# INF144 Mandatory 2 – CGJ008

## How I have solved the task

### Lempel-Ziw-Welch compression

* Initialize
  1. Populate the encoding dictionary and the decoding dictionary with the characters of the given alphabet
  2. Find the initial binary code length (block size) needed to represent a text of the given alphabet (e.g. to represent a text made up of an alphabet with 29 distinct characters, we need 5 bits)
  3. Set current block size equal to initial block size
* Compress
  1. Find the first subset of characters in the text which is not in the encoding dictionary
  2. Add this subset to the dictionary at index equal to the size of the dictionary
  3. Add the dictionary index of this subset, minus the last character, to the compression output (as binary, number of bits equal to current block size)
  4. If the dictionary size is now strictly greater than , increase block size
  5. Find the next subset, which is not in the dictionary, starting with the last character of the previous subset
  6. Repeat from 2. until all the text is compressed
* Decompress
  1. Set current block size equal to initial block size
  2. Read the first block of length equal to current block size, from the compressed source
  3. Add its translation from the decode dictionary to the output
  4. Read the next block of length equal to current block size
  5. If it has a mapping in the decode dictionary, add its translation to the output
  6. Else, add the translation of the previous block +