Project #3. Semantic

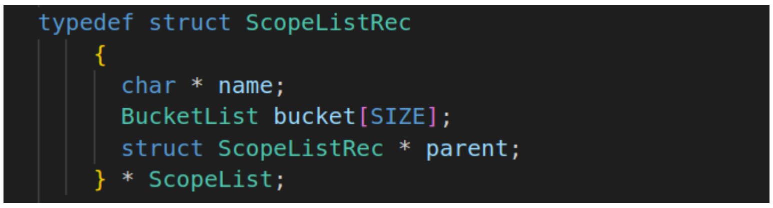
컴퓨터소프트웨어학부 2020079689 신다혜

1. Environment

- Windows 11 Pro, Ubuntu 16.04 LTS
- Visual Studio Code 1.71.2
- gcc (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0
- lex 2.6.0
- bison (GNU Bison) 3.0.4

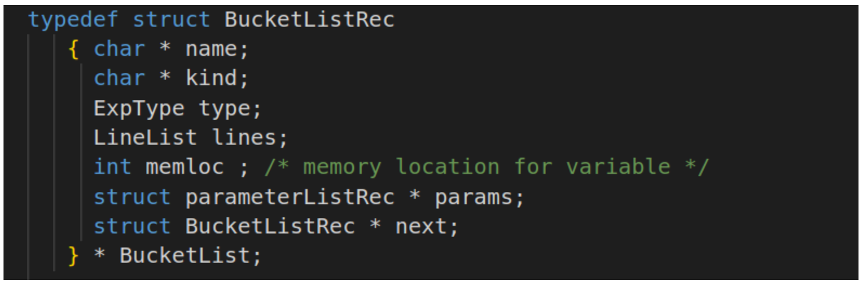
2. Implementation

- *symtab.h*



```
typedef struct ScopeListRec
    {
      char * name;
      BucketList bucket[SIZE];
      struct ScopeListRec * parent;
    } * ScopeList;
```

Scope 저장 공간



```
typedef struct BucketListRec
   { char * name;
     char * kind;
     ExpType type;
     LineList lines;
     int memloc ; /* memory location for variable */
     struct parameterListRec * params;
     struct BucketListRec * next;
   } * BucketList;
```

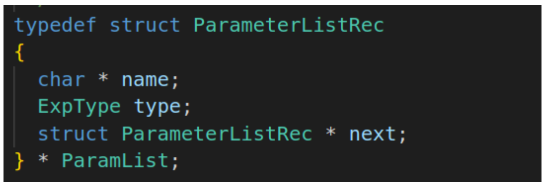
cminus에 맞게 구조체 수정.

name – symbol name

kind – Function/Variable

type – Integer/Void

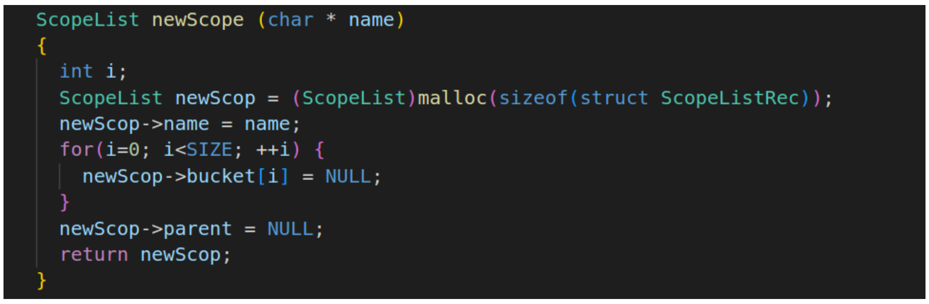
params – Function일 경우 parameter 저장 공간



```
typedef struct ParameterListRec
{
  char * name;
  ExpType type;
  struct ParameterListRec * next;
} * ParamList;
```

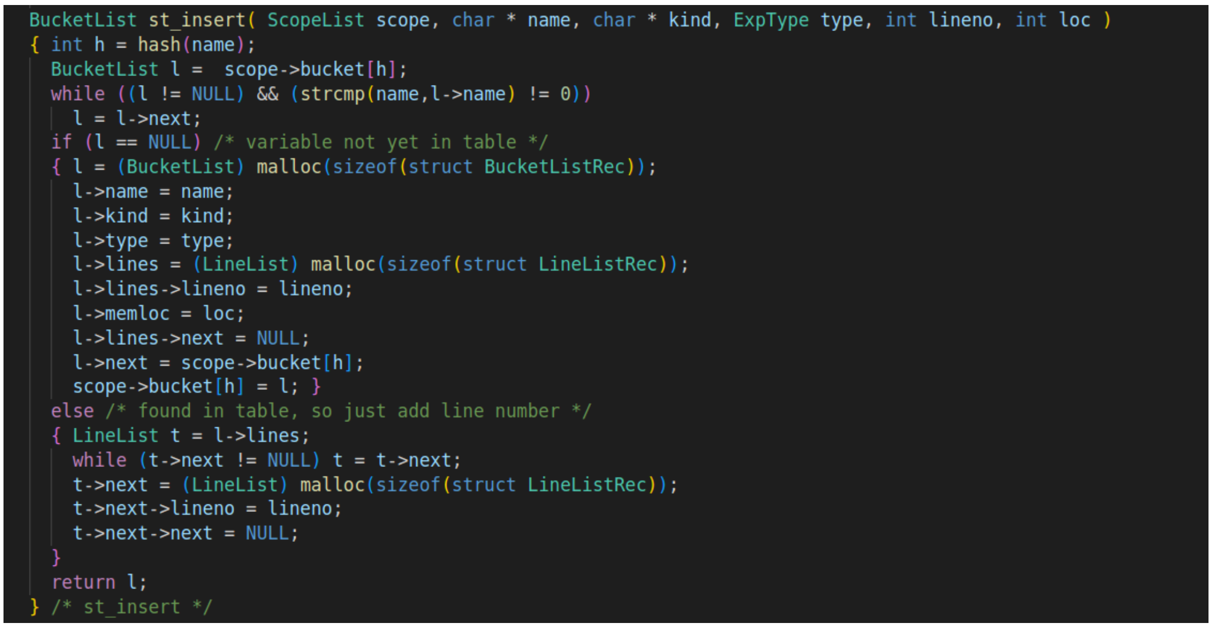
function parameter의 name과 type 저장

- *symtab.c*



```
ScopeList newScope (char * name)
{
  int i;
  ScopeList newScop = (ScopeList)malloc(sizeof(struct ScopeListRec));
  newScop->name = name;
  for(i=0; i<SIZE; ++i) {
    newScop->bucket[i] = NULL;
  }
  newScop->parent = NULL;
  return newScop;
}
```

새로운 scope 생성 후 반환하는 함수. scope 구조체 내 모든 것을 초기화.



```
BucketList st_insert( ScopeList scope, char * name, char * kind, ExpType type, int lineno, int loc )
{ int h = hash(name);
  BucketList l =  scope->bucket[h];
  while ((l != NULL) && (strcmp(name,l->name) != 0))
    l = l->next;
  if (l == NULL) /* variable not yet in table */
  { l = (BucketList) malloc(sizeof(struct BucketListRec));
    l->name = name;
    l->kind = kind;
    l->type = type;
    l->lines = (LineList) malloc(sizeof(struct LineListRec));
    l->lines->lineno = lineno;
    l->memloc = loc;
    l->lines->next = NULL;
    l->next = scope->bucket[h];
    scope->bucket[h] = l; }
  else /* found in table, so just add line number */
  { LineList t = l->lines;
    while (t->next != NULL) t = t->next;
    t->next = (LineList) malloc(sizeof(struct LineListRec));
    t->next->lineno = lineno;
    t->next->next = NULL;
  }
  return l;
} /* st_insert */
```

symbol table in cminus 구조에 맞게 수정

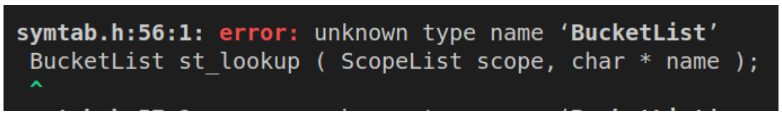
print*Tab (File->void) 함수들 – pdf에서 주어진 각 table 출력 형식에 일치하도록 작성

- *analyze.c*

insertNode, checkNode 함수에서 인자로 받은 TreeNode의 nodekind에 따라 적절한 동작을 수행하도록 switch-case로 나눠서 작성

3. Problem Shooting

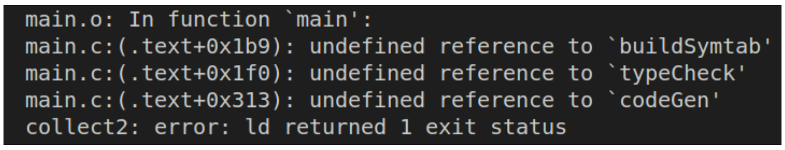
3-1. unknown type name



```
symtab.h:56:1: error: unknown type name ‘BucketList’
 BucketList st_lookup ( ScopeList scope, char * name );
 ^
```

- symtab.c에 정의된 struct BucketList, ScopeList를 symtab.h에서 정의하도록 변경

3-2. undefined reference to



```
main.o: In function `main':
main.c:(.text+0x1b9): undefined reference to `buildSymtab'
main.c:(.text+0x1f0): undefined reference to `typeCheck'
main.c:(.text+0x313): undefined reference to `codeGen'
collect2: error: ld returned 1 exit status
```

- Makefile – symtab.o, analyze.o 추가

4. Run

```
$ make
$ ./cminus_parser {filename.cm}
```

5. Expected Result

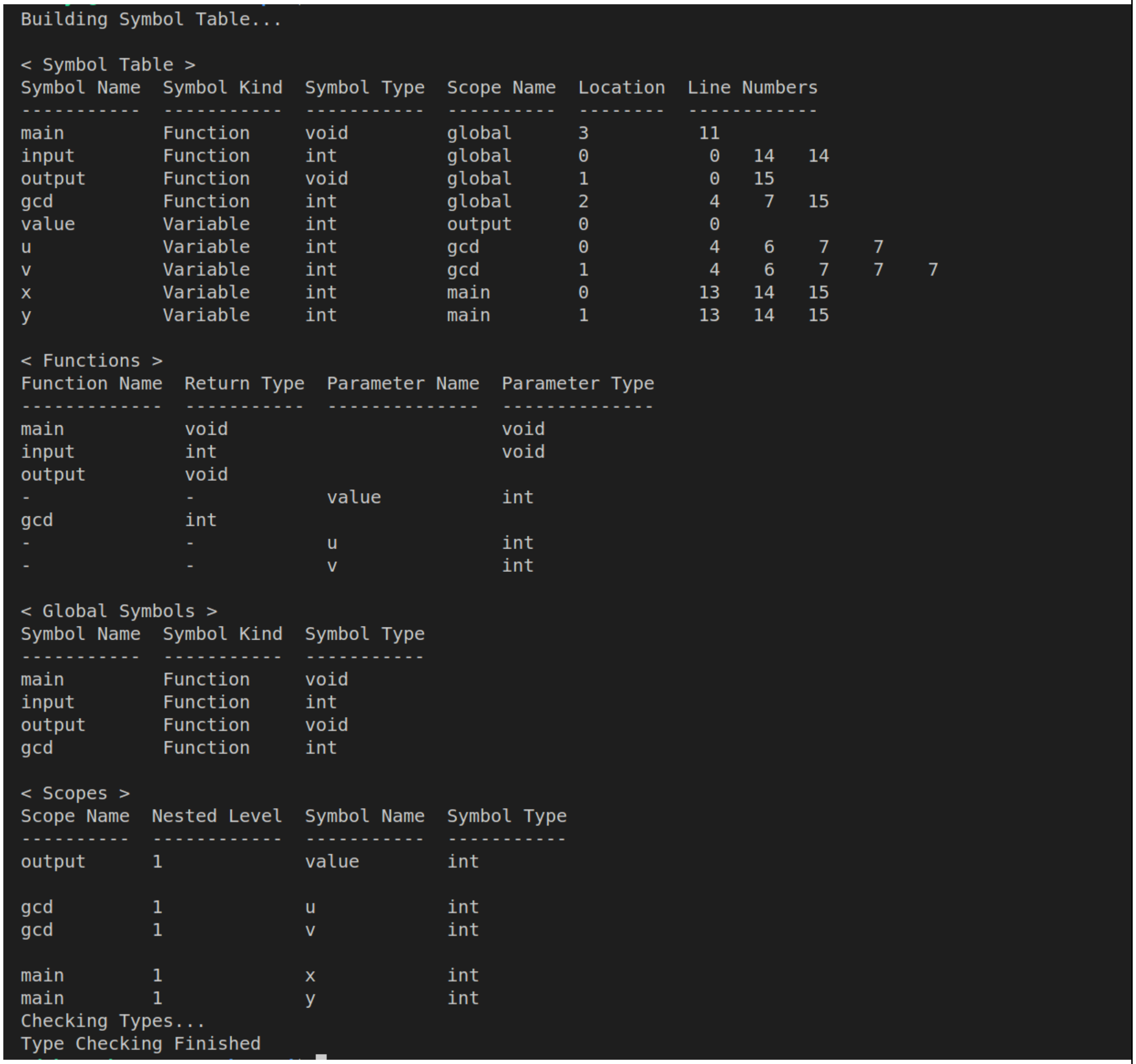
```
test.cm

/* 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));
}
```

output (console)



```
Building Symbol Table...

< Symbol Table >
Symbol Name  Symbol Kind  Symbol Type  Scope Name  Location  Line Numbers
-----------  -----------  -----------  ----------  --------  ------------
main         Function     void         global      3          11
input        Function     int          global      0           0   14   14
output       Function     void         global      1           0   15
gcd          Function     int          global      2           4    7   15
value        Variable     int          output      0           0
u            Variable     int          gcd         0           4    6    7    7
v            Variable     int          gcd         1           4    6    7    7    7
x            Variable     int          main        0          13   14   15
y            Variable     int          main        1          13   14   15

< Functions >
Function Name  Return Type  Parameter Name  Parameter Type
-------------  -----------  --------------  --------------
main           void                         void
input          int                          void
output         void
-              -            value           int
gcd            int
-              -            u               int
-              -            v               int

< Global Symbols >
Symbol Name  Symbol Kind  Symbol Type
-----------  -----------  -----------
main         Function     void
input        Function     int
output       Function     void
gcd          Function     int

< Scopes >
Scope Name  Nested Level  Symbol Name  Symbol Type
----------  ------------  -----------  -----------
output      1             value        int

gcd         1             u            int
gcd         1             v            int

main        1             x            int
main        1             y            int
Checking Types...
Type Checking Finished
```