Gebze Technical University
Computer Engineering

CSE 222 - 2019 Spring

HOMEWORK 6 REPORT

NEVRA GÜRSES
161044071

Ayşe Şerbetçi Turan

1 INTRODUCTION

1.1 Problem Definition

There is multiple files in a folder. We have to read all files and record all words in a hashmap.This hashmap structure that is name Word_Map includes String data structure to keep words as key and another hashmap class as value. Also all words have to link one added after word.Another hashmap that is name File_Map class includes String data structure as key and Arraylist data structure that includes list data structure in it as value.The key of File_Map class must be string because it keeps file names in which a word is included.The value of File_Map class that is arraylist have to keep all indexes that are positions of a word in a file. After reading all files and creating 2 hashmap classes we have to implement two operation that are name retrieving bi-grams and calculating TFIDF values that are used in Natural Language Processing operations.Retrieving bi-gram operation is list of two sequantial words that are given text.And TFID values is product of TF(Number of times that given word appears in a document) / (total number of words in the document) and IDF numbers(log(total number of documents / number of documents with given word in it).We have to solve this problems.

.

1.2 System Requirements

This program is writed with Java programming language.And Java requires following system requirements for Windows:

→Windows 10 (7u85 and above)

→Windows 8.x (Desktop)

→Windows 7 SP1.

→ Vista SP2.

→Windows Server 2008 SP2 and 2008 R2 SP1 (64-bit)

→Windows Server 2012 (64-bit) and 2012 R2 (64-bit)

→RAM: 128 MB; 64 MB for Windows XP (32-bit)

→Disk space: 124 MB.

This program is writed in Intellij IDEA.And Intellij IDEA requires following system requiremets:

→Windows 10/8/7/Vista/XP (incl. 64-bit).

→2 GB RAM minimum, 4 GB RAM recommended.

→1.5 GB hard disk space + at least 1 GB for caches.

→1024x768 minimum screen resolution.

This program also require Java SE Development Kit 11.0.2 .Java SE Development Kit (JDK) and Java SE Runtime Environment (JRE) require at minimum a Pentium 2 266 MHz processor.For the JDK, the option of installing the following features:

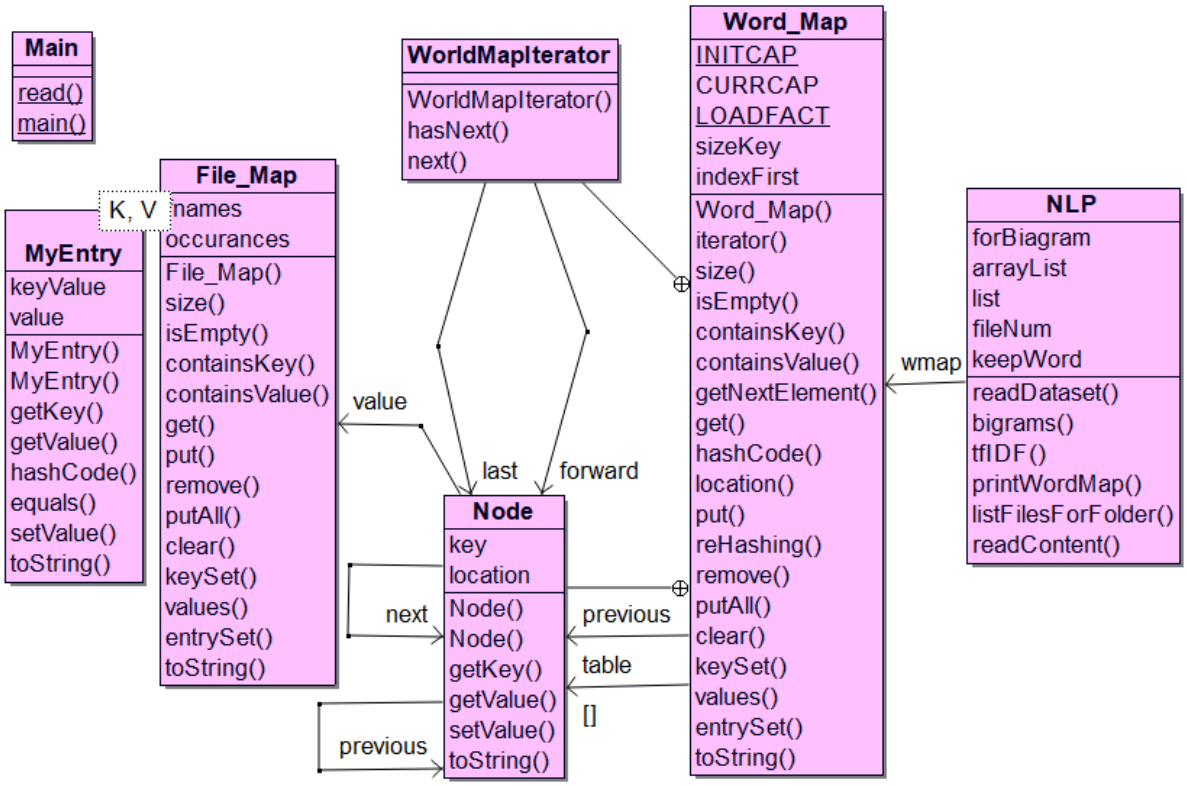
→Development Tools.

→Source Code.

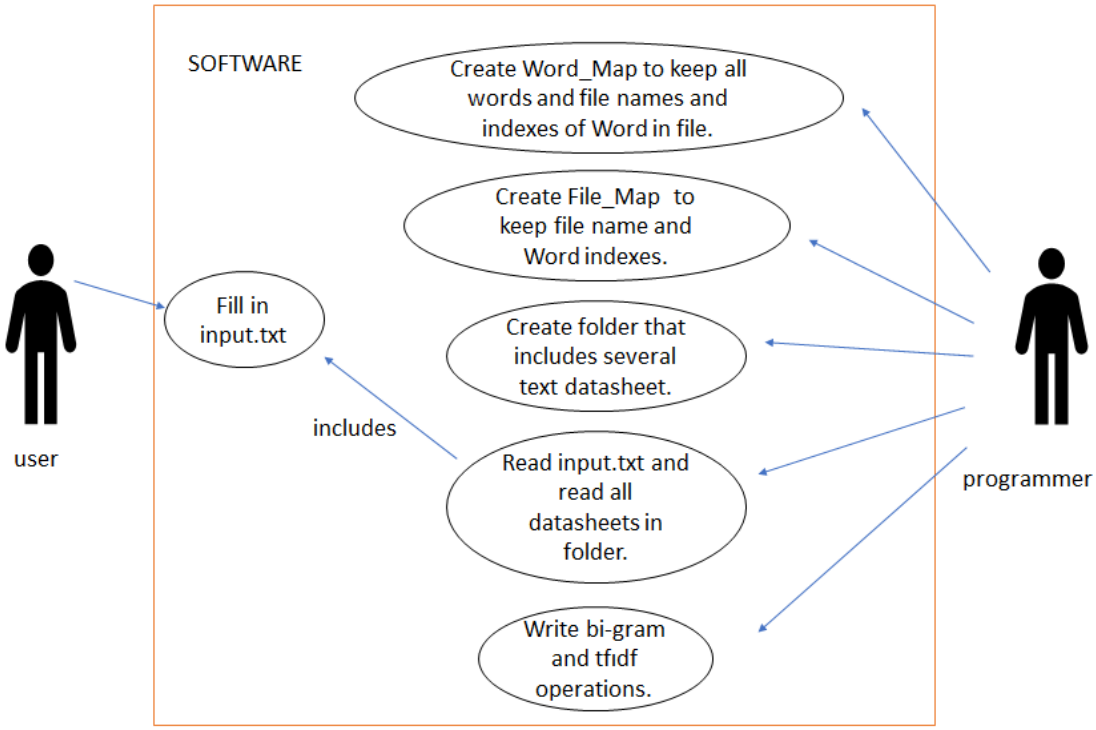
→Public Java Runtime Environment.

2 METHOD

2.1 Class Diagrams



2.2 Use Case Diagrams



2.3 Problem Solution Approach

To solve this problem ,firstly I create File_Map class that has String data structure as key and ArrayList data structure that includes List in it as value.I keep two arraylist to keep values and keys and create all methods of File_Map class.After that, I create Word_Map class that has String data structure as key and File_Map class as value.This Word_Map class includes an inner Node class to keep values and keys and to create a linked structure.Every added element in hashTable refers to one after added element.Also I create an inner iterator class to traverse all elements in hashMap in an efficient way.
After that I begin to read the all datasheets in folder.While I was reading files in NPL class, I record all words in Word_Map class as key and I record all file names and indexes of word in files as value.I do this operation in this way:
I control whether Word_map class includes processing word in it as key or not.If it include,I create a temperature File_Map class and put them file name and index of word in that file. If value of word that is a File_Map class contains current fileName as key,I add index of

current in list that is value of temperature File_Map, Actually I expand list that is value of temperature File_Map.Then I combine temperature File_Map and File_Map that is already value in Word_Map and it contains current word as key.By doing this,I upload value of word that is key.And then,I put word as key,and combined File_Map class as value in Word_Map.If Word_Map does not include current word as key,I create a File_Map and put current fileName as key and current index as value.Then I put in Word_Map current word as key and created File_Map as value.While I am doing this,I add all words in an arrayList that is data field of NPL class to use in finding biagrams.And also I add all fileNames as key and number of words in this file as value in map data structure that is data field of NPL class. To write biagrams,I traverse arrraylist that is data field.If element of arraylist equal to given word, I combine this word with one after word and add in a variable list. After I return this variable list.So,all bi-grams in that returned list.
To calculate TFIDF, I used recorded Word_Map and map data structure type that is data field.By doing this,I solve the problem.

3 RESULT

3.1 Test Cases

Firstly, I tested all methods of my File_Map and Word map class seperately.All of working true.Then read all files and then print all results on screen.Reading files processed true. After that, I print word_map iterator on screen.All words and their files and indexes is true.Finally,I printed TFIDF and bi-grams results and they are the same with output of this homework.Briefly,I tested all classes and steps.Output of homework and my output is the same.So program is working true.

3.2 Time Complexity Calculations:

FİLE_MAP CLASS:

1) **put (Object key, Object value) Method :** Time complexity of this method is O(n).If statement in this method has constant time complexity because of add in end of arraylist method of arraylist, but else statement in this method "n" time complexity because of indexOf method of arraylist.So total time complexiy 1/2+ ½(n) =>O(n).
2) **get(Object key) Method:**Time complexity of this method is O(n).This method includes indexOf method of arraylist and this method has "n" time complexity.So this method has O(n) time complexity.
3) **size() Method :** Time compexity of this method is O(1).This method include size method of arraylist and this method has constant time complexity so, also this method has constant time complexity.
4) **isEmpty() Method**:Time complexity of this method is O(1).This method include size method of arraylist that is constant time complexity.So also,this method has constant time complexity.
5) **remove(Object key) Method**:Time complexity of this method is O(n).This method include remove function of arraylist and because of shifting,remove function of arraylist O(n) time complexity.And also indexOf method of araylist has O(n) time complexity.So time complexiy is O(n).
6) **containsKey(Object key) Method**:Time complexity of this method is O(n).Because there is traversing all elements in while loop and this has n time.
7) **contaşnsValue(Object key) Method**:Time complexity of this methos is O(n). Because there is traversing all elements in while loop and this has n time.
8) **putAll(Map m) Method:**This method has O(n*m) time compexity.Because two map is combining and while loop is ends size of m.And in while loop there is a get method of arraylist that is n time.So total time complexity O(n*m).
9) **keySet() Method:** This method has O(n) time complexity.Because all elements in arraylist added a set.This operation has O(n) time complexity.
10) **values() Method:**This method has O(n) time complexity.Because all elements in arraylist added a collection.This operation has O(n) time complexity.

11) **clear() Method:** methos has O(1) time complexity.Because clear method of arraylist has O(1) time.

12) **entrySet() Method:** This method has O(n) time complexity.Because there is a while loop .

MYENTRY CLASS:

All methods of this class has O(1) time complexity.Because there is only initializing,assigning and returning operations.All of these operations has constant time complexity.

Word_Map Class

1)hashCode(String val) Method : This method has O(1) time complexity.Because there is only hash code of string is taken and this is constant time.

2)location(Object key) Method: This method finds index of a key according to linear probing.This method has O(1) best case.If index of key is empty location,it place there.But worst case is O(n) time complexity because if table is full, traverse all table in while loop.

3)reHashing() Method: This method has $O(n^2)$ time complexity.Because all elements again placing in table after rehash.And put method is n time complexity.Also there is a while loop.So total complexity O(n*n).

4)put(Object key,Object value) : Average case of this method is O(n).get method is using in this method and it has O(n) time compexity.But worst case is $O(n^2)$.If rehashing is need and this method has $O(n^2)$,also put method has $O(n^2)$.

5)putAll(Map m) Method: This method has O(n*m) time compexity.Because two map is combining and while loop is ends size of m.And in while loop there is a get method that is n time.So total time complexity O(n*m).

6)keySet() Method: This method has O(n) time complexity.Because all elements in arraylist added a set.This operation has O(n) time complexity.

7)values() Method: This method has O(n) time complexity.Because all elements added in a collection.This operation has O(n) time complexity.

8)clear() Method: This method has O(n) time complexity.Because there is a while loop to traverse all elements.

9)containsKey(Object key): This method has O(n) time complexity.Because find index of element is table has average O(n) time complexity.

10)containsValue(Object value) Method: This method has O(n) time complexity.Because there is while loop to traverse nodes and this is O(n) time compexity.

11)size() Method: This method has O(1) time complexity.Because only returns size.

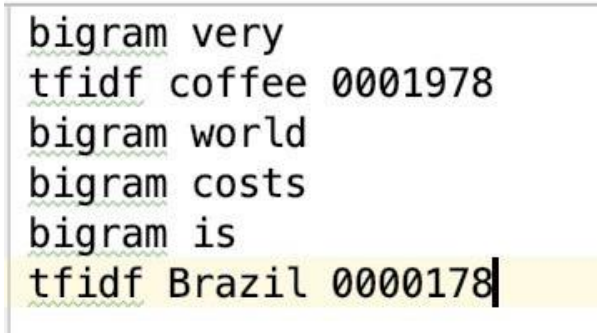
12)isEmpty() Method: This has O(1) time complexity.Because returns boolean result according to only size.

13)iterator() Method: This method has O(1) time complexity.Because only returns iterator over elements .

INNER NODE AND WorldMapIterator CLASS

All methods of this classes has O(1) time complexity.Because there is only initializing,assigning and returning operations.All of these operations has constant time complexity.

3.3 Running Results



[very attractive, very soon, very vulnerable, very difficult, very promising, very rapid, very aggressive]

0.0048781727

[world coffee, world bank, world share, world market, world cocoa, world markets, world as, world for, world made, world prices, world price, world grain, world tin]

[costs Transport, costs of, costs and, costs have]

[is high, is likely, is unlikely, is expected, is 112, is insufficient, is wrong, is unrealistic, is put, is currently, is still, is due, is one, is the, is to, is he, is not, is an, is no, is apparent, is set, is possible, is unchanged, is a, is now, is sending, is more, is keen, is also, is ending, is getting, is insisting, is unfair, is are, is precisely, is great, is beginning, is foreseeable, is trimming, is Muda, is improving, is willing, is proposing, is fairly, is some, is harvested, is trying, is heading, is imperative, is projected, is going, is very, is passed, is difficult, is flowering, is searching, is estimated, is being, is showing, is helping, is it, is often, is why, is planned, is meeting, is time, is keeping, is too, is defining, is sold, is uncertain, is down, is after, is aimed, is slightly, is forecast, is at, is caused, is depending, is committed, is faced, is in, is basically, is affecting, is downward, is sceptical, is how, is favourable, is only, is that, is well, is scheduled, is open, is concerned]

0.0073839487

```
bigram up
tfidf producers 0008477
bigram see
bigram faith
bigram for
tfidf Teck 0008304
tfidf June 0002096
```

[up stocks, up on, up with, up arrears, up to, up from, up quota, up in, up the, up only, up for, up earlier, up market, up any, up of, up some, up operating]

0.0127343

[see no, see quotas, see if, see prices, see 90, see a, see chances, see little, see any, see attitude]

[faith in, faith Nicaraguan]

[for April, for a, for 1987, for agreement, for international, for 20, for export, for Brazil, for last, for new, for some, for the, for coffee, for 95, for moving, for reintroducing, for 12, for this, for exporters, for an, for volume, for upcoming, for many, for instance, for establishing, for its, for confirmation, for May, for consultation, for spot, for about, for their, for payment, for In, for two, for other, for devaluation, for such, for subsequent, for delivery, for winning, for larger, for april, for calculating, for pay, for basing, for high, for MayJune, for quotas, for expected, for farmers, for reaching, for 1900, for difficult, for net, for Managua, for reimposed, for Colombian, for similar, for Shearson, for rain, for itself, for up, for sales, for Latin, for noncommunist, for crude, for less, for just, for Soviet, for steel, for goods, for Indian, for Thursday, for Saturday, for oil, for March, for future, for them, for September, for 55, for greater, for 676, for meetings, for industrialized, for repaying, for exports, for soluble, for calendar, for agriculture, for prices, for another, for 10, for todays, for renewed, for continued, for International, for PrudentialBache, for storage, for it, for 1986, for Colombia, for 198788, for three, for buffer, for sugar, for discussions, for wheat, for commodity, for lead, for commodities, for economically, for more, for intellectual, for aid, for industrialised, for one, for sabotaging, for shares, for 60, for Africa, for Africas, for 128]

0.0737546

0.0151505135

```
bigram just
tfidf producers 0008477
bigram International
bigram chronic
bigram it
tfidf the 0000178
tfidf bag 0007709
```

[just 125, just as, just over, just completed, just five, just 10, just waiting]

0.0127343

[International Coffee, International Monetary, International Petroleum, International Merchant, International Services, International Tin, International Cocoa, International Natural, International agreements]

[chronic shortage]

[it will, it was, it out, it did, it could, it to, it said, it isnt, it conducts, it reopens, it is, it doesnt, it has, it can, it Cardenas, it pressured, it became, it would, it more, it remains, it looks, it does, it had, it no, it cut, it as, it proves, it difficult, it impossible, it held, it little, it fell, it one, it A, it US, it considered, it reflects, it keeps]

0.0013518701

0.024239343

Process finished with exit code 0

```
bigram said
```

[said He, said the, said Colombia, said lower, said was, said The, said Delegates, said India, said Export, said assuming, said Brazils, said Near, said Producer, said that, said Some, said With, said Coffee, said Guatemala, said more, said unless, said They, said Producers, said Zimbabwean, said Administrative, said Marketing, said at, said According, said Distribution, said Resumption, said Kenyas, said This, said Agriculture, said Private, said in, said Recent, said an, said We, said ageing, said Earlier, said no, said High, said Ouko, said exports, said they, said registrations, said is, said separately, said Another, said he, said yesterday, said according, said Tempers, said Their, said If, said Retail, said No, said ICO, said from, said There, said adding, said Uganda, said Customs, said Malaba, said However, said Jacobs, said world, said stagnating, said producers, said IBC, said it, said supplies, said many, said Further, said Gold, said Costa, said on, said At, said Escalante, said Speaking, said export, said Traders, said this, said a, said lows, said prospects, said today, said 126, said Indonesia, said Latin, said if, said Indonesias, said stocks, said Especially, said to, said Jon, said one, said Brazil, said El, said Within, said West, said shade, said Exports, said production, said Opinions, said A, said In, said high, said It, said Carlos, said Dauster, said Thailand, said would, said Buyers, said after, said there, said without, said Questioned, said President, said William, said Debra, said by, said Sandra, said picking, said Machakos, said Commenting, said consumer, said these, said except, said Prices, said Withdrawals, said Countertrade, said Total, said One, said Early, said Illustrating, said Nicaragua, said objective, said All, said Other, said quotas, said On, said Colombian, said Cattle, said Arango, said Colombias, said Madagascars, said Madagascar, said Imports, said cumulative, said Were, said UIC, said Teck, said several, said Talks, said efforts, said have, said his, said over, said Since, said Major, said That, said Roldan, said commodities, said New, said Tommy, said But, said Justo, said Peru, said flood, said Peruvian, said Yesterdays, said Vietnam, said Quotas, said Zimbabwe, said Given, said Consumers, said differences, said US, said Canadian, said UK, said Japan, said France, said consuming, said when, said Bomani, said Average, said although, said An, said Growing, said exporter, said THE, said those]

Process finished with exit code 0

Output of printWordMap function in NLP that prints word_map by using iterator :

```
Word :shelf
TextFile: 0000165 Index :[398]
TextFile: 0000527 Index :[159]

Word :likely
TextFile: 0000165 Index :[401]
TextFile: 0000167 Index :[95, 380]
TextFile: 0000178 Index :[50]
TextFile: 0000194 Index :[318]
TextFile: 0000402 Index :[117]
TextFile: 0000527 Index :[435]
TextFile: 0001184 Index :[217]
TextFile: 0001234 Index :[137]
TextFile: 0003805 Index :[391]
TextFile: 0007218 Index :[152]
TextFile: 0007320 Index :[152]
TextFile: 0007709 Index :[31, 115, 237]
TextFile: 0007851 Index :[63]
TextFile: 0007915 Index :[42, 152]
TextFile: 0008304 Index :[33]
TextFile: 0008477 Index :[494, 532]
TextFile: 0008656 Index :[196]

Word :sinnificantly
TextFile: 0000165 Index :[404]

Word :price
TextFile: 0000165 Index :[392]
TextFile: 0000527 Index :[136]
TextFile: 0000639 Index :[121, 172]
TextFile: 0000764 Index :[169]
TextFile: 0000828 Index :[60, 92, 145]
TextFile: 0001180 Index :[111]
TextFile: 0001184 Index :[8, 268, 291, 355, 406]
TextFile: 0001562 Index :[120]
TextFile: 0001603 Index :[878]
TextFile: 0002972 Index :[233, 377]
TextFile: 0003195 Index :[34, 97, 449]
TextFile: 0003558 Index :[158, 171]
TextFile: 0003805 Index :[122, 164]
TextFile: 0005750 Index :[361]
TextFile: 0007485 Index :[84]
TextFile: 0007709 Index :[41, 149]
TextFile: 0007882 Index :[11, 40, 103, 233, 597, 657, 705, 869, 1076, 1094]
TextFile: 0008051 Index :[29]
TextFile: 0008364 Index :[176]
TextFile: 0008425 Index :[152]
TextFile: 0008477 Index :[208, 520]
TextFile: 0008656 Index :[117, 238]
TextFile: 0008777 Index :[43]
TextFile: 0008792 Index :[44]
```

```
Word :sources
TextFile: 0000165 Index :[409]
TextFile: 0000527 Index :[34, 147]
TextFile: 0000639 Index :[267]
TextFile: 0000764 Index :[151]
TextFile: 0000828 Index :[275]
TextFile: 0001004 Index :[569]
TextFile: 0001234 Index :[75, 110]
TextFile: 0001671 Index :[26]
TextFile: 0001978 Index :[147]
TextFile: 0002732 Index :[126]
TextFile: 0002972 Index :[23, 353, 483, 487]
TextFile: 0003195 Index :[315]
TextFile: 0003876 Index :[312]
TextFile: 0005012 Index :[36, 62, 110, 127, 183, 211]
TextFile: 0005743 Index :[20, 93]
TextFile: 0005750 Index :[31, 156, 304, 309, 357, 433]
TextFile: 0007218 Index :[50, 175, 218, 282, 503]
TextFile: 0007320 Index :[50, 175, 218]
TextFile: 0007569 Index :[23]
TextFile: 0007678 Index :[27]
TextFile: 0007709 Index :[180, 268]
TextFile: 0007882 Index :[376]
TextFile: 0008857 Index :[23]
TextFile: 0008891 Index :[27]
```

```
Word :Sandra
TextFile: 0008656 Index :[380]
TextFile: 0008777 Index :[134]
TextFile: 0008792 Index :[134]

Word :Kaul
TextFile: 0008656 Index :[381, 394, 434, 481]
TextFile: 0008777 Index :[135]
TextFile: 0008792 Index :[135]

Word :quarterly
TextFile: 0008656 Index :[391]

Word :procure
TextFile: 0008656 Index :[399]

Word :keep
TextFile: 0008656 Index :[417]
TextFile: 0008777 Index :[189]
TextFile: 0008792 Index :[189, 369]
```

```
Word :Colombia
TextFile: 0000165 Index :[263, 318]
TextFile: 0000202 Index :[71]
TextFile: 0000527 Index :[369, 433]
TextFile: 0001004 Index :[277, 321, 435]
TextFile: 0001085 Index :[90, 141, 158, 176, 217]
TextFile: 0001180 Index :[113]
TextFile: 0001327 Index :[332]
TextFile: 0001562 Index :[72]
TextFile: 0001671 Index :[57]
TextFile: 0001916 Index :[0]
TextFile: 0002827 Index :[147, 181, 191]
TextFile: 0003195 Index :[0, 162, 179, 239, 269, 292, 371]
TextFile: 0003463 Index :[0]
TextFile: 0003558 Index :[3, 35, 98, 136]
TextFile: 0003805 Index :[39, 61, 142, 240, 291, 419]
TextFile: 0004598 Index :[108]
TextFile: 0004626 Index :[3]
TextFile: 0004842 Index :[29]
TextFile: 0004997 Index :[0, 45]
TextFile: 0005172 Index :[51]
TextFile: 0005743 Index :[43, 140]
TextFile: 0006240 Index :[56, 114, 119]
TextFile: 0007985 Index :[52]
TextFile: 0008477 Index :[378]
TextFile: 0008911 Index :[66]
```