

Compiler Task #2

2017.10.17

이준수 : 20132429

1. 목적

Unix Program : YACC 과 LEX 를 이용해서 간단한 정수계산 Syntax Analysis Parser 를 제작한다.

2. YACC

calc.yacc :

```
%{
#include <stdio.h>
#include <stdlib.h>
}%

%start S
%token NUM, PLUS, STAR, LPAREN, RPAREN, NEW_LINE

%%
S : E NEW_LINE      { printf("%d\n", $1); exit(0); }
  ;

E : E PLUS T        { $$ = $1 + $3; }
  | T                { $$ = $1; }
  ;

T : T STAR F         { $$ = $1 * $3; }
  | F                { $$ = $1; }
  ;

F : LPAREN E RPAREN  { $$ = $2; }
  | NUM              { $$ = $1; }
  ;

%%
```

- C 언어 코드는 %{ ... %} 블록에서 담을 수 있다. 계산 결과를 출력하는 코드를 포함하기 때문에 stdio.h 를 include 시켜주어야 한다.
- %token 명령어는 뒤의 숫자들을 겹치지 않는 적절한 값으로 Define 해주는 역할을 담당한다. 이 정보는 이후 -d 옵션을 이용할 시 "y.tab.h 파일"에 저장된다.
- 기본 로직은 %% ... %% 블록 안에서 작성된다.

3. LEX

calc.lex :

```
%{
#include "y.tab.h"
extern int yylval;
}%

digit [0-9]
letter [a-z]
delim [ \t]
ws [delim]+

%%
{ws}      {}
"+"       { return (PLUS); }
"*"       { return (STAR); }
"\n"      { return (NEW_LINE); }
"("       { return (LPAREN); }
")"       { return (RPAREN); }
{digit}+  {
            yylval = atoi(yytext);
            return (NUM);
        }

%%
```

- lex 파일의 형식도 yacc 과 거의 동일하다. "y.tab.h" 헤더파일의 pre-define 된 값을 이용할 것이기 때문에 미리 include 한다.
- lex 에서 반복되는 패턴을 찾기 위해 정규표현식(regular expression) 을 사용한다.
- 기본 로직은 %% ... %% 블록 안에서 작성된다.
- *yylval* 에 값을 넘겨주게 되며, 토큰을 잠시 저장하는 공간이 *yytext* 변수이다.

4. 실행순서 및 결과

```
yacc -d calc.yacc
```

결과 : y.tab.h 헤더파일과 y.tab.c 파일이 생성된다.

```
lex calc.lex
```

결과 : lex.yy.c 가 생성된다.

main 로직을 담을 파일을 작성한다.

my_calc.c :

```

#include <stdio.h>
#include "y.tab.h"
int yywrap(){
    return(1);
}
void yyerror(char * error_str){
    fprintf(stderr, "%s\n", error_str);
}
int main(){
    yyparse();
    return 0;
}

```

```
gcc y.tab.c lex.yy.c my_calc.c -o output
```

결과 : output 실행파일 만들어진다.

```
./output
```

문법 적합 경우

```

myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$ ./output
1+2*3
7
myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$

```

```

myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$ ./output
(1+23)*4+2
98
myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$

```

문법 오류 경우

```

myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$ ./output
(1+233*2++)*28**
syntax error
myZZUNG@ijunsuui-MacBook-Pro:~/myworkspace/anything/compiler_lecture/lex_yacc$

```