# דו"ח הפרויקט – חלק א'

פירוט המחלקות השונות בתוכנית:

ReadFile:

מחלקה זו קוראת את כלל המסמכים מתוך מאגר הקבצים. מחלקה זו מקבלת נתיב לתיקייה בה נמצאים כל הקבצים. שיטות המחלקה:

- ***Public ReadFile(String pathDir, boolean stemm, String postingPathSaved)*** – בנאי המחלקה. מעדכן את נתיב התיקייה להיות ה-pathDir שהתקבל, ואת המשתנה הבוליאני אודות הפעלת stemming על המאגר.
- ***Public void readInsideAllFiles()*** – מתחברת לתיקייה הרלוונטית על סמך הנתיב שהתקבל בבנאי המחלקה, וקוראת מסמך-מסמך מתוך כל קובץ טקסט שמופיע בכל תת-תיקייה של הקורפוס. זורקת alert במידה ונתיב התיקייה שגוי.
- ***Private void createDoc()*** – יוצרת אובייקט של המחלקה Document עבור כל מסמך בקובץ טקסט, על פי תגיות <Doc>, </Doc> הרלוונטיות. לכל אובייקט כזה השיטה מוסיפה מזהה מסמך, כותרת, ואת הטקסט עצמו. מוסיפה כל מסמך לתור documentsSet הסטטי של המחלקה Parse.
- ***public void start()*** – השיטה מתחילה להריץ את הפונקציה המרכזית באמצעות ת'רדים שונים בו זמנית.

Parse:

מחלקה זו מפרקת את קטע הטקסט מכל מסמך ל-terms נפרדים, על סמך מגוון חוקים של השפה בנוגע למספרים, שמות, אותיות גדולות וקטנות, וכו'. שיטות המחלקה:

- ***Parse(boolean stemmer , String stopWordPath*** – בנאי המחלקה. מעדכן האם נדרש לבצע על הטקסט stemming או לא, ומעדכן את רשימת ה-stop worlds הנדרשת להסרה מתוך המאגר.
- ***private void parseDocs(LinkedList<Document> listDocument)*** – השיטה עוברת על רשימת המסמכים כולה ומפרידה את הטקסט על פי מגוון סימני פיסוק (:*;'#+...). שולחת לטיפול ייחודי בהתאם לסוג המילה שזיהתה – באם מדובר במחיר/באחוז/בטווח של מספרים או במילה.
- ***private void insertTermDic(String term, Document newDoc, int index)*** – השיטה מוסיפה את ה-term ל-2 המילונים השונים של המסמך הספציפי: באם מדובר ב-term חדש – היא קוראת לשיטה ***insertFirstOccur()***, אחרת – מעדכנת בעצמה את כמות מופעי המילה ומיקומיה במילונים הרלוונטיים.
- ***private void insertFirstOccur(String term, Document newDoc, int index)*** – השיטה מוסיפה את ה-term למילון המילים הרגיל של המסמך הספציפי, ולמילון השומר את מיקומי ההופעות של כל term עבור המסמך הספציפי.
- ***private void changeUpperCaseToLowerCase(String term, Document newDoc)*** – השיטה מסירה מהמילון של המסמך את ה-term בתצורת האותיות הגדולות בהן היה כתוב עד כה – ומוסיפה את אותם הנתונים תחת ה-term באותיות קטנות.

- ***private String numericToPrice(String num ,String sum,...)*** – השיטה מחזירה String המייצג מחיר בהתאם לחוקי השפה המפורטים בהנחיות העבודה: מוסיפה את המילים Dollar, M בהתאם לערך המילה שנקראה.
- ***private boolean equalToSum(String word)*** – השיטה מחזירה האם מדובר בביטוי המייצג מחיר או לא, לפי ה-eNum שהוגדרו – Dollars, million, trillion, billion, m, bn.
- ***private boolean isPrice(String price)*** – השיטה מחזירה true באם הביטוי כולל את הסימן $, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים.
- ***private void handlePrice (String currToken, String[] afterIndex, Document document, int index)*** – השיטה מקבלת מחרוזת המייצגת מחיר ומוסיפה אותה למילון על פי חוקי השפה המתאימים.
- ***private boolean isNumericDouble(String docToken)*** – השיטה מחזירה true באם הביטוי הינו מספר, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים.
- ***private boolean isFraction(String docToken)*** – השיטה מחזירה true באם הביטוי המספרי כולל שבר, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים.
- ***private boolean lessThanMillion (String numToken)*** – השיטה מחזירה true באם הביטוי המספרי מתאר מספר קטן ממליון, אחרת תחזיר false.
- ***private boolean lessThousand (String numToken)*** – השיטה מחזירה true באם הביטוי המספרי מתאר מספר קטן מאלף, אחרת תחזיר false.
- ***private boolean isThousand (String numToken)*** – השיטה מחזירה true באם הביטוי המספרי מתאר אלפים, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים (הוספת K עבור מחיר למשל).
- ***private boolean isMillion (String numToken)*** – השיטה מחזירה true באם הביטוי המספרי מתאר מליונים, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים (הוספת M עבור מחיר למשל).
- ***private boolean isBillion (String numToken)*** – השיטה מחזירה true באם הביטוי המספרי מתאר ביליונים, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי המסוים (הוספת B עבור מחיר למשל).
- ***private boolean isMonths (String token)*** – השיטה מחזירה true באם הביטוי הינו חודש כלשהו, אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי.
- ***private boolean isLine (String token)*** – השיטה מחזירה true באם הביטוי מכיל את הסימן '-', אחרת תחזיר false, לטובת זיהוי המשך טיפול בביטוי (תאריך, טווח מספרים, ביטוי מוכר ועוד).
- ***private String returnDouble (String numToken)*** – השיטה מקבלת ביטוי מספרי ומחזירה String המייצג אותו בהתאם להנחיות העבודה – עד דיוק של 3 ספרות לאחר הנקודה העשרונית.
- ***private String shiftLeft (String numToken ,int shift)*** – השיטה מקבלת ביטוי מספרי ומזיזה את הנקודה העשרונית מספר מיקומים שמאלה בהתאם לקלט shift.
- ***private String deleteZeroFromEnd (String num)*** – השיטה מקבלת ביטוי מספרי ומוחקת את האפסים המיותרים באם קיימים מימין לנקודה העשרונית.
- ***private boolean isNumericDate (String numToken)*** – השיטה מחזירה true באם הביטוי החלקי המייצג את היום בתאריך כלשהו הוא חוקי (בטווח הערכים 1-31 למשל), אחרת תחזיר false, לטובת המשך טיפול מתאים בביטוי.
- ***private boolean isDate (String docToken)*** – השיטה מחזירה true באם הביטוי הינו תאריך חוקי בשפה והצליחה להוסיף אותו למילון, אחרת תחזיר false.

- ***private boolean isPercent (String term)*** – השיטה מחזירה true באם הביטוי מייצג אחוז (%), אחרת תחזיר false, לטובת המשך טיפול מתאים בביטוי.
- ***private boolean isBetween (String term)*** – השיטה מחזירה true באם הביטוי מייצג טווח חוקי בשפה (יכיל את המילה between למשל), אחרת תחזיר false, לטובת המשך טיפול מתאים בביטוי.
- ***private String turnMonthToNumber (String docMonth)*** – השיטה מקבלת ביטוי המתאר חודש מסוים בשנה ומחזירה String עם מספר החודש המזהה אותו ("Jun" לדוגמה יחזיר "06"), לטובת יצירת term המכיל חודש חוקי בשפה.
- ***private boolean stopWord (String word)*** – השיטה מחזירה true באם המילה היא אחת מה- stop words הקיימות ברשימה שהתוכנית קיבלה בהתחלה, אחרת תחזיר false, בכדי שנדלג עליה ולא נטפל בה כמו מילה מן המניין.
- ***private String startEndWord (String word)*** – השיטה מקבלת ביטוי ומסירה סימני פיסוק בתחילתו או בסופו, דוגמת "," או "." או "-", לטובת יצירת term רלוונטי וחוקי בשפה.
- ***private void handleWords (String word, Document document, String currToken, int index)*** – השיטה מקבלת מילה ומטפלת בה בהתאם לתוכנה (אותיות גדולות בלבד / אותיות קטנות בלבד / הופעה ראשונה של המילה במאגר/..) – ושליחתה לפונקציית המשך הטיפול הרלוונטית.
- ***private void handlePercent (String num)*** – השיטה מקבלת מחרוזת המייצגת אחוז מספרי כלשהו ומוסיפה אותו למילון על פי חוקי השפה המתאימים.
- ***private void handleNum (String intNum)*** – השיטה מקבלת מחרוזת המייצגת ביטוי מספרי כלשהו ומוסיפה אותו למילון על פי חוקי השפה המתאימים.
- ***private void handleLine (String line)*** – השיטה מקבלת מחרוזת המייצגת ביטוי עם טווח כלשהו ומוסיפה אותו למילון על פי חוקי השפה המתאימים.
- ***private void setStopWord (String path)*** – השיטה מקבלת נתיב לתיקייה בה נמצא הקובץ הכולל את כל ה-stop words עבור התוכנית, ומוסיפה אותם לשדה Set<String> stopWords של המחלקה, לצורך המשך אפיון כלל המילים במסמכים.

:Document

מחלקה זו מייצגת אובייקט מסוג מסמך, ושומרת שדות חשובים לגביו – כותרת, מזהה מסמך, הטקסט עצמו, מילון המילים הייחודיות במסמך ביחד עם כמות הופעתן בו וכן מיקומי הופעתן. שיטות המחלקה:

- ***()*** Document – בנאי המחלקה. מאתחל את שני המילונים של המחלקה.
- ***public int uniqueTerm ()*** – השיטה מחזירה את גודל המילון של המסמך, כלומר מספר המילים הייחודיות שנקראו מתוכו.
- ***public int getTfMax ()*** – השיטה מחזירה את ערך ה-tf הגבוה ביותר במסמך, כלומר של המילה הנפוצה ביותר במסמך.
- ***public String getText ()*** – השיטה מחזירה מחרוזת הכוללת את כל הטקסט של המסמך.
- ***public String getPlaces (String term)*** – השיטה מחזירה מחרוזת עם כל המיקומים של ה-term במסמך, ומחרוזת ריקה באם ה-term לא קיים במסמך.
- ***public String getText ()*** – השיטה מחזירה את מזהה המסמך.

- ***public int getLength ()*** – השיטה מחזירה את אורך המסמך כולו (כולל כפילות של מילים, לא כולל stop words).
- ***public void setTitle (String title)*** – השיטה מעדכנת את שדה הכותרת של המסמך.
- ***public void setId (String id)*** – השיטה מעדכנת את שדה המזהה של המסמך.
- ***public void setText (String text)*** – השיטה מעדכנת את שדה המזהה של המסמך.
- ***private int calMaxTf ()*** – השיטה מחשבת את הערך המקסימלי של tf במסמך, ומעדכנת את השדה הרלוונטי במחלקה.
- ***public void clear (String id)*** – השיטה מאתחלת את השדה טקסט להיות null.

:Indexer

מחלקה זו מקבלת את כלל המילים מה-parser ויוצרת את ה- inverted index התואמים: מילון שנשמר בזכרון הראשי, והמשך טיפול עבור יצירת קבצי posting שנשמרים בדיסק. שיטות המחלקה:

- ***Indexer (boolean stemming ,String postingPath)*** – בנאי המחלקה. קולט את הנתיב לתיקייה בה יש ליצור את קבצי ה-posting, ומעדכן את שדה ה-stemming במחלקה (האם להפעילו או לא).
- ***private void createFile(){ -*** השיטה יוצרת קובץ טקסט חדש בו ייכתב כל המידע על המילים, רק במידה ולא קיים כבר קובץ כזה.
- ***private void indexAll(LinkedList<Document> listDoc) –*** השיטה עוברת על רשימת המסמכים שקיבלה ממחלקת parse ומכניסה את כל ה-terms למילון מאוחד. כמו כן – קוראת לשיטה המתאימה במחלקה Writer כדי שתכתוב את המידע לקובץ posting בדיסק.
- ***private void updateTermDic (TreeMap<String, int[]> littleDic) –*** השיטה מעדכנת את המילון הכולל של התוכנית על סמך המילים מהמילון הקטן והזמני שמקבלת – מוסיפה terms חדשים, או מעדכנת ערכים עבור terms שכבר קיים.
- ***private void createPostingFile (String path , HashMap<String, String> ChunkTermDicDocs) -*** השיטה יוצרת קובץ posting חדש עם כל המידע שקיבלה מאותה קבוצת המסמכים שאנדקסה.
- ***private TreeMap<String , int[]> newTree) –*** השיטה מחזירה מבנה נתונים חדש מסוג TreeMap, עם הגדרת Comparator מתאימה (סדר לקסיקוגרפי).
- ***private static void restart () –*** השיטה מאתחלת את תור המסמכים הנוכחי

:Stemmer

מחלקה זו מממשת את Porter Stemmer - ממירה כל מילה שמקבלת לצורת השורש שלה, גם אם אין מדובר במילה קיימת בשפה. השתמשנו בקוד פתוח מתוך האתר:
https://tartarus.org/martin/PorterStemmer/java.txt, אשר מחולק לשלבי האלגוריתם השונים.

:Writer

מחלקה זו כותבת את כל המידע לדיסק, ל-7 קבצי posting שונים לכל היותר, בחלוקה לטווחי אותיות בשפה האנגלית. שיטות המחלקה:

- ***public Writer ()*** – בנאי המחלקה. מאתחל את שדות המחלקה.
- ***public void createPostingFile (String Path, HashMap<String, String> chunkTermDicDocs)*** – הפונקציה המרכזית של המחלקה. מקבלת את הנתיב לתיקייה הכללית וכותבת את כל אחד מה-terms במבנה הנתונים שקיבלה אל תוך קובץ הכולל את טווח האותיות המתאים.
- ***public void sortPostingFiles()*** – השיטה עוברת על כל קבצי ה-posting שנוצרו בתיקייה הייעודית, וממיינת את התוכן של כל אחד מהם לפי ABC.

:Controller

מחלקה זו מציגה את הפרויקט למשתמש בתור ממשק נוח וידידותי, ולכן מתממשקת עם כל שאר המחלקות שתוארו לעיל. שיטות המחלקה:

- ***public void onStart ()*** – השיטה מפעילה את מנוע החיפוש . השיטה תעדכן את שדות המחלקה: נתיב הקורפוס ונתיב ליצירת קבצי הפוסטינג. שיטה זו תחלק את הקורפוס למסמכים, תפרסר כל מסמך ותיצור את קבצי האינדקס ההופכיים.
- ***public void Browse ()*** – השיטה תאפשר את בחירת הנתיב הרצוי מתוך חלונית המציגה את תיקיות המחשב.
- ***public void onReset ()*** – השיטה תמחק את קבצי ה-posting הקיימים בתיקיות שנוצרו בריצה הנוכחית או נותרו קיימות מריצות קודמות, וכן תאתחל את מבני הנתונים של התוכנית (תאפס את הזכרון הראשי).
- ***public void onDisplayInv ()*** – השיטה מציגה למשתמש בצורה ממויינת את המילון הקיים בזכרון הראשי – את ה-term עצמו ולצדו מספר המופעים הכולל שלו במאגר.
- ***public void onLoadInv ()*** – השיטה טוענת לזכרון הראשי את המילון שיוצרת מתוך כלל קבצי הפוסטינג הקיימים בתיקייה שהזין המשתמש בתור posting path.
- ***public void deleteFiles (String pathToDelete)*** – השיטה מוחקת את כלל קבצי ה-posting שבתוך התיקייה בנתיב שמקבלת - pathToDelete , ולבסוף גם את התיקייה עצמה.
- ***private void displayError (String error)*** – השיטה מציגה למשתמש הערה עם התוכן שמקבלת – error.
- ***private void displayInfo (String info)*** – השיטה מציגה למשתמש הערה עם התוכן שמקבלת – info.
- ***private void load (String fileNameToLoadFrom)*** – השיטה טוענת לזכרון הראשי את המילון שיוצרת מתוך קובץ הפוסטינג המסוים שמקבלת כקלט.

b. <u>התמודדות עם מגבלת הזכרון:</u>

בעקבות ההגבלה נאלצנו לשמור קבצי posting על הדיסק עצמו, בכדי להימנע ממצב בו עלינו לשמור את כל המידע על הזכרון הראשי. ראשית, נכתוב את כל המידע אודות כל ה-terms במאגר ל-7 קבצי posting שונים (ע"פ חלוקה לטווחי ABC בשפה). לאורך התהליך עצמו אנו מוסיפים את הערכים למילון הכולל (המילה עצמה, ומספר המופעים שלה בקורפוס השלם לבד, ללא פרטים נוספים אודותיה).

בכדי להיטיב עם זמני הריצה של התכנית השתמשנו ב-**Threads**: אפשרנו ריצת תהליכונים במקביל עבור קריאה ממסמכי המאגר, וכן עבור כתיבת המידע אודות כל term לקבצי ה- posting בדיסק.

כמו כן, נעזרנו **בביטויים רגולריים** בקוד בכדי לאתר סימנים שלא בשפה, ובכך הסרנו ביטויים לא רלוונטיים למילון. בהמשך, הדבר סייע מאוד לזיהוי כל מילה ומשמעותה בתהליך.

בנוסף, הורדנו **stop words** מהמאגר – מילים חסרות משמעות אשר מגדילות משמעותית את זמן הריצה והצורך בזכרון.

C. את **קבצי ה-posting** שמרנו בתור קבצי טקסט בדיסק. בסוף כל תהליך האינדוקס נשארו 7 קבצים הכוללים את המידע על כל המילים במאגר. ישנה חלוקה על פי מיון, כאשר כל קובץ מכיל מילים של טווח אותיות מוגדר (a-d, e-i, j-m, n-q, r-v, w-z), וקובץ נוסף הכולל את כל הביטויים המספריים. כל קובץ בנוי בצורה הבאה:
barring|FBIS3-29:1;237|FBIS3-72:3;22,48,57

ניתן לראות כי כל שורה מציגה את ה-term עצמו| את מזהה המסמך בו מופיע: את כמות הפעמים בהן מופיע במסמך; פירוט מיקומי הופעת ה-term במסמך. בהמשך השורה יופיעו מזהי מסמכים נוספים בהם מופיע ה-term.

<u>d. תהליך יצירת הקבצים ההופכיים:</u>

בשלב הראשון נקבל את כל המילים מה-parse בצורה מקבילית ונכניס אותן למבני הנתונים המתאימים. כל 5,000 מילים נבצע כתיבה של המידע לדיסק (גודל קובץ חלקי).

בשלב הבא, נעבור על כלל הקבצים שיצרנו בשלב הקודם וניצור מילון הכולל את מספר המסמכים בהם כל מילה הופיעה, וכן את סך המופעים במילון. כמו כן, ניצור קבצים הופכיים לפי כמות השורות.

הסיבה שהחלטנו על גודל הקבצים החלקיים (5,000 מילים) התבססה על מספר הרצות שביצענו, בהן הגדלנו והקטנו את גודל קבוצת המסמכים, תוך חיפוש אחר זמן ריצה מיטבי ווידוא את כך שהמערכת לא תקרוס.

אם היינו מבצעים את התהליך עבור כל המסמכים במאגר (חצי מליון לערך), היינו מגיעים למצב בו כל המידע שמור בזכרון, מה שהיה מוביל לקריסת המערכת.

e. <u>פריטי אינפורמציה נוספים:</u>

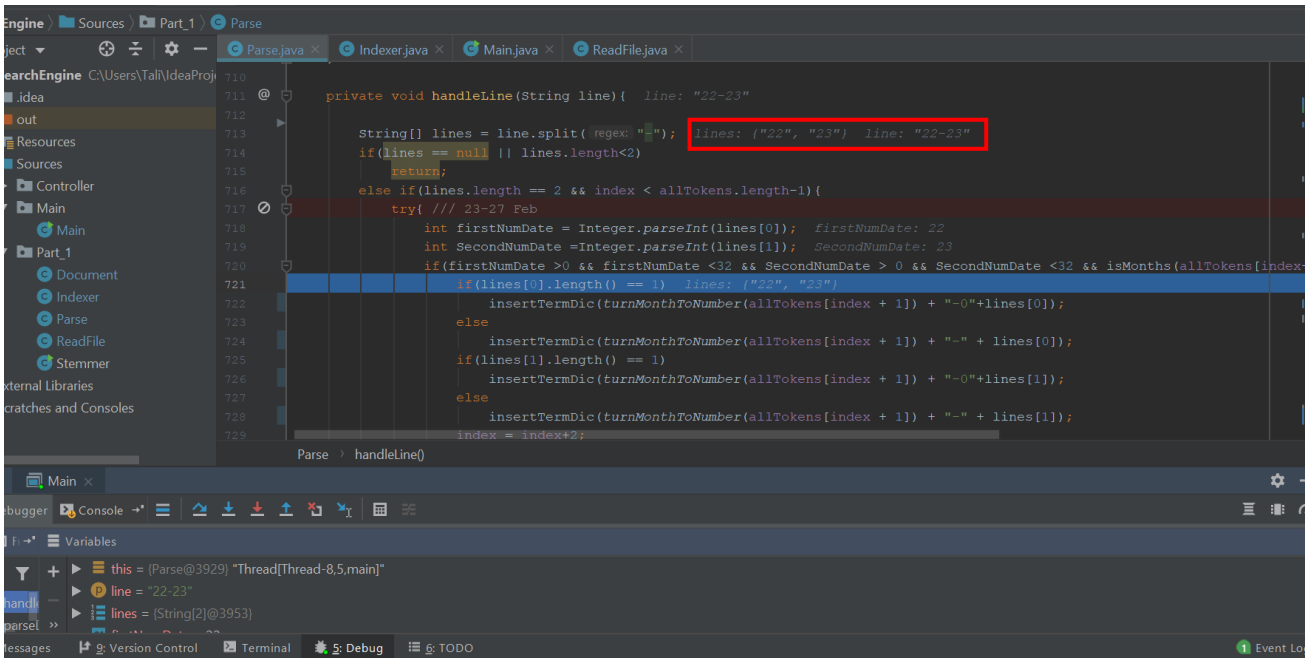
עבור כל term שמרנו את כל המיקומים המדויקים בהם הופיע בתוך כל מסמך, לטובת דירוג המסמכים בחלק ב'. למשל עבור שאילתא שתהיה מורכבת משתי מילים שונות – מסמך המכיל את שתיהן באותו המשפט – ידורג גבוה יותר ממסמך ארוך בו מילה אחת מופיעה בתחילתו, והשניה בסופו.

עבור כל מסמך שמרנו את אורכו, המיוצג על ידי מספר המילים שיש בו (כולל כפילויות, ולאחר הסרת ה- stop words). בהמשך נוכל להשתמש בערך זה לצורך נרמול תדירות המילים (בחישוב דירוג המסמכים השונים).

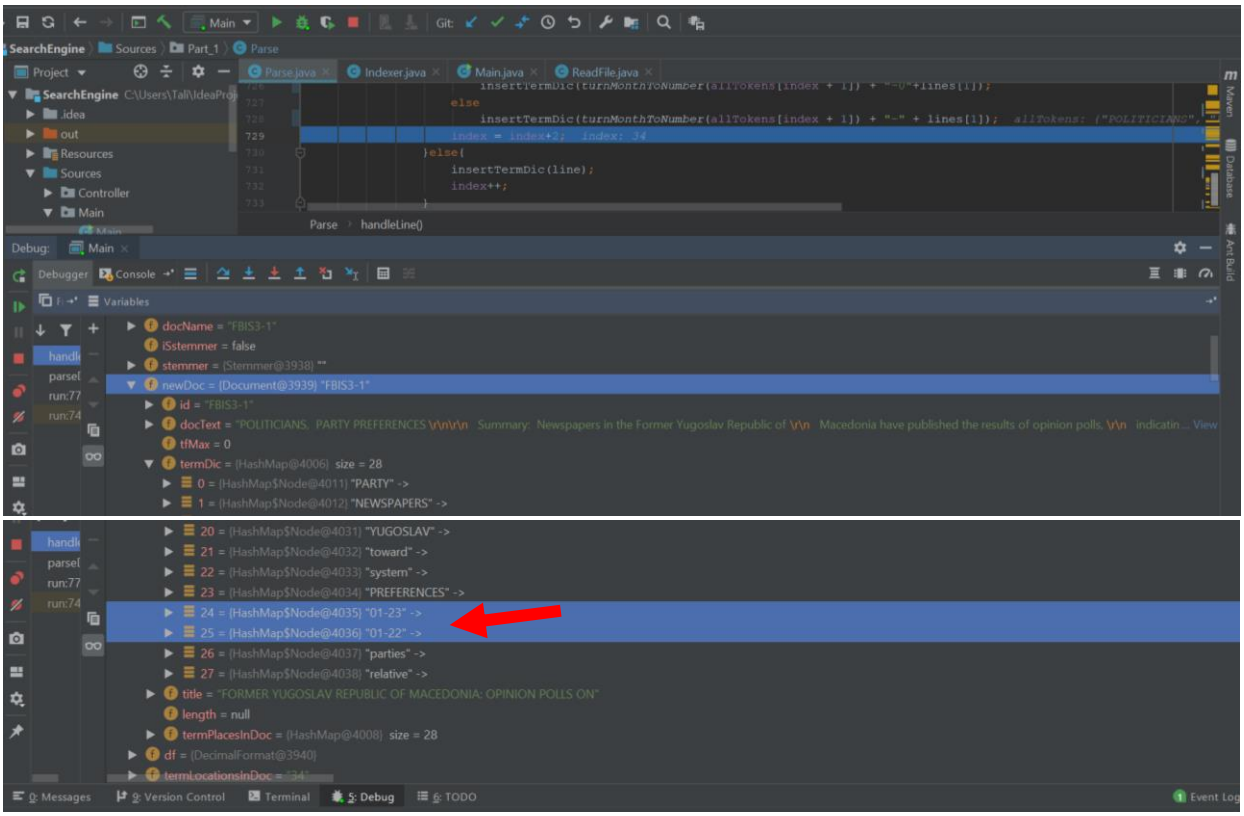
f. <u>הדגמת החוקים הנוספים שהגדרנו מתוך המחלקה Parse:</u>

1. **טווח תאריכים** – נשמור בשני terms נפרדים: התאריך ההתחלתי, והתאריך הסופי. לדוגמה:

המחלקה Parse זיהתה ביטוי הבולל "-" (January 22-23), נכנסה לשיטה המתאימה handleLine, ושמרה את כל אחד מהתאריכים ב-term נפרד:

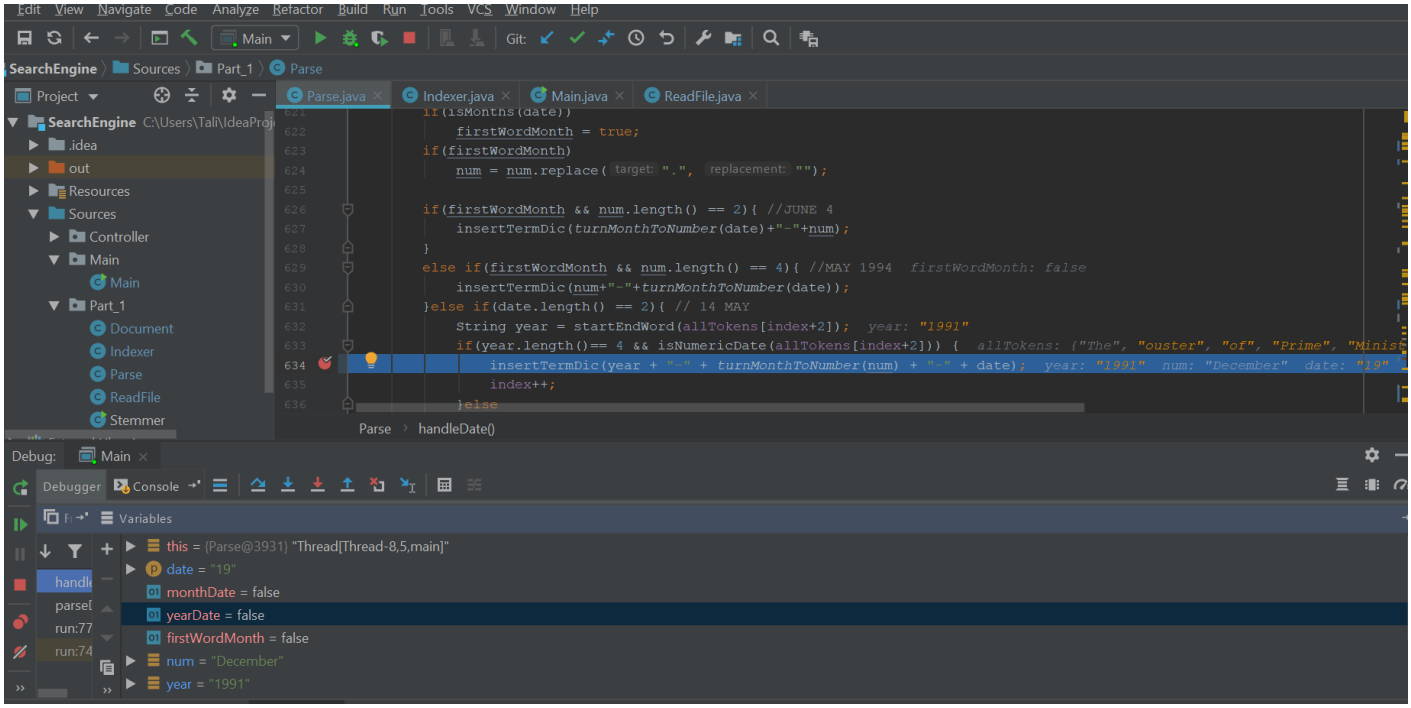



לאחר מכן נראה כי שני התאריכים (01-23, 01-22) נוספו למילון של המסמך הספציפי FBIS3-1:

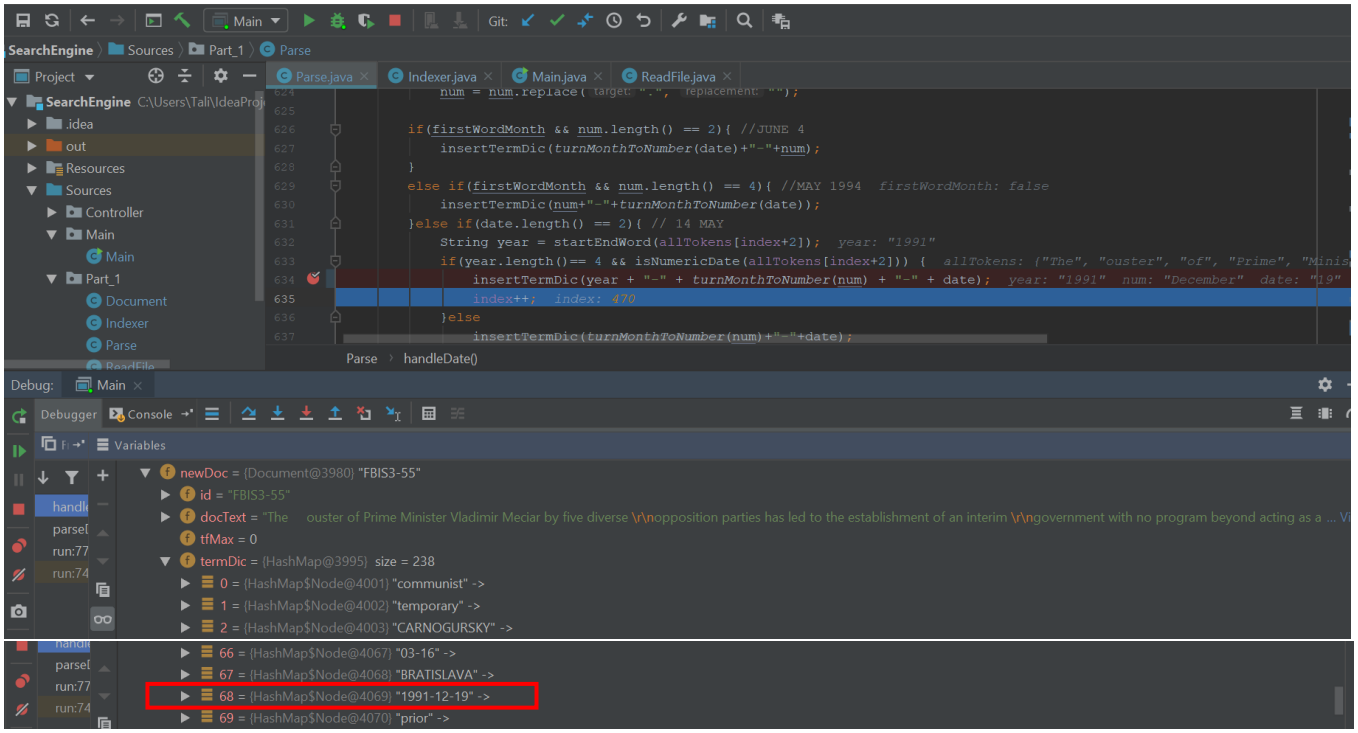


2. **תאריך שלם** – נשמור בתור term יחיד את התאריך במלואו. כך לדוגמה:

המחלקה Parse זיהתה ביטוי המתאר תאריך בפורמט שונה מאלה המוגדרים בהנחיות (הכולל גם את השנה וגם את היום המסוים בחודש: 19 December 1991), נכנסה לשיטה המתאימה handleDate, ושמרה את הביטוי בתצורה החדשה שהגדרנו (1991-12-19):




לאחר מכן נראה כי התאריך בפורמט שהגדרנו (1991-12-19) נוסף למילון של המסמך הספציפי FBIS3-35:



חלק 2: לאחר עיבוד המאגר:

a. לפני stemming ישנם 2,763,084 term שונים במאגר.

b. לאחר stemming ישנם 2,569,574 term שונים במאגר.

c. ישנם 460,448 term שונים במאגר שהם מספרים, כאשר הספירה היתה עבור כל ביטוי שהתו הראשון בו הוא ספרה כלשהי.

d. 10 ה-term השכיחים ביותר:

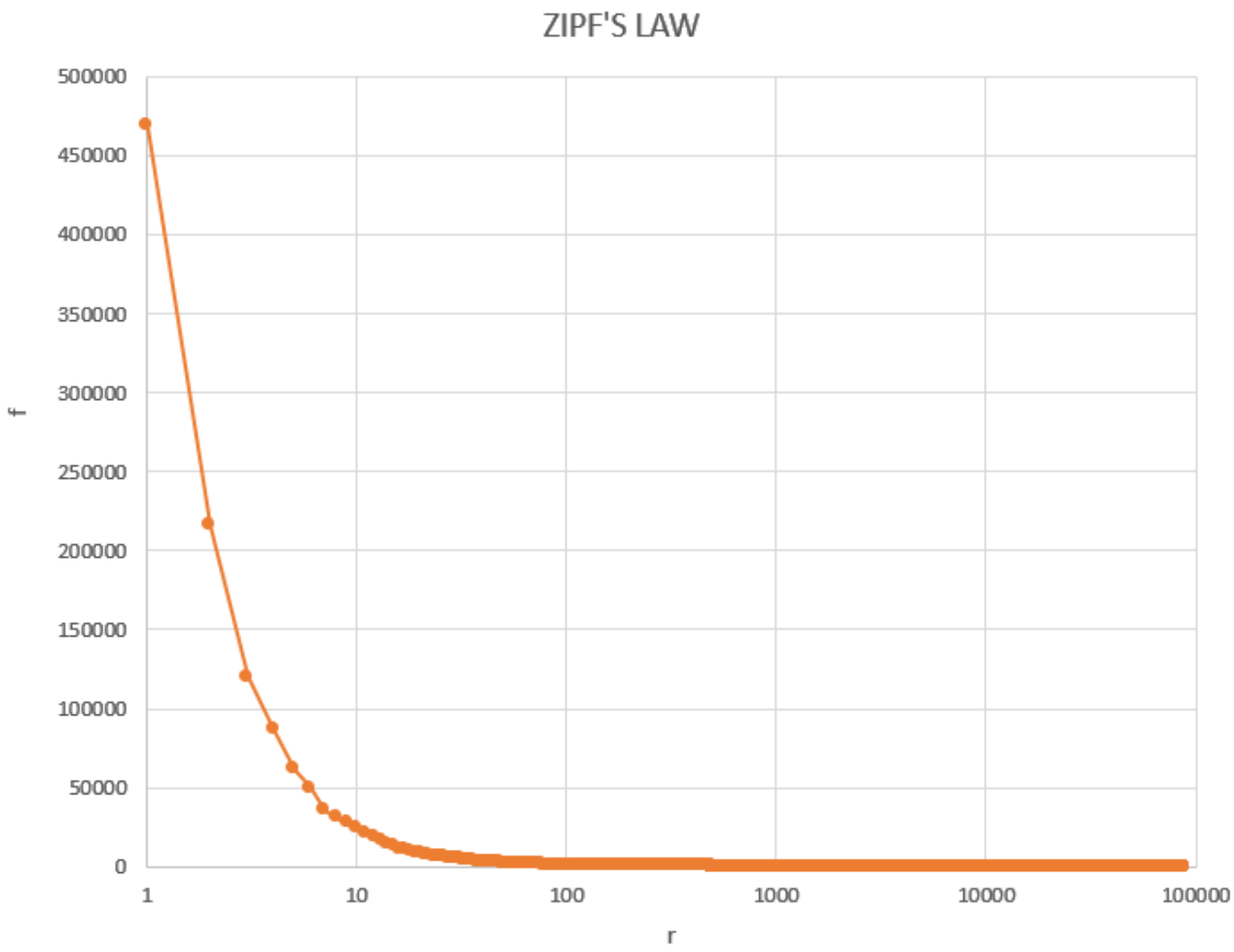
| | Terms | Frequency |
|---|---|---|
| 1 | CELLRULE | 718012 |
| 2 | TABLECELL | 578744 |
| 3 | year | 340194 |
| 4 | MR | 334686 |
| 5 | cent | 300593 |
| 6 | CHJ | 262222 |
| 7 | CVJ | 262222 |
| 8 | pounds | 256144 |
| 9 | years | 219540 |
| 10 | time | 208391 |

10 ה-term הכי פחות שכיחים במספר ההופעות שלהם במאגר כולו:

| | Terms | Frequency |
|---|---|---|
| 1 | evolutive | 1 |
| 2 | microtargets | 1 |
| 3 | POMFRET | 1 |
| 4 | poor-illumination | 1 |
| 5 | lowskill | 1 |
| 6 | comunidades | 1 |
| 7 | ORZEL | 1 |
| 8 | KALEVS | 1 |
| 9 | deciliter | 1 |

| | | |
|---|---|---|
| 10 | PORDUCTIO | 1 |

e. גרף Zipf's Law של המילים הייחודיות במאגר:




f. רשימת ה-term של מסמך FBIS3-3366 (עם תדירות של כל מילה במסמך):

- זאת בהנחה שאנו מכניסים למילון מילים שכתובות לאחר התגית [TEXT] בלבד, ולכן המילים שמופיעות לאחר <TEXT>, כמו למשל – Language, Article Type – לא נכנסות למילון במימוש שלנו.

| | Term | Frequency |
|---|---|---|
| 1 | 03-19 | 2 |
| 2 | 1994-03-19 | 1 |
| 3 | adopted | 2 |
| 4 | amended | 3 |

| | | |
|---|---|---|
| 5 | BEIJING | 1 |
| 6 | CHARTER | 3 |
| 7 | CHINESE | 4 |
| 8 | COMMITTEE | 5 |
| 9 | CONFERENCE | 4 |
| 10 | CONSULTATIVE | 4 |
| 11 | CPPCC | 4 |
| 12 | decided | 1 |
| 13 | effect | 1 |
| 14 | EIGHTH | 3 |
| 15 | NATIONAL | 4 |
| 16 | PEOPLE | 4 |
| 17 | POLITICAL | 4 |
| 18 | proposed | 1 |
| 19 | RESOLUTION | 1 |
| 20 | SESSION | 3 |
| 21 | STANDING | 1 |
| 22 | today | 1 |

g. גודל ה-posting – נפח אחסון נדרש עבור כל אחת מהריצות :

עם stemming: 1,538,690.311 KB

ללא stemming: 1,729,315.339 KB

h. משך הזמן שלקח למנוע לבנות את האינדקס על הקורפוס כולו: 11 דקות.