**Name: Siddhi Kothekar**

**Roll No.19**

**Batch – A2**

**Practical No. 1**



# Aim (E1): Design a lexical analyzer

Write a lex code to identify the tokens such as keywords, identifiers, operators, constants (Int, float &amp; character), special symbols and strings for C language using LEX. Use File for the input.

**Program:**

%{

#include<stdio.h> #include<string.h> int i = 0;

%}

digit [0-9] letter [a-zA-Z]

symbol [,;|&%(){}<>]

dot [\.] lbrace \[ rbrace \] string (\".\*\") keyword

(abstract|assert|boolean|break|byte|case|catch|care|class|continue|default|do|double|else|enum|exte nds|final|finally|float|for|if|implements|import|instanceof|int|interface|long|native|new|null|packag e|private|protected|public|return|short|static|super|switch|this|throw|throws|try|void|while)

/\* Rules Section \*/

%%

{keyword} { printf("Keyword: %s\n", yytext); }

{letter}({letter}|{digit})\* { printf("Identifier: %s\n", yytext); }

"&&"|"=="|"+"|"-"|"\*"|"/" { printf("Operator: %s\n", yytext); }

{symbol} { printf("Symbol: %s\n", yytext); }

{dot} { printf("Symbol: %s\n", yytext); }

{lbrace} { printf("Symbol: %s\n", yytext); }

{rbrace} { printf("Symbol: %s\n", yytext); }

{digit}+ { printf("Constant: %s\n", yytext); }

{string} { printf("String: %s\n", yytext); }

\n { /\* Ignore newlines \*/ }

[ \t]+ { /\* Ignore spaces and tabs \*/ }

%%

int main() {

yyin = fopen("input.txt", "r"); if (yyin == NULL) { perror("Error opening file");

return 1;

} yylex(); fclose(yyin); return 0;

}

int yywrap()

{

return 1;

}

**Output:**



# Aim(E2): Question Paper Analyzer

Write a Lex program to find the parameters given below. Consider as input a question paper of an examination.

1. Count the number of questions.
2. Number of questions that have sub-part and how many donot.
3. Count the total marks.
4. Date of examination
5. Semester
6. Count different types of questions- Eg: What, Discuss, etc.
7. Numbers of words, lines, small letters, capital letters, digits, and special characters.

**Program:**

%{

#include <stdio.h> #include <ctype.h> int count\_with\_subparts = 0; // Questions with subparts int count\_without\_subparts = 0; // Questions without subparts

int total\_marks = 0; // Total marks int lines = 0, words = 0, s\_letters = 0, c\_letters = 0, spl\_char = 0, total = 0;

%}

%%

[Q][0-9]+ { count\_without\_subparts++; }

[Q][0-9]+\([a-zA-Z]\) { count\_with\_subparts++; }

[0-9]+Marks { total\_marks += atoi(yytext); }

[0-9]+[/][0-9]+[/][0-9]+ { printf("\nExam Date: %s", yytext); }

("I"|"II"|"III"|"IV"|"V"|"VI"|"VII") { printf("\nSemester: %s", yytext); }

What.\*\? { printf("\n%s is a 'What' question", yytext); }

Discuss.\*\? { printf("\n%s is a 'Discuss' question", yytext); }

Explain.\*\? { printf("\n%s is an 'Explain' question", yytext); }

\n { lines++; words++; }

[\t ] { words++; }

[A-Z] { c\_letters++; }

[a-z] { s\_letters++; }

. { spl\_char++; }

%%

int main(void) {

FILE \*file = fopen("E2.txt", "r");

yyin = file;

yylex();

printf("\nAnalysis of the Question Paper:"); printf("\n-------------------------------");

printf("\nTotal Questions: %d", count\_without\_subparts + count\_with\_subparts / 2); printf("\nQuestions with Subparts: %d", count\_with\_subparts); printf("\nQuestions without Subparts: %d", count\_without\_subparts); printf("\nTotal Marks: %d", total\_marks); printf("\nTotal Words: %d", words); printf("\nTotal Lines: %d", lines); printf("\nSmall Letters: %d", s\_letters); printf("\nCapital Letters: %d", c\_letters);

printf("\nSpecial Characters: %d", spl\_char);

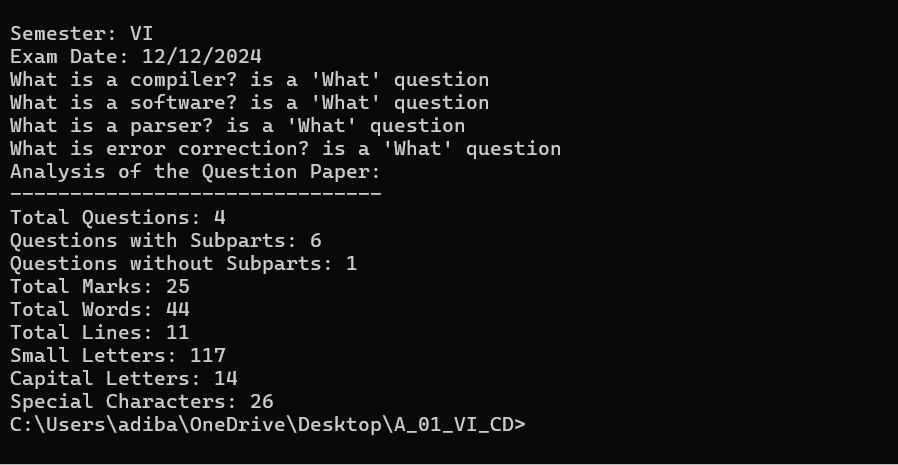
fclose(file); return 0;

}

int yywrap() { return 1;

}

**Output:**



# Aim(E3): Program Cleaner

Write a Lex Program which takes C program from file and write the same C program in another file after removing the comments.

**Program:**

%{

#include <stdio.h>

#include <stdlib.h>

FILE \*outFile;

%}

%%

\/\/.\* { /\* Ignore single-line comments \*/ }

\/\\*([^\*]|\\*+[^\*/])\*\\*\/ { /\* Ignore multi-line comments \*/ }

.|[\n] { fprintf(outFile, "%s", yytext); }

%%

int yywrap() { return 1;

}

int main(int argc, char \*argv[]) {

if (argc != 3) { fprintf(stderr, "Usage: %s <input\_file> <output\_file>\n", argv[0]); return 1;

}

FILE \*inFile = fopen(argv[1], "r"); if (!inFile) { perror("Error opening input file"); return 1;

}

outFile = fopen(argv[2], "w");

if (!outFile) { perror("Error opening output file");

fclose(inFile); return 1;

}

yyin = inFile; yylex(); fclose(inFile);

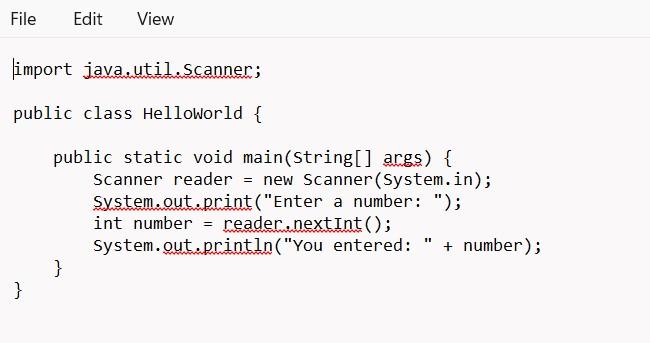
fclose(outFile);

printf("Comments removed and output written to %s\n", argv[2]); return 0;

}

**Output:**





# E4: Do as directed

Write a LEX specification to take the contents from a file

1. Add 3 to number divisible by 7
2. Add 4 to number divisible by 2
3. Convert the alphabetical list to numbered list

**Program:**

%{

#include <stdio.h> #include <stdlib.h>

int line\_count = 1;

%}

%%

[0-9]+ { int num = atoi(yytext); if (num % 7 == 0) num += 3; if (num % 2 == 0) num += 4; printf("%d\n", num);

}

[a-zA-Z]+ {

printf("%d. %s\n", line\_count++, yytext);

}

\n { printf("\n"); }

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

%%

int yywrap() { return 1; }

int main(int argc, char \*argv[]) {

if (argc != 2) { fprintf(stderr, "Usage: %s <input\_file>\n", argv[0]); return 1;

}

FILE \*inFile = fopen(argv[1], "r"); if (!inFile) { perror("Error opening input file"); return 1;

}

yyin = inFile; yylex(); fclose(inFile); return 0;

}

**Output:**

