

## PROBLEM SET 1

### PART – 1:

- In the first part of the assignment, this program reads a line one by one using `BufferedReader` and then scans that each of the lines. Then it converts into the number of tokens of that line with the help of `split()`.
- With the help of `Regex`, tokens are scattered into the words, special characters and count those words and characters and store them into the List.
- At the end of the file, our whole List will contain all the words. This List can be transferred to the `SortedList` so that we will get unique words. Meanwhile, this program also takes care for the special characters, and again store them into one another List and convert them into the `SortedList`, plus, it also counts the number of blank lines appeared at each and every line.
- Finally, it just prints all the `SortedList` and shows the special characters.
- For Example :

```

C:\Windows\system32\cmd.exe

E:\Darpan Spring 2013\Algorithm\Assignment_1>javac Ngrams.java
Note: Ngrams.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

E:\Darpan Spring 2013\Algorithm\Assignment_1>java Ngrams
----- Part 1 -----
There were 812794 words.
There were 35950 blank lines.
There are [", !, &, ', .., ., 3, 2, 1, 7, 6, 5, 4, ;, :, 9, 8, ?, ], [] Punctuations
----- Part 2 -----

E:\Darpan Spring 2013\Algorithm\Assignment_1>

```

### PART -2:

- In the second part of the assignment, first of all this program converts each of the line into the number of tokens and store them into the List. After that if there are special characters appear or the word is in the Upper Case, then it filters the special characters and convert them into the Lower Case with the help of `String.toLowerCase()` method.
- This program uses the Map Data Structure to count the number of grams. Each and every line, it updates its value, this Map is like `Map<String,Integer>` , in which its key is tokens and its values are the number of repetitions.
- Once the whole file has been finished, we get the Map of whole tokens. For n-grams this program passes this map to the function “`ngrams()`” as a parameter and this `ngrams` returns the List of n-grams.
- With the help of `Comparable` Interface, it has used the `comapreTo()` method to sort them according to the values of Map. And it gives the expected output.
- For Example :

```
CA: C:\Windows\system32\cmd.exe
----- Part 2 -----
For 1-grams -----
There are UNIQUE : 31125 grams
the - [ 23318 ]
I - [ 20041 ]
and - [ 16856 ]
to - [ 15534 ]
of - [ 15043 ]
you - [ 12383 ]
a - [ 12316 ]
my - [ 10711 ]
in - [ 9483 ]
is - [ 8331 ]
For 2-grams -----
There are UNIQUE : 293547 grams
I am - [ 1844 ]
I have - [ 1612 ]
I will - [ 1567 ]
in the - [ 1495 ]
to the - [ 1373 ]
of the - [ 1322 ]
my lord - [ 1221 ]
to be - [ 828 ]
I do - [ 823 ]
it is - [ 722 ]
For 3-grams -----
There are UNIQUE : 519338 grams
I pray you - [ 246 ]
I will not - [ 213 ]
I know not - [ 162 ]
I do not - [ 160 ]
I am a - [ 141 ]
I am not - [ 137 ]
I would not - [ 125 ]
my good lord - [ 113 ]
I have a - [ 97 ]
I will be - [ 96 ]
For 4-grams -----
There are UNIQUE : 520147 grams
I know not what - [ 38 ]
I do beseech you - [ 31 ]
I do not know - [ 29 ]
I would not have - [ 26 ]
Give me thy hand - [ 25 ]
Ay my good lord - [ 25 ]
with all my heart - [ 25 ]
I can tell you - [ 23 ]
What is the matter - [ 23 ]
With all my heart - [ 22 ]
E:\Darpan Spring 2013\Algorithm\Assignment_1>
```

Normally, This program takes 2-3 seconds to finish the part 1. And for the part 2 it takes probably around 10-15 seconds to complete the process.