# Trie

*"Prefix Tree"*

Prerequisite: Recursion, Tree

Md. Saidul Hoque Anik
onix.hoque.mist@gmail.com

# How many words are there in Dictionary?

# How many words are there in Dictionary?

The 20-volume Oxford English Dictionary contains full entries for 1,71,476 words in current use, and 47,156 obsolete words
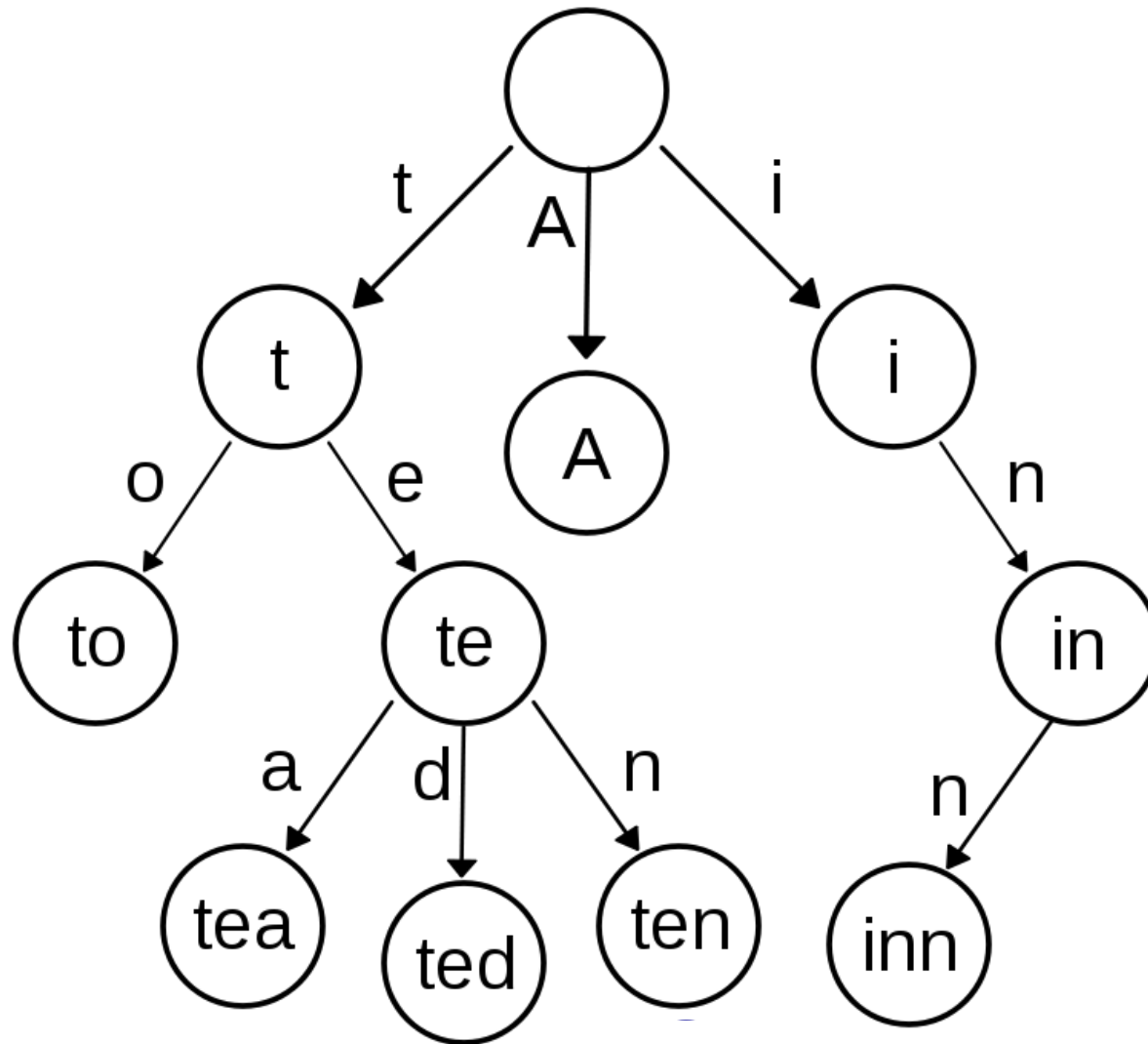
# How many words are there in the Internet?
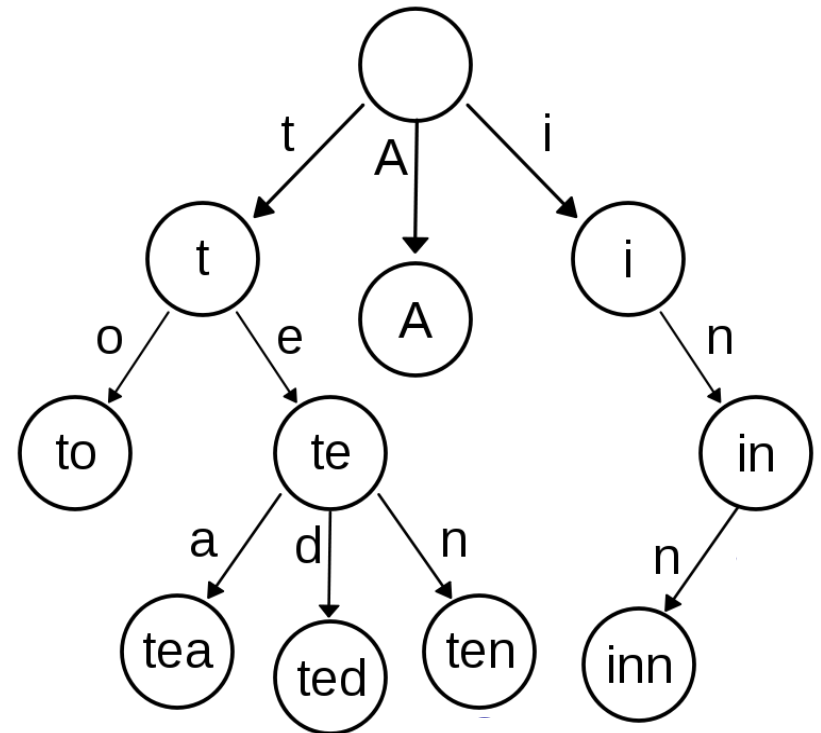
# Tree for Retrieval

# Node Structure

| Node |
|------|
| Node * children[SIZE_OF_ALPHABET]; |
| bool isLeaf; |

# Task

Insert the following words in a Trie

- Car
- Carbon
- Carpet
- Carbide
- Carpal
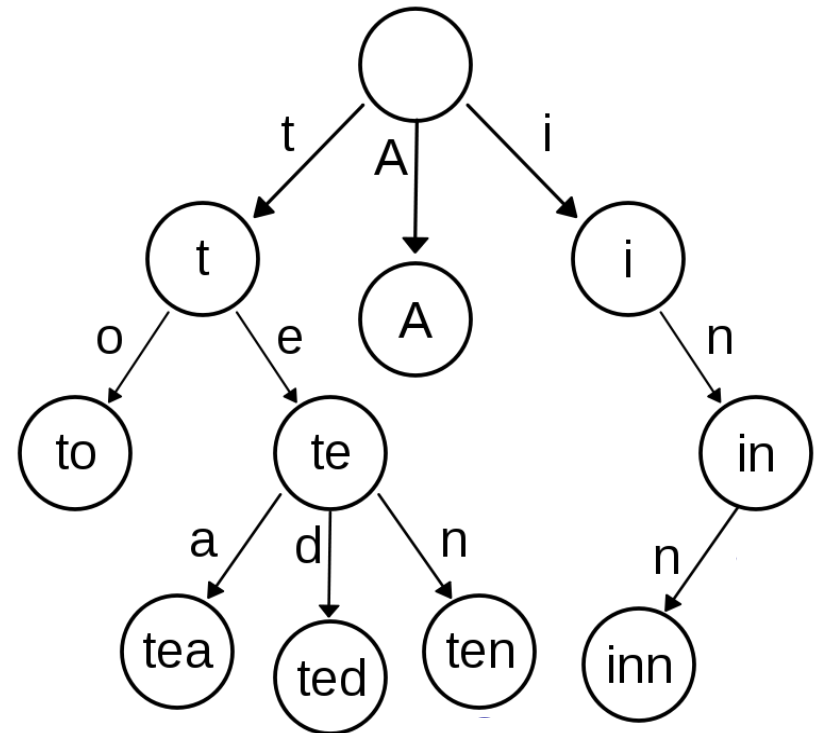
# Insertion

```
void insert(String s)
{
    for(every char in string s)
    {
        if(child node belonging to current char is null)
        {
            child node=new Node();
        }
        current_node=child_node;
    }
}
```

# Task

Insert the following words in a Trie

- a
- answer
- any
- by
- bye
- the
- their
- there

# Display

Perform Pre-order traversal in Trie

# Display

Perform Pre-order traversal in Trie

```cpp
void display(struct TrieNode* root, char str[], int level)
{
    if (isLeafNode(root))
    {
        str[level] = '\0';
        cout << str << endl;
    }
    for (int i = 0; i < alpha_size; i++)
    {
        if (root->children[i])
        {
            str[level] = i + 'a';
            display(root->children[i], str, level + 1);
        }
    }
}
```

# Search

```
boolean check(String s)
{
    for(every char in String s)
    {
        if(child node is null)
        {
            return false;
        }
    }
    return true;
}
```

# Search

```
bool search(struct TrieNode *root, string key)
{
    struct TrieNode *temp = root;

    for (int i = 0; i < key.length(); i++)
    {
        int index = key[i] - 'a';
        if (!temp->children[index])
            return false;
        temp = temp->children[index];
    }
    return (temp != NULL && temp->isEndOfWord);
}
```

# Deletion

Possible Cases

1.  Key may not be there in trie. Delete operation should not modify trie.
2.  Key present as unique key (no part of key contains another key (prefix), nor the key itself is prefix of another key in trie). Delete all the nodes.
3.  Key is prefix key of another long key in trie. Unmark the leaf node.
4.  Key present in trie, having atleast one other key as prefix key. Delete nodes from end of key until first leaf node of longest prefix key.

# Deletion

```cpp
TrieNode* remove(TrieNode* root, string key, int depth = 0)
{
    if (!root)
        return NULL;

    if (depth == key.size()) {

        if (root->isEndOfWord)
            root->isEndOfWord = false;

        if (isEmpty(root)) {
            delete (root);
            root = NULL;
        }
        return root;
    }
```

# Deletion

```
// If not last character, recur for the child
// obtained using ASCII value
int index = key[depth] - 'a';
root->children[index] = remove(root->children[index], key, depth + 1);

// If root does not have any child (its only child got
// deleted), and it is not end of another word.
if (isEmpty(root) && root->isEndOfWord == false) {
    delete (root);
    root = NULL;
}

return root;
}
```

# Reference

1. https://www.hackerearth.com/practice/data-structures/advanced-data-structures/trie-keyword-tree/tutorial/

2. https://en.wikipedia.org/wiki/Trie

3. https://www.geeksforgeeks.org/trie-insert-and-search/

4. https://www.geeksforgeeks.org/trie-delete/