A quick overview of the Indexer code by Lucas Ferreira

The review was made for the **decodeFile( )** method of the **Encoder** class as the simplest and most obvious one.

The method used the following approach - a two-dimensional matrix (table) was compiled in which each word or suffix from the **mappingFile** was matched with a corresponding code:

cipherMatrix[i][0] – word/suffix
cipherMatrix[i][1] – corresponding code

The following approach was used for decoding:
- if the code was **0**, it was replaced with **[???]**
- otherwise, a full table search was performed and when the corresponding code was found, we extracted the token
- then we checked the token
    - if it is a suffix, we remove the previous space and the first two characters "**@@**" and add to the result
    - if it is a punctuation mark, we remove the previous space and add to the result
    - if it is not a suffix or a punctuation mark, we simply add the word to the result

Thus, when searching for each token, it was necessary to sort through the table with codes. As a result, the complexity of the algorithm is **$O(n^2)$**

This method can be greatly simplified if you get the token directly using the predefined **token-index** == **token-code** mapping:

var currentKey = cipherMatrix[c][0];

Thus, the complexity of the algorithm will be reduced to **O(n)**, which will significantly affect its performance.

To handle an error related to an incorrect code or its format, we will wrap the code iteration cycle in a **try – catch** block.

The resulting code can be viewed in the fork of the original repository at the link:
https://github.com/artem-totality/Indexer-little-review