A picture containing logo

Description automatically generated

King Fahd University of petroleum and minerals

Ics202 PROJECT

[Dictionary data structure]

MOHAMMED ALMUBARAK – 202024880

HUSSAIN ALKHALAIF – 202035120

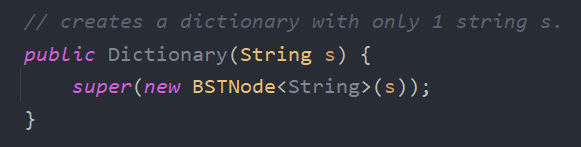
The dictionary holds a list of words (strings) to be used in a spell checker.

**1- Initialization**

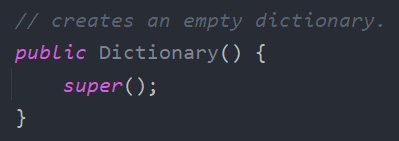
First of all, we create a dictionary class that extends AVLTree<String> class, then we initialize the dictionary using three methods:

1. a single string, [public Dictionary(String s)]
2. an empty dictionary [public Dictionary( )]
3. a text file having strings, each on a new line. [public Dictionary(File f)]

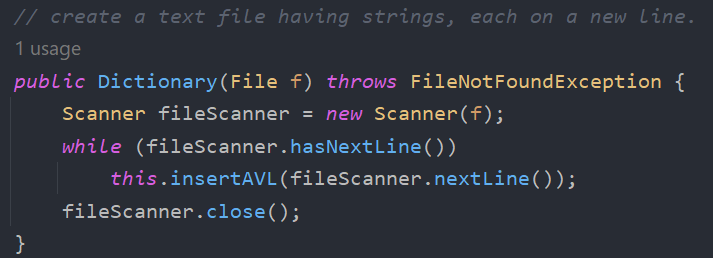
in the first method, we create a dictionary with only one string which is will be the root in the avl tree.



in the second method, we create an empty dictionary.



In the third method, we are reading the text file and insert each line to the AVL tree, and the method should throw an exception if the file does not exist.



**2- Add new word**

This method adds a new word to the dictionary, it searches for the given word, if it does not exist, then it will add it to the tree, if it exists, it should throw an exception.

Text

Description automatically generated

For search, we are using search method from BST class, to insert, we use insertAVL from AVLTree class.

**3- Search for word**

This method of type Boolean takes one string, it searches for the given word, if it’s found, then will return true, if it not found, return false.

For search, we are using search method from BST class.

Text

Description automatically generated

**4- Remove word**

This method takes one string to delete a word from the dictionary, it searches for the given word, if it’s found, it will delete the word from the AVL tree, if it not found, then it should throw an exception.

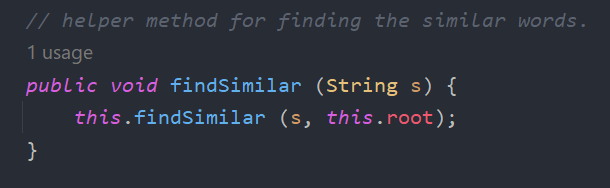
For search, we are using search method from BST class, to delete, we use deleteAVL from AVLTree class.

Text

Description automatically generated

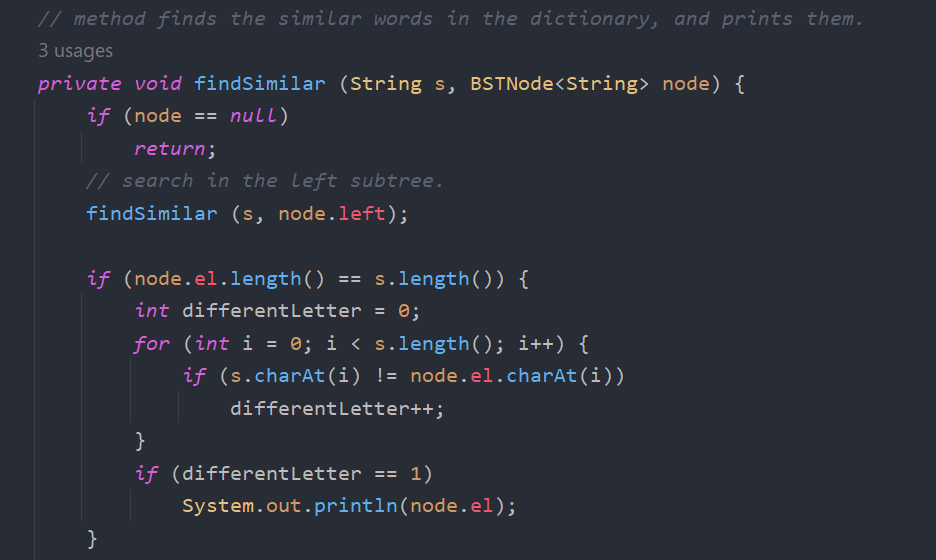
**5- Find similar word:**

This method takes one string to find the similar words from the dictionary, it searches for the words that have the same length or differ by one letter in the length, if it’s found, it should be printed.



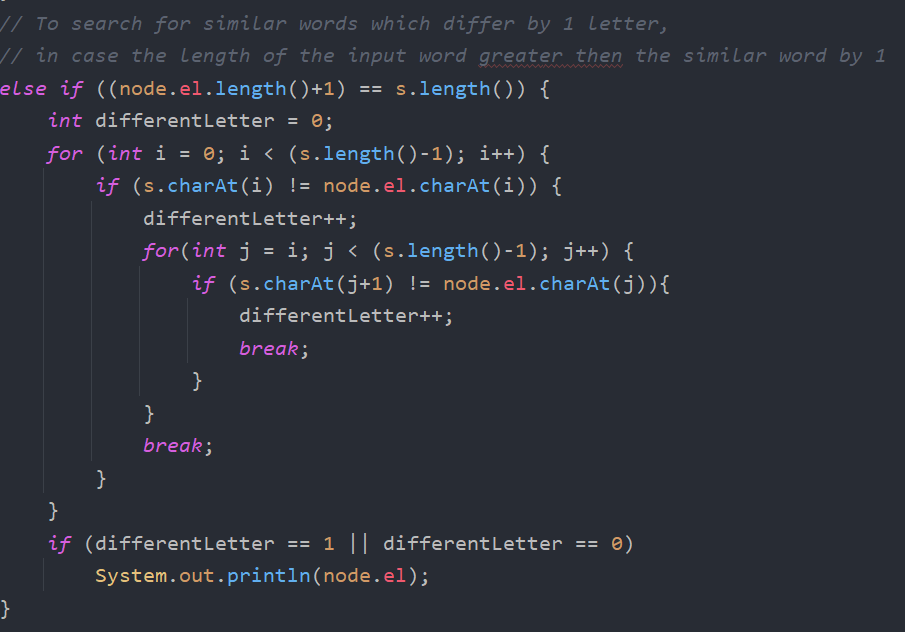
Helper method to call the recursive method with the root to start searching.

**findSimilar method: Part 1**



In this part we call the method with moving to left, and we check if the input word length is equal to the word which we found, then if its equal, then we initialize a variable called differentLetter to count how many letter the two words are different from each other, we check for each character, if we finish and we get differentLetter = 1, then print the founded word.

**findSimilar method: Part 2**



This part if the previous condition (same length) not true, then check if the input word length greater than the founded word by 1, then we initialize a variable called differentLetter to count how many letter the two words are different from each other, we check for each character, if we get a different character, then we skip that character for the input word, then compare each character, if we finish and we get differentLetter = 1 or = 0 (which means there is no difference or the different character is the last character in the input word), then print the founded word.

**findSimilar method: Part 3**

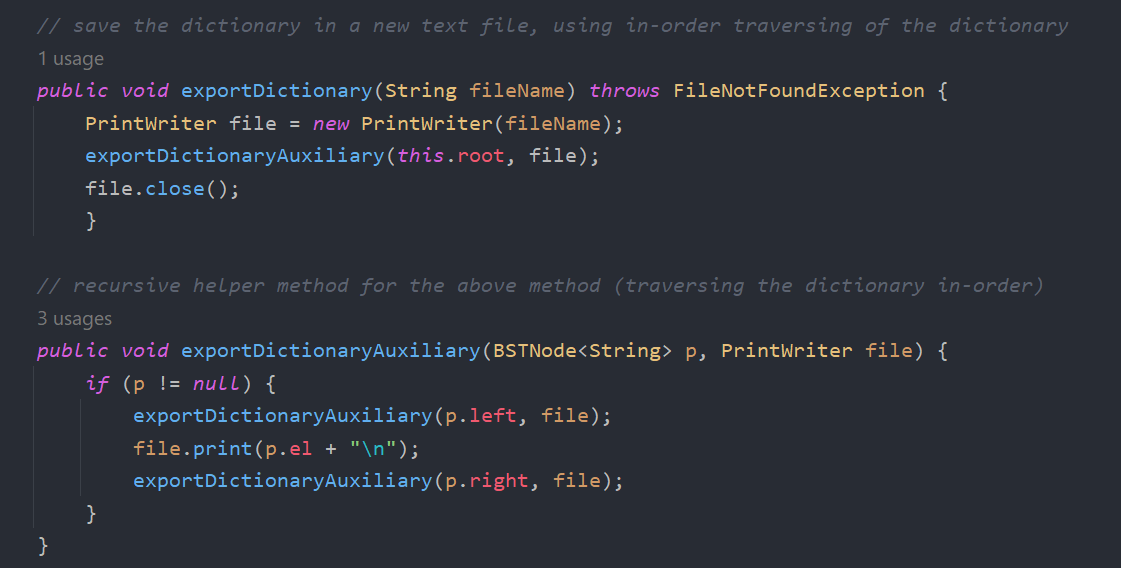


This part if the previous condition (input word’s length grater by 1) not true, then check if the founded word length greater than the input word by 1, then we initialize a variable called differentLetter to count how many letter the two words are different from each other, we check for each character, if we get a different character, then we skip that character for the founded word, then compare each character, if we finish and we get differentLetter = 1 or = 0 (which means there is no difference or the different character is the last character in the founded word), then print the founded word.

Then search in the right subtree.

**6- Save the updated dictionary:**

After adding or removing words from the dictionary, we save it as a new txt file.



**All the exceptions:**

* **WordAlreadyExistsException:**

Text

Description automatically generated

* **WordNotFoundException:**

Graphical user interface, text

Description automatically generated

* **Catches in the test class:**

Text

Description automatically generated

**Test class:**

Text

Description automatically generated

Text

Description automatically generated

1. **Initialization**

* **A single string.**

Text

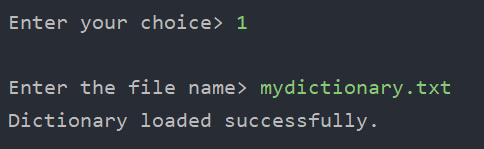
Description automatically generated

* **Empty.**

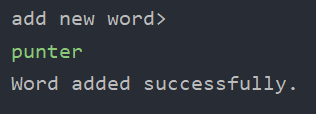
A screenshot of a computer

Description automatically generated with medium confidence

* **A text file having strings.**



1. **Add new word**



Text

Description automatically generated

1. **Search for word**

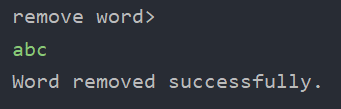
A screenshot of a computer

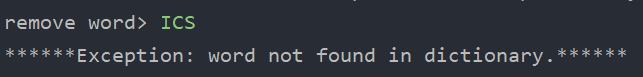
Description automatically generated with low confidence

Graphical user interface, text

Description automatically generated

1. **Remove word**





**5) Find similar words**

Text

Description automatically generated

**6) Save the file**

Graphical user interface, text

Description automatically generated with medium confidence