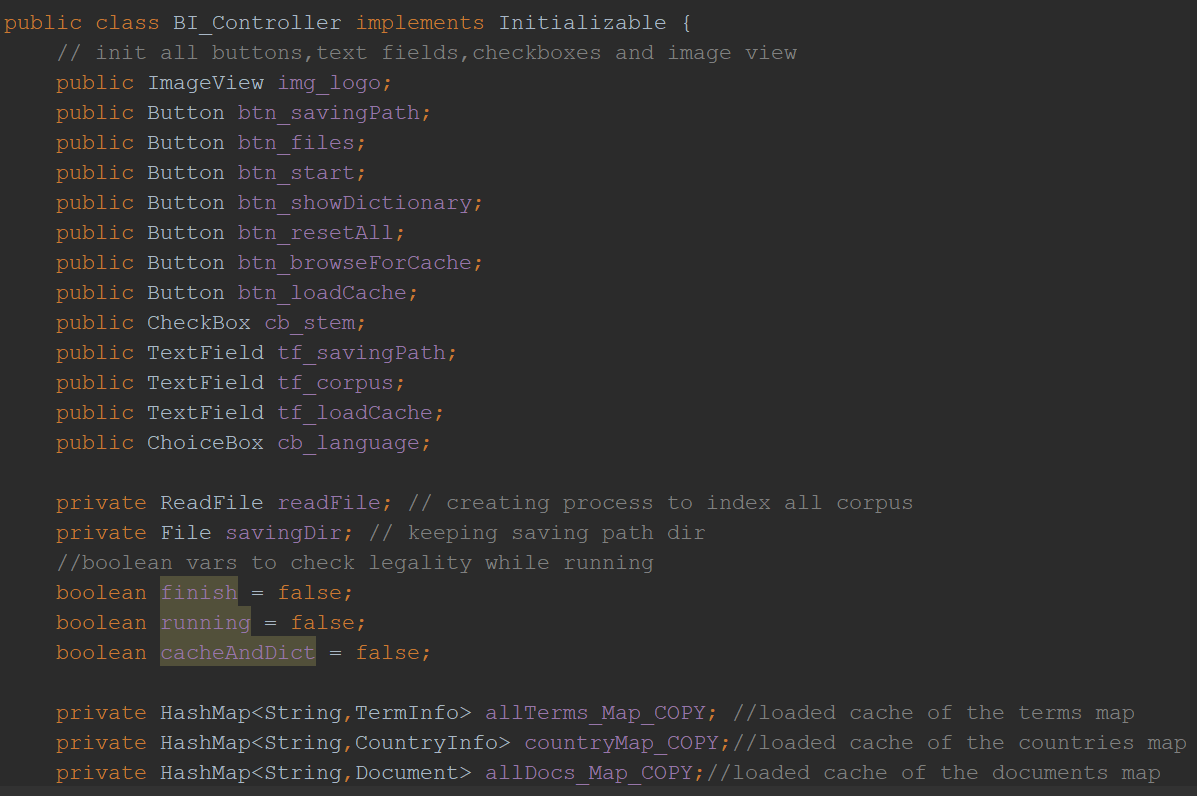
אחזור מידע – דו"ח מנוע חיפוש לאחר סיום חלק 1

מגישים: בן מועלם 308064047  
 עידן רווח 204634893

**תיאור מחלקות:**  
  
Controller Package:  
  
**• מחלקת BI-Controller:**

המחלקה שאחראית על ניהול ממשק המשתמש והתקשורת של המשתמש עם המערכת

משתנים בוליאנים אשר בודקים את תקינות ריצת ממשק המשתמש \*הסבר למטה

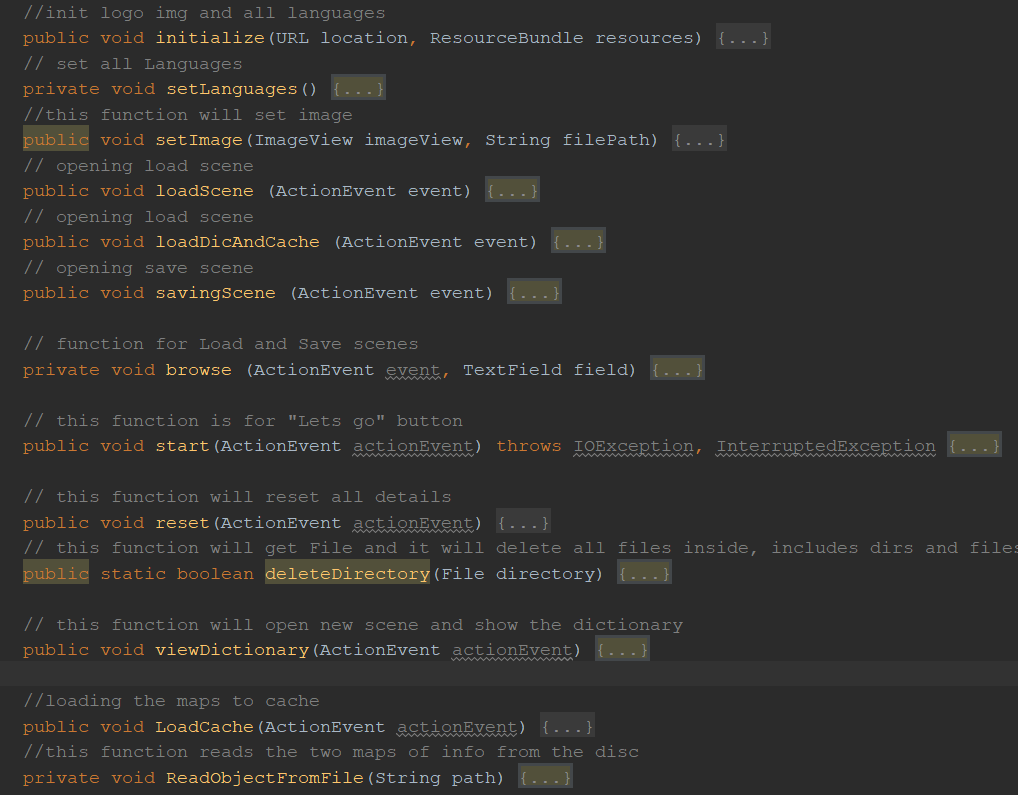
אובייקטים של התמונה, כפתורים, doStem,שורות הכתיבה ושפת המסמכים (קשורים לfxml)

המילונים של הterms ושל המדינות אשר נטענים מהזיכרון

\* יש צורך במשתנים הבוליאנים הללו מכיוון שנרצה לדוגמא למנוע מהמשתמש ללחוץ על כפתור הפעלת האינדקס (כפתור ה-Let's Go) בזמן שאנו יוצרים את האינדקס. המשתנים הנוספים מונעים לחיצות לא רצויות על הכפתורים הנוספים וכאשר הם ישתנו לערך true נדע שתהליך האינדוקס אכן הסתיים ושאפשר להמשיך בפעולות הרצויות.

**פונקציות של הController**

פונקציית אתחול חלון הממשק



לאחר יצירת האינדקס, נוכל לאפשר שפות בממשק

פונקציות לטעינת החלונית לבחירת נתיב במחשב (טעינת שני הנתיבים לשמירה וטעינה ונתיב למילון השמור)

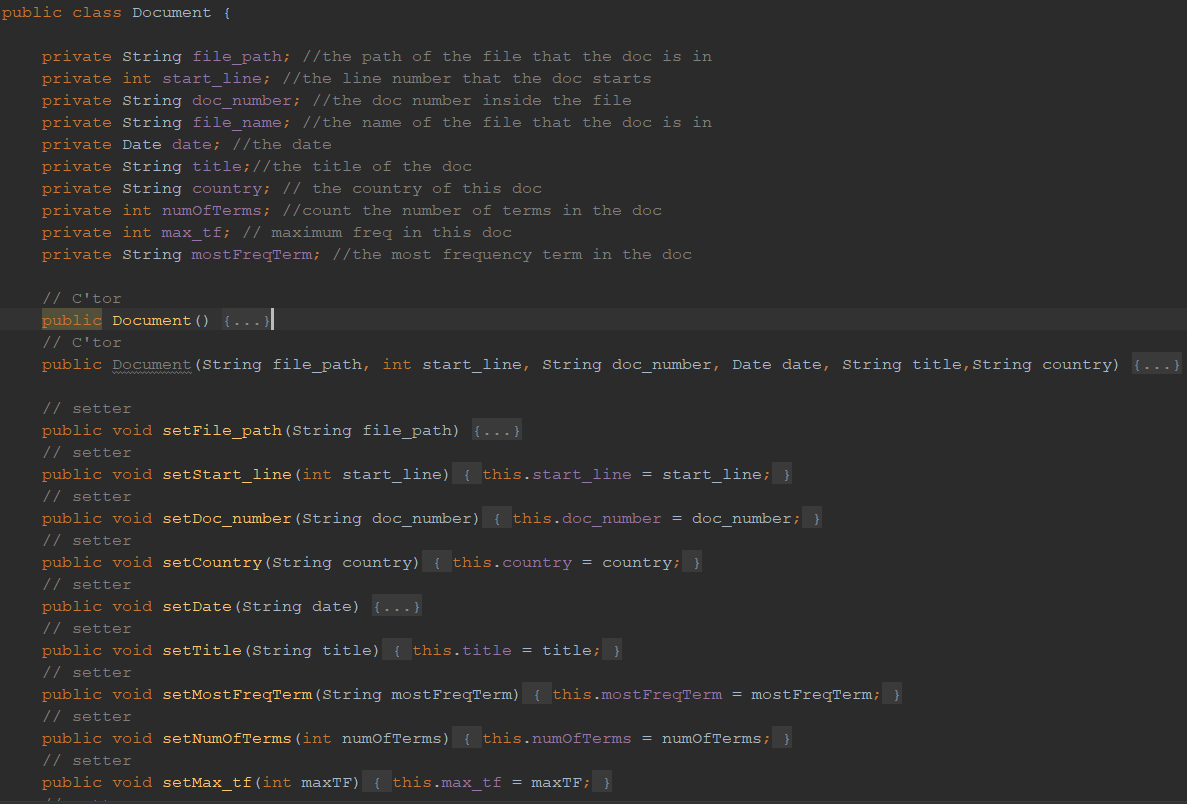
הפונקציה הראשית של הcontroller אשר מפעילה את תהליך קריאת הקבצים ויצירת האינדקס בהמשך

שאר הפונקציות הן לכל כפתור בממשק המשתמש, לדוגמא:

הפונקציה LoadCache מופעלת כאשר המשתמש לחץ על כפתור ה-load, היא בודקת את התנאי שאכן המשתמש הזין נתיב לתיבת הטקסט ולאחר מכן מעביר את בקשת הטעינה לפונקציית העזר ReadObjectFromFile שקוראת את שלושת המילונים (מילון הטרמים, מילון הערים ומילון המסמכים) מתוך קבצים השמורים בנתיב שניתן, ומעלה אותם לזיכרון הRAM.

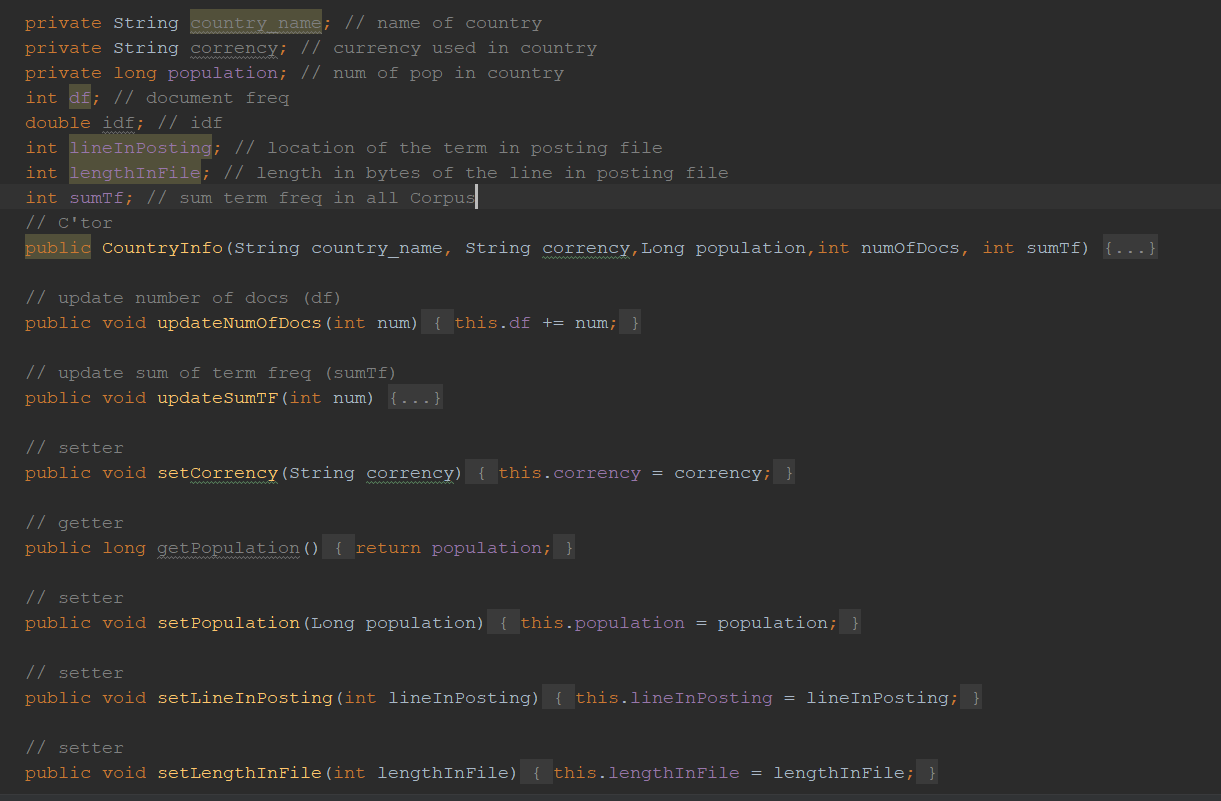
Model Package:  
  
**• מחלקת Document:**

זהו אובייקט אשר מחזיק את כל המידע לגביי המסמכים שנמצאים בקורפוס, לדוגמא שם הקובץ בו המסמך נמצא, מספר המסמך, תאריך המסמך וכ'ו

\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

הפונקציות שנמצאות באובייקט זה הן getters & setters לשדות האובייקט בלבד.

**• מחלקת CountryInfo:**  
  
זהו אובייקט אשר מחזיק את כל המידע לגביי מדינות אשר נמצאות בקורפוס

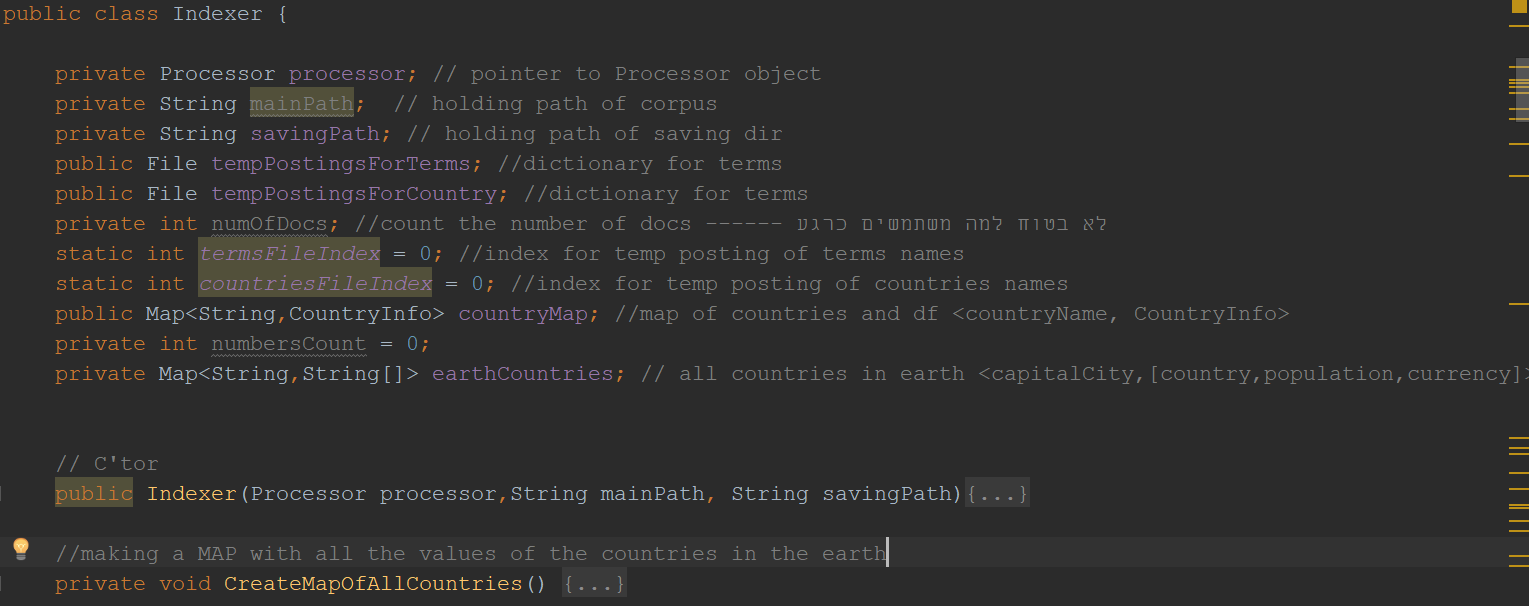
\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

כל הפונקציות שנמצאות באובייקט זה הן getters & setters

(פונקציות הupdate הן בעצם גם setters מכיוון שהן בודקות אם הערך הוא לא מינוס אחד שזה הערך ההתחלתי, ומעדכנות בהתאם)

**• מחלקת Indexer:**

מחלקה זו אחראית על ייצור האינדקס (יצירת קבצי הפוסטינג)

\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

termsFileIndex & countriesFileIndex – שני אינדקסים אשר מחזיקים את מספר הקובץ הבא לפוסטינג הזמני של אותו סוג קובץ (טרם/מדינה)

countryMap – מפה המחזיקה זוגות של <שם העיר, המידע על העיר>. היא מתמלאת במהלך ריצת הindexing כך שבכל סבב איטרציות בודקים אם המדינה קיימת במפה ואם כן מעדכנים לה את המידע הנדרש.

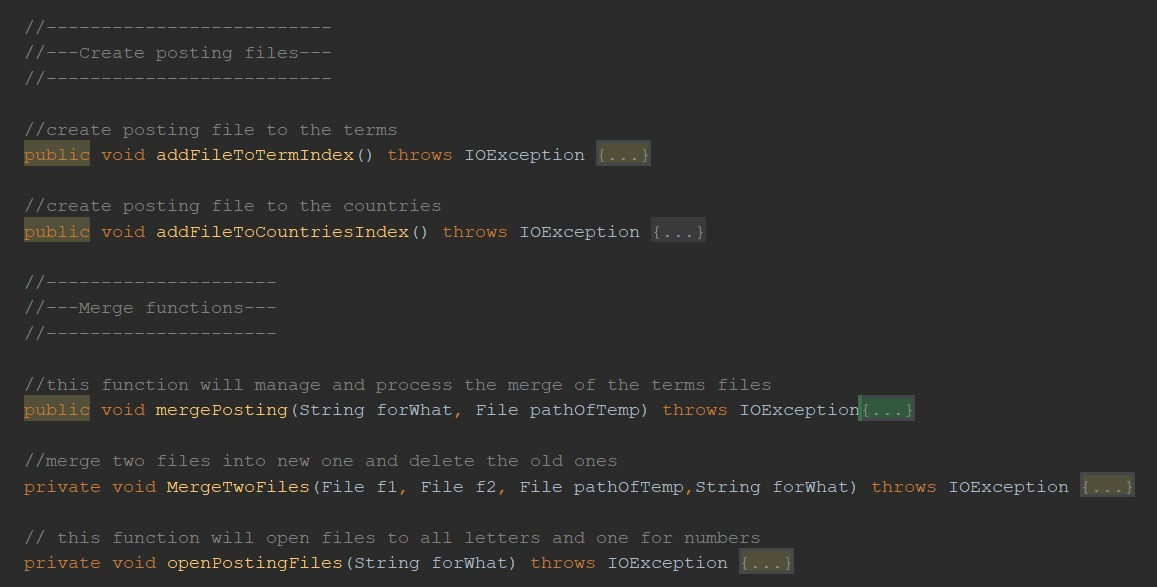
earthCountries – מפה המחזיקה את שם העיר מול מערך של [שם מדינה, מטבע, גודל האוכלוסיה]. מפה זאת בעצם נקראת מתוך קובץ הjson שנמצא לנו בתיקיית ה Resources (תהליך זה קורה בפונקציה האחרונה בתמונה לעיל) ושולפת את המידע על המדינות בעולם ובכך נוכל לבדוק על כל עיר אם יש לנו מידע נדרש כמו מטבע וגודל אוכלוסייה ולצמד אותם ביחד.

תהליך קבלת מידע על ערים:

• אם עיר מסויימת שקיבלנו מהמסמכים נמצאת במפה earthCountries, נשלוף ממנה את המידע לגביי המדינה של העיר, גודל האוכלוסייה והמטבע המקומי. (אם העיר נמצאה במפתח של המפה ניתן להבין מכך שעיר זו היא למעשה עיר בירה מכיוון שמפה זו מחזיקה ערי בירה בלבד.)

• אם העיר לא נמצאה בearthCountries, ניגש לAPI נוסף (בעזרת התחברות לurl של האתר <http://getcitydetails.geobytes.com/GetCityDetails?fqcn> ) ונשלוף בעזרתו את המידע לגביי אותה עיר. המידע שיוחזר אלינו הוא מסוג שJSON PARSER יודע לתפעל ולכן נשתמש בו כדי לחתוך את החלקים הרצויים לנו (שם מדינה, גודל אוכלוסייה ומטבע).

**המשך פונקציות indexer:**

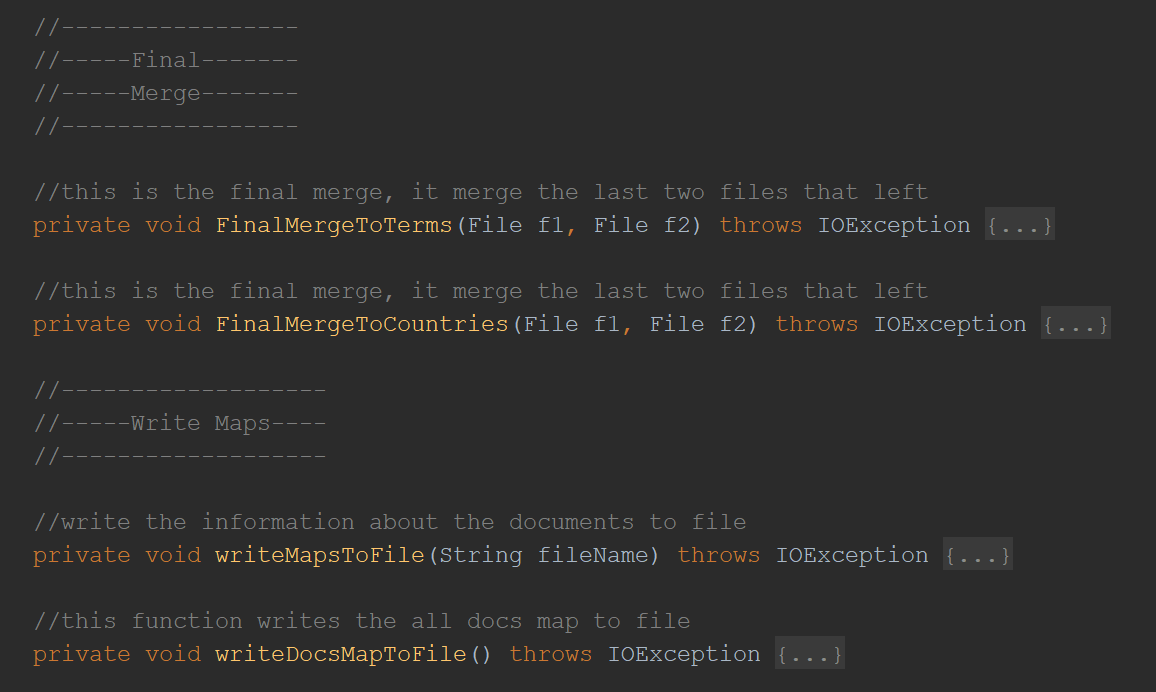


פונקציה שנקראת לאחר יצירת כל קבצי הפוסטינג הזמניים ומאתחלת את ניהול איחוד הפוסטינגים לכדי קובצי פוסטינג סופיים. הפונקציה קוראת לפונקציה mergeTwoFiles כל עוד קיימים יותר משני קבצים זמניים, ולאחר מכן כאשר נשארו בדיוק שני קבצים היא קוראת לפונקציה FinalMerge

פונקציות אשר לוקחות את המפות של הטרמים/מדינות הנוכחיות (שמתמלאות מאיטרציה של 185 קבצים) וכותבות אותן לקובץ פוסטינג זמני

פונקציה אשר מקבלת כפרמטר שני קבצים (קבצי פוסטינג זמניים) ומאחדת בינהם לכדי קובץ חדש יחיד. האיחוד מתבצע בצורה ממויינת ובנוסף הטרמים אשר זהים בשני המסמכים מתאחדים לשורה יחידה עם כמות מסמכים משותפת

פונקציה שיוצרת את קבצי הפוסטינג הסופיים (ריקים מתוכן) אשר יתמלאו בהמשך הפונקציות



הסבר לפונקציות אלו בעמוד הבא

פונקציות שכותבות את המילונים הסופיים של הטרמים, המדינות והמסמכים לתוך קובצים (כך שנוכל לטעון את המילונים בעתיד)

הסבר מעמיק לתהליך יצירת הפוסטינג:

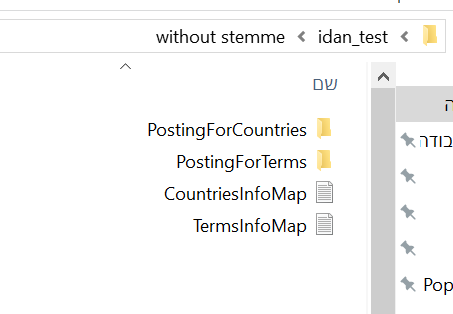
תהליך יצירת קבצי הפוסטינג זהה בשני המקרים של טרמים וערים לכן נקח לדוגמא את הטרמים עצמם (לא המדינות).

לאחר כל איטרציה של 185 קבצים מהקורפוס, המידע אשר עבר דרך מחלקת הParse מגיע למחלקת הindexer במילון המתאים ועל כל איטרציה שכזו נפתח קובץ פוסטינג זמני שלתוכו נרשמים כל הטרמים שהיו באיטרציה זו בצורה ממויינת (כולל המסמכים בהם הופיעו והמיקומים שלהם, ניתן לראות דוגמא לכך בהמשך הדו"ח).

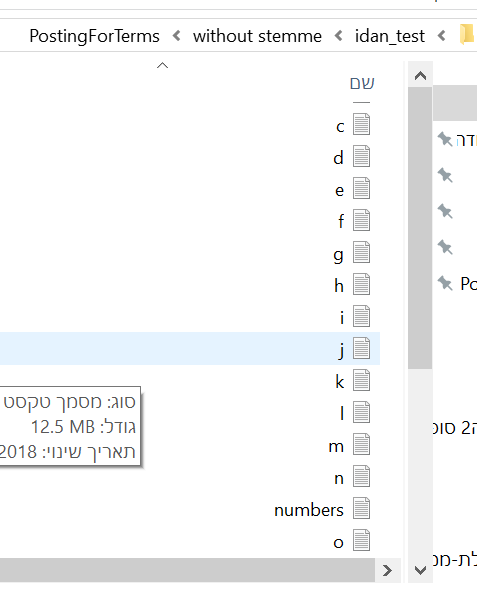
לאחר שהסתיימו כל האיטרציות (אמורות להיות 10 איטרציות – סדר גודל של 1850 קבצים בקורפוס ולכן נוצרו 10 קבצי פוסטינג זמניים) נקראת הפונקציה **mergePosting** אשר מנהלת את איחוד קבצי הפוסטינג הזמניים, היא מריצה את הפונקציה **mergeTwoFiles** אשר מקבלת כארגומנטים שני קבצי פוסטינג ומאחדת אותם ע"י כך שהיא עוברת שורה שורה על שניהם במקביל ומאחדת בצורה ממויינת. פונקציה זו נקראת כל עוד קיימים יותר משני קבצי פוסטינג זמניים, ובצורה זו אנו מאחדים בצורה מדורגת את המידע.

כאשר נשארו בדיוק שני קבצי פוסטינג זמניים (אשר מכילים את תוכן 10 הקבצים) הפונקציה **FinalMerge** נקראת. פונקציה זו מאחדת בין שני קבצי פוסטינג אלו ועל כל שורה שהיא סיימה, היא כותבת אותה לקובץ הפוסטינג הסופי המתאים לה, ע"י כך שהיא בודקת את האות הראשונה של אותו טרם (קבצי הפוסטינג הסופיים הם למעשה האותיות א"ב האנגליים וקובץ נוסף למספרים)

לבסוף, הפונקציה **writeMapsToFile** נקראת והיא רושמת את שני המילונים של המדינות/טרמים לקובץ .txt

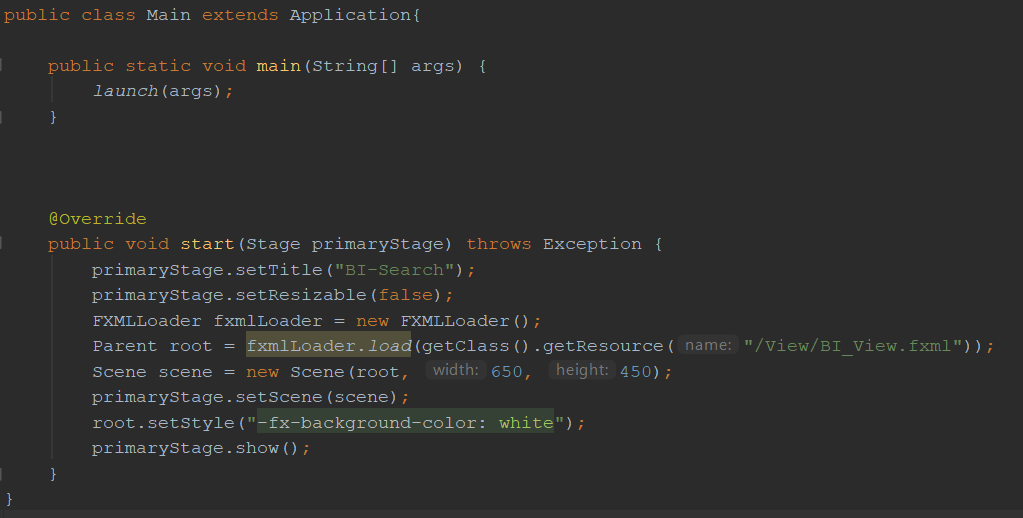


בתמונה זו ניתן לראות את תיקיות הפוסטינג הסופיות שנוצרות ובנוסף את שני קבצי המילונים:



בתמונה זו ניתן לראות את קבצי הפוסטינג הסופיים לפי ה-א"ב האנגלי (הטרמים ממוינים בתוך כל קובץ)

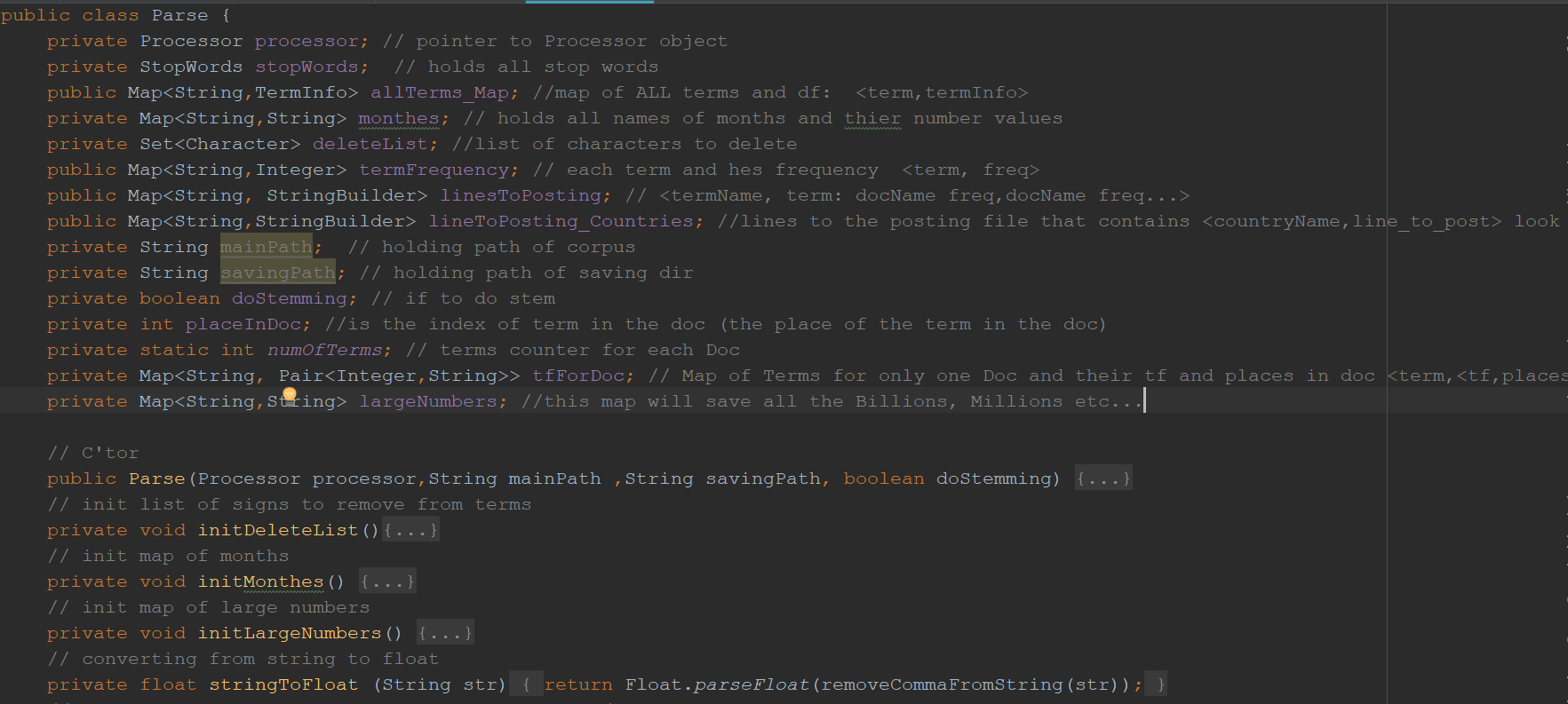
**• מחלקת Main:**

מחלקת הmain היא המחלקה הראשית בה מתחילה ריצת התוכנית. הmain קורא לפונקציה start ולמעשה משם נפתח ממשק המשתמש.

**• מחלקת Parse:**

מטרתה של מחלקה זו היא לעבור על כל הterms בכלל המסמכים ולשמור אותם כפי שהוגדר בתרגיל. טיפלנו במקרים כגון : מספרים גדולים, אותיות גדולות\קטנות, אחוזים, מחירים ותאריכים.

במחלקת Processor אנו עוברים על רשימת המסמכים ומשם שולחים כל מסמך (רק את הטקסט שלו) לפונקציה בשם “**ParseThisDoc**” במחלקת הParse.  
בפונקציה זו נעבור על כלל הterms ובעזרת פונקציות עזר רבות ננסה למצוא האם הterm שייך למקרה מסוים. אם כן, נשלח אותו לפונקציית “**UpdateTermFreq**” לפי איך שהתבקש ואחרת נשלח אותו לפונקציה איך שהוא (מקרה של term רגיל).   
מטרתה של פונקציה זו, היא לקבל term ולעדכן אותו במספר מילונים: allTermsMap, termFrequency, tfForDoc.

\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

אתחול המפות

allTerms\_Map – מפה המחזיקה את כל הטרמים שנאספים במהלך האיטרציות. מחזיקה זוג סדור של <שם הטרם, מידע על הטרם>

months – מפה המחזיקה את החודשים הלועזיים כך ש: <January, 01>, בעזרתה נוכל להמיר שמות חודשים מהטקסט למספרים.

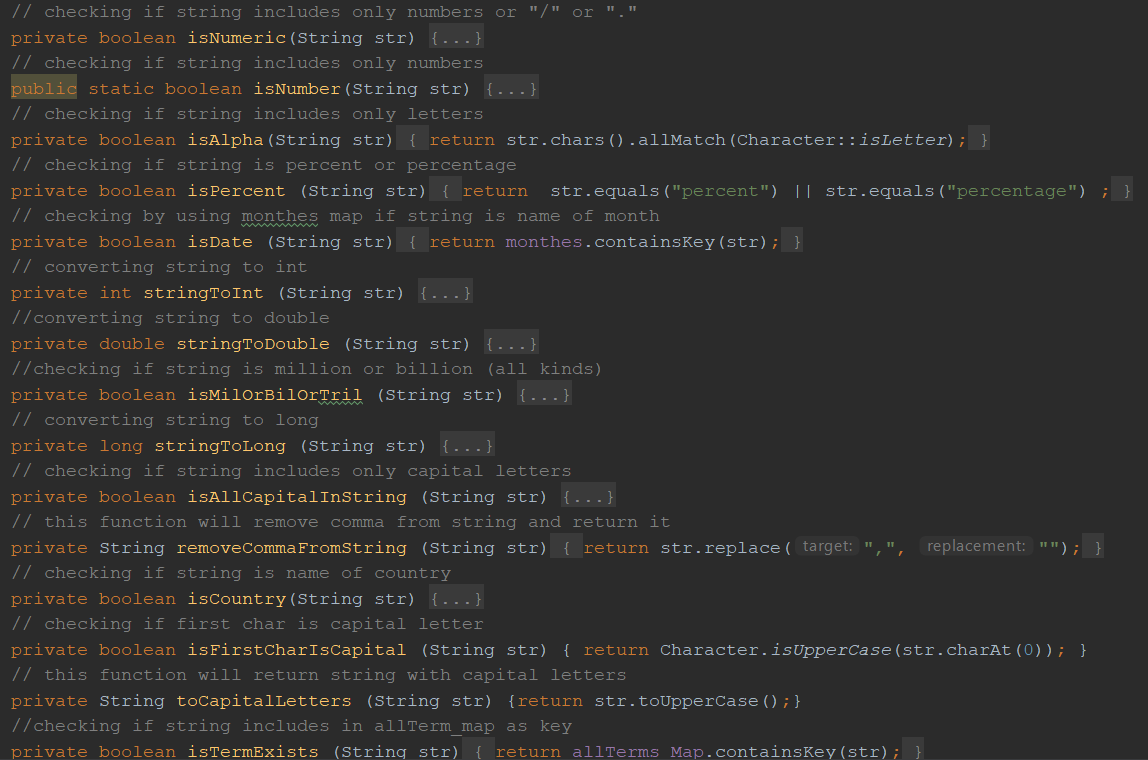
largeNumbers – בדומה לחודשים, מפה זו מחזיקה זוגות סדורים כך ש: <Billion, B>

deleteList – סט של char שלדעתנו "מפריעים" לטרם עצמו לדוגמא הסימנים הבאים: "," ";" "~" "—" "^" וכ'ו

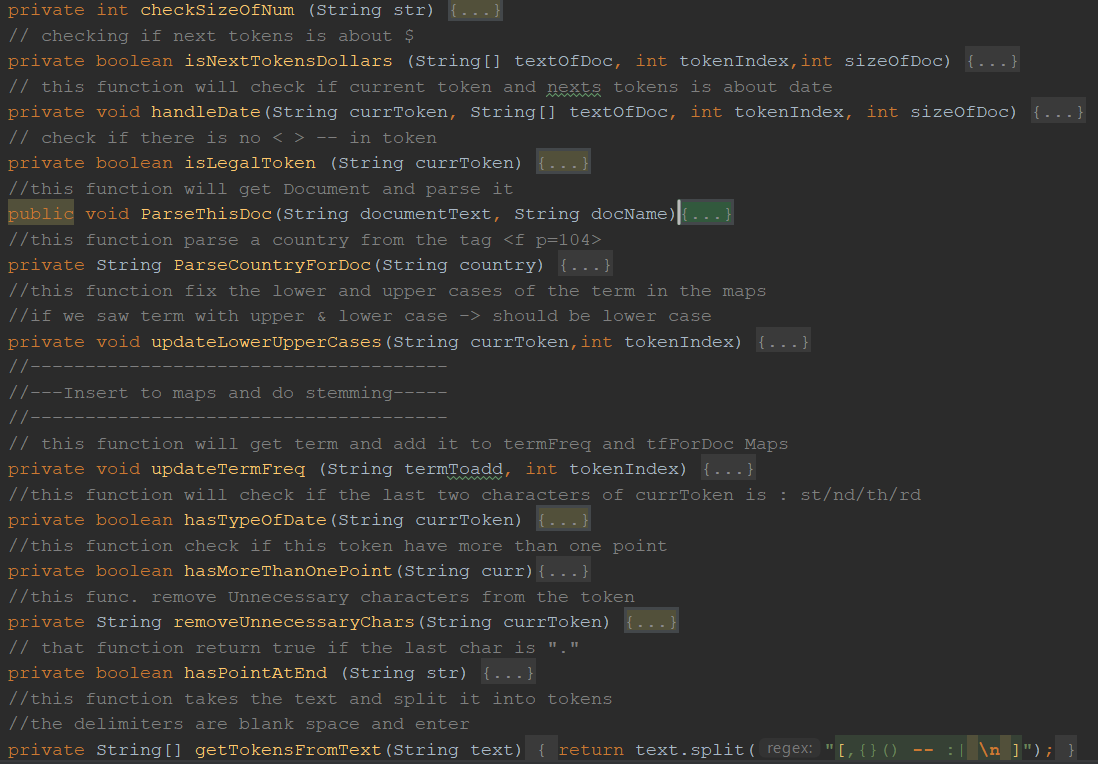
termFrequency – מפה של <שם הטרם, התדירות שלו>, מפה זו מאותחלת בכל סבב איטרציות.

lineToPosting & lineToPosting\_Countries – מפות המחזיקות זוג סדור של < שם הטרם, השורה שתירשם לפוסטינג>, מפות אלו עוזרות לנו בכך שאנו מעדכנים אותן בכל פעם שמופיע הטרם במסמך כלשהו. (ניתן לראות כיצד נראית שורה בפוסטינג בהמשך).

tfForDoc – מפה אשר מחזיקה את התדירות של טרם במסמך הספציפי (מפה זו מתאפסת בכל מסמך חדש). בנוסף לתדירות של הטרם במסמך, המפה מחזיקה את המיקומים המדוייקים של הטרם באותו מסמך (ניתן לראות את שורת הפוסטינג שמופיעה בהמשך הדו"ח). משפה של זוג סדור שהvalue הוא זוג סדור: <שם טרם, <תדירות,מיקומים במסמך>>

\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

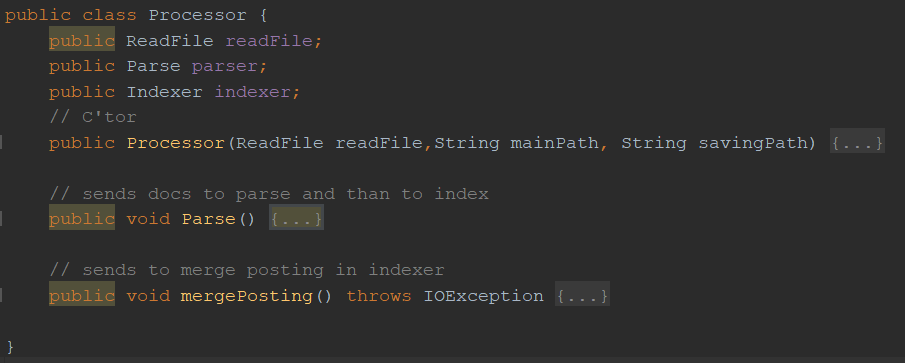
פונקציות עזר לביצוע תהליך הparse



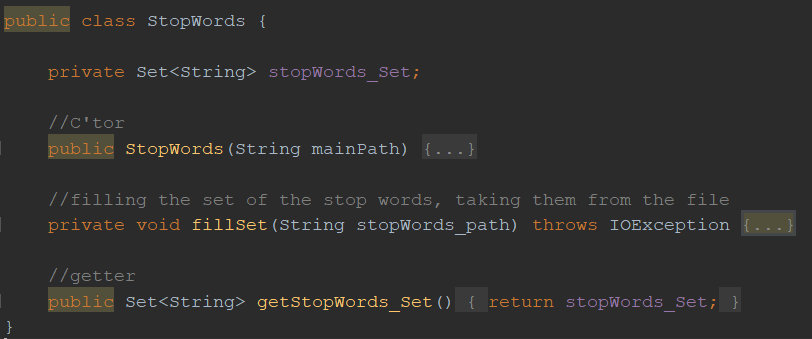
פונקציה שבודקת את תקינות העיר של המסמך ומורידה ממנה סימנים מיותרים. אם היא מספר, ה"עיר" נזרקת

הפונקציה שמעדכנת את המפות עם הטרם החדש

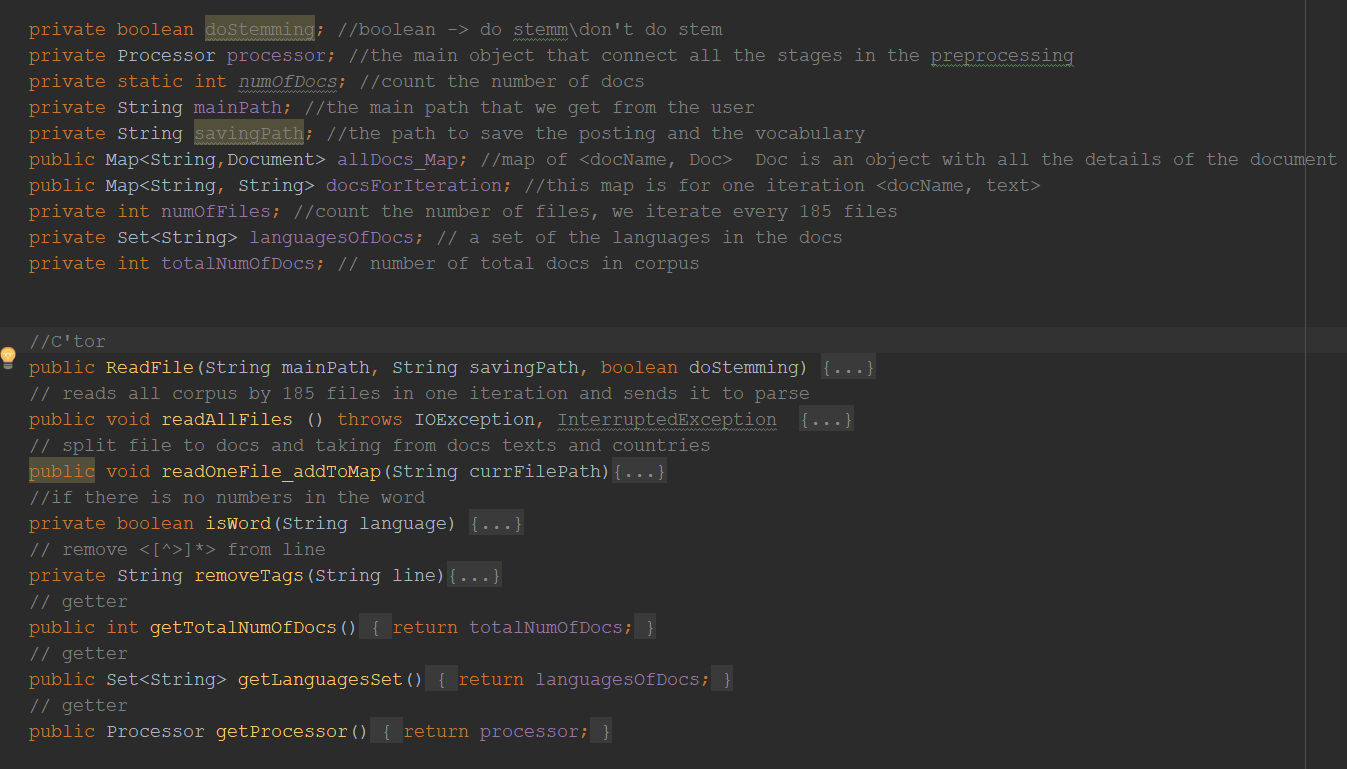
הפונקציה הראשית שמבצעת את הParse

**• מחלקת Processor:**מחלקה זו נבנתה לשם ניהול כל תהליך קריאת הקבצים, מעבר על הטרמים (Parse) ושליחה לאינדוקס. למעשה לאחר שהread file מסיים לקרוא 185 קבצים, מחלקת הprocessor יודעת לעבור לפונקציה parse שנמצאת בParser וכאשר תהליך זה מסתיים המידע עובר למחלקת הindexer ולבסוף לאיחוד קבצי הפוסטינג.

**• מחלקת StopWords:**

****מחלקה זו אחראית על ניהול הstop words, כאשר נקראת מחלקת הparse היא למעשה קוראת למחלקה זו. היא מקבלת בבנאי את הנתיב בו נמצא קובץ הstop words, שולפת ממנו את המידע ומכניסה אותו לתוך השדה מסוג set שיש לה (תהליך זה מתבצע בפונקציה fillSet).

**• מחלקת ReadFile:**  
  
מחלקה זו אחראית על קריאת הקבצים ומעבר על המסמכים שבתוך כל קובץ.

  
\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

פונקציות עזר, לדוגמא removeTags מקבלת סטרינג ומורידה ממנו את התגית פתיחה ותגית סיום

docsForIteration – מפה המחזיקה את <שם המסמך, הסטרינג תחת התגית טקסט> של קבוצת המסמכים שנמצאים באותה איטרציה של 185 קבצים. מפה זו מועברת ישירות לParse.

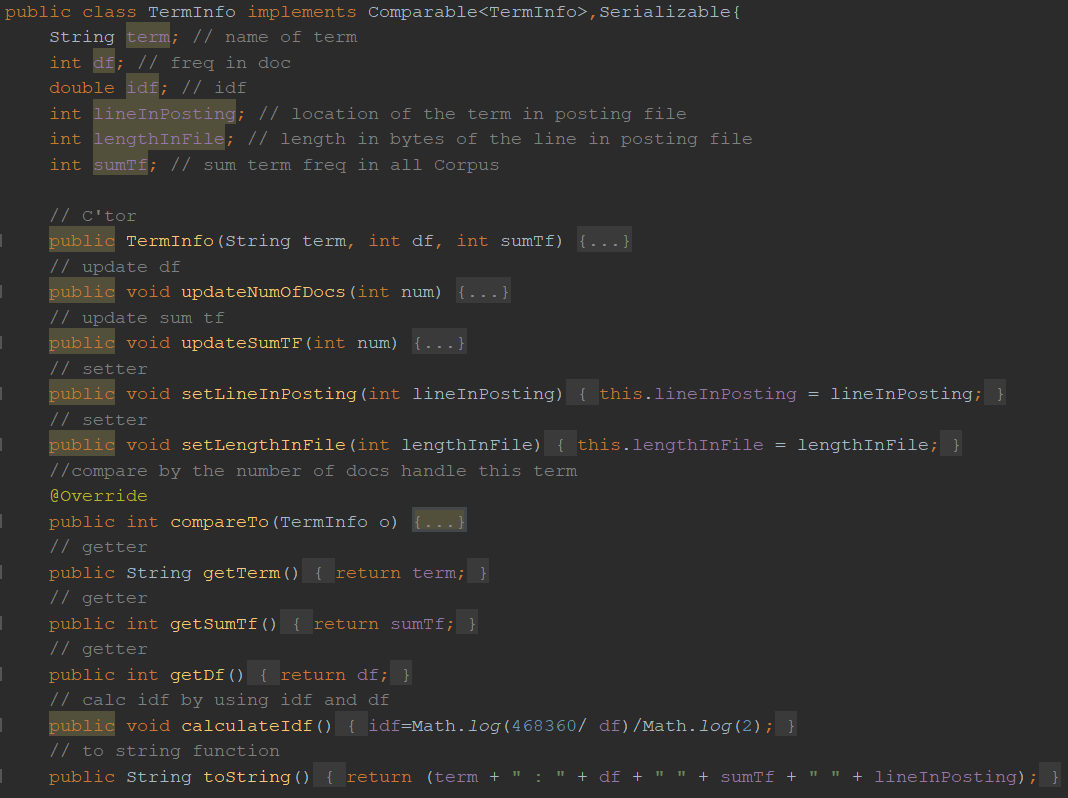
languagesOfDocs – set המחזיק את כל השפות בקורפוס שנמצאות תחת התגית <F P=105> , לאחר סיום ריצה ויצירת האינדקסים השות מועברות לממשק המשתמש ומוצגות לו.

readAllFiles – פונקציה אשר מנהלת את הריצה על כל התקיות בקורפוס ומבצעת את האיטרציות של 185 קבצים בכל סבב.

readOneFile\_addToMap – פונקציה אשר נקראת מהפונקציה הראשית (readAllFiles) ומבצעת סיווג של המידע שנמצא באותו קובץ (מפרידה לפי מסמכים -> לכל מסמך מפרידה את איזור הטקסט, שולפת את השפה, שם המסמך, עיר, וכותרת ומוסיפה אותם למקומות המתאימים)

**• מחלקת TermInfo:**

זהו אובייקט אשר מחזיק את כל המידע לגביי טרמים אשר נמצאים בקורפוס. מילון האינדקס מחזיק <term name, TermInfo>

****\* הסברים למשתנים והפונקציות נמצאים תחת ההערות בתמונה

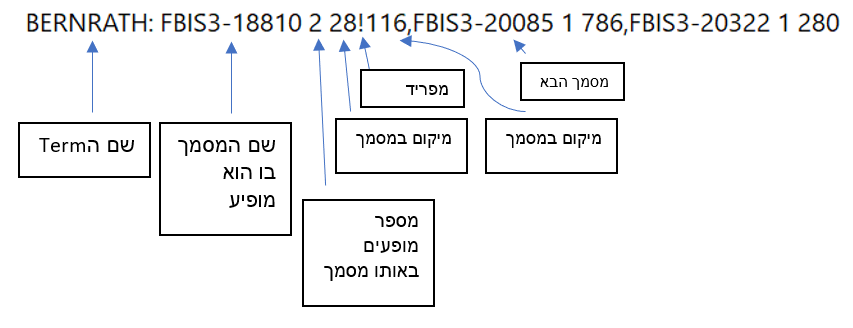
**• מחלקת Stemmer:**  
נלקחה כמו שהיא מהקובץ שנשלח.

View Package:

מחזיקה את קובץ הfxml של הפרויקט, נקראת “BI\_View.fxml”.

- בכל הנוגע לקבצי הPosting, אנו בחרנו לעבוד באיטרציות של 185 קבצים.  
כלומר, אנו קוראים 185 קבצים במחלקת הReadfile-, לאחר מכן נבצע על אותם קבצים Parse אשר מממש את התכונות שהתבקשנו לבדוק ויוצר בהמשך מפות של terms ומדינות לאותה איטרציה .

לאחר מכן ניצור temp Posting File על פי המילונים שיצרנו בשלב הParse, ומכאן נחזור חלילה עד אשר יגמרו הקבצים במאגר.

בחרנו לחלק לאיטרציות כאלו מכיוון שזיכרון המחשב הינו מוגבל ואינו יכול לשאת את קריאה ושמירה של המידע מתוך המאגר בשלב אחד בלבד. לאחר ניסויים ובדיקות על המנוע שבנינו ועל הקורפוס שקיבלנו, ומצאנו כי עבור איטרציה של 185 קבצים נקבל זמן ריצה אופטימלי מול החזקת כמות זיכרון אפשרית של משאבי המחשב. לכן, עבור הCorpus הנוכחי, אנו יוצרים 10 קבצי טקסט זמניים הממוספרים מ0 עד 9 שכל אחד מהם ממויין ומוצג לפי הדוגמה הבאה:

בנוסף, בקבצי הפוסטינג של הערים הוספנו חוק שלנו שכאשר יש מדינה שהופיעה במסמך מסויים אך ורק בתגית <F P=104> , העיר תופיעה בפוסטינג עם שם המסמך ו- tf = 0



במסמך זה העיר איסטנבול הופיעה בתגית בלבד

- בחרנו להוסיף לקבצי הפוסטינג את המידע הבא:

• שם הטרם – כדי לדעת איזה טרם מחזיק את המידע שנמצא בהמשך השורה

• שם מסמך – בכדי לדעת את המסמך המחזיק באותו הטרם

• tf לאותו מסמך – בכדי לדעת כמה פעמים חזרה המילה באותו המסמך (חוזקת המילה במסמך)

• מיקום המילה במסמך – בכדי לדעת בדיוק איפה המילה נמצאת באותו מסמך (ניתן להעזר במידע זה בזמן האחזור כאשר נרצה לדעת אם שתי מילים קרובות אחת לשניה בטקסט עצמו)

**- פירוט על 2 הרחבות חוקים למחלקת Parse שהוספנו:**

1) במידה ובמסמך נמצא “September 80” נדאג לשמור את הterm כ “1980-9”. כלומר, אם נקבל שם/מספר של חודש ולאחריו מספר שנע בין 31 ל-99, נוכל להסיק כי מדובר בחודש בשנה מסוימת בין 1931-1999. מימשנו זאת ע"י שימוש במילון אשר יחזיק כמפתח את שמות החודשים (עם אותיות גדולות וקטנות) והערך הוא המספר המסמל את החודש. דרך מילון זה ודרך בדיקה שערכו של המספר הוא בין 31 ל99, שרשרנו את 2 הערכים יחד לפי הסדר הנכון והוספנו "19" לפני המספר המסמל את השנה.

2) במידה ובמסמך נמצא “4th of July” נדאג לשמור את הterm כ"4-4".   
כלומר, אם נזהה מספר שנע בטווח הימים בשנה ולאחריו st\nd\th\rd ולאחריו of ולאחריו חודש, נדע כי מדובר בתאריך. מימשנו זאת ע"י פונקציית עזר שנקראת “hasTypeOfDate” שמטרתה לבדוק כי המשתנה שקיבלה מכיל מספר יום בחודש ולאחריו st\nd\th\rd (מחזירה true במידה וכן). בנוסף לפונקציה זו, בדקנו כי גם המילה הבאה במסמך היא of ואז גם לאחריה מדובר בחודש. במידה וכל התנאים מתקיימים נוכל לשמור את רצף המילים האלו כterm יחיד שמסמל תאריך.  
  
• בנוסף הוספנו חוק אשר מוריד מterm את כל הסימנים המיותרים שנצמדו אליו (ניתן לראות את הפונקציה בתמונות שצורפות בתחילת הדו"ח) – סימנים שהיו מיותרים לדעתנו הם set של char . לדוגמא כמה מהסימנים: "~" "," "#" "^" "—" "\" ";" וכ'ו. הרעיון מאחורי חוק זה הוא למנוע מסימנים מיותרים להיצמד לעיקר המילה (ובכך ליצור טרם חדש ולא נחוץ עם סימנים לידו). מימשנו זאת באמצעות פונקציה אשר רצה בלולאה כל עוד קיימים עדיין סימנים כלשהם מהset הנתון, ואם קיימים למחוק אותם.

קוד נוסף שהשתמשנו בו במהלך בניית המנוע:

1. במחלקת האינדקסר השתמשנו בקובץ countries.json , קובץ זה מכיל מידע לגביי מדינות בעולם. שלפנו קובץ זה דרך האתר <https://restcountries.eu> אשר מספק שירותי מידע לגביי מדינות.

ביצענו קריאה של הקובץ הנתון בעזרת json של חברת גוגל אשר נלקח מאתר <https://mvnrepository.com/repos/central> והוספנו אותו לפרוייקט בעזרת המייבן. אובייקט הjson יודע לקרוא קבצים מסוג סיומת .json ובכך אנו שולפים את המידע שנחוץ לנו מתוך הקובץ. לבסוף שמרנו offline קובץ שמכיל את כל המידע הנחוץ לנו על ערי הבירה והמדינות שלהן וכאשר אנחנו מגיעים לשלב האינדקס אנו קוראים את האובייקט מפה מתוך הקובץ שהכנו מראש.

1. השתמשנו בAPI נוסף למדינות, דרך האתר <http://geobytes.com/> . אם העיר לא נמצאת במפה של העירים שלנו (יש הסבר בסעיף מעל + במחלקת האינדקס בחלק " תהליך קבלת מידע על ערים" ) ניגש לאתר זה ונשלוף ממנו מידע.

**פלטים נדרשים:**

1. 845608 terms נמצאים במילון ללא ביצוע Stem.  
2. 820450 terms נמצאים במילון לאחר ביצוע Stem.  
3. 259940 terms הם מספרים מתוך המילון.  
4. קיימות 47 מדינות שונות במאגר

5. 468 ערים שונות, מתוכן 333 ערים אינן ערי בירה (135 ערי בירה במאגר)

6. שם המסמך: FBIS3-60342

שם העיר: MOSCOW כמות הפעמים שהיא מופיעה במסמך: 295

מיקומי המילה במסמך:

9!23!46!67!128!425!445!473!497!498!563!711!1917!2364!

3412!3883!4251!4368!4395!4468!4613!4801!4805!5361!5365

!5546!5573!5603!5608!5734!5782!5816!5852!5866!5910!6294

!6315!6489!6557!6563!6568!6699!6751!7015!7039!7111!7294

!7360!7799!7967!7982!8024!8112!8199!8391!8485!8596!8597

!8618!8710!9518!9960!10222!10228!10231!10237!10239!

10247!10254!10262!10269!10273!10274!10280!10287!10291

!10304!10609!10697!10935!10954!11116!11143!11269!11391

!11492!11508!11699!11714!11814!11829!11833!11940!11963

!12103!12226!12233!12394!12412!12430!12477!12482!

12487!12496!12501!12505!12521!12525!12540!12544!12966

!13135!13277!17282!17836!18257!19733!22464!26032!

26459!26682!26830!26972!27063!27256!27285!27321!

27540!27692!27872!27930!28219!28228!28234!28268!

28287!28300!28313!28340!28372!28403!28418!28426!28431

!28491!28719!28722!28766!28866!28894!28895!28898!

28909!28937!28990!29047!29062!29099!29107!29113!29135

!29142!29161!29276!29343!29452!29469!29491!29505!29515

!29516!29530!29539!29744!29843!29846!29887!29915!

29925!29934!29945!29951!29969!29972!30002!30007!30012

!30078!30095!30112!30127!30143!30188!30271!30289!30319

!30334!30359!30448!30498!30505!30620!30671!30694!30710

!30742!30750!30801!30858!30864!30904!31090!31151!

31153!31274!31345!31408!31456!31473!31584!31602!

31632!31646!31660!31735!31743!31783!31856!31891!

31925!31954!32009!33741!35894!36248!39339!40158!

40722!42929!42966!42980!42999!43050!43084!43195!

43253!43296!43330!43401!43556!43587!43609!43688!

43735!43792!43837!43912!44020!44088!44349!44582!

49280!49721!56501!57527!63513!63925!64471!64511!64800

!64995!65005!65008!65014!65016!65019!65022!65028!65032

!65199!65351!65502!65670!66004!66327!66349!66723!66864

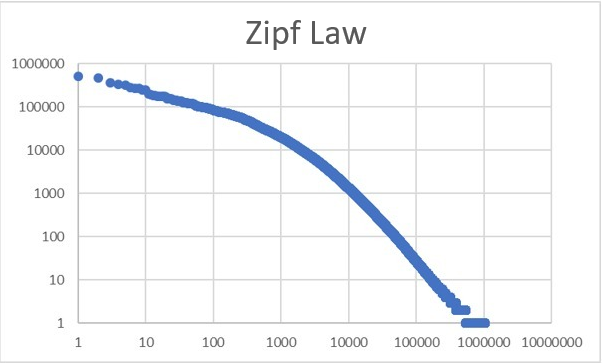
!66909!67011!67148!67529!67740!67929!68014

7.  
עשרת הterms השכיחים ביותר במאגר: עשרת הterms הכי פחות שכיחים:   
 (כמובן שיש עוד terms עם תדירות זהה)

|  |
| --- |
| **year : 523768** |
| **mr : 468639** |
| **cent : 365073** |
| **government : 356341** |
| **pounds : 311756** |
| **people : 300789** |
| **said : 267078** |
| **time : 263230** |
| **CHJ : 262222** |
| **CVJ : 262222** |

|  |
| --- |
| **L986.6 : 1** |
| **manufactuere : 1** |
| **MARWARDI : 1** |
| **O5 : 1** |
| **PERFIT : 1** |
| **PERFOMA : 1** |
| **QAA : 1** |
| **QABIL : 1** |
| **QADERI : 1** |
| **U.D.I : 1** |
| **U.P.I : 1** |

8.

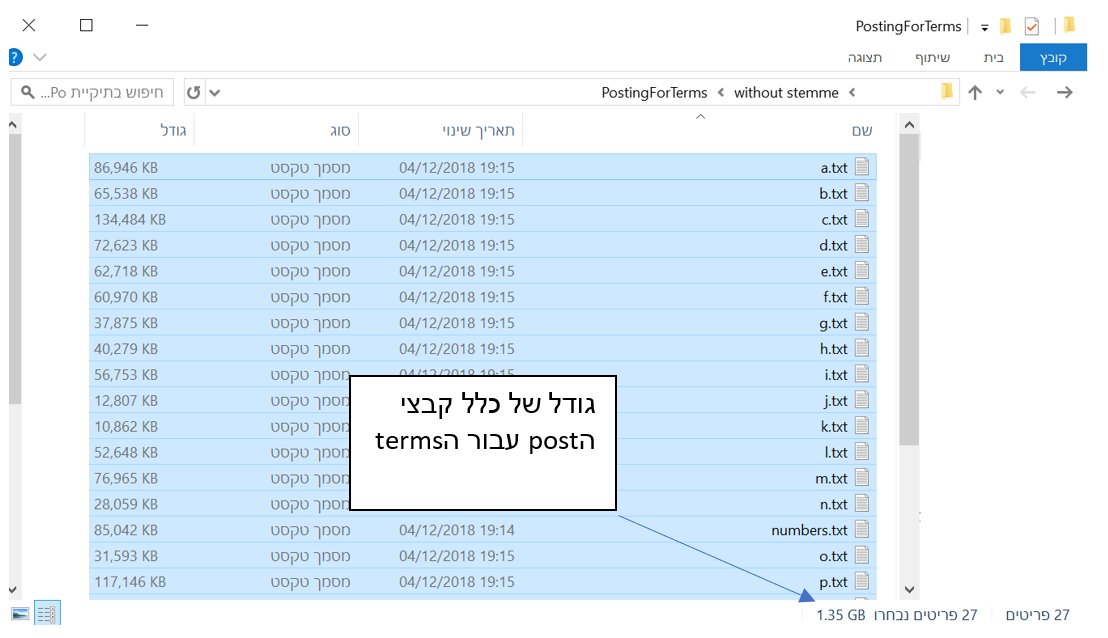
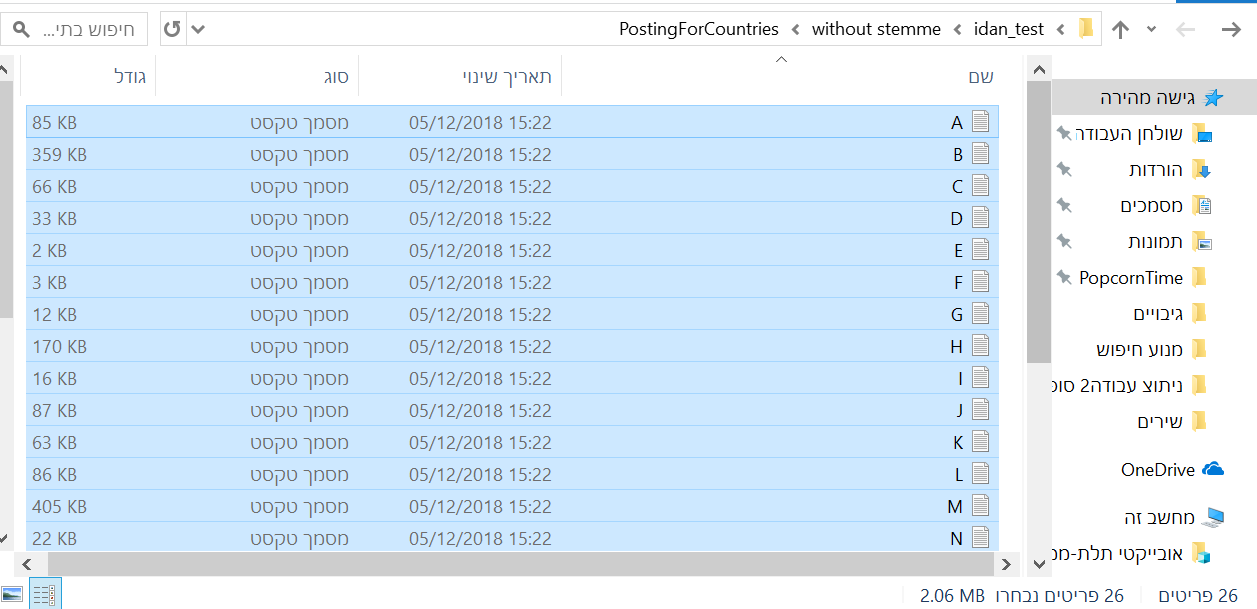


בנינו גרף לוגריתמי כאשר בציר הX מופיעים הטרמים ובציר Y התדירות של אותו term.  
אכן העקומה דומה בצורתה לZipf’s Law. ניתן לראות כי העקומה אינה לינארית במלואה אך קרובה. ניתן לראות בגרף שקיימים טרמים רבים שמופיעים פעמים בודדות ואפילו פעם אחת עפ"י הקצה הימני של הגרף.

|  |
| --- |
| **2 : 03-19** |
| **1.994K : 1** |
| **adopted : 2** |
| **article : 1** |
| **BEIJING : 1** |
| **BFN : 1** |
| **charter : 3** |
| **chinese : 5** |
| **committee : 5** |
| **conference : 4** |
| **consultative : 4** |
| **CPPCC : 4** |
| **decided : 1** |
| **eighth : 3** |
| **language : 1** |
| **national : 4** |
| **people : 4** |
| **political : 4** |
| **proposed : 1** |
| **resolution : 1** |
| **second : 3** |
| **session : 3** |
| **standing : 1** |
| **text : 1** |
| **the : 2** |
| **today : 1** |
| **type : 1** |
| **XINHUA : 1** |

9.  
רשימת הterms עבור מסמך FBIS3-3366 והתדירות של כל מילה במסמך:

10.   
עבור כלל הterms :

  
  
לכן סה"כ גודל קבצי הpost בKB הינו : 1350000.  
  
עבור המדינות:  
  
  
לכן סה"כ גודל קבצי הpost בKB הינו : 2060.

גודל של כלל קבצי הpost עבור מדינות