**Implement Trie**

/\*

Your Trie object will be instantiated and called as such:

Trie\* obj = new Trie();

obj->insert(word);

bool check2 = obj->search(word);

bool check3 = obj->startsWith(prefix);

\*/

class Node{

public:

Node \*links[26];

bool flag = false;

Node(){}

bool containsKey(char ch){

return (links[ch-'a'] != NULL);

}

void put(char ch , Node \*n){

links[ch-'a'] = n;

}

Node \*get(char ch){

return links[ch-'a'];

}

void setend(){

flag = true;

}

bool isend(){

return flag;

}

};

class Trie {

public:

Node \* root;

Trie() {

root = new Node();

}

void insert(string word) {

Node\* n = root;

for(int i = 0 ; i<word.size() ; i++){

if(!n->containsKey(word[i])){

Node\* temp = new Node();

n->put(word[i],temp);

}

n = n->get(word[i]);

}

n->setend();

}

bool search(string word) {

Node \*n = root;

for(int i =0;i<word.size();i++){

if(!n->containsKey(word[i])){

return false;

}

n = n->get(word[i]);

}

return n->isend();

}

bool startsWith(string prefix) {

Node \*n = root;

for(int i =0;i<prefix.size();i++){

if(!n->containsKey(prefix[i])){

return false;

}

n = n->get(prefix[i]);

}

return true;

}

};