```cpp
class Trie {
private:
  struct node {
    char val; map<char, node*> childs_mp; ll deg; node *parent; int cnt;
  };
  bool erase_leaf(node *ptr) {
    if (ptr->val != '¥0') {
      do {
        char v = ptr->val;
        ptr->cnt--;
        ptr = ptr->parent;
        if (ptr->childs_mp[v]->cnt == 0) {
          delete(ptr->childs_mp[v]);
          ptr->childs_mp.erase(v);
        }
      } while (ptr != root);
      ptr->cnt--;
      return true;
    }
    else return false;
  }
  node *root;
  bool multi_flag;
public:
  Trie(bool multi_flag) {
    root = new node{ '¥0',{}, 0, nullptr, 0 };
    Trie::multi_flag = multi_flag;
  }
  void add(string s) {
    node *a = root;
    Loop(i, s.length()) {
      char c = s[i];
      if (a->childs_mp.find(c) == a->childs_mp.end()) {
        node *node_buf = new node{ c,{},a->deg + 1, a, 0 };
        a->childs_mp[c] = node_buf;
      }
      a->cnt++;
      a = a->childs_mp[c];
    }
    if (a->childs_mp.find('¥0') == a->childs_mp.end()) {
      node *nil = new node{ '¥0',{}, a->deg + 1, a, 0 };
      a->childs_mp['¥0'] = nil;
    }
    a->cnt++;
    a = a->childs_mp['¥0'];
    a->cnt++;
    if (!multi_flag && a->cnt >= 2) erase_leaf(a);
  }
  bool find(string s) {
    node *a = root;
    Loop(i, s.length()) {
      char c = s[i];
      if (a->childs_mp.find(c) == a->childs_mp.end()) return false;
      else a = a->childs_mp[c];
    }
    if (a->childs_mp.find('¥0') != a->childs_mp.end()) return true;
    else return false;
  }
  bool erase(string s) {
    node *a = root;
    Loop(i, s.length()) {
      char c = s[i];
      if (a->childs_mp.find(c) == a->childs_mp.end()) return false;
      else a = a->childs_mp[c];
    }
    if (a->childs_mp.find('¥0') != a->childs_mp.end()) {
      if (erase_leaf(a)) return true;
      else return false;
    }
    else return false;
  }
```

```
72  };
```