מעבדה בתכנות מערכות סיכום ממצה

חלק ראשון קדם אסמבלר:

קדם אסמבלר קלאסי בדיקת שגיאות מאקרו, פריסת מאקרו, תוכן מאקרו נבדק במעברים.

אסטרטגיה לפני הגישה למעברים: דרך יעילה למדי להתמודד עם תהליכי קומפילציה, היא לפרק את חלק ניתוח התחביר, מהלוגיקה של המעברים עצמם, כלומר באופן מופשט להשתמש בפונקציה יחידה, שתדאג לניתוח תחביר של כל שורה שהיא תקבל כארגומנט.

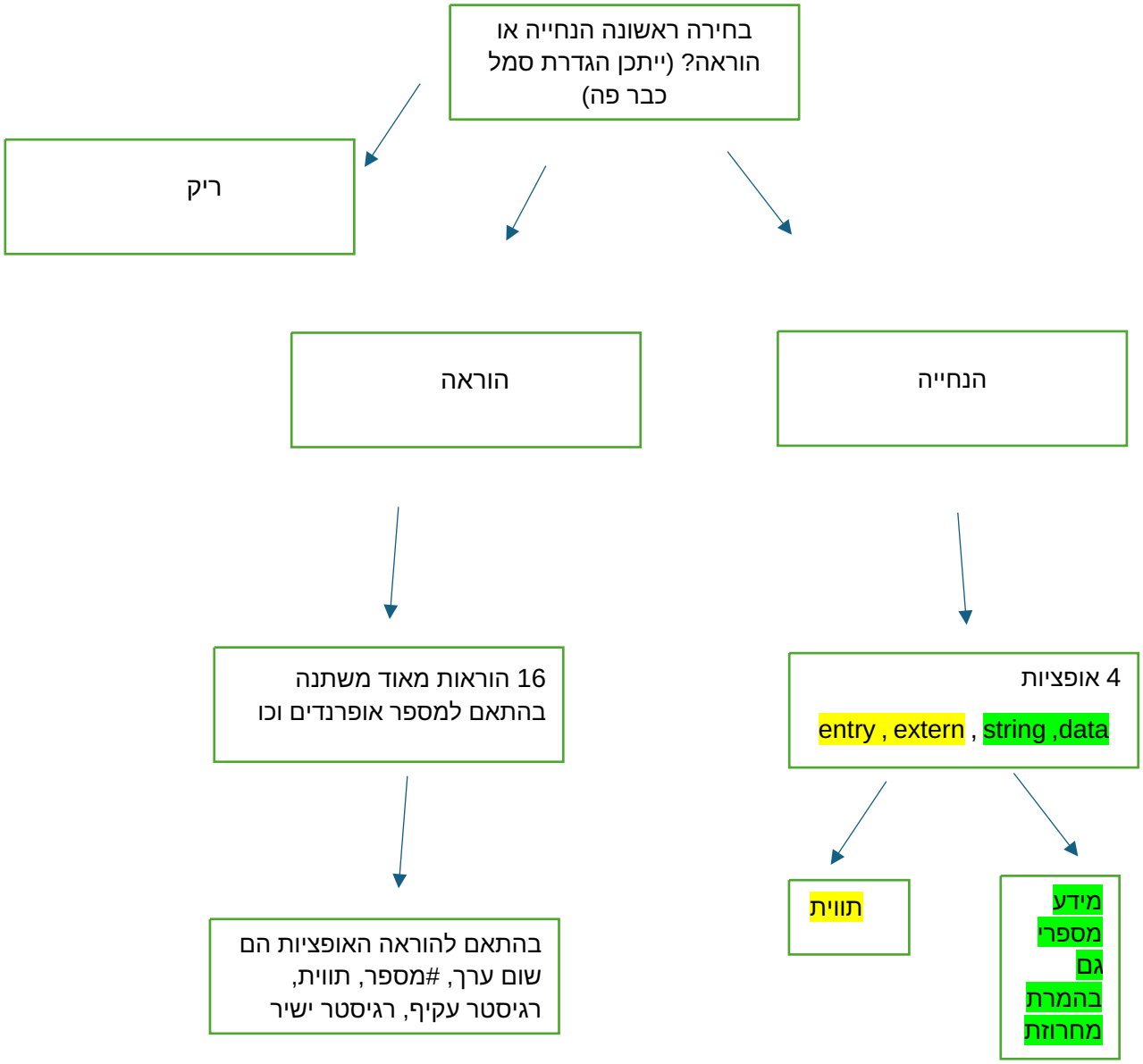
על מנת שהפונקציה תהיה מובנת ותעבוד טוב, יצרתי קובץ בלעדי לניתוח תחביר של שורה ובו קודדתי פונקציות שמכסות מקרים רבים שנתקל בהם, הן בקלט תקין והן בקלט עם שגיאות סינטקס, בעצם פונקציות עזר אלו מתנקזות לפונקציה יחידה בקובץ שתנתח את התחביר ולה "נקרא" במעברים, כך נעבור על כל האופציות ונקבל מידע על כלל המקרים האפשריים.

לדוגמה יצרנו בנפרד שני פונקציות עזר, אחת שבודקת הנחייה מסוג data. ואחרת מסוג string. כמובן, הבדיקות הם שונות ויש להתייחס בהתאם, אבל בסוף שניהן יהיו בפונקציה הראשית לתחביר.

כל זה נחמד, אבל לאן הולך המידע? מאחר ואנו כל פעמים סורקים שורה ועוברים הלאה, אין לנו צורך באחסון כל השורות במבנה נתונים אשר "גודל" בכל שורה, כלומר יש לנו אלמנט של חילוף בדיקת שורה -> תוצאות סינטקס-> הכנסת תוצאות למבנה נתונים ושימוש ולבסוף איפוס מבנה -> חזרה בדיקת שורה.

עץ סינטקס זה הפתרון במקרה זה אבל לא במובן הקלאסי של עץ, אלא עץ מתחלף כפי שציינתי, אין ברצוננו להוסיף צמתים ולגדול, אלא רק להגדיר תכונות ומבני נתונים לשימוש חד פעמי פר שורה, נעזרתי ב enums על מנת לייצג תכונות, כי הרי בסוף קל לעבוד עם מספרים שמייצגים תכונות מסוימות, ומבני נתונים גם כן בהתאם להוראה/ הנחייה, שמו של העץ זה עץ מתוייג וכך זה מרגיש בעבודה שכל תכונה היא תגית (אינדקס מספרי).

הצגה מופשטת של תתי החלוקות בעץ והאפשרויות השונות בעמוד הבא .



זו הפשטה, שדות ומבני נתונים יותר קונקרטיים בקוד.

מעברים: החלקים הללו קצרים יחסית, לוגיים בעיקרם ומפורטים היטב בקוד ויותר מובנים שם מאשר במלל פה, לכן לא ארחיב שלא לצורך, רק אציין שפה החוזק של ההפשטה בא לידי ביטוי, השורה מתפרקת לתכונות וקל לזהות ולעבוד את השורה.(כלי סופר חזק בכללי לזיהוי כפילויות וNLP בכללי).