We would like you to write a decompressor for LZW, a lossless compression algorithm. LZW is relatively simple compression algorithm which was the default compression algorithm on UNIX for a long time and is still used in GIF images and some PDFs.

Wikipedia provides a good description of the algorithm, which you should read. You'll discover that there are a few variations to the LZW algorithm. We are interested in the fixed width variation and we have written a compressor that uses this algorithm. The key details are summarised below:

1. We used 12-bit fixed with codes. Packed such that if the encoder was to output the codes 72 followed by 500 it would write out the 3 bytes 00000100 10000001 11110100.
2. If the file being compressed results in an odd number of codes then the last code is padded to 16-bits e.g. if the last code is 400 then the encoder would output 00000001 10010000
3. The dictionary is initialised with 256 entries, which map all the possible values for a byte to a string of length 1 containing a byte of that value. 0 -> "\x00", 1 -> "\x01" ... 65 -> "A", 66 -> "B" etc.
4. When all the possible codes have been used, the dictionary is reset to the initial dictionary described above and new entries are added as normal.

The attached zip file contains 4 example compressed files for you to try your decoder on. They all decompress to human readable text. There are no magic numbers or headers on the file so they will not be compatible with the standard UNIX tools.

We'd prefer to see the solution in Golang, but if you're more comfortable in C or C++ feel free to use one of those languages.

As well as looking for a submission that works, we are interested in the quality of the code particularly:

- Readability of the code. The code should be clear and easily understandable, even with few comments.
- The structure of the code. For example would it be possible to use the decoder as a library with in a larger application.
- Resources usage. You are not required to optimize the code, but we are looking for a implementation which could be used with very large files.

Good Luck!