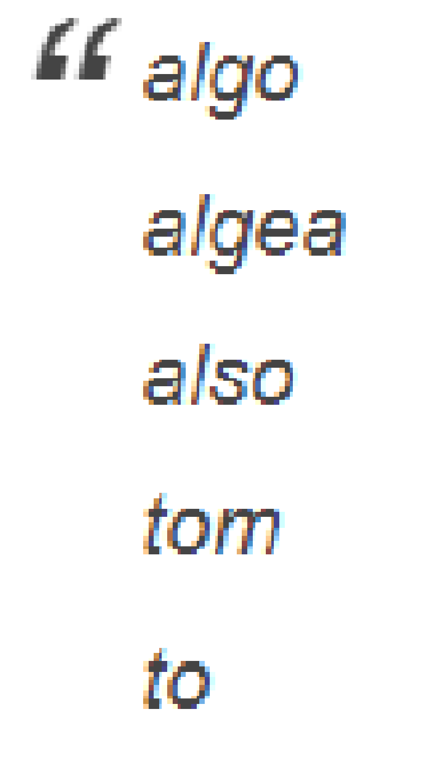
Trie

Q queries of 3 types :

1 . Insert a string in a set
2 . Delete a string from the set
3 . Check whether a given string is in the set or not



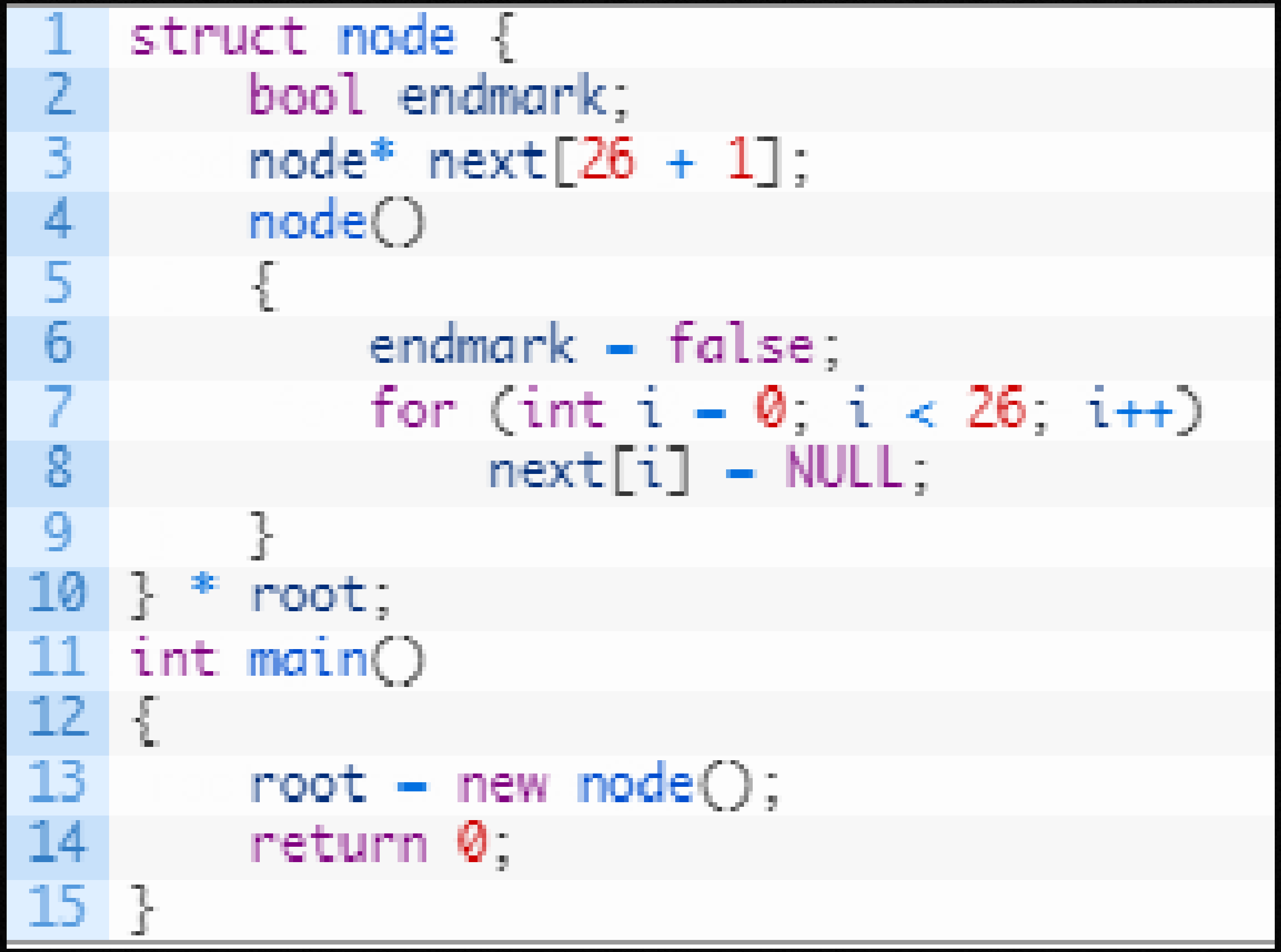
“ *algo*

algea

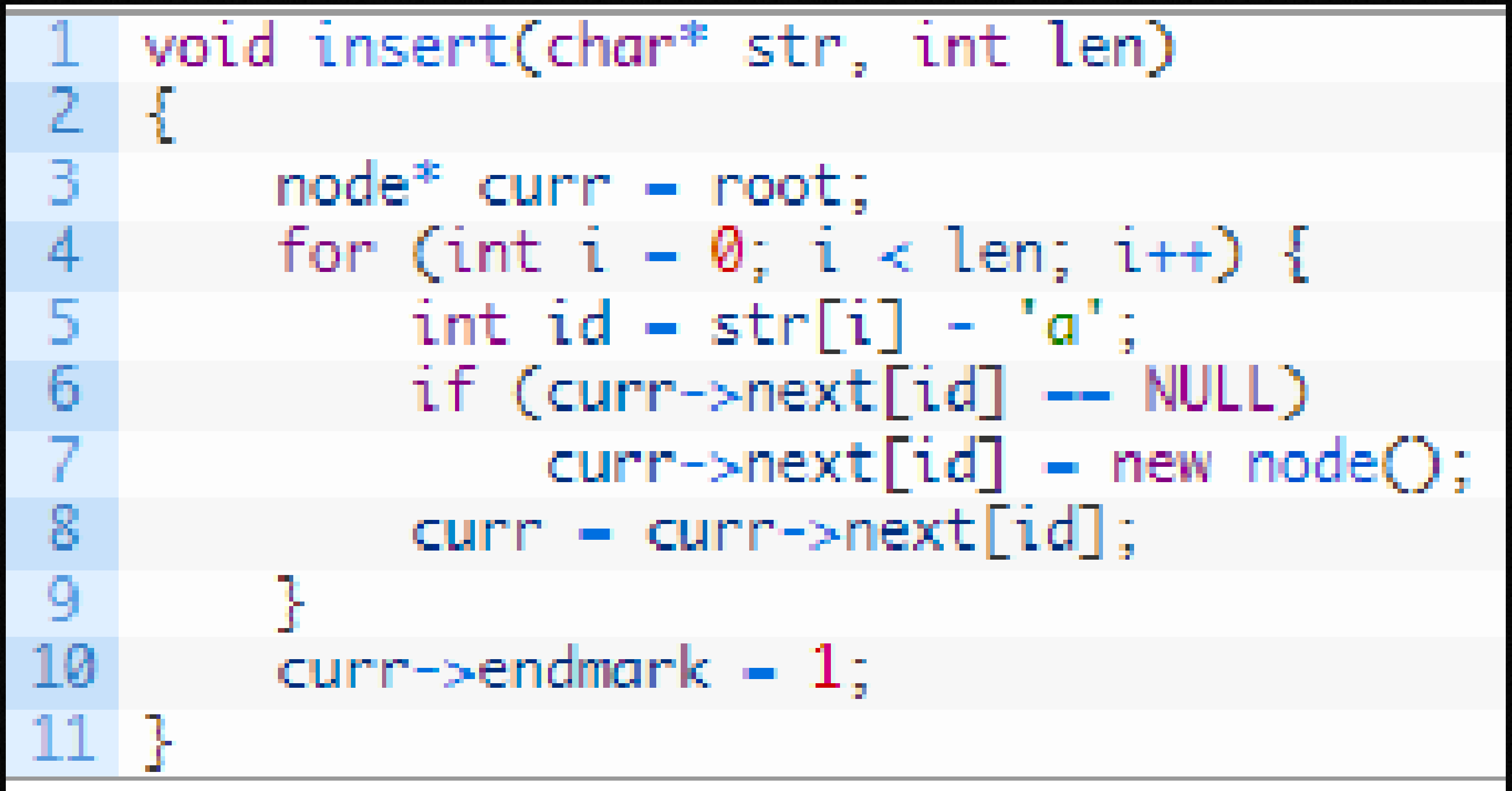
also

tom

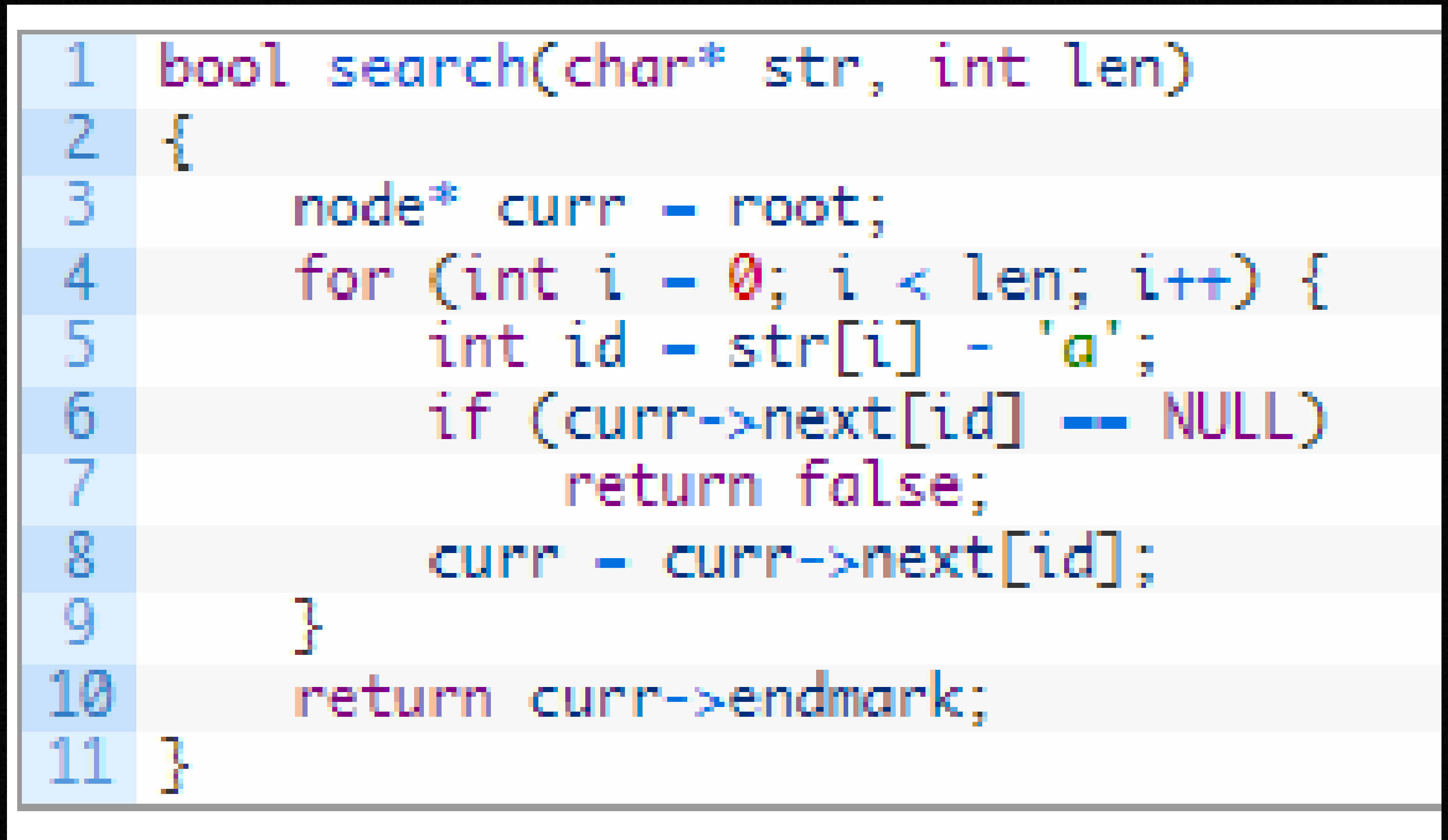
to



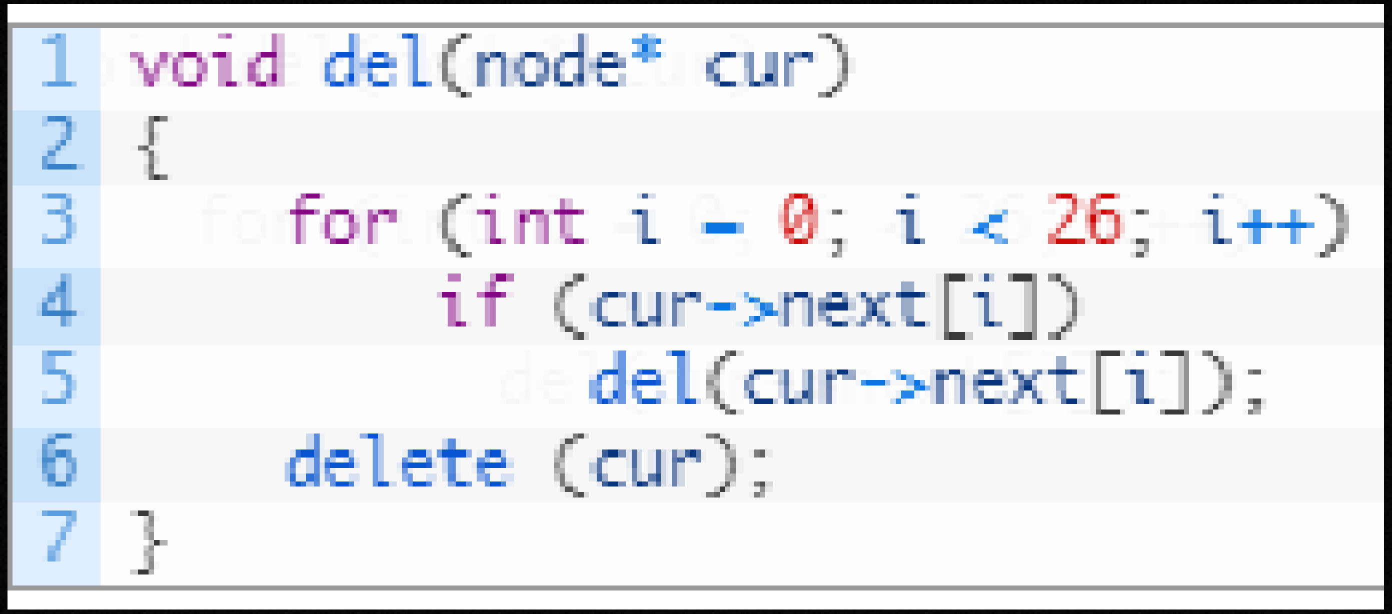
```
struct node {
    bool endmark;
    node* next[26 + 1];
    node()
    {
        endmark = false;
        for (int i = 0; i < 26; i++)
            next[i] = NULL;
    }
} * root;
int main()
{
    root = new node();
    return 0;
}
```



```
void insert(char* str, int len)
{
    node* curr = root;
    for (int i = 0; i < len; i++) {
        int id = str[i] - 'a';
        if (curr->next[id] == NULL)
            curr->next[id] = new node();
        curr = curr->next[id];
    }
    curr->endmark = 1;
}
```



```
bool search(char* str, int len)
{
    node* curr = root;
    for (int i = 0; i < len; i++) {
        int id = str[i] - 'a';
        if (curr->next[id] == NULL)
            return false;
        curr = curr->next[id];
    }
    return curr->endmark;
}
```



```cpp
void del(node* cur)
{
    for (int i = 0; i < 26; i++)
        if (cur->next[i])
            del(cur->next[i]);
    delete (cur);
}
```

Time Complexity

len = length of a string .

L = sum of length of each inserted string .

K = number of alphabets

Memory : O(L*K)

Time : O(L)