

main.c



```
1 #include<stdio.h>
2 #include<stdlib.h>
3 #include<stdbool.h>
4
5 #define ALPHABET_SIZE 26
6
7 struct TrieNode{
8     struct TrieNode * children[ALPHABET_SIZE];
9     bool isEndOfWord;
10
11 };
12
13 struct TrieNode *createNode(){
14     struct TrieNode *node = (struct TrieNode *) malloc(sizeof(struct TrieNode));
15     node->isEndOfWord = false;
16     for(int i = 0; i < ALPHABET_SIZE; i++)
17         node->children[i] = NULL;
18     return node;
19
20 };
21
22 void insert(struct TrieNode *root, const char *word){
23     struct TrieNode *current = root;
24     while(*word){
25         int index = *word - 'a';
26         if(!current -> children[index])
```

main.c



```
27         current->children[index] = createNode();
28         current = current->children[index];
29         word++;
30     }
31     current->isEndOfWord= true;
32 }
33
34 - bool search (struct TrieNode * root, const char *word){
35     struct TrieNode *current = root;
36
37 -     while(*word){
38         int index = *word - 'a';
39         if(!current->children[index])
40             return false;
41         current = current->children[index];
42         word++;
43     }
44     return current->isEndOfWord;
45 }
46
47 - int main(){
48     struct TrieNode *root = createNode();
49     insert(root, "hello");
50     insert(root, "world");
51     insert(root, "hi");
52     insert(root, "trie");
```

main.c

Run

Output

```
37 -     while(*word){  
38         int index = *word -'a';  
39         if(!current->children[index])  
40             return false;  
41         current = current->children[index];  
42         word++;  
43     }  
44     return current->isEndOfWord;  
45 }  
46  
47 int main(){  
48     struct TrieNode *root = createNode();  
49     insert(root, "hello");  
50     insert(root, "world");  
51     insert(root, "hi");  
52     insert(root, "trie");  
53  
54     printf("search 'hello': %s\n", search (root, "hello") ? "Found" : "Not  
      Found");  
55     printf("search 'hi': %s\n", search (root, "hi") ? "Found" : "Not Found");  
56     printf("search 'her': %s\n", search (root, "her") ? "Found" : "Not Found");  
57     printf("search 'world': %s\n", search (root, "world") ? "Found" : "Not  
      Found");  
58     return 0;  
59 }  
60
```

search 'hello': Found
search 'hi': Found
search 'her': Not Found
search 'world': Found
==== Code Execution Successful ===