой сигнал. Также нужен метод, считывающий файл до конца. Возвращаемое значение этого метода - список объектов типа Beep.

Beep - должен содержать продолжительность (duration) сигнала и его частоту (frequency).

*5. SnowdenStreamWriter*

Класс, расшифровывающий и помещающий в отдельный файл все зашифрованные данные, которые с его помощью записываются. Алгоритм шифрования, местоположение и имя файла с расшифрованными данными выбрать по желанию.

*6. UrlEncodingWriter*

StreamWriter (UTF-8), который выполняет кодирование URL при записи.

*7. HtmlContentsWriter*

StreamWriter (UTF-8), который кодирует специальные символы HTML при записи.

*8. CsvWriter и CsvReader*

а) CsvWriter должен иметь методы Write(DataTable dt) и Write(DataRow dr). Данные из DataTable (DataRow) записываются в поток в виде значений, разделенных запятой (comma-separated values, csv).

б) CsvReader должен иметь методы DataRow ReadRow() и DataTable ReadTable().

Для простоты работать только с числовыми типами данных. Полученный .csv файл должен читаться с помощью MS Excel.

Первый ряд должен содержать заголовки столбцов.

Рекомендуется перед реализацией сохранить таблицу Excel в формат CSV и тщательно изучить результат, в дальнейшем пользоваться им как эталоном.

\* Добавить поддержку строковых данных, строки не должны содержать двойные кавычки (") и запятые.

\*\* Добавить поддержку строковых данных, содержащих двойные кавычки и запятые.

*\* 9. TypeWriter*

Должен содержать метод WriteDeclaration(Object instance). При вызове анализирует тип переданного значения и с помощью Reflection формирует .cs файл с описанием этого типа. Допускается пропускать using директивы. При добавлении необходимых using файл должен компилироваться.