```
 1 # step 1 : Loading and reading the Data using os.path
 2 import os
 3 import sys
 4 with open(os.path.join(sys.path[0], "ID1(reading and
   its  correlates2014).txt"), "r") as f:
 5     f.close()
 6
 7
 8 # step 2 :Split by Whitespace and preserve the
   punctuations.
 9 file=open("ID1(reading and its  correlates2014).txt"
   , "r")
10 text=file.read()
11 words=text.split()
12 print(words[:])
13 #punctuation is preserved (e.g. “wasn’t” and “armour-
   like“),
14 # which is nice. We can also see that end of
15 #sentence punctuation is kept with the
16 # last word (e.g. “thought.”)
17
18
19 # Step3 :Split by Whitespace and obliterate the
   punctuations.
20 #want the words, but without the punctuation like
   commas
21 #and quotes. We also want to keep contractions
   together.
22 #load the text file, split it into words by white
   space,
23 #then translate each word to remove the punctuation.
24 import string
25 table= str.maketrans("","", string.punctuation)
26 stripped = [w.translate(table) for w in words]
27 print(stripped[:])
28
29 #printing huge number of punctuations in python
30 print(string.punctuation)
31
32
33
34 #Normalizing Case
35 words=[word.lower() for word in words]
36 print(words[:])
```

```
37
38 words=[word.capitalize() for word in words]
39 print(words[:])
40
41
42
43 print("An end to this task; lets go for NLTK")
44
45
46
47 # Remember, the simpler, the better.
48 # Simpler text data, simpler models, smaller
49 # vocabularies. You can always make things
50 # more complex later to see if it results in
51 # better model skill
52 # we'll look at some of the tools in the NLTK (a
   Python library
53 # written for working and modeling text.)
54 # library that offer more than simple string
   splitting.
55 # NLTK = Natural Language Tool-Kit
56 import nltk
57 #nltk.download()
58 filename = 'ID1(reading and its  correlates2014).txt'
59 file= open(filename, "r")
60 text= file.read()
61 print(text)
62 file.close()
63 #NLTK provides the sent_tokenize() function to split
64 #text into sentences.
65 from nltk.tokenize import sent_tokenize
66 sentences = sent_tokenize(text)
67 print(sentences[5])
68 total_sentences=len(sentences)
69 print(total_sentences)
70 for sentence in range (total_sentences):
71     print("Sentence ",sentence, "is",": ",sentences[
   sentence])
72 #NLTK provides a function called word_tokenize()
73 #for splitting strings into tokens (nominally words).
74 from nltk.tokenize import word_tokenize
75 words = word_tokenize(text)
76 print(words[51])
77 total_words=len(words)
```

```
 78 print(total_words)
 79 for word in range(total_words):
 80     print("word ",word,"is",words[word])
 81
 82 #Our next step is to filtering out the punctuation.
 83 filename="ID1(reading and its  correlates2014).txt"
 84 file=open("ID1(reading and its  correlates2014).txt"
    ,"r")
 85 text= file.read()
 86 from nltk.tokenize import word_tokenize
 87 words = word_tokenize(text)
 88 only_words = [word for word in words if word.isalpha
    ()]
 89 print(only_words[:])
 90
 91
 92 #how to remove stopwords?
 93 print("the task tries to eliminate the words
    extracted "
 94       "here called stop words")
 95 from nltk.corpus import stopwords
 96 stop_words=stopwords.words("english")
 97 print(stop_words)
 98 #lets compare your tokens to the stop words and
    filter them out
 99 print("lets creat a corpus with no stop words")
100 words=[word for word in words if not word in
    stop_words]
101 print(words[:])
102
103
104 #lets analyize for stem of the corpus with no stop
    words.
105 file_name = "ID1(reading and its  correlates2014).
    txt"
106 file= open("ID1(reading and its  correlates2014).txt
    ","rt")
107 text=file.read()
108 from nltk.tokenize import word_tokenize
109 tokens= word_tokenize(text)
110 total_token=len(tokens)
111 print(total_token)
112 for words in range (total_token):
113     print("The word number ", words, " is ", tokens[
```

```
113 words])
114
115 from nltk.stem.porter import PorterStemmer
116 porter= PorterStemmer()
117 stemmed=[porter.stem(word) for word in tokens]
118 print(stemmed)
119 total_stem=len(stemmed)
120 print(total_stem)
121 for item in range (total_stem) :
122     print("The stem, number ", item, " is ",stemmed[
    item])
123
124
125 # Extracting Method (lets you take a code fragment
126 # that can be grouped, move it into a separated
127 # method, and replace the old code with a call to
    the method.
128 # When you extract the method you need to check for
129 # variables. If there is one output variable, it
130 # is used as a return value for the extracted
131 # method. In case there are multiple output
132 # variables, the Extract Method refactoring may
133 # not be applied, and the error message appears.
134
135
136
137
```