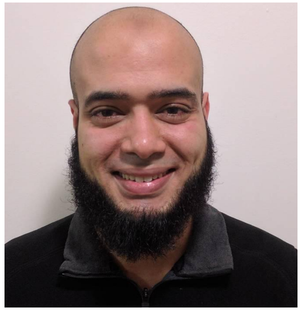
Data Structures
Trie Homework 1

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Problem #1: Iterative version

- void insert(string str)
- bool word_exist(string str)
-
- The recursive version of the lecture is slow when we have a lot of queries
- Rewrite an iterative version of these functions
- *Tip: for all programs assume trie is based on 26 letters unless otherwise mentioned*

Problem #2: minimal prefix

- string first_word_prefix(const string &str)
- Given a string, find the smallest trie (full) word that is a prefix for the given str
    - If there is no word, just return the original input
- E.g. Assume trie has words {xyz, xyzeA, a, bc}
- Input ⇒ output
    - x ⇒ x   [no trie word is prefix for x
    - xyzabc ⇒ xyz is the smallest prefix to xyzabc
- This is a sub-problem from this leetcode problem

Problem #3: Is suffix

- Write a trie that has 2 methods only
  - void insert(string str)
  - bool suffix_exist(string str)
- Suffix_exist returns true if any inserted word has such suffix

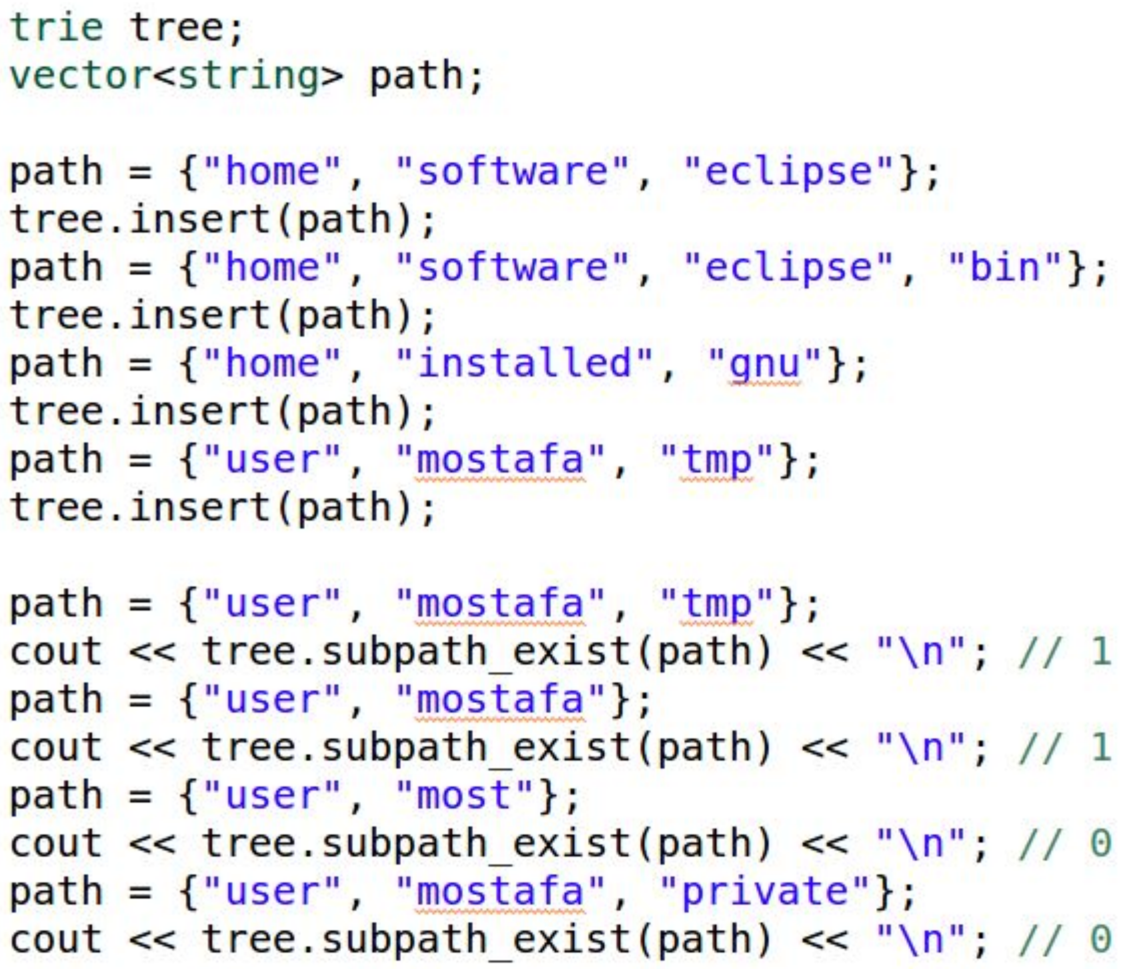
Problem #4: Memory Efficient

- Imagine that we need to use 256 (not just 26) letters. This means too much memory pointers that could be null
- Another efficient representation is using map<> so that we only store the existing nodes
    - In other words, we move from array of pointers to self-balancing trees of pointers
    - Better memory, but log256 to access element instead of O(1), actually better!
- Change the data structure to have a map instead of the array!

Problem #5: OS Paths

- Assume for simplicity a system path represented as vector of strings
  - /home/software/eclipse/bin ⇒ {"home", "software", "eclipse", "bin"}
- Design a trie
  - void insert(const vector<string> &path) ⇒ adds a path to the trie
  - bool subpath_exist(const vector<string> &path) ⇒ True if such sub-path exists

Problem #5: OS Paths



```
trie tree;
vector<string> path;

path = {"home", "software", "eclipse"};
tree.insert(path);
path = {"home", "software", "eclipse", "bin"};
tree.insert(path);
path = {"home", "installed", "gnu"};
tree.insert(path);
path = {"user", "mostafa", "tmp"};
tree.insert(path);

path = {"user", "mostafa", "tmp"};
cout << tree.subpath_exist(path) << "\n"; // 1
path = {"user", "mostafa"};
cout << tree.subpath_exist(path) << "\n"; // 1
path = {"user", "most"};
cout << tree.subpath_exist(path) << "\n"; // 0
path = {"user", "mostafa", "private"};
cout << tree.subpath_exist(path) << "\n"; // 0
```

“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”