Aayush Shah
19BCE245
1 October 2022

Compiler Construction

Practical 7

To implement lexical analyse to recognize all distinct token classes

- **Code :**

lexer.l

```
%{
#include  "y.tab.h"
#include<stdio.h>
int line = 1;
%}

%%
"\n" {line++;}
"\t"|" " {}
"int"|"float"|"double"|"char"   {return DT;}
"==" {return EQ;}
"=" {return EQ1;}
"!=" {return NE;}
"<=" {return LE;}
">=" {return GE;}
"++"|"--" {return UNARY;}
[-+*/><!();{}:=] {return yytext[0];}
"&"[&]? {return AND;}
"|"[|]? {return OR;}
[0-9]+  {return INT_CONST;}
[']. ['] {return CHAR_CONST;}
"true"|"false"  {return BOOL_CONST;}
["].*["]  {return STR_CONST;}
"while" {return WHILE;}
"do"  {return DO;}
"switch"  {return SWITCH;}
"case"  {return CASE;}
"default"  {return DEFAULT;}
"break" {return BREAK;}
"for" {return FOR;}
([_a-zA-Z](0-9)*)+  {return ID;}
%%
```

parser.l

parser.y

Syntax: Bison | Language | Run | Stop | Run Settings... | Back/Forward | View

lexer.l | parser.y

```
%{
#include<stdio.h>
int yylex();
void yyerror(char *msg);
void yywrap();
int
    count_for=0,count_if=0,count_while=0,count_switch=0,count_do_while
    =0;
extern int line;
%}
%token WHILE DO SWITCH CASE DEFAULT BREAK DT ID INT_CONST BOOL_CONST
    CHAR_CONST STR_CONST LE GE EQ EQ1 NE AND OR FOR EQSN IDSET

%right '='
%left AND OR
%left '<' '>' LE GE EQ NE
%left '+''-'
%left '*''/'
%right UNARY
%left '!'

%%

PROG : DT ID '(' ')' BLK;

BLK :   '{' BS '}';

BS : SS
    |
    ;

SS  :   S SS
    |   S
    ;

S   :   DO_WHILE_STMT
```

Symbol | Spaces: 4 | Line 18, Column 1

parser.y

Syntax: Bison

Language Run Stop Run Settings... Back/Forward View

lexer.l parser.y

```
    |   WHILE_STMT
    |   SWITCH_STMT
    |   BLK
    |   E ';'
    |   ';'
    ;

SWSS: SS BRK
    | BRK
    ;

BRK : BREAK ';'
    |
    ;


;

WHILE_STMT : WHILE '(' E ')' BLK  {count_while++;};

DO_WHILE_STMT   :   DO BLK WHILE '(' E ')' ';' {count_do_while++;};

SWITCH_STMT :   SWITCH '(' E ')' '{' CASE_STMTS DEFAULT_STMT
    '}' {count_switch++;}
            ;

CASE_STMTS  :   CASE_STMT CASE_STMTS
            |   CASE_STMT
            ;

CASE_STMT   :   CASE CONST ':' SWSS
            ;

DEFAULT_STMT    :   DEFAULT ':' SWSS
                |
                ;
```

Symbol Spaces: 4 Line 18, Column 1

parser.y

Syntax: Bison | Language | Run | Stop | Run Settings... | Back/Forward | View

lexer.l | parser.y

```
                ;

E   :   E '+' E
    |   E '-' E
    |   E '*' E
    |   E '/' E
    |   E '<' E
    |   E '>' E
    |   E EQ E
    |   E NE E
    |   E GE E
    |   E LE E
    |   ID
    |   CONST
    ;

CONST   :   INT_CONST
        |   CHAR_CONST
        |   BOOL_CONST
        |   STR_CONST
        ;

%%

int main() {
    yyparse();
    return 0;
}

void yyerror(char *msg) {
    printf("Error Message :- %s at line %d\n", msg, line);
}

void yywrap() {
    printf("Number of do while loop is %d\n", count_do_while);
    printf("Number of while loop is %d\n", count_while);
    printf("Number of for loop is %d\n", count_for);
    printf("Number of switch statements is %d\n", count_switch);
}
```

Symbol | Spaces: 4 | Line 85, Column 23

input.c

```c
int main() {
    do {
        switch(a) {
            case 1:
                ;
                break;
            case 2:
                ;
                break;
            default:
                ;
                break;
        }
    } while(i < 245);

    while (1<3){

    };
    while (2==3){

    };
    while (5<a){

    };
}
```

output

```
[Aayushs-MBP: codes-final/ $ bison -dy parser.y ]
[Aayushs-MBP: codes-final/ $ flex lexer.l ]
[Aayushs-MBP: codes-final/ $ gcc lex.yy.c y.tab.c ]
[Aayushs-MBP: codes-final/ $ ./a.out < input.c ]
Number of do while loop is 1
Number of while loop is 3
Number of for loop is 0
Number of switch statements is 1
Aayushs-MBP: codes-final/ $
```