Project #2. Parser

컴퓨터소프트웨어학부 2018008813 이동균

Environment

Ubuntu 18.04

main.c

NO_ANALYZE와 TraceParse를 True로 바꿔준다.

globals.h

Yacc/globals.h에 담긴 globals.h를 가져오고 필요한 node의 type들을 추가해주고, 이에 맞게 struct treeNode의 구조를 바꿔준다.

util.h

util.c에서 새로 추가된 newDecNode를 사용할 수 있도록 util.h에 선언해준다.

util.c

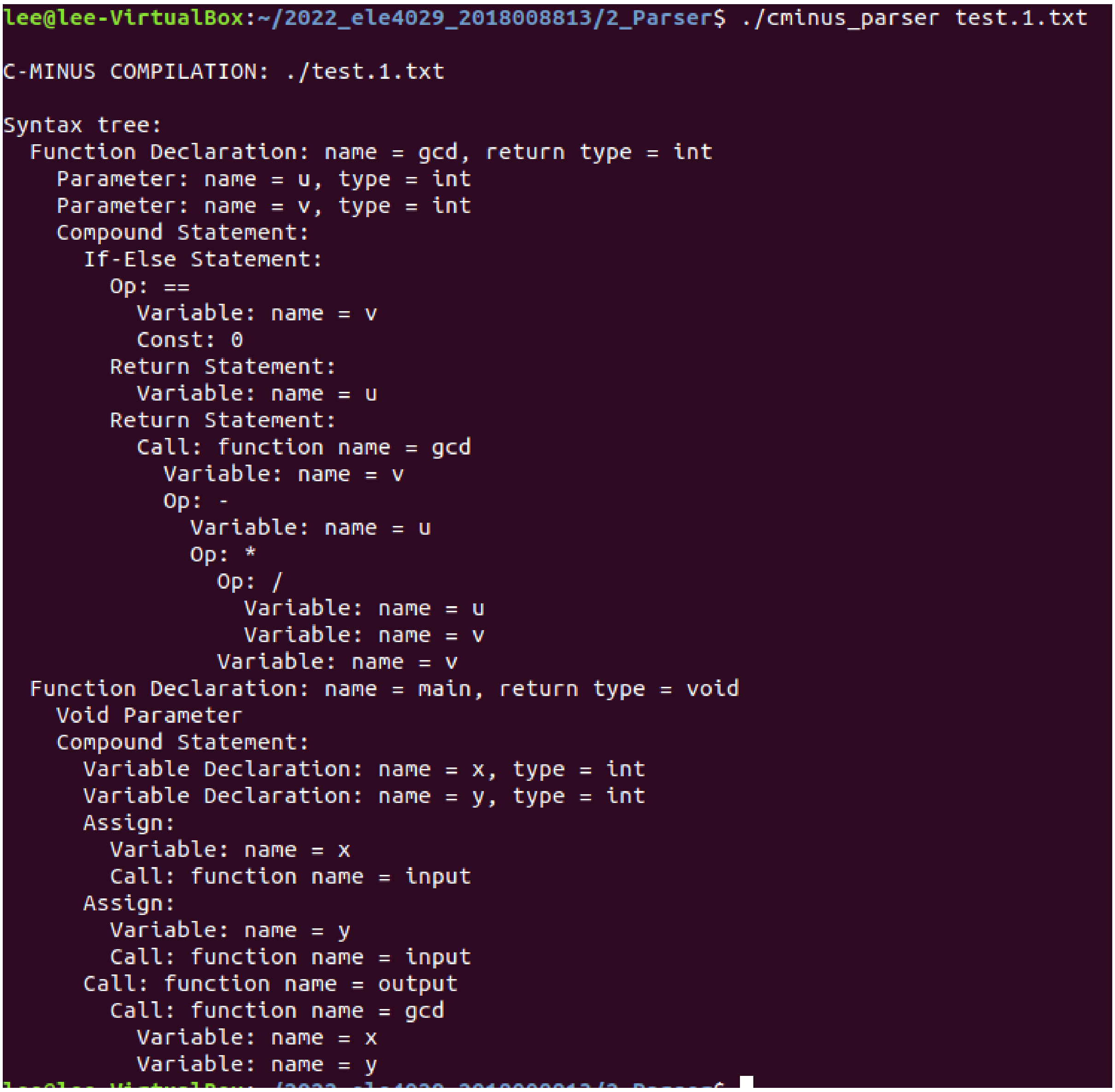
기존 함수와 유사하게 newDecNode 함수를 정의해주고, printTree에서는 새로 추가한 type들이 올바른 형식에 맞게 출력되도록 수정한다.

cminus.y

사용하는 token들을 추가해주고 BNF Grammer Rule에 따라 코드를 작성한다.

Result

test.1.txt

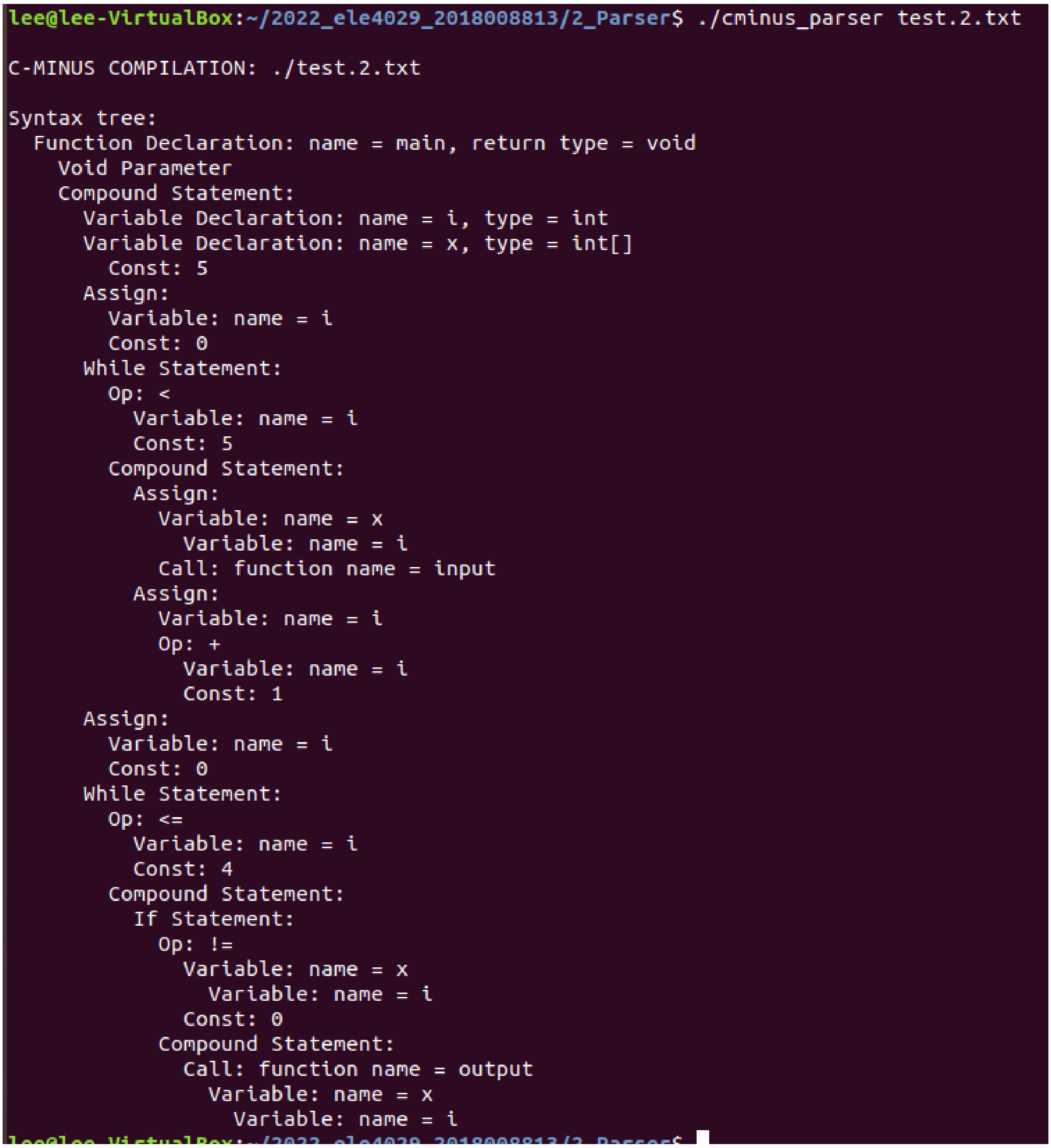


```
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$ ./cminus_parser test.1.txt

C-MINUS COMPILATION: ./test.1.txt

Syntax tree:
  Function Declaration: name = gcd, return type = int
    Parameter: name = u, type = int
    Parameter: name = v, type = int
    Compound Statement:
      If-Else Statement:
        Op: ==
          Variable: name = v
          Const: 0
        Return Statement:
          Variable: name = u
        Return Statement:
          Call: function name = gcd
            Variable: name = v
            Op: -
              Variable: name = u
              Op: *
                Op: /
                  Variable: name = u
                  Variable: name = v
                Variable: name = v
  Function Declaration: name = main, return type = void
    Void Parameter
    Compound Statement:
      Variable Declaration: name = x, type = int
      Variable Declaration: name = y, type = int
      Assign:
        Variable: name = x
        Call: function name = input
      Assign:
        Variable: name = y
        Call: function name = input
      Call: function name = output
        Call: function name = gcd
          Variable: name = x
          Variable: name = y
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$
```

test.2.txt



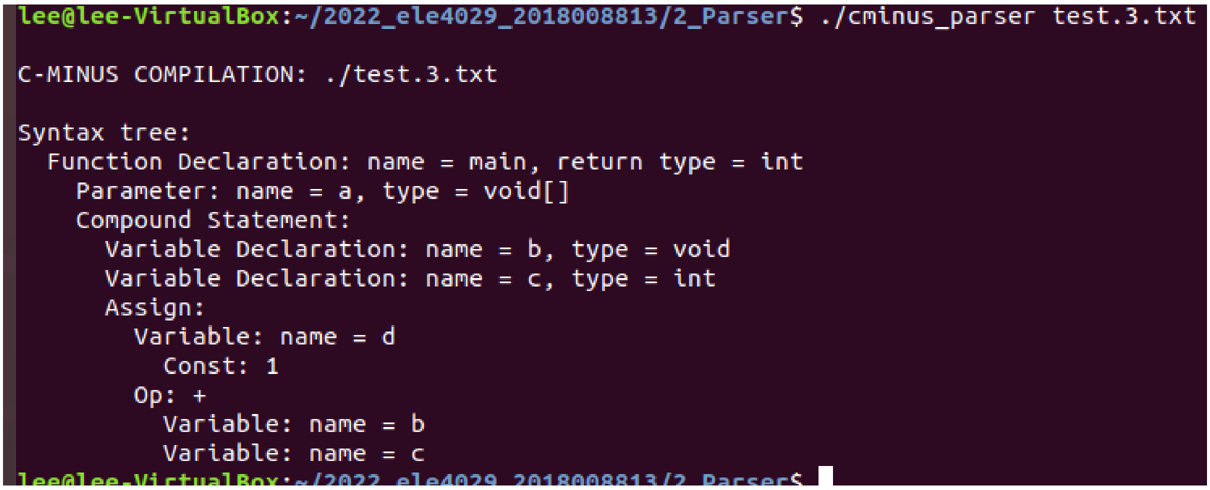
```
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$ ./cminus_parser test.2.txt

C-MINUS COMPILATION: ./test.2.txt

Syntax tree:
  Function Declaration: name = main, return type = void
    Void Parameter
    Compound Statement:
      Variable Declaration: name = i, type = int
      Variable Declaration: name = x, type = int[]
        Const: 5
      Assign:
        Variable: name = i
        Const: 0
      While Statement:
        Op: <
          Variable: name = i
          Const: 5
        Compound Statement:
          Assign:
            Variable: name = x
              Variable: name = i
            Call: function name = input
          Assign:
            Variable: name = i
            Op: +
              Variable: name = i
              Const: 1
      Assign:
        Variable: name = i
        Const: 0
      While Statement:
        Op: <=
          Variable: name = i
          Const: 4
        Compound Statement:
          If Statement:
            Op: !=
              Variable: name = x
                Variable: name = i
              Const: 0
            Compound Statement:
              Call: function name = output
                Variable: name = x
                  Variable: name = i
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$ 
```

결과는 위와 같이 PDF와 동일하게 출력됩니다.

test.3.txt



```
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$ ./cminus_parser test.3.txt

C-MINUS COMPILATION: ./test.3.txt

Syntax tree:
  Function Declaration: name = main, return type = int
    Parameter: name = a, type = void[]
    Compound Statement:
      Variable Declaration: name = b, type = void
      Variable Declaration: name = c, type = int
      Assign:
        Variable: name = d
          Const: 1
        Op: +
          Variable: name = b
          Variable: name = c
lee@lee-VirtualBox:~/2022_ele4029_2018008813/2_Parser$
```

Semantic error가 있어도 제대로 parsing되는 것을 확인할 수 있습니다.