Partner1:
Name: Jiaqi Fan
PID: A12584051
Partner2:
Name: Xuanru Zhu
PID: A91064234
Date: 2/15/2017
PA3

Refactor Report

Introduction:

Code refactor is the process of restructuring existing computer code - changing the factoring-without changing its external behavior. Refactoring improves nonfunctional attributes of the software. The purpose of this process is to let third party people who look at you code and they could easily understand the code better without digging into the code and look up the library usage inside your code. For example, better design of your code make if else statements easily to look. Sometimes when I look at my own code it is easily to get lost when there are several if else statement in the function and I could mess up the "{ }" for which if which else. Also the importance of handling magic number. Sometimes meaningless number appear inside our code and it could take long time for me to figure out what is that number stand for. Helper method is really good thing to reduce the length of each method and make the code easier to read.

Changes We Made:

In the function DictionaryTrie::insert(std::string word, unsigned int freq). We looked at the length of the method there are three if else statements inside that made the method quite long however, it is difficult to reduce the length by using helper method because all the check condition is necessary. However, we could make the if else statement easily to read by label the if else statements and use newline after each if else statement. This will not affect the functionality of the code but helpful for the style of the code. We alse added more comments to explain the code more detail. We also label the if else statement to make the code easier to read. Another change in this method is that we found out that we have too much if else statements and we could reduce the number of if else statement by combining the if else together and make into a big if else statement.

In the function DictionaryTrie::find(std::string word) const. We find out that we have very less comment in this method which is really hard for other people to read and understand what are we doing in the code. Comment should be like explaining code to grandma, and able to let them understand then that considered as good comment. We also label the if else statement to make the code easier to read. Another change in this method is that we found out that we have too much if else statements and we could reduce the number of if else statement by combining the if else together and make into a big if else statement.

All other methods that we wrote are fine. For example, the predictComplete method we used several helper methods to reduce the length of the body. We have eSearch method and autoComplete method. Both methods are not very long and logic are clear with all the comment clearly explained what the code is doing. The reason we Designed this way is the logic of the code is more clearout. One thing that we changed is there are two lines that are over 80 characters so we reduce the length in one line and we put them on the second line. Also we combined the two separate for loops into one because the logic are similiar.

Conclusion:

The goal for all the changes that we made are for bettering coding style and make it easier for people to read. I think when third party people looked at our code it is very clear that our through are represented in the code.