Assignment 1: Sentence Segmentation

**Method of designing sentence segmentation program:**

While reading input file, if line is blank, for example blank line after paragraph, then it has not been considered in data string. For eliminating this blank line re.search function and \S (one non-whitespace character) has been used. Search function takes \S and line as input and returns true if line has at least one non-whitespace character. If it returns true then only that line will be concated to data string.

Sentence segmentation is done using regular expression. Regular expression has been developed using positive lookbehind and negative lookbehind. If it finds a one whitespace it will check as explained below:

**(?<!\w\.\w\.)** checks whether whitespace precede by words like (**i.e.**)**.** If yes then it is not end of sentence. It is negative lookbehind.

**(?<!\w\.\w\.\w\.\w\.)** checks if whitespace is precede by words like (**U.S.I.A.**) in sentence. If yes then it is not end of sentence.

**(?<![A-Z]\.)** checks if whitespace is precede by words like (**J.**). If yes then it is not end of sentence.

**(?<!MR.|Mr.|\.\.\.)** checks if whitespace is precede by words like (**Mr.**) or(**Mr.**) or(**…**). If yes then it is not end of sentence.

**(?<!\;\"|\,\")** this checks for quotation. So if quotation mark is preceded by (**,**) or (**;**) then it is not end of sentence.

**(?<=\.|\?|\]|\")** this checks for positive lookbehind. So if whitespace is preceded by (**.**) or (**?**) or (**]**) or (**“**) then it is end of sentence.

Split function splits data into sentences. For sentence number, value of i has been incremented.

**What program covers and what are limitations of program?**

Program covers abbreviations such as G.N.P., O.A.S. It doesn’t split sentences on those abbreviations when they are used in the middle of sentence. Program correctly split sentences having “fanatical… leaders”. Program works for Mr. and MR. But program gives error when abbreviation comes at the end of the sentence. That time it does not split sentence after abbreviation, for example line number 168 in output.txt file.

But when compared with NLTK, it is seen that program is consistent in dealing with MR. and Mr., while NLTK sometimes has given wrong sentence breaking on those words (false positive).

Since program has hard coding of abbreviation of Mr. and MR., it will not work for Dr. and will break sentence (false positive). This is another limitation of code.

Further, program is unable to break sentence after (*II.*) If we try to add that to positive lookbehind, it gives compile time error “*lookbehind assertion has to be fixed width.*”

Another limitation of program is it is giving false positive on the quotes that come in middle of sentence, for example: *This is good example of “false negative” isn’t it?* Here since space is precede by (“) which is not precede by either (,) or (;), by regular expression, it gives two sentences in output “This is good example of *“false negative””* and *“isn’t it?”.* But program works very well on other quotes.

**How could I have improved program if given more time?**

For improvement, I could have added more abbreviations (two characters followed by (.) for example, BE.) in the regex. Also, rather than hard coding them I could have come up with regex pattern which will match those abbreviations.

Further, sentence splitting over quotations could have been improved. I could have added checking for preceding character whether if it is alphabet or not. So if it is alphabet, we can conclude that is not end of sentence and this is quote in the middle of sentence.