TITLE OF PRESENTATION
“TRIE TREES”

PRESENTED BY

11 Bhoomika Tolia

04 Prathmesh Araikar

05 Atharv Bhale

07 Aarohi Bhand

72 Ayush Mishra

GUIDE

Prof. Devika Verma

CONTENTS:

INTRODUCTION

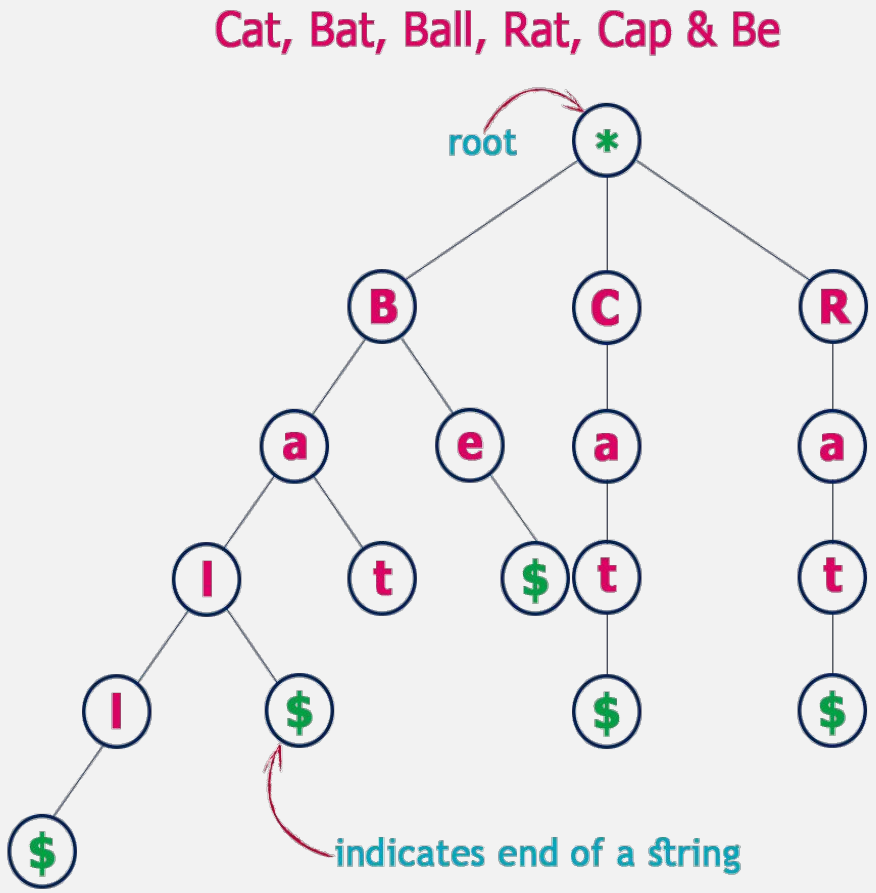
- Trie is a sorted tree-based data-structure that stores the set of strings. It has the number of pointers equal to the number of characters of the alphabet in each node. It can search a word in the dictionary with the help of the word's prefix. Trie is also known as the digital tree or prefix tree. The position of a node in the Trie determines the key with which that node is connected.For example, if we assume that all strings are formed from the letters 'a' to 'z' in the English alphabet, each trie node can have a maximum of 26 points.Properties of the Trie for this set of the string are:The root node of the trie always represents the null node.Each child of nodes is sorted alphabetically.Each node can have a maximum of 26 children (A to Z).Each node (except the root) can store one letter of the alphabet.

CREATION OF TRIE CLASS

```
public class Trie {
static final int ALPHABET_SIZE = 26; // Lowercase characters only

static class TrieNode
{
TrieNode[] children = new TrieNode[ALPHABET_SIZE];
boolean isEndOfWord;
};
static TrieNode root; Part 1
```

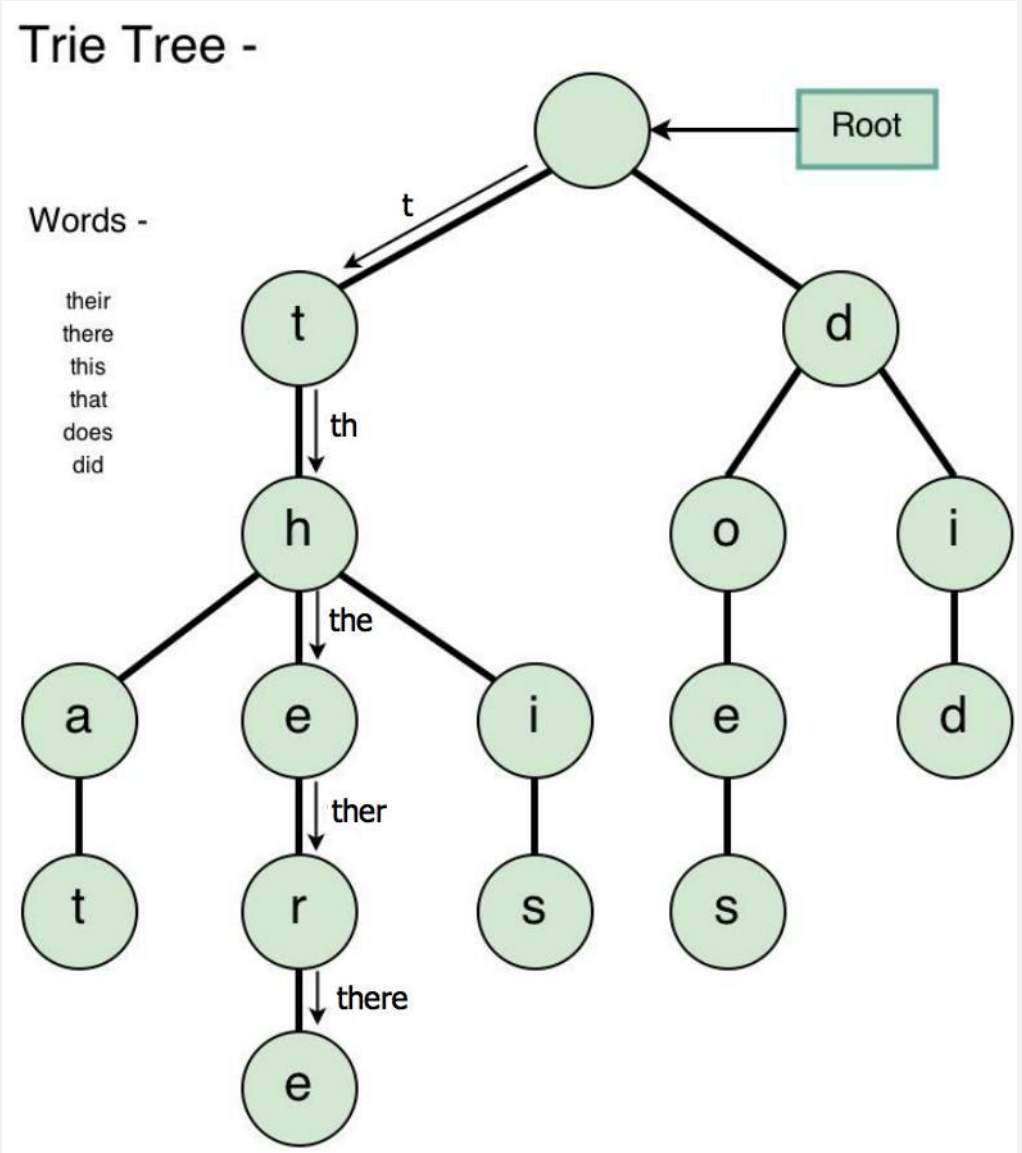
INSERTION FUNCTION

```
static void insert(String key)
{
int level;
int length = key.length();
int index;
TrieNode pCrawl = root;
for (level = 0; level < length; level++)
{ index = key.charAt(level) - 'a';
if (pCrawl.children[index] == null)
pCrawl.children[index] = new TrieNode();
pCrawl = pCrawl.children[index];          }
pCrawl.isEndOfWord = true;  }
```



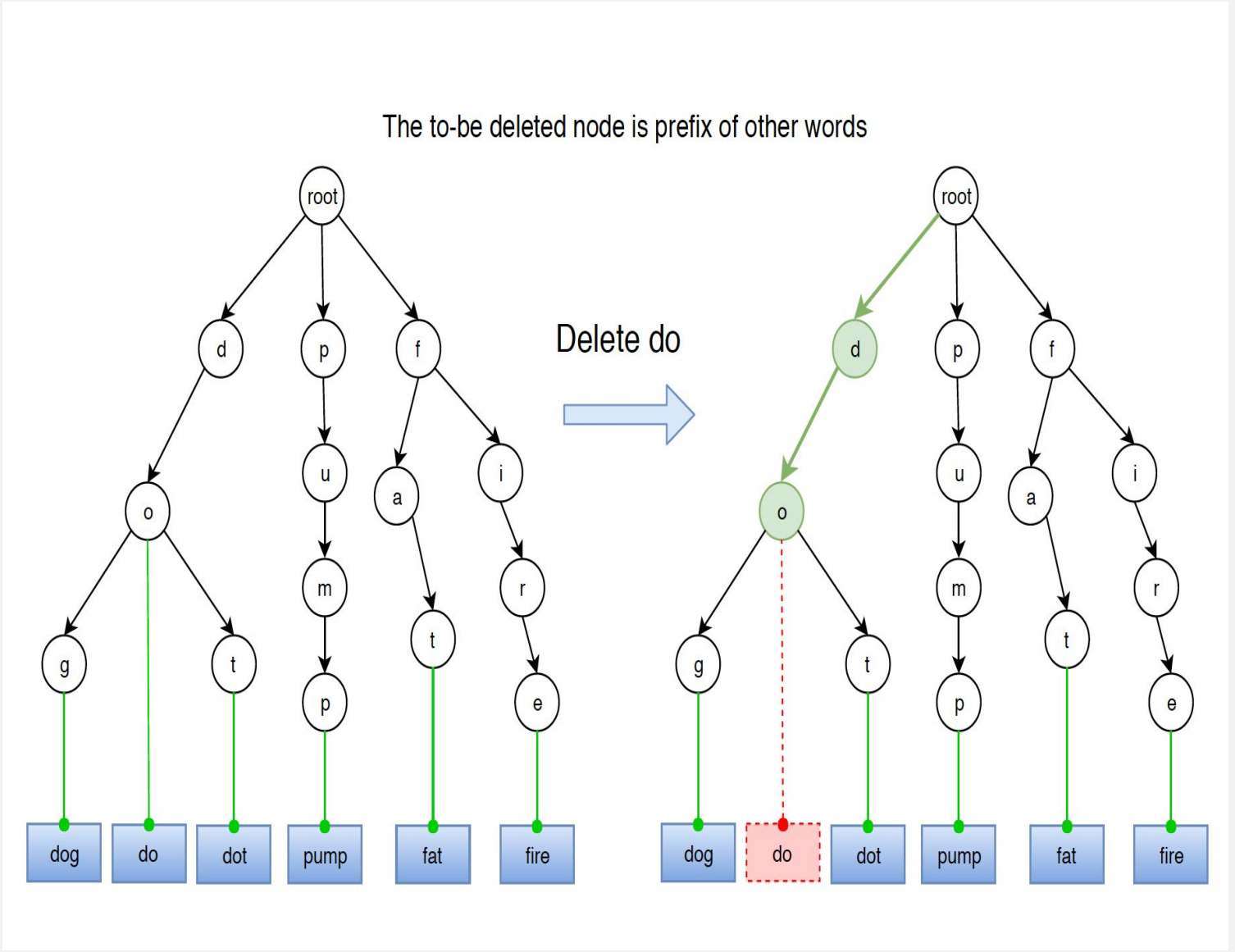
SEARCH FUNCTION

```
static boolean search(String key)  {
        int level;
        int length = key.length();
        int index;
        TrieNode pCrawl = root;
        for (level = 0; level < length; level++) {
                index = key.charAt(level) - 'a';
                if (pCrawl.children[index] == null)

                        return false;
                pCrawl = pCrawl.children[index];
        }
        return pCrawl.isEndOfWord;
}
```



Deletion

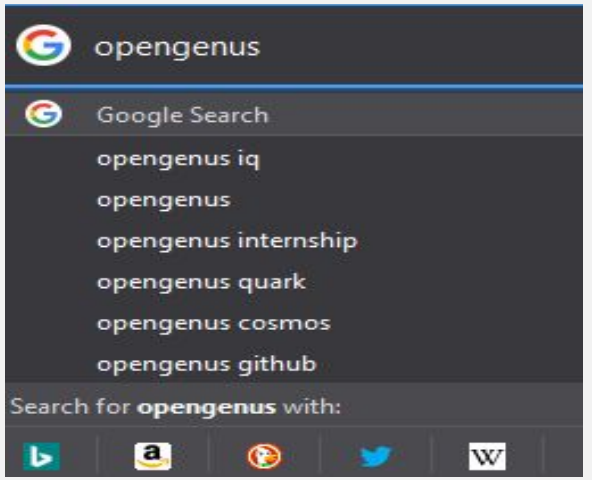
```
 static TrieNode remove(TrieNode root, String key, int
depth){
      if (root == null)
          return null;
       if (depth == key.length()) {
          if (root.isEndOfWord)
              root.isEndOfWord = false;
          if (isEmpty(root)) {
              root = null;
          }
          return root;
      }


      int index = key.charAt(depth) - 'a';
      root.children[index] =
          remove(root.children[index], key, depth + 1);
      if (isEmpty(root) && root.isEndOfWord == false){
          root = null;
      }
    return root;
  }
```



APPLICATIONS

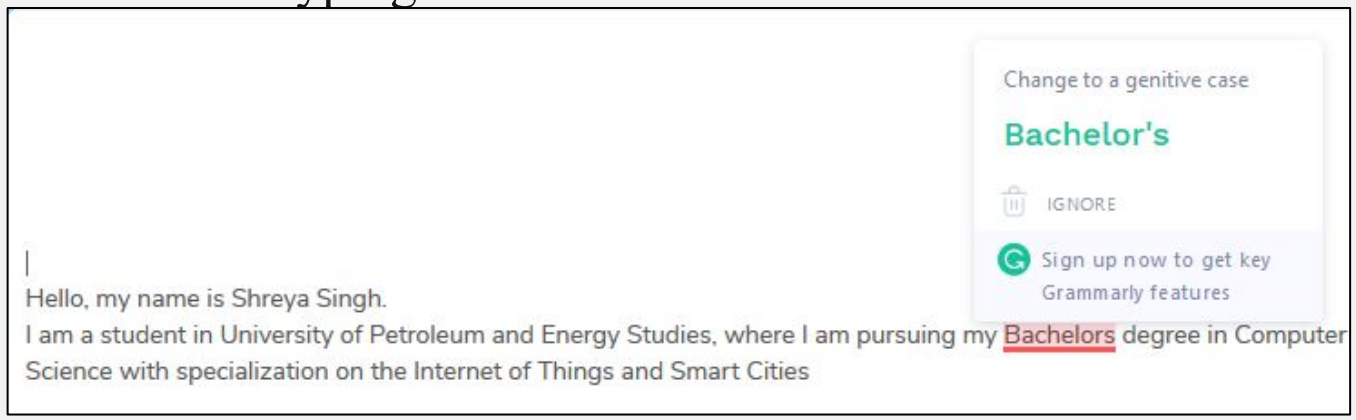
- **Auto Complete:**

  Autocomplete allows the browser to predict the value. When a user starts to type in a field, the browser should display options to fill in the field, based on earlier typed values.



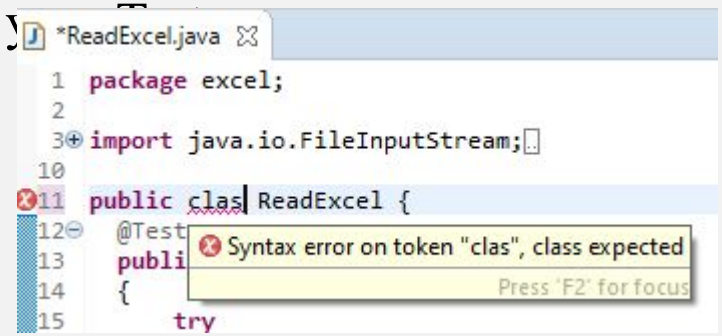
- **Spell Checkers / Auto Correct:**

  a software function that automatically makes or suggests corrections for mistakes in spelling or grammar made while typing:



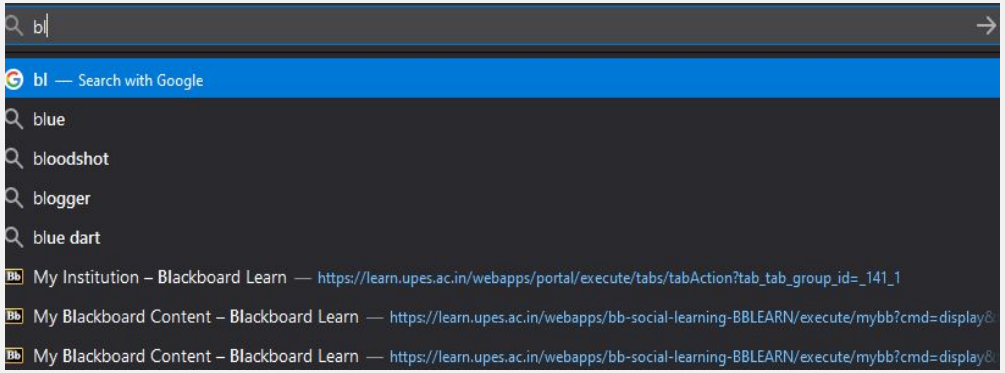
- **Source code editors:**

You can choose from the available suggestions and adopt it by typing the Tab or Enter key. With auto-complete enabled the editor identifies and underlines incorrect syntax which can result in compilation errors. Hover your mouse over the code, and a tool-tip describes the error. You can fix these errors early and save time in y



- **Browser History:**

The History feature keeps tabs on your Internet browsing for as long as you're online. Browser application designers realized that people needed a way of knowing where they'd been and what they'd read or seen online over a long Internet session.



PRO'S & CON'S

- **Advantages:**
- 1. The insert and the search algorithm have the best time complexity, i.e., O(n), faster than even the best of BST.
  2. All words can be easily printed in alphabetical order, which is difficult if we use hashing.
  3. Prefix search is easily doable.

- **Disadvantages:**
- 1. A lot of memory is used in storing the pool of strings in a Trie, and as more and more words are inserted, space complexity explodes.

CONCLUSION

- In this article, we looked at all the algorithms that concern a Trie DS along with Trie's advantages and disadvantages which will enable readers to make informed decisions while choosing Trie and also while using them as well.