# Exercise 2

Implement the LZW compression algorithm as follows:

1. The alphabet is 8-bit. It is formed by characters with ASCII codes from 1 to 255. There are no clear codes or stop codes. At the beginning, the dictionary is initialized with characters with ASCII codes from 1 to 255.
2. When the dictionary is full, it is not updated any more.
3. The Least Significant Bit First packing order is used.

The program should be organized in the following way:

1. The class `InputBitStream`, declared in the file InputBitStream.h, provides an interface for working with input bit streams.
2. The class `OutputBitStream`, declared in the file OutputBitStream.h, provides an interface for working with output bit streams.
3. The class `FileInputBitStream`, declared in the file FileInputBitStream.h and implemented in the file FileInputBitStream.cpp, implements an input bit stream, which reads data from a file.
4. The class `FileOutputBitStream`, declared in the file FileOutputBitStream.h and implemented in the file FileOutputBitStream.cpp, implements an output bit stream, which writes data to a file.
5. The function `void Encode()`, declared in the file LZW.h and implemented in the file LZW.cpp, performs encoding of the input stream according to the LZW algorithm and write the output to the output stream.
6. The function `void Decode()`, declared in the file LZW.h and implemented in the file LZW.cpp, performs decoding of the input stream according to the LZW algorithm and write the output to the output stream.
7. The program should compress the file original.txt to the file encoded.dat and then decompress the file encoded.dat to the file decoded.txt. (If the program is correct, the files original.txt and decoded.txt are the same.) The code length should be taken as a command line parameter and passed to the functions `Encode()` and `Decode()`.

References:

http://en.wikipedia.org/wiki/Lempel%E2%80%93Ziv%E2%80%93Welch