Programming Assignment : 01

Aditya Ganesh Khedekar (Matrikel-Nr: 245974)
Sarvesh Kulkarni (Matrikel-Nr: 245891)
Siddhi Belgamwar (Matrikel-Nr: 246167)
Amita Rohidas Kashid (Matrikel-Nr: 248642)

Text:
Today is sunny. She is a sunny girl. To be or not to be. She is in Berlin today. Sunny Berlin! Berlin is always exciting!

Tasks:

a) Standard & Whitespace Tokenizer
b) Standard Tokenizer and (custom) Stopword Filter
c) Custom Analyzer [Standard Tokenizer; Lowercase, Stopword, and PotterStemmer Filter]

Task (a):

To perform this task, given input text was passed to 2 functions performing whitespace tokenization and standard tokenization respectively. An object of whitespace analyzer and standard analyzer are created (provided by lucene 7.4) which uses the input string to be tokenized. Each object performs its task of tokenizing (whitespace – tokenizes based on whitespace; standard – tokenizes based on grammar following the rules from Unicode standard annex 29) and returns a stream of tokens which are then converted to string and printed on the console.

After performing the given task, it is observed that where whitespace tokenizer created tokens solely based on whitespace, the standard tokenizer behaved differently. It not only changed all the characters to lower alphabets but also removed all the punctuations. It also got rid of stopwords like “to”, “is”, “a”, “or”, “be”, “in”.

Task (b):
For part B, we created **stopWord** method to remove stopwords from the text, like "was," "is," "in," "to," and "be”. A **CharArraySet** is created from the list of stopwords which is used to identify and remove stopwords during the tokenization process. A **StandardAnalyzer** instance is generated, with the stopSet(list of stopwords) being used as an input parameter. We create empty list **result** to store the final tokens after the stopword elimination process. **TokenStream** named stream is created using the provided **analyzer** and the input text.The method iterates through the tokens in the "TokenStream," adding them to the "result" list after removing stopwords.

Task (c):

1. **Setting the Stopword File Path**

- First of all, for this task we created a *stopwords.txt* file and listed all the given stopwords in it.
- Then we get the file path of this file in our program.

2. **Custom Analyzer Creation and Configuration**
   - In next step, we built a Custom Analyser with the following processing step respectively:
     Standard Tokenizer → Lowercase Filter → Custom Stopwords Filter → Porter Stemmer
3. **Tokenization and Token Processing**
   - At last, we tokenize the given string/text with help of the built custom analyser and print each token.

Output:
Programming Assignment 01:

Task 'a':

Tokens generated by Standard Tokenizer:
today
sunny
she
sunny
girl
she
berlin
today
sunny
berlin
berlin
always
exciting

Tokens generated by Whitespace Tokenizer:
Today
is
sunny.
She
is
a
sunny
girl.
To
be
or

not
to
be.
She
is
in
Berlin
today.
Sunny
Berlin!
Berlin
is
always
exciting!

Task 'b':

Tokens generated by Standard Tokenizer:
today
sunny
she
sunny
girl
she
berlin
today
sunny
berlin
berlin
always
exciting

Tokens generated by own Stopwords Filter:
today
sunny
she
a
sunny
girl
or
not
she
berlin
today

sunny
berlin
berlin
always
exciting

Task 'c':

Tokens generated by Custom Analyzer:
todai
sunni
she
a
sunni
girl
or
not
she
berlin
todai
sunni
berlin
berlin
alwai
excit