**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 1**

#include <stdio.h>

#include <string.h>

int state = 0, bcount = 0;

FILE \*fp;

int reset() { bcount = 0; state = 0; fseek(fp, -1, SEEK\_CUR); }

int isDigi(char ch) {

if(ch >= '0' && ch <= '9') return 1;

return 0;

}

int isIden(char ch) {

if((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z') || ch == '\_') return 1;

return 0;

}

int isOper(char ch) {

if(ch == '+' || ch == '-' || ch == '\*' || ch == '/' || ch == '%' || ch == '=') return 1;

return 0;

}

int isSymb(char ch) {

if(ch == '{' || ch == '}' || ch == '(' || ch == ')' || ch == ';' || ch == ',') return 1;

return 0;

}

int isWhit(char ch) {

if(ch == ' ' || ch == '\t' || ch == '\n') return 1;

return 0;

}

int isKeyw(char \*b) {

if(!strcmp(b, "main") || !strcmp(b, "void") || !strcmp(b, "int")) return 1;

return 0;

}

void main() {

char buffer[128], ch;

fp = fopen("input\_1.txt", "r");

while((ch = fgetc(fp)) != EOF) {

buffer[bcount++] = ch;

switch(state) {

case 0:

if(isDigi(ch)) state = 1;

if(isIden(ch)) state = 2;

if(isOper(ch)) state = 3;

if(isSymb(ch)) state = 5;

if(isWhit(ch)) bcount = 0;

break;

case 1:

if(!isDigi(ch)) {

buffer[bcount - 1] = '\0';

printf("Number: \t%s\n", buffer); reset();

}

break;

case 2:

if(!isIden(ch) && !isDigi(ch)) {

buffer[bcount - 1] = '\0';

if(isKeyw(buffer)) printf("Keyword: \t%s\n", buffer);

else printf("Identifier: \t%s\n", buffer);

reset();

}

break;

case 3:

if(ch == '/' && buffer[bcount - 2] == '/') state = 5;

if(!isOper(ch)) {

buffer[bcount - 1] = '\0';

printf("Operator: \t%s\n", buffer); reset();

}

break;

case 4:

if(ch == '\n') reset();

case 5:

buffer[bcount - 1] = '\0';

printf("Symbol: \t%s\n", buffer); reset(); break;

}

}

fclose(fp);

}

**// Input File**

void main()

{

// hello world

int a = 10, b;

a = a \* b / 5; // Simple Operation comment

}

**// Output**

Keyword: void

Keyword: main

Symbol: (

Symbol: )

Symbol: {

Keyword: int

Identifier: a

Operator: =

Number: 10

Symbol: ,

Identifier: b

Symbol: ;

Identifier: a

Operator: =

Identifier: a

Operator: \*

Identifier: b

Operator: /

Number: 5

Symbol: ;

Symbol: }

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 2**

%{

#include <stdio.h>

int comment = 0;

%}

%%

"//".\*\n { comment++; }

[ \n\t] { ; }

[+-]?[0-9]+(\.[0-9]+)?(E[+-]?[0-9]+) { printf("%s\tExponent Number\n", yytext); }

([+-]?[0-9]+)\.[0-9]+ { printf("%s\tFloating Number\n", yytext); }

[+-]?[0-9]+ { printf("%s\tNumber\n", yytext); }

(void|main|printf|int|float) { printf("%s\tReserved Keyword\n", yytext); }

[a-zA-Z\_][a-zA-Z0-9\_]\* { printf("%s\tLiteral\n", yytext); }

[=\*/+\-%] { printf("%s\tArithmetic Operator\n", yytext); }

(==|<|<=|>|>=|!=) { printf("%s\tRelational Operator\n", yytext); }

[(){};,] {printf("%s\tSpecial Operator\n", yytext); }

%%

void main() {

yyin = fopen("input.c", "r"); yylex();

printf("\n%d Comments Ignored\n", comment); fclose(yyin);

}

**// Input File**

void main(){

// hello world

int a = 7, b = 7.35, c = 7.35E2 \* 7E10;

a = a \* b;

}

**// Output**

void Reserved Keyword

main Reserved Keyword

( Special Operator

) Special Operator

{ Special Operator

int Reserved Keyword

a Literal

= Arithmetic Operator

7 Number

, Special Operator

b Literal

= Arithmetic Operator

7.35 Floating Number

, Special Operator

c Literal

= Arithmetic Operator

7.35E2 Exponent Number

\* Arithmetic Operator

7E10 Exponent Number

; Special Operator

a Literal

= Arithmetic Operator

a Literal

\* Arithmetic Operator

b Literal

; Special Operator

} Special Operator

1 Comments Ignored

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 3**

%{

#include <stdio.h>

int lines = 0, char\_count = 0, words = 0;

%}

%%

\n { lines++; char\_count++; }

[a-zA-Z\_]\* {words++; char\_count += yyleng; }

. { char\_count++; }

%%

void main() {

yyin = fopen("input.txt", "r"); yylex();

printf("Statistics:\n%d\tLines\n%d\tCharacters\n%d\tWords\n", lines, char\_count, words);

fclose(yyin);

}

**// Input File**

hello World

aeiou

abcf gejk

**// Output**

Statistics:

4 Lines

32 Characters

5 Words

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 4**

%{

#include <stdio.h>

%}

%%

abc { printf("ABC"); }

.|\n { printf("%s", yytext); }

%%

void main() {

yyin = fopen("input.txt", "r"); yylex();

fclose(yyin);

}

**// Input**

helloabccdabfc

**// Output**

helloABCcdabfc

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 5**

%{

#include <stdio.h>

int vowels = 0, cons = 0;

%}

%%

[aeiouAEIOU] { vowels++; }

[a-zA-Z] { cons++; }

(.|\n) { ; }

%%

void main() {

yyin = fopen("input.txt", "r"); yylex();

printf("Statistics:\n%d\tVowels\n%d\tConsonants\n", vowels, cons);

fclose(yyin);

}

**// Input**

hello World

aeiou

abcf gejk

**// Output**

Statistics:

10 Vowels

13 Consonants

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 6**

**// Lex File**

%{

#include <stdio.h>

#include "y.tab.h"

extern int yylval;

%}

%%

[0-9]+ { return NUM; }

[a-zA-Z] { return ID; }

[+\-\*/()] { return yytext[0]; }

. { return other; }

\n { return '\n'; }

%%

**// YACC File**

%{

#include <stdio.h>

int yylex(); void yyerror();

%}

%token NUM ID other

%left '+' '-'

%left '\*' '/'

%%

start : T '\n' { printf("Valid Arithmetic Expression\n"); return 0; };

T : T '+' T

| T '-' T

| T '\*' T

| T '/' T

| '(' T ')'

| NUM

| ID

;

%%

void yyerror() {

printf("Error. Failed to parse.\n");

}

void main() {

printf("Enter arithmetic expression: ");

yyparse();

}

**// Output #1**

Enter arithmetic expression: (a+b\*(a/2+3))

Valid Arithmetic Expression

**// Output #2**

Enter arithmetic expression: (a+b(c+/

Error. Failed to parse.

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 7**

**// Lex File**

%{

#include <stdio.h>

#include "y.tab.h"

extern int yylval;

%}

%%

[0-9] { return digit; }

[a-zA-Z] { return alpha; }

. { return other; }

\n { return '\n'; }

%%

**// YACC File**

%{

#include <stdio.h>

int yylex();

void yyerror();

%}

%token digit alpha other

%%

start : T '\n' { printf("Valid Computer Identifier\n"); return 0; };

T : alpha U

;

U : U alpha

| U digit

|

;

%%

void yyerror() {

printf("Error. Failed to parse.\n");

}

void main() {

printf("Enter identifier: ");

yyparse();

}

**// Output #1**

Enter identifier: abc123

Valid Computer Identifier

**// Output #2**

Enter identifier: 12ac

Error. Failed to parse.

**// Output #3**

Enter identifier: abc$#123

Error. Failed to parse.

**// Emmanuel Jojy**

**// 53 S7 CSE A**

**// Experiment 8**

**// Lex File**

%{

#include <stdio.h>

#include "y.tab.h"

extern int yylval;

%}

%%

[0-9]+ { sscanf(yytext, "%d", &yylval); return NUM; }

[+\-\*/()] { return yytext[0]; }

. { return other; }

\n { return '\n'; }

%%

**// YACC File**

%{

#include <stdio.h>

int yylex(); void yyerror();

%}

%token NUM other

%left '+' '-'

%left '\*' '/'

%%

start : T '\n' { printf("Result: %d\n", $$); return 0; };

T : T '+' T { $$ = $1 + $3; }

| T '-' T { $$ = $1 - $3; }

| T '\*' T { $$ = $1 \* $3; }

| T '/' T { $$ = $1 / $3; }

| '(' T ')' { $$ = $2; }

| NUM { $$ = $1; }

;

%%

void yyerror() {

printf("Error. Failed to parse.\n");

}

void main() {

printf("Enter arithmetic expression: ");

yyparse();

}

**// Output #1**

Enter arithmetic expression: 2+3\*5

Result: 17

**// Output #2**

Enter arithmetic expression: (2+3)\*5+(6\*7+(2\*3))

Result: 73

**// Output #3**

Enter arithmetic expression: 2+3/

Error. Failed to parse.