Data Structures and Algorithms

HOMEWORK 6 REPORT

RUNNING COMMANDS AND RESULTS

```
Original String: 'Hush, hush!' whispered the rushing wind.
Preprocessed String: hush hush whispered the rushing wind

The original (unsorted) map:
Letter: h - Count: 7 - Words: [hush, hush, hush, hush, whispered, the, rushing]
Letter: u - Count: 3 - Words: [hush, hush, rushing]
Letter: s - Count: 4 - Words: [hush, hush, whispered, rushing]
Letter: w - Count: 2 - Words: [whispered, wind]
Letter: i - Count: 3 - Words: [whispered, rushing, wind]
Letter: p - Count: 1 - Words: [whispered]
Letter: e - Count: 3 - Words: [whispered, whispered, the]
Letter: r - Count: 2 - Words: [whispered, rushing]
Letter: d - Count: 2 - Words: [whispered, wind]
Letter: t - Count: 1 - Words: [the]
Letter: n - Count: 2 - Words: [rushing, wind]
Letter: g - Count: 1 - Words: [rushing]


The sorted map:
Letter: p - Count: 1 - Words: [whispered]
Letter: t - Count: 1 - Words: [the]
Letter: g - Count: 1 - Words: [rushing]
Letter: w - Count: 2 - Words: [whispered, wind]
Letter: r - Count: 2 - Words: [whispered, rushing]
Letter: d - Count: 2 - Words: [whispered, wind]
Letter: n - Count: 2 - Words: [rushing, wind]
Letter: u - Count: 3 - Words: [hush, hush, rushing]
Letter: i - Count: 3 - Words: [whispered, rushing, wind]
Letter: e - Count: 3 - Words: [whispered, whispered, the]
Letter: s - Count: 4 - Words: [hush, hush, whispered, rushing]
Letter: h - Count: 7 - Words: [hush, hush, hush, hush, whispered, the, rushing]
```

IMPLEMANTATION OF HOMEWORK

info class: This class for holding and counting words. It takes split words from myMap class. Then it storages these words in words array. While this storage process also counts the words. Fro printing process it has toString method to print words and its counts.

myMap class: This class for implementing map structure. It takes preprocessed sentence and splits to words form. For key structure it splits again the words to letters. It sorts the letters in the map structure according to the order in which they were read. Also puts the

values from the info class in their place on the map. After this process prints the unsorted map.

mergeSort class: This class for sorting myMap according to merge algorithm. Transfers the count and word values contained in the map to the info array. Puts the keys in the map into the aux arraylist for sorting. After these operations, it puts both structures into the sort function and divides them into two parts, right and left. It sorts the separated parts according to the merge algorithm recursively. After the count values are sorted, the aux values with the same number are also sorted in the order in which they were read. Finally the two sorted structures are placed in the SortedMap and printed.

TestClass: Preprocess the given sentence for the map structure. Then prints unsorted sorted maps.