TSL - Symbol Table & AST & Code Generation

Oh조

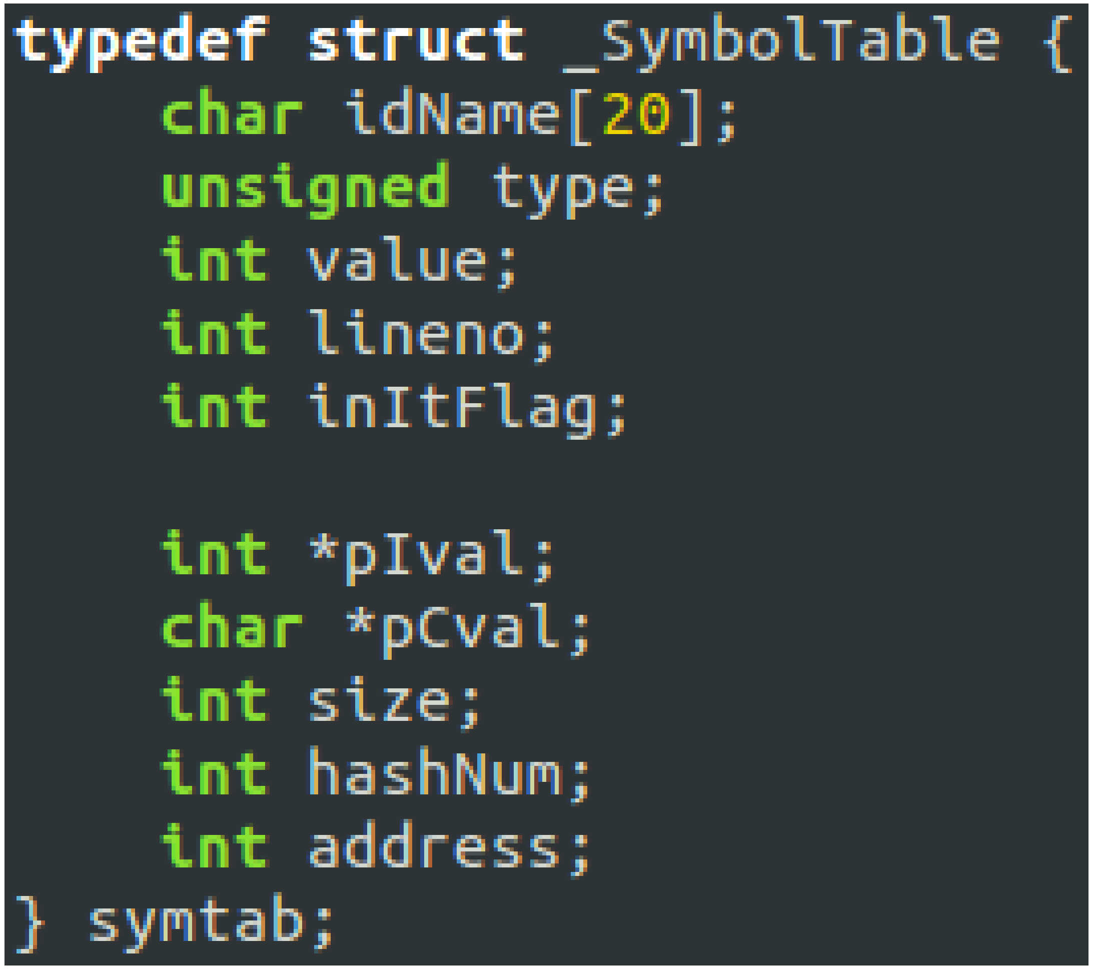
권성철(200924413)

조용래(201224540)

발표 순서

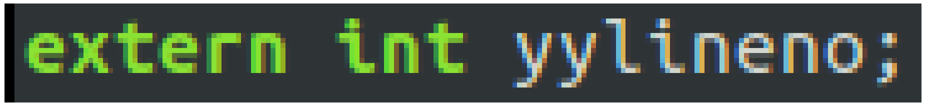
1. Symbol Table
2. AST
3. Code Generation
4. 시연
5. Q&A

Symbol Table - 구조



```
typedef struct _SymbolTable {
    char idName[20];
    unsigned type;
    int value;
    int lineno;
    int inItFlag;

    int *pIval;
    char *pCval;
    int size;
    int hashNum;
    int address;
} symtab;
```

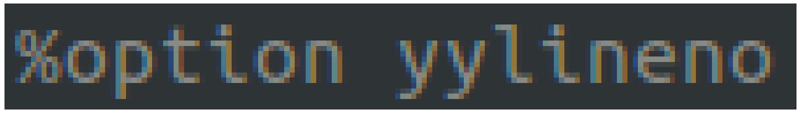
Symbol Table - lineno, hash



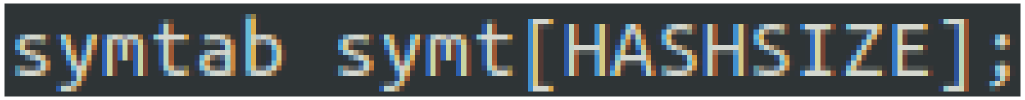
```
extern int yylineno;
```

yacc

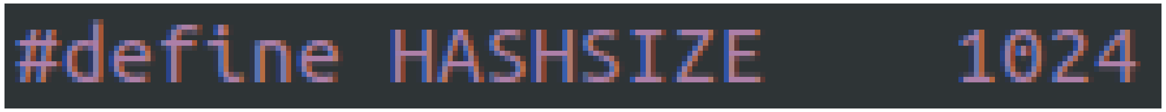
lex



```
%option yylineno
```

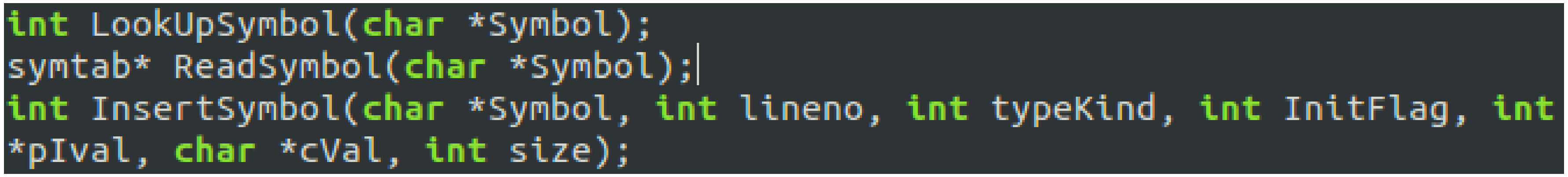


```
symtab symt[HASHSIZE];
```



```
#define HASHSIZE    1024
```

Symbol Table – 인터페이스 함수

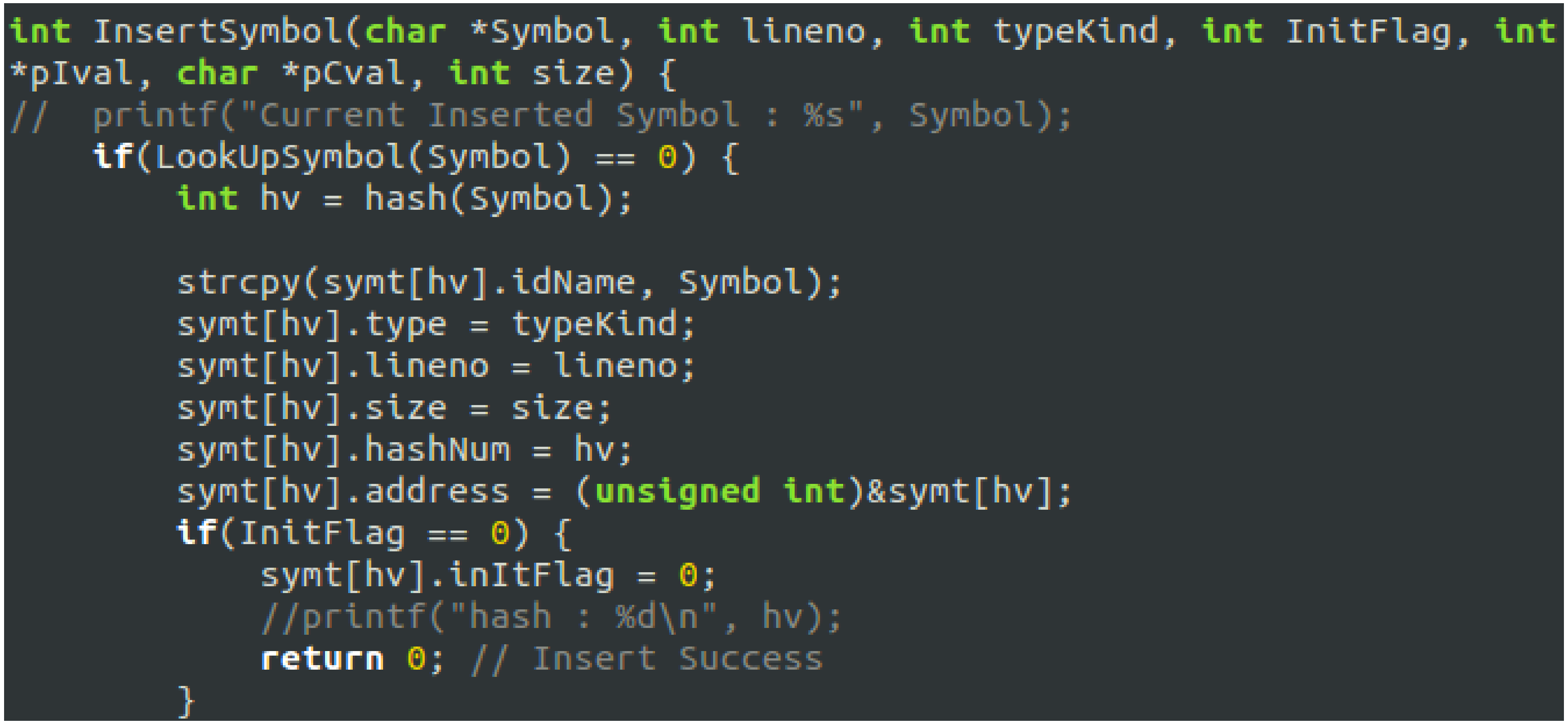


```
int LookUpSymbol(char *Symbol);
symtab* ReadSymbol(char *Symbol);
int InsertSymbol(char *Symbol, int lineno, int typeKind, int InitFlag, int
*pIval, char *cVal, int size);
```

Inserted Elements

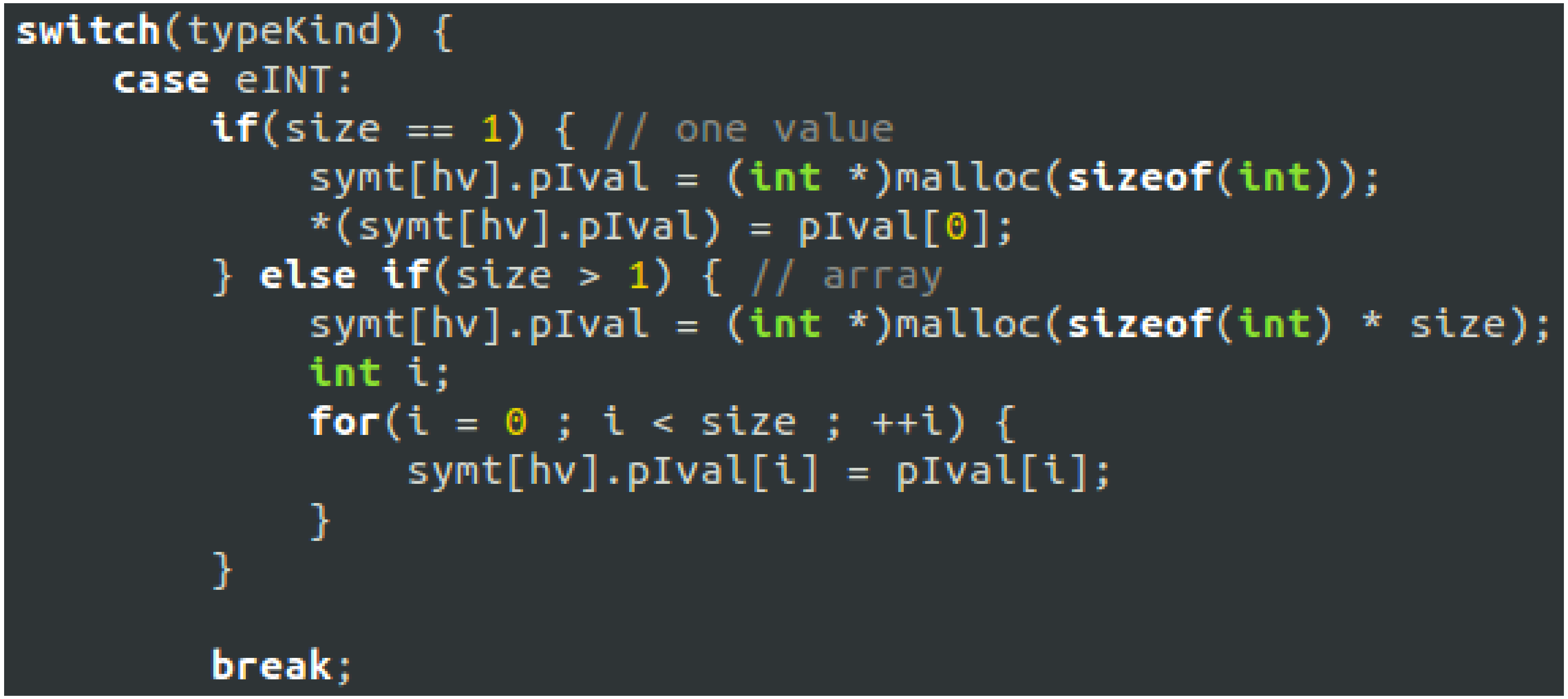
- Name
- Line number
- Type
- InitFlag
- Init Value
- Size

Symbol Table - InsertSymbol



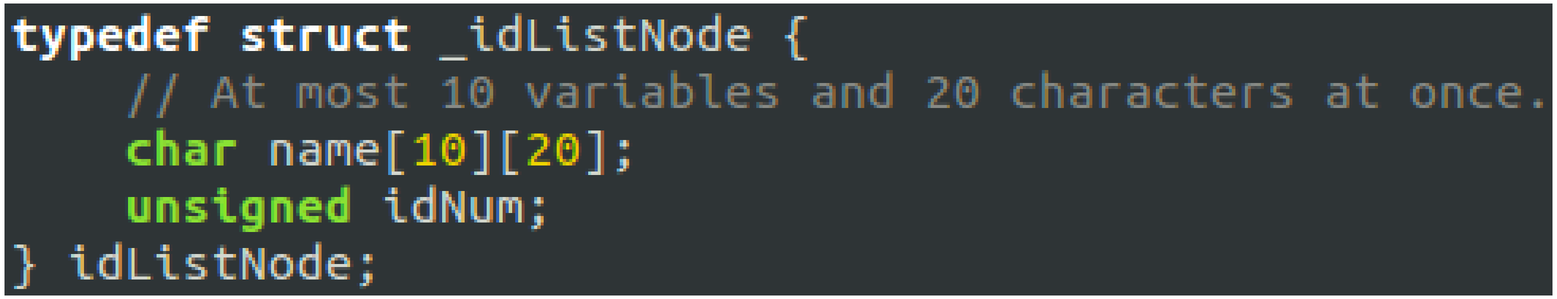
```
int InsertSymbol(char *Symbol, int lineno, int typeKind, int InitFlag, int
*pIval, char *pCval, int size) {
//  printf("Current Inserted Symbol : %s", Symbol);
    if(LookUpSymbol(Symbol) == 0) {
        int hv = hash(Symbol);

        strcpy(symt[hv].idName, Symbol);
        symt[hv].type = typeKind;
        symt[hv].lineno = lineno;
        symt[hv].size = size;
        symt[hv].hashNum = hv;
        symt[hv].address = (unsigned int)&symt[hv];
        if(InitFlag == 0) {
            symt[hv].inItFlag = 0;
            //printf("hash : %d\n", hv);
            return 0; // Insert Success
        }
```

Symbol Table - InsertSymbol

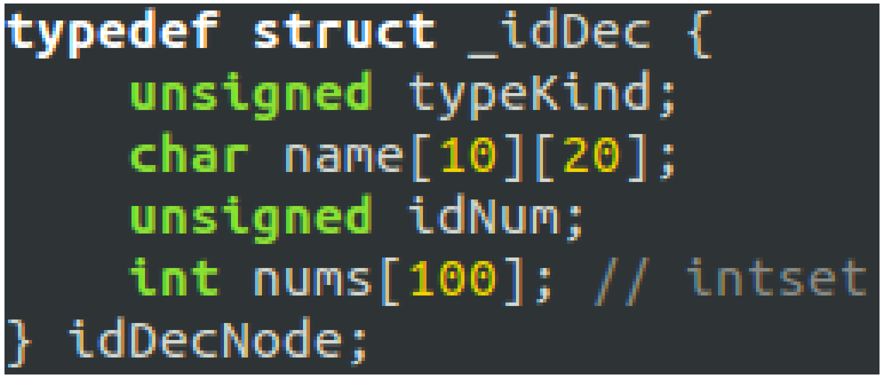


```
switch(typeKind) {
    case eINT:
        if(size == 1) { // one value
            symt[hv].pIval = (int *)malloc(sizeof(int));
            *(symt[hv].pIval) = pIval[0];
        } else if(size > 1) { // array
            symt[hv].pIval = (int *)malloc(sizeof(int) * size);
            int i;
            for(i = 0 ; i < size ; ++i) {
                symt[hv].pIval[i] = pIval[i];
            }
        }

        break;
```

Symbol Table - 변수 선언

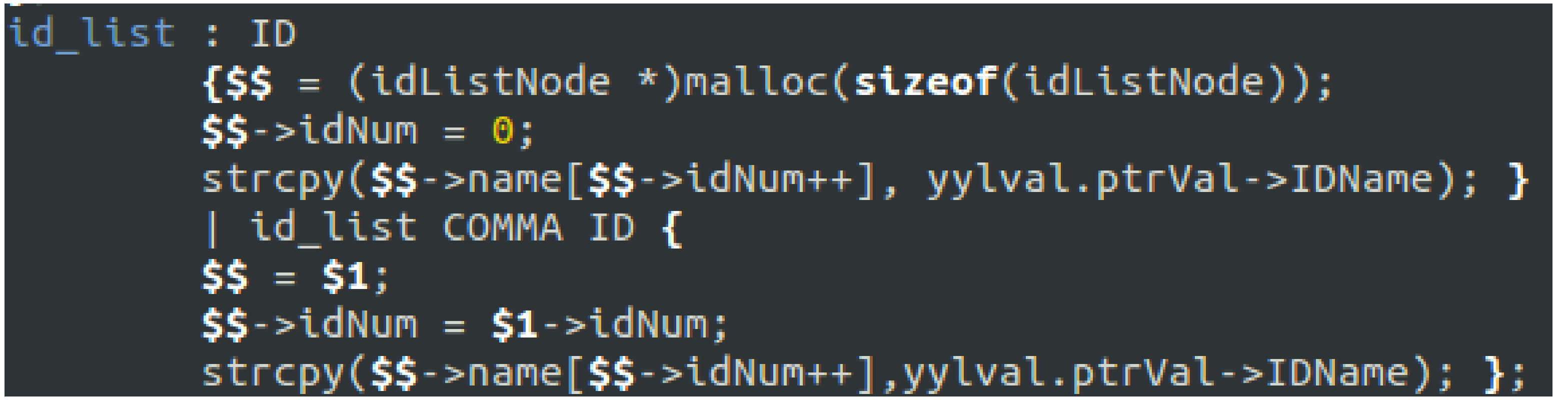


```
typedef struct _idListNode {
    // At most 10 variables and 20 characters at once.
    char name[10][20];
    unsigned idNum;
} idListNode;
```

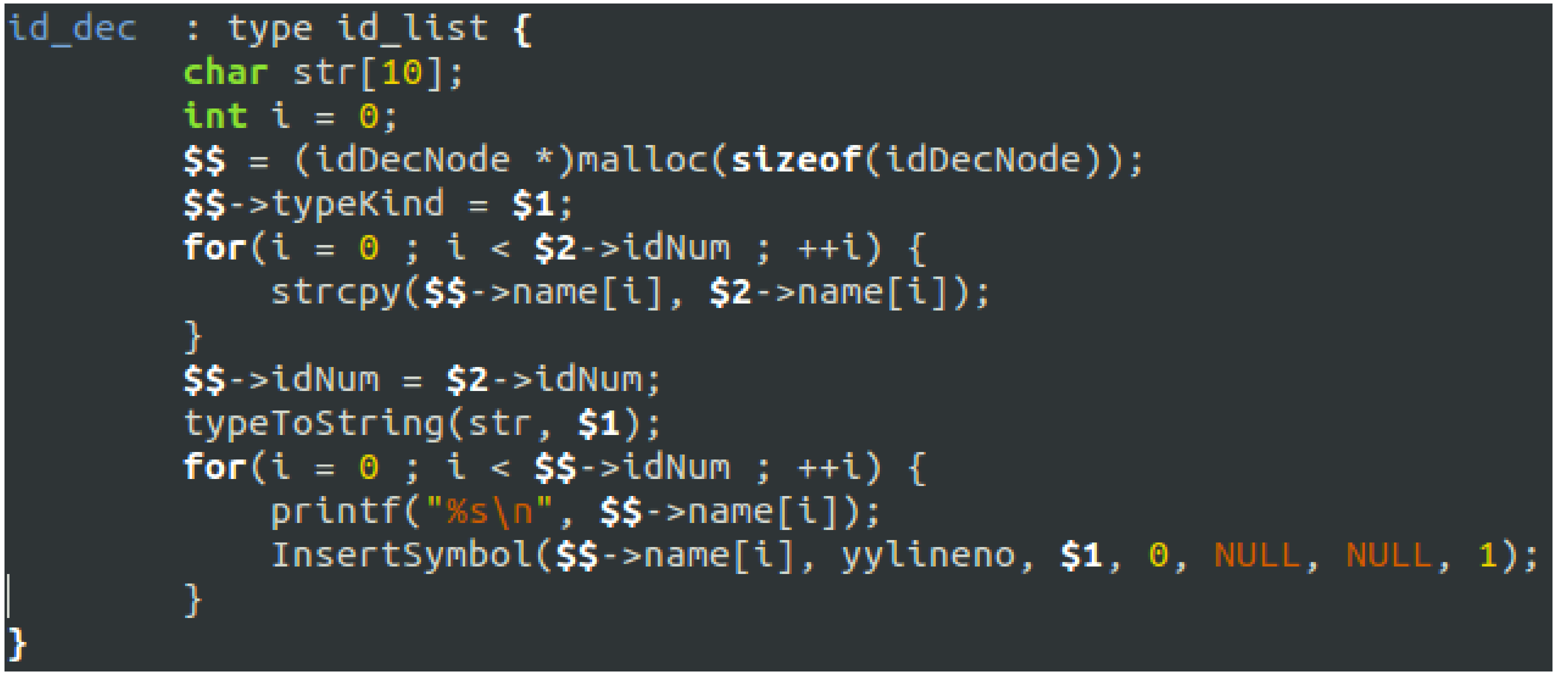


```
typedef struct _idDec {
    unsigned typeKind;
    char name[10][20];
    unsigned idNum;
    int nums[100]; // intset
} idDecNode;
```

Symbol Table - 변수 선언

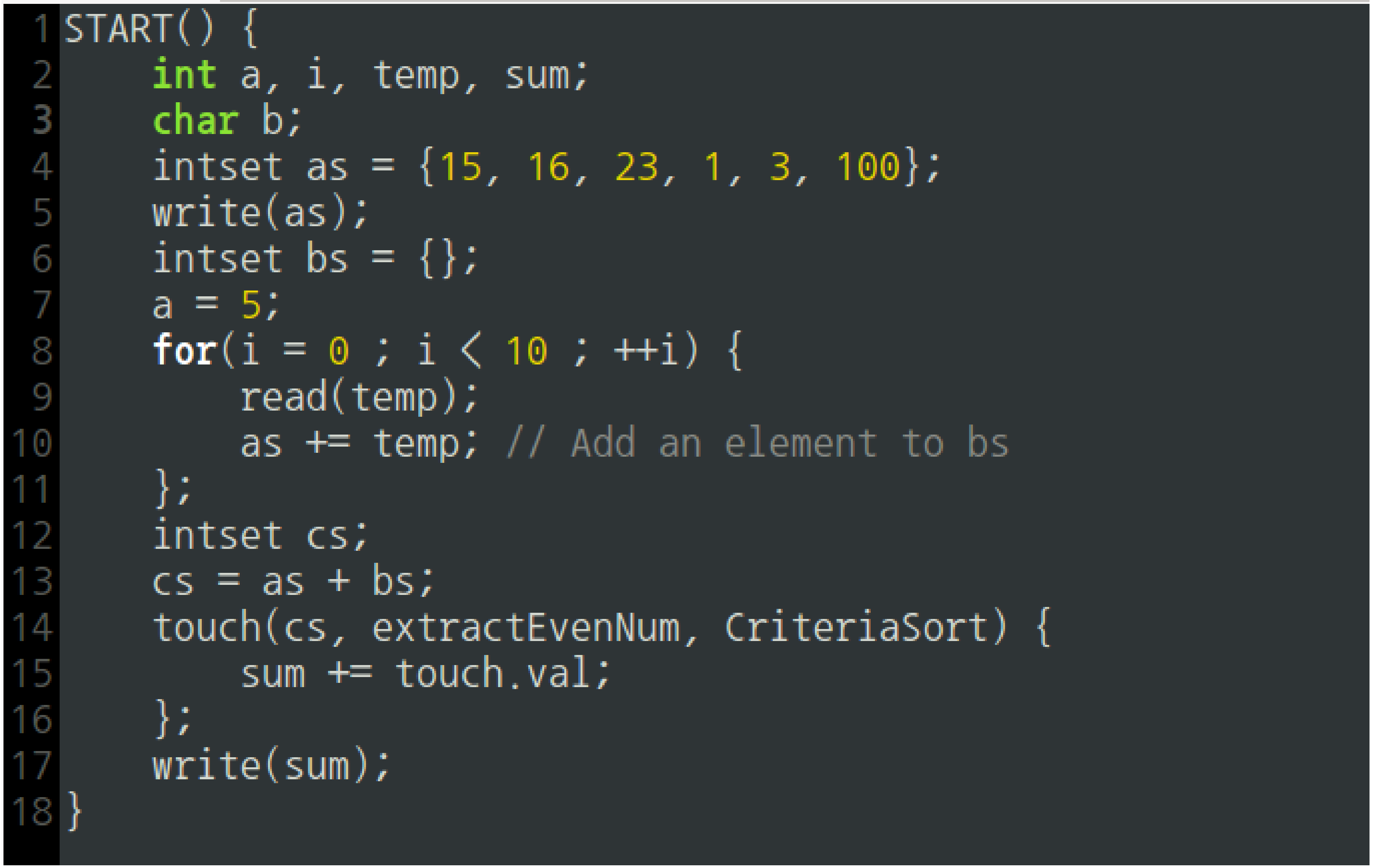


```
id_list : ID
        {$$ = (idListNode *)malloc(sizeof(idListNode));
        $$->idNum = 0;
        strcpy($$->name[$$->idNum++], yylval.ptrVal->IDName); }
        | id_list COMMA ID {
        $$ = $1;
        $$->idNum = $1->idNum;
        strcpy($$->name[$$->idNum++],yylval.ptrVal->IDName); };
```



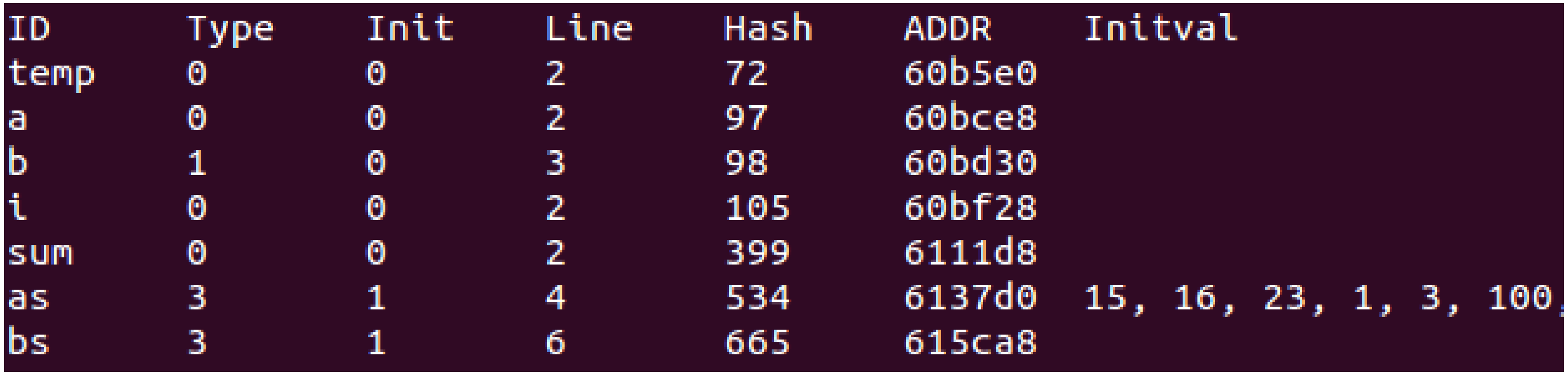
```
id_dec  : type id_list {
        char str[10];
        int i = 0;
        $$ = (idDecNode *)malloc(sizeof(idDecNode));
        $$->typeKind = $1;
        for(i = 0 ; i < $2->idNum ; ++i) {
            strcpy($$->name[i], $2->name[i]);
        }
        $$->idNum = $2->idNum;
        typeToString(str, $1);
        for(i = 0 ; i < $$->idNum ; ++i) {
            printf("%s\n", $$->name[i]);
            InsertSymbol($$->name[i], yylineno, $1, 0, NULL, NULL, 1);
        }
}
```

Sample Code



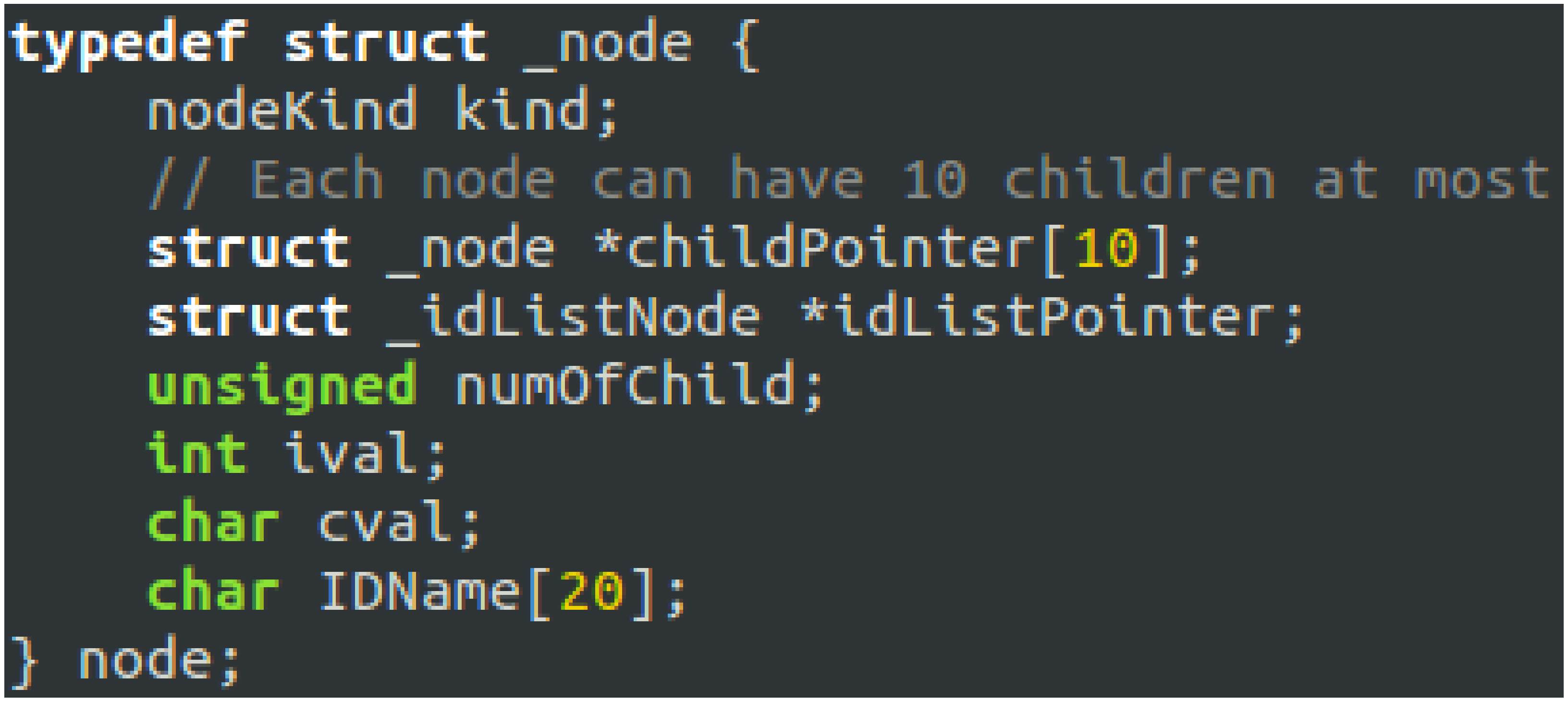
```
START() {
    int a, i, temp, sum;
    char b;
    intset as = {15, 16, 23, 1, 3, 100};
    write(as);
    intset bs = {};
    a = 5;
    for(i = 0 ; i < 10 ; ++i) {
        read(temp);
        as += temp; // Add an element to bs
    };
    intset cs;
    cs = as + bs;
    touch(cs, extractEvenNum, CriteriaSort) {
        sum += touch.val;
    };
    write(sum);
}
```

Symbol Table - 구축결과



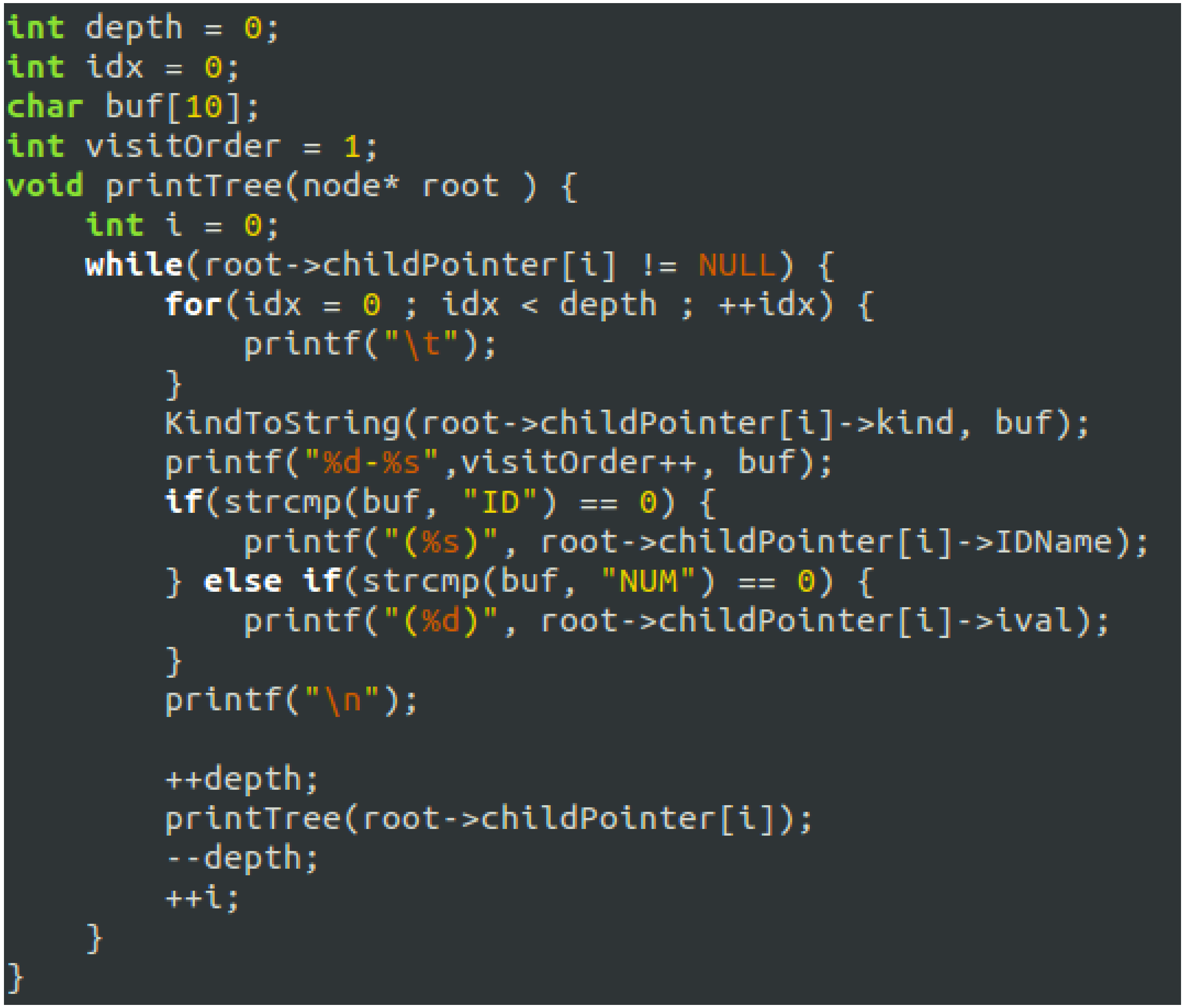
```
ID        Type      Init      Line      Hash      ADDR      Initval
temp      0         0         2         72        60b5e0
a         0         0         2         97        60bce8
b         1         0         3         98        60bd30
i         0         0         2         105       60bf28
sum       0         0         2         399       6111d8
as        3         1         4         534       6137d0    15, 16, 23, 1, 3, 100,
bs        3         1         6         665       615ca8
```

AST - 노드 구조



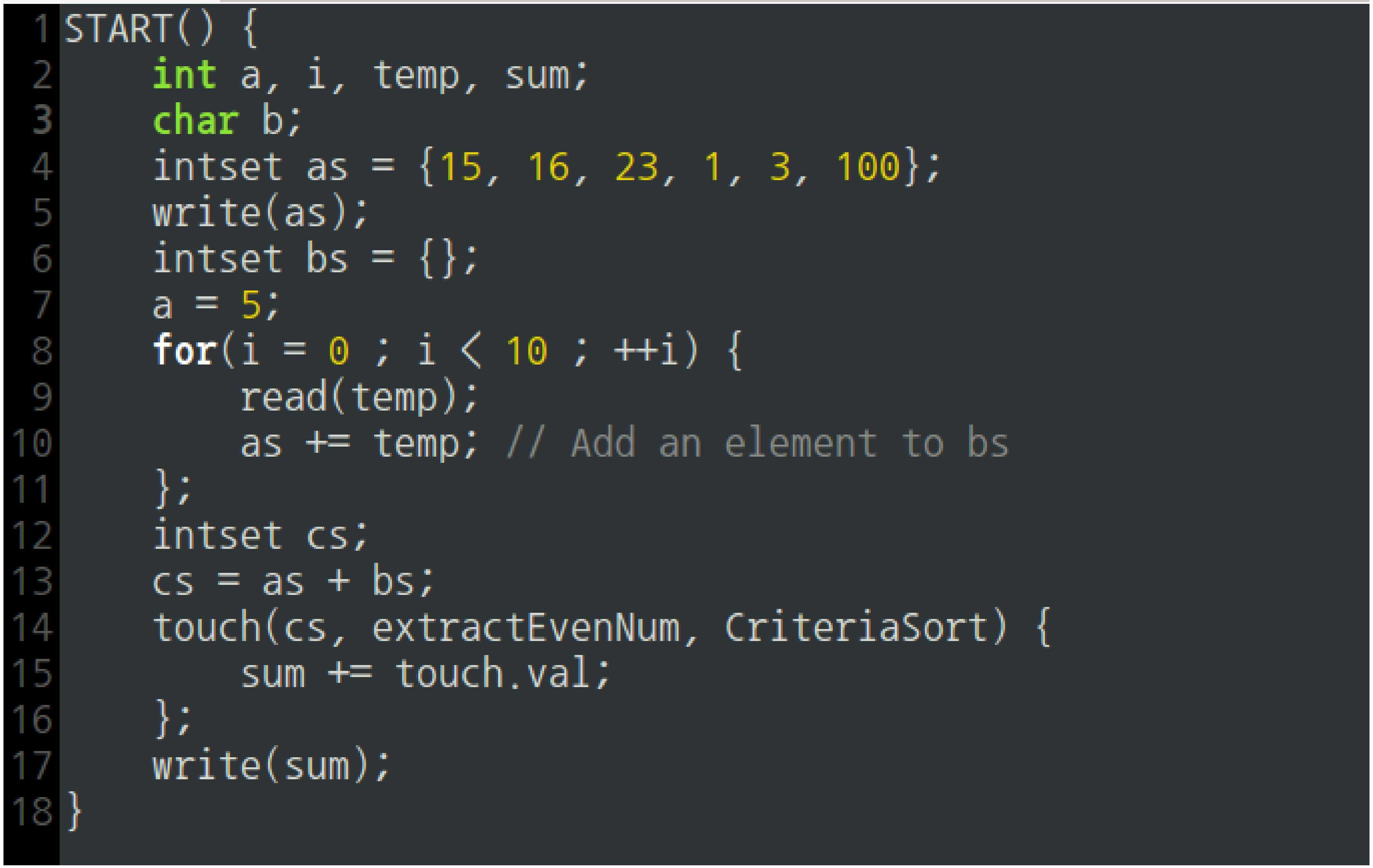
```c
typedef struct _node {
    nodeKind kind;
    // Each node can have 10 children at most
    struct _node *childPointer[10];
    struct _idListNode *idListPointer;
    unsigned numOfChild;
    int ival;
    char cval;
    char IDName[20];
} node;
```

AST - 순회 함수

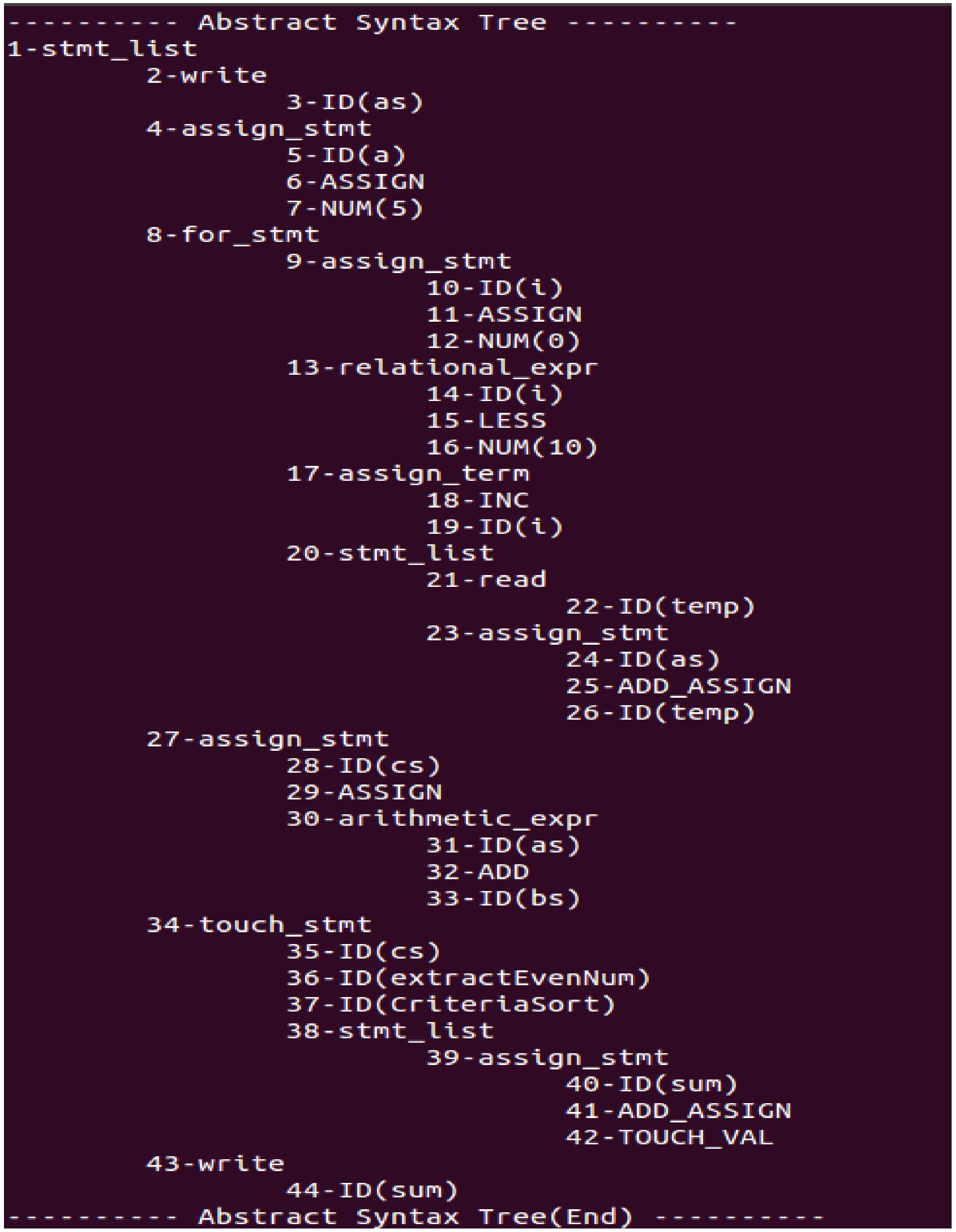


```c
int depth = 0;
int idx = 0;
char buf[10];
int visitOrder = 1;
void printTree(node* root ) {
    int i = 0;
    while(root->childPointer[i] != NULL) {
        for(idx = 0 ; idx < depth ; ++idx) {
            printf("\t");
        }
        KindToString(root->childPointer[i]->kind, buf);
        printf("%d-%s",visitOrder++, buf);
        if(strcmp(buf, "ID") == 0) {
            printf("(%s)", root->childPointer[i]->IDName);
        } else if(strcmp(buf, "NUM") == 0) {
            printf("(%d)", root->childPointer[i]->ival);
        }
        printf("\n");

        ++depth;
        printTree(root->childPointer[i]);
        --depth;
        ++i;
    }
}
```

Sample Code

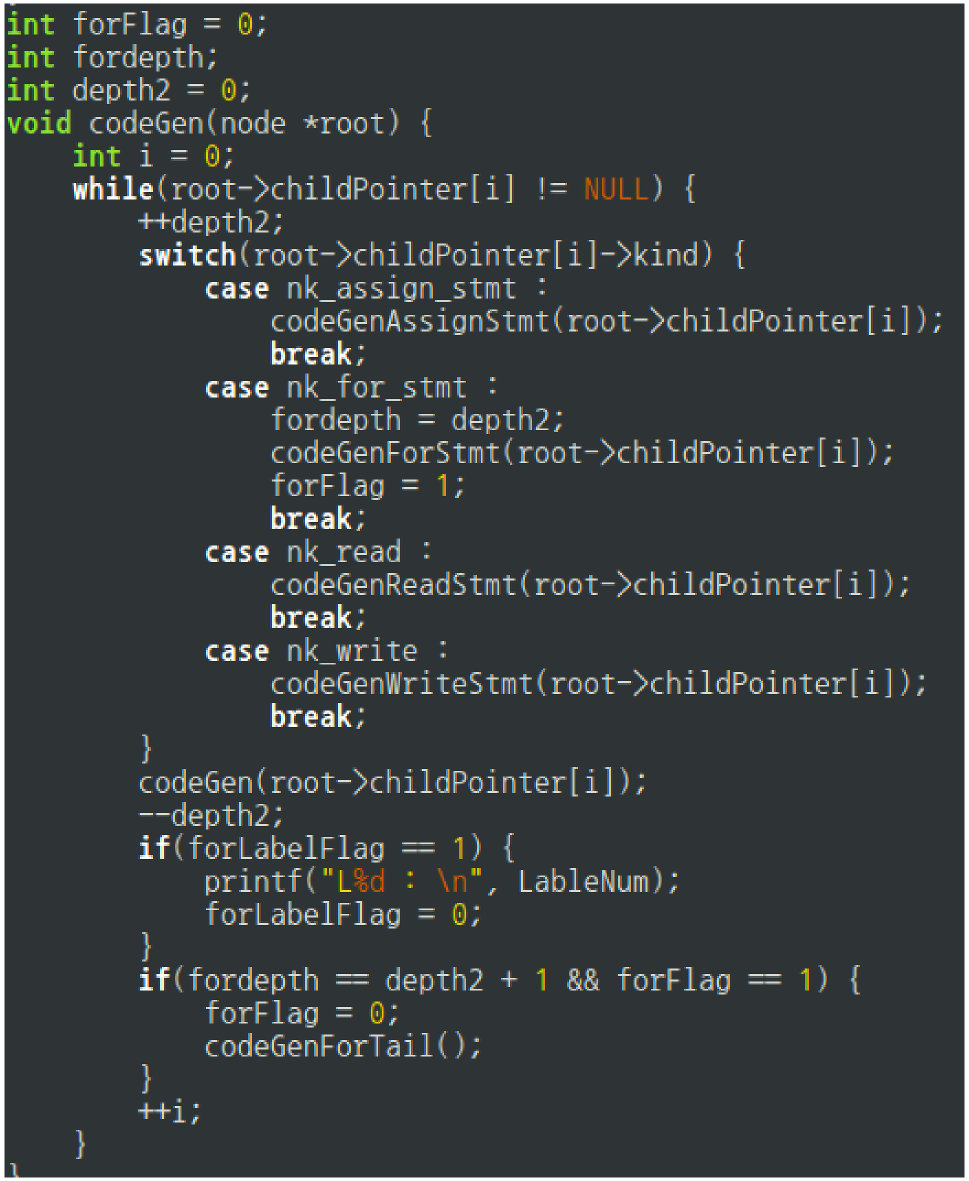


```
START() {
    int a, i, temp, sum;
    char b;
    intset as = {15, 16, 23, 1, 3, 100};
    write(as);
    intset bs = {};
    a = 5;
    for(i = 0 ; i < 10 ; ++i) {
        read(temp);
        as += temp; // Add an element to bs
    };
    intset cs;
    cs = as + bs;
    touch(cs, extractEvenNum, CriteriaSort) {
        sum += touch.val;
    };
    write(sum);
}
```



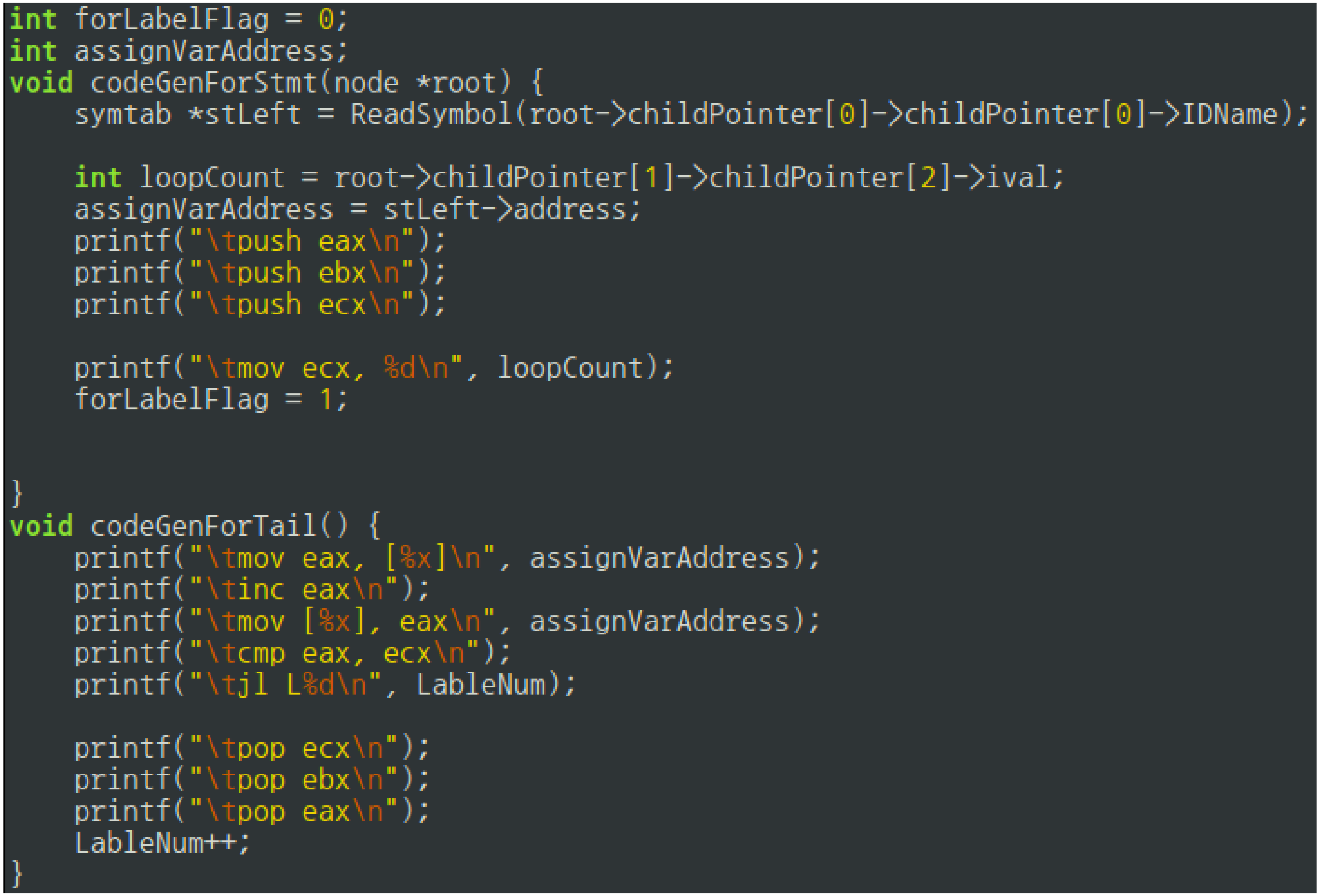
```
---------- Abstract Syntax Tree ----------
1-stmt_list
        2-write
                3-ID(as)
        4-assign_stmt
                5-ID(a)
                6-ASSIGN
                7-NUM(5)
        8-for_stmt
                9-assign_stmt
                        10-ID(i)
                        11-ASSIGN
                        12-NUM(0)
                13-relational_expr
                        14-ID(i)
                        15-LESS
                        16-NUM(10)
                17-assign_term
                        18-INC
                        19-ID(i)
                20-stmt_list
                        21-read
                                22-ID(temp)
                        23-assign_stmt
                                24-ID(as)
                                25-ADD_ASSIGN
                                26-ID(temp)
        27-assign_stmt
                28-ID(cs)
                29-ASSIGN
                30-arithmetic_expr
                        31-ID(as)
                        32-ADD
                        33-ID(bs)
        34-touch_stmt
                35-ID(cs)
                36-ID(extractEvenNum)
                37-ID(CriteriaSort)
                38-stmt_list
                        39-assign_stmt
                                40-ID(sum)
                                41-ADD_ASSIGN
                                42-TOUCH_VAL
        43-write
                44-ID(sum)
---------- Abstract Syntax Tree(End) ----------
```

Code Generation – 순회 함수



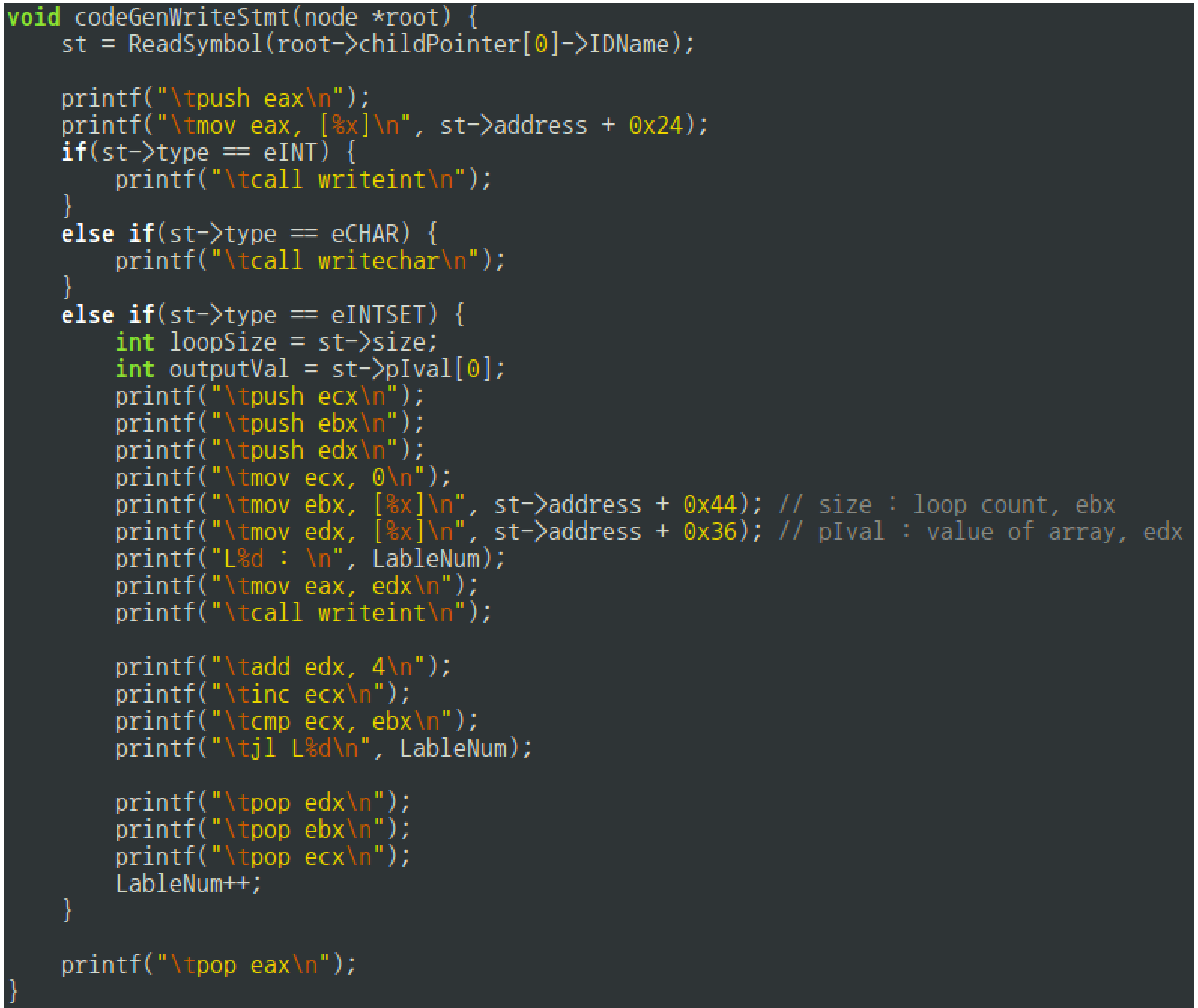
```c
int forFlag = 0;
int fordepth;
int depth2 = 0;
void codeGen(node *root) {
    int i = 0;
    while(root->childPointer[i] != NULL) {
        ++depth2;
        switch(root->childPointer[i]->kind) {
            case nk_assign_stmt :
                codeGenAssignStmt(root->childPointer[i]);
                break;
            case nk_for_stmt :
                fordepth = depth2;
                codeGenForStmt(root->childPointer[i]);
                forFlag = 1;
                break;
            case nk_read :
                codeGenReadStmt(root->childPointer[i]);
                break;
            case nk_write :
                codeGenWriteStmt(root->childPointer[i]);
                break;
        }
        codeGen(root->childPointer[i]);
        --depth2;
        if(forLabelFlag == 1) {
            printf("L%d : \n", LableNum);
            forLabelFlag = 0;
        }
        if(fordepth == depth2 + 1 && forFlag == 1) {
            forFlag = 0;
            codeGenForTail();
        }
        ++i;
    }
}
```

Code Generation 예 - for문



```
int forLabelFlag = 0;
int assignVarAddress;
void codeGenForStmt(node *root) {
    symtab *stLeft = ReadSymbol(root->childPointer[0]->childPointer[0]->IDName);

    int loopCount = root->childPointer[1]->childPointer[2]->ival;
    assignVarAddress = stLeft->address;
    printf("\tpush eax\n");
    printf("\tpush ebx\n");
    printf("\tpush ecx\n");

    printf("\tmov ecx, %d\n", loopCount);
    forLabelFlag = 1;


}
void codeGenForTail() {
    printf("\tmov eax, [%x]\n", assignVarAddress);
    printf("\tinc eax\n");
    printf("\tmov [%x], eax\n", assignVarAddress);
    printf("\tcmp eax, ecx\n");
    printf("\tjl L%d\n", LableNum);

    printf("\tpop ecx\n");
    printf("\tpop ebx\n");
    printf("\tpop eax\n");
    LableNum++;
}
```

Code Generation 예 - write문



```
void codeGenWriteStmt(node *root) {
    st = ReadSymbol(root->childPointer[0]->IDName);

    printf("\tpush eax\n");
    printf("\tmov eax, [%x]\n", st->address + 0x24);
    if(st->type == eINT) {
        printf("\tcall writeint\n");
    }
    else if(st->type == eCHAR) {
        printf("\tcall writechar\n");
    }
    else if(st->type == eINTSET) {
        int loopSize = st->size;
        int outputVal = st->pIval[0];
        printf("\tpush ecx\n");
        printf("\tpush ebx\n");
        printf("\tpush edx\n");
        printf("\tmov ecx, 0\n");
        printf("\tmov ebx, [%x]\n", st->address + 0x44); // size : loop count, ebx
        printf("\tmov edx, [%x]\n", st->address + 0x36); // pIval : value of array, edx
        printf("L%d : \n", LableNum);
        printf("\tmov eax, edx\n");
        printf("\tcall writeint\n");

        printf("\tadd edx, 4\n");
        printf("\tinc ecx\n");
        printf("\tcmp ecx, ebx\n");
        printf("\tjl L%d\n", LableNum);

        printf("\tpop edx\n");
        printf("\tpop ebx\n");
        printf("\tpop ecx\n");
        LableNum++;
    }

    printf("\tpop eax\n");
}
```

시연

시연 영상

Q&A

감사합니다.