

# Bison – An automated tool to create LR(1) parsers

מקבלים דקדוק (כלומר כללי גזירה)

מחזירים parser

# Bison

- הקלט של Bison הוא קובץ ובו כללי הגזירה.
- הפלט הוא מימוש של LR(1) parser בשפת C
- כשהדקדוק לא ניתן ל LR(1) parsing, כלומר שקיימים
  - shift / reduce conflicts
  - reduce / reduce conflicts
- מודפסת אזהרה, והקונפליקט נפתר בצורה שרירותית על ידי bison

# Bison example – calculator grammar

```
%start program
```

```
%%
```

```
program:      E
```

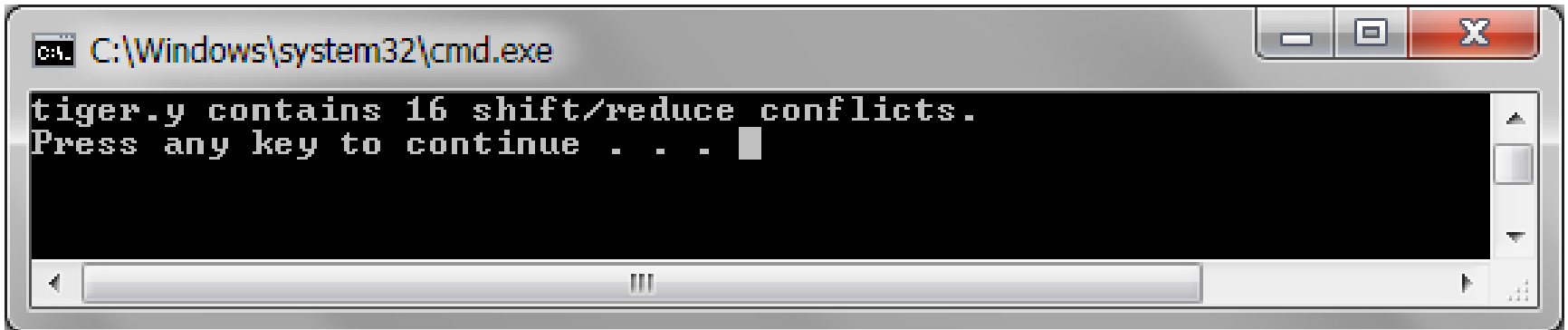
```
E:           INT  
           |   FLOAT  
           |   E PLUS E  
           |   E MINUS E  
           |   E TIMES E  
           |   E DIVIDE E  
           |   LPAREN E RPAREN
```

```
%%
```

- משתנה ההתחלה נקרא program
- הכלל הראשון כאן הוא למעשה:  
program  $\rightarrow$  E\$ (הדולר לא מופיע)
- הקבועים PLUS, MINUS etc.
- מופיעים כ #define בקובץ נפרד

# Bison example – calculator grammar

- כשמריצים את bison עם דקדוק המחשבון, מקבלים את ההודעה הבאה:



```
C:\Windows\system32\cmd.exe  
tiger.y contains 16 shift/reduce conflicts.  
Press any key to continue . . .
```

- כלומר, bison הצליח לבנות parser לדקדוק שלנו, אבל היו קונפליקטים שהוא פתר בעצמו, וחובה של מתכנן הדקדוק לבדוק כיצד הם נפתרו

# Bison example – calculator grammar

- בהנחה ששם הקובץ שלנו הוא tiger.y נוצר באותו directory קובץ שנקרא tiger.output והוא מכיל מידע רב על תהליך הבניה שהתרחש, ובפרט, כיצד פתר את הקונפליקטים שהיו בדקדוק:

State 11 contains 4 shift/reduce conflicts.

State 12 contains 4 shift/reduce conflicts.

State 13 contains 4 shift/reduce conflicts.

State 14 contains 4 shift/reduce conflicts.

# Bison example – calculator grammar

## shift/reduce conflicts in state 11

state 11

```
E -> E . PLUS E    (rule 4)
E -> E PLUS E .    (rule 4)
E -> E . MINUS E    (rule 5)
E -> E . TIMES E     (rule 6)
E -> E . DIVIDE E    (rule 7)
```

```
PLUS      shift, and go to state 6
MINUS     shift, and go to state 7
TIMES     shift, and go to state 8
DIVIDE    shift, and go to state 9
```

```
PLUS      [reduce using rule 4 (E)]
MINUS     [reduce using rule 4 (E)]
TIMES     [reduce using rule 4 (E)]
DIVIDE    [reduce using rule 4 (E)]
$default  reduce using rule 4 (E)
```

Grammar

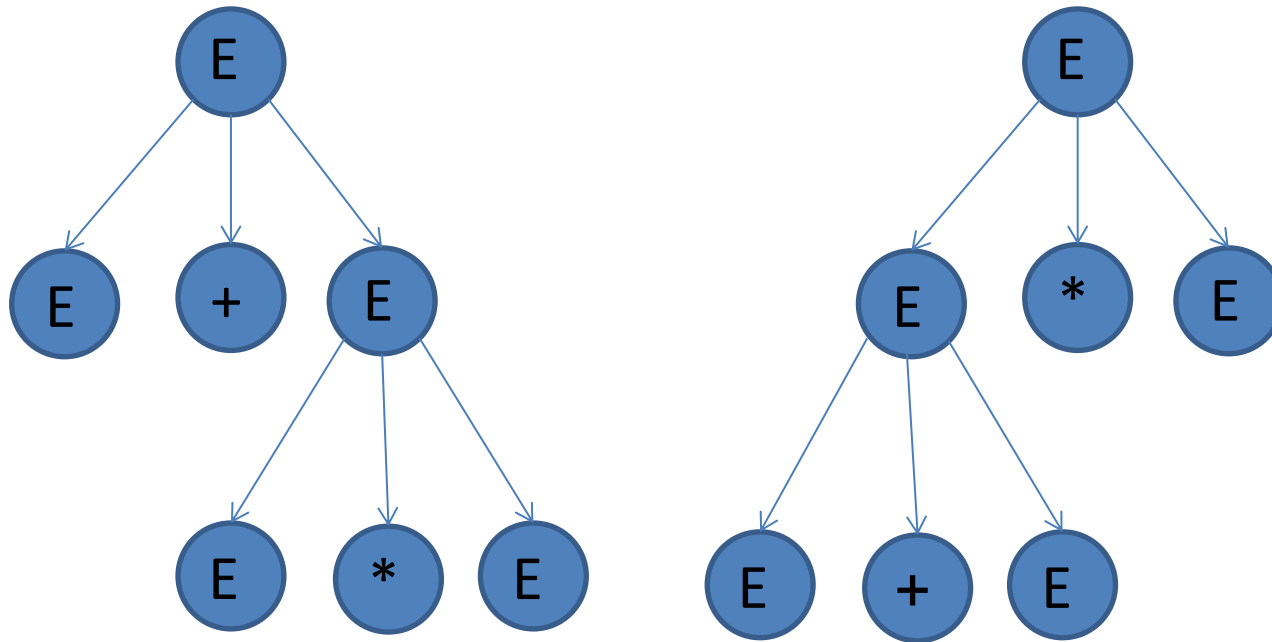
```
rule 1    program -> E
rule 2    E -> INT
rule 3    E -> FLOAT
rule 4    E -> E PLUS E
rule 5    E -> E MINUS E
rule 6    E -> E TIMES E
rule 7    E -> E DIVIDE E
rule 8    E -> LPAREN E RPAREN
```

# אסוציאטיביות של אופרטורים

## קדימויות בין אופרטורים

- עד עכשיו התרכזנו בעיקר בנושא השייכות של מילה לשפה. למשל, בהינתן ביטוי בשפת המחשבון, האם הוא חוקי או לא:  $4+3*8$
- כמובן, שבסופו של דבר, נרצה גם לחשב את ערך הביטוי, ולכן נחזור לעצי גזירה אותם ראינו בפעם שעברה.
- איזה עץ גזירה נרצה שיבנה לביטוי  $4+3*8$ ?

אסוציאטיביות של אופרטורים  
קדימויות בין אופרטורים  
שני עצי גזירה לביטוי  $4+3*8$





# איך אפשר להגדיר קדימויות אופרטורים ב bison

- היזכרו בדקדוק המחשבון ללא קדימויות אופרטורים. היו שם מצבים עם קונפליקטים שפתרנו על ידי בחירת reduce. שם למעשה נתנו קדימות לאופרטור הראשון משמאל שראינו (לא המצב הרצוי) אפשר לשנות את כניסות הטבלה כך שתותאם לקדימות הרגילה של אופרטורים.
- ב bison אפשר להגדיר קדימויות אופרטורים ואסוציאטיביות בצורה מובנית

# הגדרת אסוציאטיביות אופרטורים וקדימויות ב

## bison

```
%left PLUS MINUS  
%left TIMES DIVIDE
```

```
%start program
```

```
%%
```

```
program:      E
```

```
E:           INT  
            | FLOAT  
            | E PLUS E  
            | E MINUS E  
            | E TIMES E  
            | E DIVIDE E  
            | LPAREN E RPAREN
```

```
%%
```

# אכיפת תכנות בטוח על ידי תכונות אופרטורים

- שימו לב להערת אזהרה שנותן visual c++

```
int d = 8 + 4 >> 2;
```

```
warning C4554: '>>' : check operator precedence for possible error;  
use parentheses to clarify precedence
```

ב bison ניתן לאכוף כתיבת סוגריים עבור אופרטורים  
לא אסוציאטיביים כמו חיסור:  $8-(4-2) \neq (8-4)-2$

```
%nonassoc MINUS
```

```
%left PLUS
```

```
%left TIMES DIVIDE
```

אכיפת תכנות בטוח על ידי bison:  
אילוץ כתיבת סוגריים באופרטורים לא אסוציאטיביים

- כשהגדרנו שפעולת החיסור היא חסרת

אסוציאטיביות:

```
%nonassoc MINUS  
%left PLUS  
%left TIMES DIVIDE
```

- נקבל בהרצת הקלט  $44 - 33 - 22$  שגיאה, מכיוון שרוצים לאלץ את המתכנת לפרט איך לבצע את החיסור

# Bison הסברים נוספים

- פורמט קובץ הקלט `tiger.y`
- קבצי הפלט `tiger.output` `tiger.tab.h` `tiger.tab.c`