

**Студент: А.А.Довженко
Группа: М80-207Б
Номер по списку: 6**

Тема: Синтаксический анализ.

Лабораторные работы N6-7.

Вариант 6.

v06=>(cond((< \$zero \$dec)\$dec)(\$bool \$id))

Распечатка грамматики.

\$v06

**\$dec \$zero \$id \$bool
() < cond**

#

COND -> HCOND CLAUS) #1

**HCOND -> (cond #2 |
HCOND CLAUS #3**

CLAUS -> (BOOL E) #4

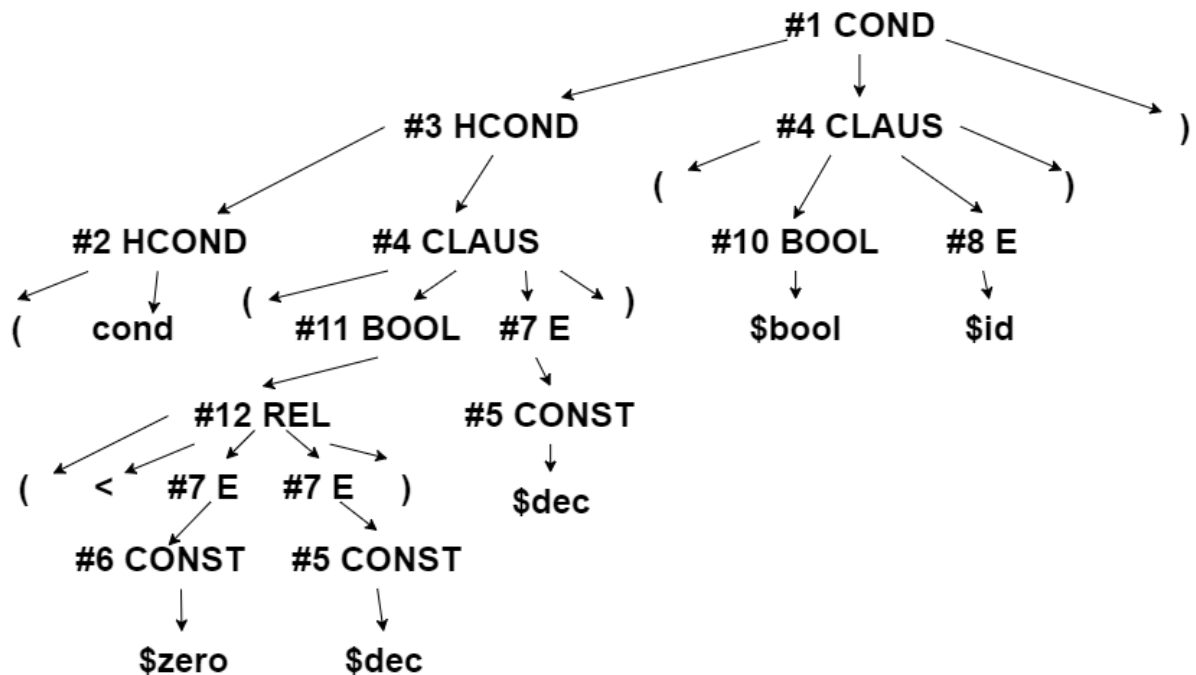
**CONST -> \$dec #5 |
\$zero #6**

**E -> CONST #7 |
\$id #8 |
COND #9**

**BOOL -> \$bool #10 |
REL #11**

REL -> (< E E) #12

Дерево разбора.



Скриншот запуска парсера для своего варианта задания с трассировкой сверток.

karma@DESKTOP-K0CDBM7: /mnt/c/Users/Karma/Desktop/sp18/lab07

```
karma@DESKTOP-K0CDBM7: /mnt/c/Users/Karma/Desktop/sp18/lab07$ ./a.out
Input grammar name>v06
Grammar: v06.txt
Source>(cond(((< 0 5.0)1.01e+0)(#t hello!)))
Source: temp.ss
  1|(cond(((< 0 5.0)1.01e+0)(#t hello!)))
  2|
```

AAD2018

```
<- (
<- cond
  HCOND -> ( cond #2
<- (
<- (
<- <
<- $zero
  CONST -> $zero #6
    E -> CONST #7
<- $dec
  CONST -> $dec #5
    E -> CONST #7
<- )
  REL -> ( < E E ) #12
  BOOL -> REL #11
<- $dec
  CONST -> $dec #5
    E -> CONST #7
<- )
  CLAUS -> ( BOOL E ) #4
  HCOND -> HCOND CLAUS #3
<- (
<- $bool
  BOOL -> $bool #10
<- $id
  E -> $id #8
<- )
  CLAUS -> ( BOOL E ) #4
<- )
  COND -> HCOND CLAUS ) #1
Good source!
```

Скриншот запуска парсера для своего файла coin.ss из лабораторной №3 в грамматике mlisp18 **без** трассировки сверток.

```
Выбрать karma@DESKTOP-K0CDBM7: /mnt/c/Users/Karma/Desktop/sp18/lab07
karma@DESKTOP-K0CDBM7: /mnt/c/Users/Karma/Desktop/sp18/lab07$ ./a.out
Input gramma name>mlisp18
Gramma:mlisp18.txt
Source>coin
Source:coin.ss
1|(define dd 2)
2|(define mm 12)
3|(define LAGEST-COIN 10)
4|
5|(define (cc amount largest-coin)
6|  (cond((or (= amount 0)(= largest-coin 1))
7|    1)
8|    ((not (and (> amount 0) (> largest-coin 0)))
9|      0)
10|    (else (+ (cc amount (next-coin largest-coin)) (cc (- amount largest-coin) largest-coin)
11|      ))
12|  )
13|)
14|
15|(define (count-change amount)
16|  (cc amount LAGEST-COIN)
17|)
18|
19|(define (next-coin coin)
20|  (cond((= coin 10) 5)
21|    ((= coin 5) 3)
22|    ((= coin 3) 2)
23|    ((= coin 2) 1)
24|    (else 0)
25|  )
26|)
27|
28|(define (GR-AMOUNT) (+ (* 100 mm) dd))
29|
30|(display " AAD variant 6")(newline)
31|(display " 1-2-3-5-10")(newline)
32|(display "count__change for 100 \t= ")
33|(display (count-change 100))(newline)
34|(display "count__change for ")
35|(display (GR-AMOUNT))
36|(display " \t= ")
37|(display(count-change (GR-AMOUNT)))(newline)
38|
Good source!
```

Выводы.

В процессе выполнения лабораторных работ я вспомнила ход построения деревьев разбора. С подобной задачей я сталкивалась на первом курсе, когда строила дерево разбора для арифметического выражения. Те навыки пригодились мне в понимании данной работы.

Суть состоит в том, что задана определенная грамматика, согласно которой мы используем продукции, чтобы из листьев дерева слева направо вышло данное выражение. Несмотря на то, что эти работы показались мне достаточно несложными, тем не менее, на мой взгляд, они весьма полезны. Задания лабораторных работ выполнены в полном объеме.