package search;  
  
import java.util.ArrayList;  
import java.util.List;  
import java.util.concurrent.Callable;  
import java.util.concurrent.ExecutorService;  
import java.util.concurrent.Executors;  
import java.util.concurrent.atomic.AtomicInteger;  
  
public class Application {  
 static final int *n* = 10000000;  
 static final int *m* = 4;  
 static char *target*[] = new char[*n*];  
 static char *pattern*[] = new char[*m*];  
// static final int numCores = Runtime.getRuntime().availableProcessors();  
 static final int *numCores* = 2;  
  
 public static void main(String[] args) {  
  
 *loadData*();  
  
// testMethod();  
  
 AtomicInteger returnCount = *executer*();  
  
 System.*out*.println("Count : " + returnCount.get());  
  
 }  
  
 static AtomicInteger executer() {  
  
 ExecutorService executorService = Executors.*newFixedThreadPool*(*numCores*);  
 int step = *target*.length / *numCores*;  
 int startIndex, endIndex;  
 List<Callable<AtomicInteger>> tasks = new ArrayList<>();  
 AtomicInteger count= new AtomicInteger();  
 List<Integer> indexList = new ArrayList<>();  
  
 for (int i = 0; i< *numCores*; i++) {  
 startIndex = i \* step;  
 if (i == *numCores* - 1) {  
 endIndex = *target*.length;  
 } else {  
 endIndex = (i+1) \* step;  
 }  
  
 int finalStartIndex = startIndex;  
 int finalEndIndex = endIndex;  
 Callable<AtomicInteger> task = () -> {  
 int j;  
 for (j= finalStartIndex; j< finalEndIndex; j++) {  
  
 if(*target*[j] == *pattern*[0] ) {  
 boolean match = false;  
 for (int k = 0;k< *pattern*.length;k++) {  
 if ((j+k) <= j) {  
 if(*target*[j+k] == *pattern*[k]) {  
 match = true;  
 } else {  
 match = false;  
 break;  
 }  
 }  
  
 }  
 if (match == true) {  
 count.incrementAndGet();  
 indexList.add(j);  
 }  
 }  
  
 if (Thread.*interrupted*()) {  
 return null;  
 }  
 }  
 return count;  
 };  
  
 tasks.add(task);  
 }  
  
 try {  
 executorService.invokeAll(tasks);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 } finally {  
 executorService.shutdown();  
 }  
  
 System.*out*.println(indexList.toString());  
 return count;  
 }  
  
 static void loadData() {  
 String str = "akwoejtreegkrwhvowpwtreelkkgakdjcnbxtreebsiochq";  
 String strPattern = "tree";  
 *pattern* = strPattern.toCharArray();  
 *target* = str.toCharArray();  
 }  
  
 private static void testMethod() {  
 List<Integer> indexList = new ArrayList<>();  
 int count = 0;  
 for (int j = 0; j< *target*.length - *pattern*.length; j++) {  
  
 if(*target*[j] == *pattern*[0]) {  
 boolean match = false;  
 for (int k = 0;k< *pattern*.length;k++) {  
 if(*target*[j+k] == *pattern*[k]) {  
 match = true;  
 } else {  
 match = false;  
 break;  
 }  
 }  
 if (match == true) {  
 count++;  
 indexList.add(j);  
 }  
 }  
  
 }  
 System.*out*.println(count);  
  
 System.*out*.println(indexList.toString());  
 }  
}