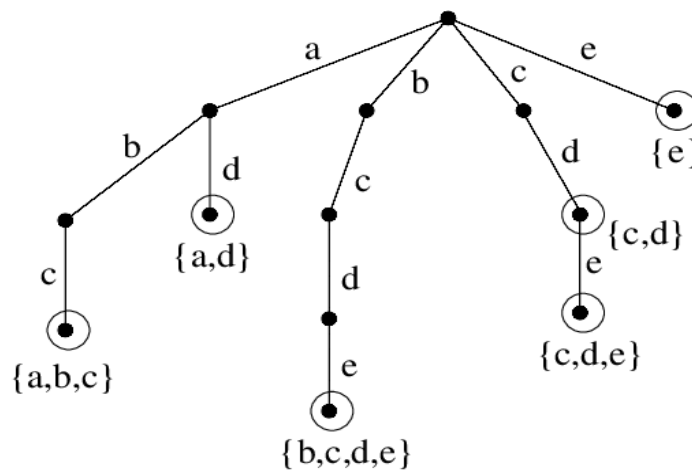- Assignment Name: Lab10.
- Description: Display the words in a trie in lexicographic order.
- Total Marks: 10.
- Deadline: Depends on respective lab hrs.
- Question Title: Display the words in a trie in lexicographic order.
- Marks allotted: 10.

A specific key from a collection can be sorted and searched for using the k-ary search tree type called "Trie." Search complexity can be reduced to the ideal level using Trie.

A trie is an alphabet-letter-storing data structure that resembles a tree. Words and strings can be extracted from the structure by travelling down a branch path of the tree by organising the nodes in a specific way.



In this Lab we will print the words in the trie in lexicographic order and implement it using DFS traversal technique.

```
#define ALPHABET_SIZE (26)


struct TrieNode
{
    struct TrieNode *children[ALPHABET_SIZE];
    bool isEndOfWord;
};
```

You are required to implement the following function:
1. Void insert(struct TrieNode *root , char *key);
   - A function that traverses through the trie and inserts the keys of the words into it . You need not call the function anywhere it is already being called in the main function.
   - **[NOTE : Same words can be repeated Multiple Times but only one copy will exist in the Trie and will be only displayed once while printing].**
2. Void dfs_word(struct TrieNode *root , char* word , int* plent);
   - A function that recursively traverses through the trie . The parameter word will append the current letter at present index and will print the word if it's a leaf

node. plent is the length of the current word that allows us to keep track of the length of the word.

**Functions to initialise Trie have already been implemented in the boilerplate code. USING THE BOILERPLATE CODE PROVIDED IS MANDATORY.**

- Description:
- Input format:
  - First line of input consists of an integer N, representing the number of words to be inserted into the Trie.
  - The following N lines represent the words that will be inserted
  - **Character of the words will be 'a' to 'z' only**
- Output format:
  - Lexicographically ordered words in N lines. Using the display method

Sample Test Cases :

- TC #1:
  - Input:
    ```
    3
    bimbo
    jumbo
    bamboo
    ```
  - Expected output:
    ```
    bamboo
    bimbo
    jumbo
    ```
- TC #2:
  - Input:
    ```
    8
    marco
    a
    ant
    p
    marc
    polo
    any
    mon
    ```

  - Expected Output:
    ```
    a
    ant
    any
    marc
    marco
    mon
    p
    polo
    ```