# Ransom Note Application

## Requirement:

You are a dognapper ( a kidnapper of dogs … not in charge of dog naps) and have to regularly write ransom notes. You are also a smart dognapper, you know that cutting out words from a newspaper or a magazine is the way to go … this way you are untraceable.

 So here is your challenge, you need to write a program that can verify if a newspaper/magazine has all the words you need to write the ransom note.

## Solution:

This is a simple Java application built with maven and added with java util logging to display the output.

Steps :

* Load magazine and ransom note contents into the corresponding maps, using **parseFile** function.
* magazineMap and ransomNoteMap will contain the Keys as Words and values as occurrences of the words.
* ParseFile function will take two arguments, one is filepath and second one is IsMagazine boolean value.
* If file type is Magazine, value will be true. Otherwise isMagazine value will be false.
* Once Maps are loaded successfully, parse the magazine map to check if ransom note can be created or not  using **parseMagazineToContainRansomNote**function
* If magazine map size is less than ransom note map size, hence Fail immediately.
* This means that the magazine does not contain all the unique words needed for the ransom note.
* If magazine map size is greater than ransom note map size, get the ransom note key set and iterate over ransom note map.
* If ransomnote key is available in magazine map, compare corresponding key values in both the maps.
* If occurrences of words in magazine map is greater than or equal to ransom note map, iterate the map further. Otherwise, return a false value.
* This application will return true value, if all the ransom note key values available in the magazine map.

### Big O Notation : O(N)

## Alternate solutions:

### Solution 1:

* Parse the input files and load the contents as words into two separate lists.
* Iterate over the ransom note words list and lookup the word in magazine words list.
* Once the word is found, remove the word from magazine words list.

This would be a costly operation as the number of iterations would be very high as we need to iterate over both the list completely before considering the case to be false.

### Solution 2 :

* Load corresponding magazine and ransom note into maps.
* Get  the ransom note key set and iterate over ransom note map.
* If ransom note key is available in magazine map, decrease corresponding word occurrences value by 1 and iterate further.

This solution is not very effective as the number of iterations to identify the false scenario will be more.

### Why our solution is better ?

* The considered solution is better than these approaches as it can identify a false scenario in less number of steps and iterations.
* If the key set in ransom note map is larger than magazine map, the overall maps need not be checked at all.
* Also, since we are iterating over the maps and comparing the values, false case can be identified easily.

## Assumptions:

It is assumed that all the special characters like full stop, comma etc. will always be available in the magazine and can be ignored in the checks.

## Compile and Steps to run :

### Compile:

The application is a maven project and can be built as below:

$ mvn clean install

This will compile the application and build an executable jar under target directory.

The application can also be compiled and executed directly in any of the IDEs(Eclipse/Netbeans).

### Run:

$ java -jar RansomNoteApplication-1.0.jar \"Path to Magazine\" \"Path to RansomNote.

To enable fine logging , provide another argument as “-d”

Example:

$ java -jar RansomNoteApplication-1.0.jar \"Path to Magazine\" \"Path to RansomNote” “-d”