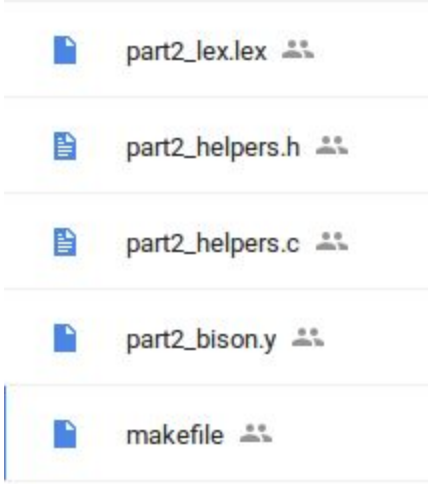
שיטות הידור 046266 – פרויקט – חלק 2

סער אליעד 204402598
אנדריי סמצ'קוב323287029

התכנית שלנו בונה עץ גזירה לשפה דמויית C.

התכנית כוללת את הקבצים הבאים:



דהיינו, הקבצים שסופקו לנו בתוספת makefile, קובץ lex וקובץ bison שכתבנו.

הגדרנו את YYSTYPE בצורה הבאה:

```
/* common struct to use for bison and flex*/
typedef struct {
    pNode node;
} yystype_t;
```

כאשר pNode הוא typedef של פוינטר לnode שניתן לנו עם קבצי ההגשה, כלומר:

```
typedef struct node {
    char * type;
    char * value;
    struct node *sibling;
    struct node *child;
} ParserNode;

typedef ParserNode* pNode;
```

בחרנו להשתמש YYSTYPE הנ"ל כדי לאפשר גמישות בהוספה של משתנים נוספים בעתיד (למרות למטרת התרגיל הנוכחי לא נדרשו כאלו משתנים, ולכן נשאר בו רק node).

מהלך ריצת התכנית הוא כדלהלן:

- כאשר lex מזהה אסימונים הוא יוצר להם node ומעביר אותם לparser.
- בכל פעם שה- parser של bison מבצע reduce , אנחנו:
  - משרשרים את כל הבנים , ראש הרשימה הוא הבן השמאלי ביותר
  - יוצרים node אב שמכיל את כל הבנים שלו , כלומר את הבן השמאלי כבן.
- כשמסיימים , מעבירים את עץ הגזירה שנוצר למשתנה גלובלי, לטובת הדפסה.

הערה: ניסינו לבצע את שרשור כל הבנים באמצעות מאקרו לפי מספר הארגומנטים, אך מכיוון ש $1 הוא משתנה פנימי של מאקרו, ויתרנו על הרעיון ואנחנו משרשרים אותם ידנית. מניחים שזה יהיה שימושי גם לקראת

פתירת קונפליקטים:

- הגדרנו אסוציאטיביות לאופרטורים באמצעות %right, %left לפי חוקי c++.
- הגדרנו עדיפות לאופרטורים לפי סדר ההופעה, כפי שנלמד בתרגול, ולפי חוקי c++.
- פתירת קונפליקטים if else, פתרנו בדיוק כמו שמוצע באופציה הראשונה בתיעוד של bison, כלומר באמצעות שימוש ב-%precedence . – (כלומר, else יהיה צמוד לif הקרוב ביותר שפתוח.)

הנחות מימוש:

- Bottom up parsing
- Positive numbers only
- else יהיה צמוד ל - if הקרוב ביותר שפתוח