**Lab 5\_3**

**SpellCheck Application**

For the SpellCheck application, various Collection classes can be used to store the dictionary of words – the words are read in from ‘words.txt’. It counts the number of misspelt words found in the text you are spell checking (alice30.txt ). A larger text file war-and-peace.txt is also given.

Use IntelliJ Profiler to generate % of time and actual time (in ms) for contains() method of your chosen Collection class – code as given uses a LinkedList.

Complete the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Structure used to store dictionary** | **% of Time for contains() method** | **Time in ms for contains() method** | **Big Oh for contains() method**  **n – size of the dictionary** |
| LinkedList | 95 | 6,920 | O(n) |
| Collection | 96 | 7,465 |  |
| Array List | 92 | 3,546 | O(n) |
| TreeSet | 55 | 187 | O(log n) |
| HashSet | 62 | 233 | O(1) |

Obtained with \_\_ Intel Core i7\_\_\_ processor, \_\_\_2.8\_\_GHz, Java Version \_22.0.2\_\_, Windows 11 (or specify if not…)

Try different Collection classes and see the different values you will get for the contains() method. Use the larger text file if the smaller file give values that are too small.

What Collection class would you recommend for the SpellCheck application?

\_\_HashSet\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Explain your answer:

The contains() method operates in O(1) time on average, making it highly efficient for checking the existence of words in the dictionary. The performance metrics indicate that it has the lowest time in milliseconds for the contains() method compared to other data structures. HashSet is straightforward to implement and manage, making it suitable for a SpellCheck application where quick lookups are essential.