שמואל הנגיד

סוגי הערות:

* <טקסט\_מקור>] <חילוף>+ (אחד או יותר)
  + <חילוף>: <טקסט\_חדש> <מקור>+
  + <מקור>: טקסט בבולד עם רווחים סביבו (ספציפית אחד מהמקורות האפשריים), \*? ?(שם של מקור מותר) ?\*?
* חסר
  + **זיהוי:** מכיל את המילה "חסר" באיטליקס
* נוסח הפנים על פי X
  + משנה את המקור של הטקסט המקורי של ההערה
* מחיקות? (strikethrough)
  + לשמור מה נמחק ומה תוקן, אלא אם מילה זהה
  + **זיהוי:** מכיל טקסט בStrikethorugh, או תבנית מוכרת אחרת ("נכתב X", "החל לכתוב X")
* הושלם לפי (להסיר)
  + כמו "נוסח הפנים לפי", משנה את ההתייחסות לטקסט המקור
* ..
  + חסרות אותיות
  + לשמור ככה
* נכתב X, סומן למחיקה ותוקן ...
  + לשמור כמו מחיקה – מה נמחק ולמה תוקן
* והמעתיק החל לכתוב... זיהה את שגיאתו...
  + לשמור אם המילה שנמחקה היא לא אותה מילה
* בספק:
  + לשמור כהערה עם סימון שהיא בספק
* יתר הבית מחוק, ומסתבר שסדר המילים הוחלף
  + ניסוח לא טוב, להגיד לאורי והוא יתקן
* בין השיטים?
  + לפעמים כותבים גם "שיטין", או "בין השורות", או "מעל/מתחת לשורה" וכו'
* "סדר המילים" – אם זה מופיע אז להניח שתוקן, להקפיץ לאורי
* בשוליים נוסף:
  + אם כתוב "מלשון X" – לא מעניין
  + אם כתוב נ"א או נ – מעניין
* [!] (להסיר)
* *ללא כותרת* (על הכתובת) – לא מעניין
* להוסיף אפשרות לשמור comment על אפרט (לדוגמה הערות בitalics אחרי חילוף)
* אולי להוסיף אופציה לשמור שהחילוף תוקן
* מונחים
  + אפרט – הפסקה שמתארת את כל החילופים
  + למה – הטקסט לפני הסוגר (הטקסט המקורי) Lemma
  + חילוף שורה – לקרוא לזה Line Apparatus
  + חילופים שמתייחסים לאותה Lemma – Lemma Apparatus
  + תיקון שמתייחס ללמה ספציפית – Variant
  + וואריאנט מורכב מText, Comment, Source
* קטגוריות חילופים
  + כתיב מלא/חסר: אם נוספה (או הוחסרה) אות אהו"י בתוך המילה
    - לחפש כל מילה להתאמה בכתיב מלא / חסר?
  + חילוף אות כללי: חילוף אות אבל עם אותו מספר תווים במילה
  + חילוף מילה
  + (לשים לב מה קורה כשיש יותר ממילה אחת ביחד)
  + חסר
  + שינוי סדר (אותן מילים בסדר אחר)
  + מחיקות
  + (שילובים של וואריאנטים – לדוגמה גם שינוי סדר וגם כתיב מלא)
* נשמע שאני אצטרך לעשות את העיבוד הזה אחרי הparsing, כי הוא דיי כבד ומורכב. אז צריך לשמור את הפורמטינג עדיין של הטקסט של הוואריאנטים

שאלות לאורי:

* סוגי הערות
  + איזה להשאיר ואיזה לזרוק?
  + הבהרות על הערות לא ברורות
* האם קריטי לתת את הטקסט המקורי המלא במקרים שיש רק חלק (לדוגמה "ושלח ... עבירה"] *חסר*)
* אם הטקסט הוא מכמה מקורות שונים, איך יודעים ביחס למי החילוף?
  + האם צריך לעבור גם על השיר עצמו ולדעת מאיזה מקור כל שורה נלקחה?
  + אפשר גם לעבור על ה"מקורות" ושם יש רשימה של המקורות, ובסוגריים יופיע צירוף "נוסח הפנים לבתים X" ואז פירוט הבתים שעבורם הגוף לקוח ממקור זה), גם "נוסח הפנים לכתובת", יכול להיות גם "נוסח הפנים לשיר" אם זה עבור כל השיר
* איך להבדיל בין סימן קריאה אמיתי לבין סימן קריאה של תיקון? תיקון תמיד בתוך [!]?
  + אין סימני קריאה בטקסט המקורי
  + כל דבר שמופיע באפרט הוא *sic*
* האם תמיד יש פסקת פתיחה? איך לזהות אותה ולהפריד אותה מהשאר?
  + תמיד יש כתובת, המקור מצויין במקורות

תהיות:

* איך מתעלמים מהערות באיטליקס בלי להשתמש בdocx?
  + אולי לעשות Preprocessing שמאחד את כל הRuns עם אותו עיצוב, ואז להמיר את זה לטקסט עם תוספות עיצוב (כוכביות, underscore)
* Preprocessing של השירים:
  + פסקת פתיחה – איך לדעת אם יש? (האם תמיד יש?)
  + אם כן, למצוא פסקה ריקה (ירידת שורה)
  + להתחיל לעבור שורה-שורה, תוך מספור שלהן, ולבדוק תמיד בסוף השורה אם יש החלפת מקור
    - לתעד על כל שורה: מספר שורה, תוכן, מקור
* האם לשמור הבדלים ”סימטריים" או חד כיווניים?
  + לדוגמה שינוי – לכתוב את שני הכיוונים, מחיקה / הוספה – לכתוב לאחד הוספה ולאחר החסרה

אחרי עוד מחשבה, נראה לי שייתכן שזה יהיה קל יותר ליצור ממש parser בשביל הדבר הזה מאשר regexים פשוטים. נראה שpyparsing זה הדבר הכי פשוט, ואחרי זה Lark. ההתלבטות שלי עכשיו היא איך לעשות את הpreprocessing שיהפוך את זה להכי נוח – כרגע אני חושב בכיוון של להמיר את הסגנונות לinline markup כמו \*\* או \_, ולהפוך הכל לשורת טקסט יחידה כזאת. אני חושב שאפשר להסיר את כל הרווחים העודפים עבור דברים שהם עם markup ולהוסיף אותם לבד, אבל לשמור אותם בטקסט רגיל. אני מתלבט לגבי איך לשלב את המקורות – אולי מראש לעשות split על רווחים ולכתוב כל אחד מהם מוקף ב\*\* בנפרד? זה יעזור להגדיר אותם בצורה קלה יותר. אם אני עושה את זה ככה, אני עדיין אצטרך בסוף לחלץ את המידע שלי מתוך אובייקט הפירסור.

בנוסף, אני צריך להבין אם אני יכול להגדיר מראש ככה סוגי הערות (כנראה שכן). כלומר המבנה יהיה משהו בסגנון של:

* פסקת-חילופים: אחד-או-יותר(חילופי-שורה)
* חילופי-שורה: <מספר> + רשימה-מפוצלת(חילוף)
* חילוף: <טקסט-מקור>] אחד-או-יותר(תיקון)
* טקסט-מקור: מילים (...)? מילים
* תיקון: אחד-מ(שכתוב, מחיקה, הוספה, הערה כללית, ... [**צריך להגדיר**]) אחד-או-יותר(מקור)
* מקור: \*אחד-או-יותר([א-ת1-9])\*
* שכתוב: ...
* מחיקה: ...
* הערה כללית: ...

שלב הpreprocessing הבא:

* עבור כל run רציף שזוהה
  + אם הוא עם עיצוב, הסר את הרווחים סביבו
  + אם הוא בולד, פצל אותו לפי רווחים
  + הוסף לו עיצוב סביבו
  + הוסף רווח בסוף
  + אם הוא רגיל, וודא שיש לו רווח או טאב בסוף

אובייקטים טובים לשימוש:

* Delimited List
* Group

### 15.08.2025

משימות פתוחות:

* לכתוב פרסור של מקורות
* לכתוב זיהוי סוגי הערות
* לכתוב תרגום לJSON שיישמר בDB
* לפרסר את שירי הדוגמה ולהעלות לDB
* להדגים שליפות על השירים
* להדגים ויזואליזציה של ההבדלים?

#### פרסור סוגי הערות

*יש לנו חילוץ של כל אפרט, עם הטקסט שלו יחד עם הפורמטינג. עכשיו אנחנו רוצים על כל אחד כזה להגיד מה ההערה, ולתרגם אותו לפורמט המתאים.*

*פורמט כללי של הערה כנראה יהיה:*

* *שם השיר*
* *שורה*
* *למה*
* *סוג*
* *כתב-יד (שבו התיקון נמצא)*
* *כתב-יד ייחוס (ביחס למי התיקון הזה)*
* *הערות (טקסט הערה חופשי)*
* *תוכן – פרטים בהתאם לסוג*

*יהיה dataclass גנרי של אפרט (וואריאנט), ואז סוגים שיורשים ממנו ומוסיפים את התוכן הייחודי שלהם.*

*יהיה צריך לבנות כנראה מעיין עץ-החלטה שמזהה איזה סוג תיקון זה. אולי לעשות שזה לא יהיה dataclass ושיהיו לו פונקציית validate או extract כזאת שמקבלת טקסט, ואם הוא תואם למבנה אז יוצרת את האובייקט?*

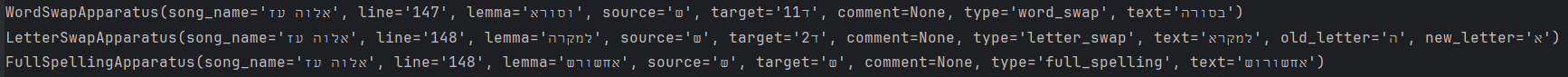
*עץ החלטות:*

* *אם מכיל את "חסר" באיטליקס – חסר*
* *אם מכיל strikethrough או תבנית מחיקה אחרת – מחיקה*
* *בדיקה אם מכיל אותה כמות מילים*
  + *אם לא – חילוף כללי. לשים את הטקסט הראשון בטקסט החילוף, ואם יש איטליקס אז כל מה שאחריו בהערה.*
* *בדיקת diff של האותיות – אם רק התווספו או רק הוחסרו אותיות והן כולן אותיות אהו"י בתוך המילה (לא בקצה) – חילוף כתיב חסר/מלא*
* *בדיקה אם כל המילים באותו אורך – אם לא, חילוף מילה כללי.*
* *אם כן – בדיקה אם התחלפה בדיוק אות אחת (אחת הוחסרה ואחת נוספה) – אם כן, חילוף אות*
* *אם לא – חילוף מילה כללי*

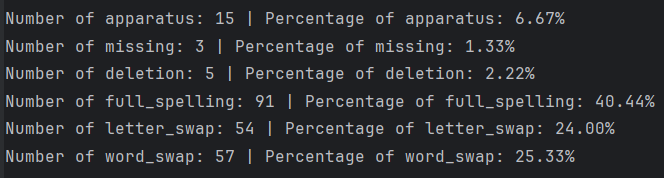
בעיות כרגע:

* להסיר full spelling מקצה המילה. ובכלל, האם לא אמור להיות נכון רק אם אותיות רק התווספו או רק נעלמו?
* להסיר סימני פיסוק
  + דוגמה: שורה 35, גופו -> גופה
  + שורה 52: צר -> צור: (בספק)
  + שורה 136: ושור -> ושורר: (בספק)
* שורה 10: ולי -> ויש. זוהה כחילוף אות. צריך לבדוק איכשהו את המיקום של החילוף ולהבין אם האות זזה.
* חילוף סדר מילים (בלי שום שינוי במילים)
* להתייחס במיוחד גם לשינוי סדר אותיות במילה? אפשר לזהות על ידי החסרה והוספה של אותה אות במילה שהיא באותו אורך.
  + לדוגמה, שורה 59: בעמדיה -> בעמידה, מזוהה כהחסרה של י' והוספה שלה במקום אחר
  + שורה 128: קטורה -> טהורה, מזוהה כחילוף אות, הוחסרה ק' והתווספה ה'. בגלל שהט' משותפת. אמור להיות חילוף מילה. צריך להתייחס מפורשות למיקום של השינוי?
  + שורה 144: ולתומכים -> ולתמוכים. זוהה כחיסור והוספה של מ'. עכשיו מזוהה כהחלפת מילה – האם רוצים התייחסות מיוחדת?
* לא טיפלתי במקרי מחיקה מיוחדים ("סומן למחיקה")
* לשפר חילוץ הערה לכל מה שהחל מאיטליקס, לעשות את זה במקום אחד גנרי בהתחלה
* לחלץ את המקור האמיתי לכל שורה מתוך פסקת המקורות
* להגדיר עם אורי הבדל בין חילוף מילה לבין חילוף אות – האם כל חילוף של אות בודדת הוא חילוף אות? או שרק חילופים בין אותיות מסוימות? (ו <--> י, ב <--> ר, וכו')
  + יש הרבה חילופי אותיות שמזוהים כחילוף מילה.
    - שורה 134: תפיל -> הפיל
    - שורה 145: ופארה -> ופורה (כנראה בגלל ששתיהן אותיות אהוי?)
* שורה 134: מקור ש(ב"ק)?
* כשיש טקסט רגיל ואז איטליקס ואז שוב רגיל, הפרסר לוקח את האחרון במקום את הראשון
  + שורה 114: ויכרו -> ויברו
  + שורה 101: *בשוליים נוסף:* מלשון מך – לא לקחת טקסט שאחרי איטליקס? (יש יוצאי דופן)

דוגמה לאיך נראות התוצאות אחרי הקיטלוג לסוגים:



דוגמה לססטיסטיקה פשוטה של הקטגוריות על אלוה עז:



שיפור לparsing כדי לקבל את החלקים השונים לפי סדר:

import pyparsing as pp

from pyparsing.unicode import pyparsing\_unicode as ppu

nums = pp.nums

alphasnums = ppu.Hebrew.alphas + nums + "'?!\":,;.ֵַּׄׄ"

# A Hebrew word

hebrew\_word = pp.Word(alphasnums)

# Sentence core (no formatting)

hebrew\_sentence = pp.OneOrMore(hebrew\_word | "..." | "[!]")

# Format wrappers (these don't name results yet)

italic\_sentence = pp.Suppress("\_") + hebrew\_sentence + pp.Suppress("\_")

strike\_sentence = pp.Suppress("~") + hebrew\_sentence + pp.Suppress("~")

# Wrap them in named Dicts so each element is tagged

italic\_sentence = pp.Group(italic\_sentence).setResultsName("italic", listAllMatches=True)

strike\_sentence = pp.Group(strike\_sentence).setResultsName("strike", listAllMatches=True)

hebrew\_sentence = pp.Group(hebrew\_sentence).setResultsName("regular", listAllMatches=True)

# Now define the complex sentence as a sequence of parts

complex\_sentence = pp.OneOrMore(italic\_sentence | strike\_sentence | hebrew\_sentence)