We would like you to write a decompressor for LZW, a lossless compression algorithm. The original algorithm is quite straightforward and uses a fixed-length code (it is a refined version using variable-length codes that is used as the compression for GIF images).

For this task we are only interested in the fixed-length version, described in detail on Wikipedia.

To test your decompressor we have provided a ZIP file containing 4 examples that all decompress to human readable text. These examples were compressed using an LZW compressor we wrote and they will not decompress using standard UNIX tools because they lack the required headers.

Our compressor was implemented based on the Wikipedia page. The key details are also summarised below:

1. We used 12-bit fixed codes
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 output would be 00000001 10010000
3. The dictionary is initialised with all strings of length one
   e.g.

1-byte string	Code
0x00	0000 00000000
...	...
0x41 (A)	0000 01000001
0x42 (B)	0000 01000010
...	...
0xFF	0000 11111111

4. When the dictionary is full, it is reset to the initial dictionary described in step 3 and new entries are added as normal.

Please complete the task using C or C++.

As a final thought, how could you improve a naive compressor that repeatedly loops over the entire dictionary to find the longest string that matches the input?

Thank you and good luck!