package trietest;

import java.util.\*;

class TrieNode

{

char content;

boolean isEnd;

int count;

LinkedList<TrieNode> childList;

/\* Constructor \*/

public TrieNode(char c)

{

childList = new LinkedList<TrieNode>();

isEnd = false;

content = c;

count = 0;

}

public TrieNode subNode(char c)

{

if (childList != null)

for (TrieNode eachChild : childList)

if (eachChild.content == c)

return eachChild;

return null;

}

}

class Trie

{

private TrieNode root;

/\* Constructor \*/

public Trie()

{

root = new TrieNode(' ');

}

/\* Function to insert word \*/

public void insert(String word)

{

if (search(word) == true)

return;

TrieNode current = root;

for (char ch : word.toCharArray() )

{

TrieNode child = current.subNode(ch);

if (child != null)

current = child;

else

{

current.childList.add(new TrieNode(ch));

current = current.subNode(ch);

}

current.count++;

}

current.isEnd = true;

}

/\* Function to search for word \*/

public boolean search(String word)

{

TrieNode current = root;

for (char ch : word.toCharArray() )

{

if (current.subNode(ch) == null)

return false;

else

current = current.subNode(ch);

}

if (current.isEnd == true)

return true;

return false;

}

/\* Function to remove a word \*/

public void remove(String word)

{

if (search(word) == false)

{

System.out.println(word +" does not exist in trie\n");

return;

}

TrieNode current = root;

for (char ch : word.toCharArray())

{

TrieNode child = current.subNode(ch);

if (child.count == 1)

{

current.childList.remove(child);

return;

}

else

{

child.count--;

current = child;

}

}

current.isEnd = false;

}

}

/\* Class Trie Test \*/

public class TrieTest

{

public static void main(String[] args)

{

Scanner scan = new Scanner(System.in);

/\* Creating object of AATree \*/

Trie t = new Trie();

System.out.println("Trie Test\n");

char ch;

/\* Perform tree operations \*/

do

{

System.out.println("\nTrie Operations\n");

System.out.println("1. insert ");

System.out.println("2. delete");

System.out.println("3. search");

int choice = scan.nextInt();

switch (choice)

{

case 1 :

System.out.println("Enter string element to insert");

t.insert( scan.next() );

break;

case 2 :

System.out.println("Enter string element to delete");

try

{

t.remove( scan.next() );

}

catch (Exception e)

{

System.out.println(e.getMessage()+" not found ");

}

break;

case 3 :

System.out.println("Enter string element to search");

System.out.println("Search result : "+ t.search( scan.next() ));

break;

default :

System.out.println("Wrong Entry \n ");

break;

}

System.out.println("\nDo you want to continue (Type y or n) \n");

ch = scan.next().charAt(0);

} while (ch == 'Y'|| ch == 'y');

}

}

**Output:**

Trie Test

Trie Operations

1. insert

2. delete

3. search

1

Enter string element to insert

cafe

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

1

Enter string element to insert

coffee

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

1

Enter string element to insert

mug

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

1

Enter string element to insert

sugar

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

3

Enter string element to search

sugar

Search result : true

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

3

Enter string element to search

spoon

Search result : false

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

2

Enter string element to delete

sugar

Do you want to continue (Type y or n)

y

Trie Operations

1. insert

2. delete

3. search

3

Enter string element to search

sugar

Search result : false

Do you want to continue (Type y or n)

n