1 FKlub Stream

Many years ago, key f-club scientists created the official linux kernel module used by most f-club members, *F-Kernel Modul*.[1]

There is, however, no FStream in the .NET class library. Your job is to create a class called FStream with similar functionality. Listing 1 shows how to create a filestream and how read lines using a streamreader.

```
using FileStream s = File.OpenRead("lines.txt");
using StreamReader sr = new StreamReader(s);

string line;
while (null != (line = sr.ReadLine()))
{
    Console.WriteLine(line);
}
```

Listing 1: StreamReader example

FileStream is a subclass of Stream, and StreamReader is parameterized with the stream to read. ReadLine reads characters from the underlying stream until it meets a newline-character in the stream.

If the file |lines.txt| containes the lines:

```
    ff
    ffff
    ffffffff
    ffffffffffffffff
```

those would be printed to the screen.

Assingment: Create the class FStream as a subclass of System.IO.Stream, and let it produce an infinite number of f's only interrupted by newlines (otherwise readline would fail).

Do not worry too much about the using statement above. Using *using* ensures appropriate closing of the stream. This is not too important in a program of the above size.

The old F Kernel Module was created by a team of the most brilliant minds of the F-Club; working with f's is not trivial. You are encouraged to work as a team!

[1]More info here:https://www.fklub.dk/diverse/fkernelmodul

1.1 Background

The Stream-class is the basic abstraction for streams in .NET.

The official documentation for the abstract class Stream writes[2]:

> *Stream is the abstract base class of all streams. A stream is an abstraction of a sequence of bytes, such as a file, an input/output device, an inter-process communication pipe, or a TCP/IP socket. The Stream class and its derived classes provide a generic view of these different types of input and output, and isolate the programmer from the specific details of the operating system and the underlying devices.*
>
> *Streams involve three fundamental operations:*
>
> *You can read from streams. Reading is the transfer of data from a stream into a data structure, such as an array of bytes.*
>
> *You can write to streams. Writing is the transfer of data from a data structure into a stream.*
>
> *Streams can support seeking. Seeking refers to querying and modifying the current position within a stream. Seek capability depends on the kind of backing store a stream has. For example, network streams have no unified concept of a current position, and therefore typically do not support seeking.*
>
> *Some of the more commonly used streams that inherit from Stream are FileStream, and MemoryStream.*
>
> *Depending on the underlying data source or repository, streams might support only some of these capabilities. You can query a stream for its capabilities by using the CanRead, CanWrite, and CanSeek properties of the Stream class.*
>
> *The Read and Write methods read and write data in a variety of formats. For streams that support seeking, use the Seek and SetLength methods and the Position and Length properties to query and modify the current position and length of a stream.*

[2]https://docs.microsoft.com/en-us/dotnet/api/system.io.stream?view=net-5.0