Trie

package main

import "fmt"

type Node struct {

Children map[byte]\*Node

IsEnd bool

}

type Trie struct {

root \*Node

}

func (t \*Trie) Insert(s string) {

node := t.root

for i := 0; i < len(s); i++ {

if node.Children[s[i]] == nil {

node.Children[s[i]] = &Node{Children: map[byte]\*Node{}}

}

node = node.Children[s[i]]

}

node.IsEnd = true

}

func (t \*Trie) Contains(s string) bool {

if t.root == nil {

return false

}

node := t.root

for i := 0; i < len(s); i++ {

if node.Children[s[i]] != nil {

node = node.Children[s[i]]

} else {

return false

}

}

return node.IsEnd

}

func (t \*Trie) Prefix(s string) {

for i := 0; i < len(s); i++ {

t.Insert(s[i:])

}

}

func (t \*Trie) Suffix(s string) {

for i := len(s); i >= 0; i-- {

t.Insert(s[:i])

}

}

func main() {

t := &Trie{&Node{Children: map[byte]\*Node{}}}

t.Insert("gold")

t.Prefix("bigbang")

t.Suffix("internet")

fmt.Println(t.Contains("big"))

}