# 컴파일러설계

학번 : 2021029889
이름 : 오치현

## 컴파일 환경 및 방법

노트북 : M2 MAC PRO
터미널 : Warp
방법 : make clean; make all; ./cminus_parser ./example/{test_file};

## 어떻게 구현했고 어떻게 동작하는지

처리하는 형태는 크게 리스트, 연산, 선언, 그룹이 있습니다.
먼저, 선언의 경우 식별자, 숫자, 타입, 연산 기호가 있습니다. 이 경우에 해당하는 형식의 타입 노드를 생성 후 라인 넘버와 값을 집어넣습니다. 연산의 경우 트리를 형성해갑니다. 연산자가 높은 것은 낮은 것을 참조하도록 만들고, 높은 것은 right reculsive 하게 만듭니다. 그룹의 경우 해당 위치의 노드를 엔트리 포인트로 다시 재지정하여(e.t. $$ = $1;) 다음 shift 시에 생성되도록 합니다. 또는 RHS 에 대해 각 노드를 자식 노드로 지정하고, 해당 타입의 엔트리 포인트를 자신으로 하여 노드를 생성합니다. 리스트의 경우 left reculsive하고, sibling 으로 원소를 연결합니다.
자세한 사항은 코드에 나와있습니다.

# 결과 화면

첫 번째 테스트 코드

```
/* A program to perform Euclid's
   Algorithm to computer gcd */

int gcd (int u, int v)
{
   if (v == 0) return u;
    else return gcd(v,u-u/v*v);
    /* u-u/v*v == u mod v */
}

void main(void)
{
    int x; int y;
    x = input(); y = input();
    output(gcd(x,y));
}
```

첫 번째 테스트 결과

```
C-MINUS COMPILATION: example/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
```

두 번째 테스트 코드

```
void main(void)
{
    int i; int x[5];

    i = 0;
    while( i < 5 )
    {
        x[i] = input();

        i = i + 1;
    }

    i = 0;
    while( i <= 4 )
    {
        if( x[i] != 0 )
        {
            output(x[i]);
        }
    }
}
```

두 번째 테스트 결과

```
./cminus_parser example/test.2.txt

C-MINUS COMPILATION: example/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
```

세 번째 테스트 코드

```
cat example/test.3.txt
/* multiple statement TEST */

int test(void)
{
    return 1;
}

void main(void)
{
    int a; int b; int c; int d[100];

    a = 0;
    while (a < 100) {
        d[a] = 0;
        a = a + 1;
    }

    a = 0;
    while (a < 100)
    {
        b = 0;
        while (b < 100)
        {
                else if (b < 60)
                {
                    d[b] = d[b - a + 30];
                }
                else
                {
                    c = 1;
                    while (c < 10)
                    {
                        d[c] = d[b] + 99;
                    }
                }
            }
            else
            {
                if (b > 90)
                {
                    if (test())
                    {
                        output(1);
                    }
```

세 번째 테스트 결과

```
./cminus_parser example/test.3.txt

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

Syntax tree:
  Function Declaration: name = test, return type = int
    Void Parameter
    Compound Statement:
      Return Statement:
        Const: 1
  Function Declaration: name = main, return type = void
    Void Parameter
    Compound Statement:
      Variable Declaration: name = a, type = int
      Variable Declaration: name = b, type = int
      Variable Declaration: name = c, type = int
      Variable Declaration: name = d, type = int[]
        Const: 100
      Assign:
        Variable: name = a
        Const: 0
      While Statement:
        Op: <
          Variable: name = a
          Const: 100
        Compound Statement:
          Assign:
            Variable: name = d
              Variable: name = a
            Const: 0
          Assign:
            Variable: name = a
            Op: +
              Variable: name = a
              Const: 1
      Assign:
        Variable: name = a
        Const: 0
      While Statement:
        Op: <
          Variable: name = a
          Const: 100
        Compound Statement:
          Assign:
            Variable: name = b
            Const: 0
            While Statement:
              Op: <
                Variable: name = b
                Const: 30
              Compound Statement:
                Assign:
                  Variable: name = d
                    Variable: name = b
                  Op: +
                    Variable: name = d
                      Variable: name = a
                    Const: 60
                If-Else Statement:
                  Op: <
                    Variable: name = b
                    Const: 60
                  Compound Statement:
                    Assign:
                      Variable: name = d
                        Variable: name = b
                      Variable: name = d
                        Op: +
                          Op: -
                            Variable: name = b
                            Variable: name = a
                          Const: 30
                    Compound Statement:
                      Assign:
                        Variable: name = c
                        Const: 1
                      While Statement:
                        Op: <
                          Variable: name = c
                          Const: 10
                        Compound Statement:
                          Assign:
                            Variable: name = d
                              Variable: name = c
                            Op: +
                              Variable: name = d
                                Variable: name = b
                              Const: 99
                  Compound Statement:
                    If-Else Statement:
                      Op: >
                        Variable: name = b
                        Const: 90
                      Compound Statement:
                        If-Else Statement:
                          Op: <
                            Variable: name = a
                            Const: 30
                          Compound Statement:
                            If-Else Statement:
                              Op: <
                                Variable: name = b
                                Const: 30
                              Compound Statement:
                                Assign:
                                  Variable: name = d
                                    Variable: name = b
                                  Op: +
                                    Variable: name = d
                                      Variable: name = c
                                    Const: 3
                                If-Else Statement:
                                  Op: <
                                    Variable: name = b
                                    Const: 60
                                  Compound Statement:
                                    Assign:
                                      Variable: name = d
                                        Variable: name = b
                                      Op: +
                                        Variable: name = d
                                          Variable: name = a
                                        Const: 100
                                  Compound Statement:
                                    Assign:
                                      Variable: name = d
                                        Variable: name = a
                                      Op: -
                                        Variable: name = d
                                          Variable: name = b
                                        Const: 100
                                If-Else Statement:
                                  Op: <
                                    Variable: name = a
                                    Const: 60
                                  Compound Statement:
                                    If-Else Statement:
                                      Op: <
                                        Variable: name = b
                                        Const: 60
                                      Compound Statement:
                                        Call: function name = test
                                      Compound Statement:
                                        Call: function name = outpu
                                          Const: 1
                                      Compound Statement:
                                        Call: function name = outpu
                                          Variable: name = d
                                            Variable: name = b
                                    Compound Statement:
                                      Assign:
                                        Variable: name = d
                                          Variable: name = error
                                        Op: -
                                          Variable: name = d
                                            Variable: name = error
                                          Variable: name = error
                                  Assign:
                                    Variable: name = b
                                    Op: +
                                      Variable: name = b
                                      Const: 1
```

네 번째 테스트 코드

```
cat example/test.4.txt
/* paren in operation test */

int test(int a, int b)
{
    int result;
    result = 3 * ((6 - 7 * (5 - 1)) + 88 - (((3 - 1) * a + (b - 0) / 4) - 2 * 9) - (4 - 3 * (6 * 7 * (2 + 1)))) + 4;
    return result;
}

void main(void)
{
    int a; int b; int c; int d;
    a = 100; b = 200;

    output(test(a, b));
}
```

네 번째 테스트 결과

```
./cminus_parser example/test.4.txt

C-MINUS COMPILATION: example/test.4.txt

Syntax tree:
  Function Declaration: name = test, return type = int
    Parameter: name = a, type = int
    Parameter: name = b, type = int
    Compound Statement:
      Variable Declaration: name = result, type = int
      Assign:
        Variable: name = result
        Op: +
          Op: *
            Const: 3
            Op: -
              Op: -
                Op: +
                  Op: -
                    Const: 6
                    Op: *
                      Const: 7
                      Op: -
                        Const: 5
                        Const: 1
                  Const: 88
                Op: -
                  Op: +
                    Op: *
                      Op: -
                        Const: 3
                        Const: 1
                      Variable: name = a
                    Op: /
                      Op: -
                        Variable: name = b
                        Const: 0
                      Const: 4
                  Op: *
                    Const: 2
                    Const: 9
              Op: -
                Const: 4
                Op: *
                  Const: 3
                  Op: *
                    Op: *
                      Const: 6
                      Const: 7
                    Op: +
                      Const: 2
                      Const: 1
            Const: 4
      Return Statement:
        Variable: name = result
  Function Declaration: name = main, return type = voi
    Void Parameter
    Compound Statement:
      Variable Declaration: name = a, type = int
      Variable Declaration: name = b, type = int
      Variable Declaration: name = c, type = int
      Variable Declaration: name = d, type = int
      Assign:
        Variable: name = a
        Const: 100
      Assign:
        Variable: name = b
        Const: 200
      Call: function name = output
        Call: function name = test
          Variable: name = a
          Variable: name = b
```