

**Compiler Design Lab**

**SUBMITTED BY: SUBMITTED TO:**

Name: Harsh Agrawal Dr. Ajay Prasad

SAP ID: 500107136

Enrollment No: R2142220326

Course- B.Tech. (CSE)

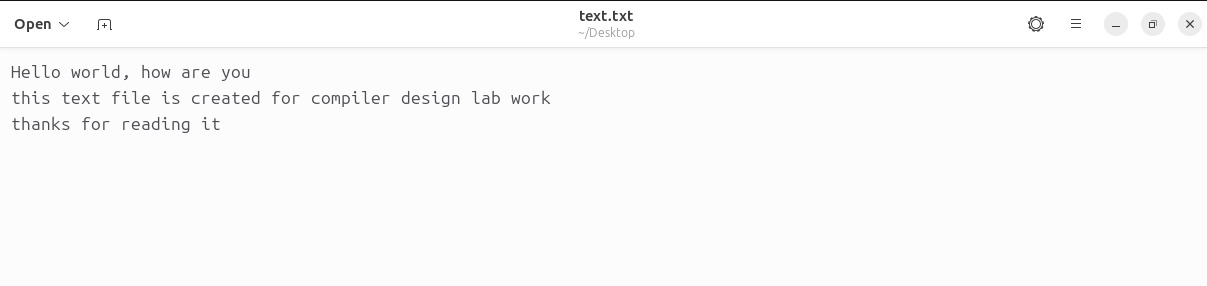
Batch: 4 (AI/ML) (Non- Hons)

**Lab Assignment 2**

1. WAP in both C and lex to count the number of characters, words, spaces, and end of lines in a given input file.

Sol-

Text file screenshot-



C code-

#include <stdio.h>

#include <ctype.h>

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

if (argc < 2) {

printf("Usage: %s <input\_file>\n", argv[0]);

return 1;

}

FILE \*file = fopen(argv[1], "r");

if (!file) {

printf("Could not open file %s\n", argv[1]);

return 1;

}

int characters = 0, words = 0, spaces = 0, lines = 0;

char ch;

int in\_word = 0;

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

characters++;

if (isspace(ch)) {

if (ch == ' ') spaces++;

if (ch == '\n') lines++;

in\_word = 0;

} else if (!in\_word) {

words++;

in\_word = 1;

}

}

fclose(file);

printf("Characters: %d\n", characters);

printf("Words: %d\n", words);

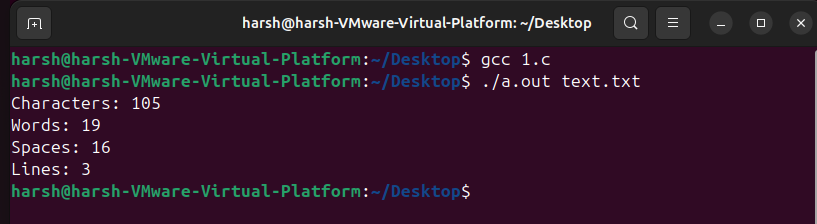
printf("Spaces: %d\n", spaces);

printf("Lines: %d\n", lines);

return 0;

}

Output-



Lex code-

%{

#include <stdio.h>

int characters = 0;

int words = 0;

int spaces = 0;

int lines = 0;

%}

%%

[ \t]+ { spaces += yyleng; characters += yyleng; } // Match spaces and tabs

\n { lines++; characters++; } // Match new lines

[^\n \t]+ { words++; characters += yyleng; } // Match words (non-whitespace characters)

. { characters++; } // Match all other characters

%%

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

if (argc < 2) {

printf("Usage: %s <input\_file>\n", argv[0]);

return 1;

}

FILE \*file = fopen(argv[1], "r");

if (!file) {

printf("Could not open file %s\n", argv[1]);

return 1;

}

yyin = file;

yylex();

fclose(file);

printf("Characters: %d\n", characters);

printf("Words: %d\n", words);

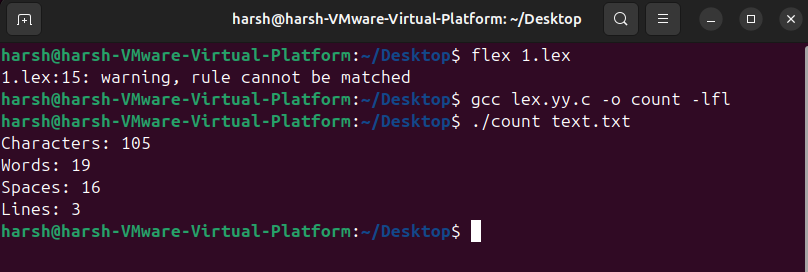
printf("Spaces: %d\n", spaces);

printf("Lines: %d\n", lines);

return 0;

}

Output-



1. WAP in both C and lex to count number of comment lines in a given C program.

Sol-

Text file screenshot-



C code-

#include <stdio.h>

#include <string.h>

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

if (argc < 2) {

printf("Usage: %s <c\_program\_file>\n", argv[0]);

return 1;

}

FILE \*file = fopen(argv[1], "r");

if (!file) {

printf("Could not open file %s\n", argv[1]);

return 1;

}

int comment\_lines = 0;

char line[1024];

int in\_multiline\_comment = 0;

while (fgets(line, sizeof(line), file)) {

char \*trimmed = line;

// Trim leading whitespace

while (\*trimmed == ' ' || \*trimmed == '\t') trimmed++;

if (in\_multiline\_comment) {

comment\_lines++;

if (strstr(trimmed, "\*/")) {

in\_multiline\_comment = 0;

}

} else if (strncmp(trimmed, "//", 2) == 0) {

comment\_lines++;

} else if (strncmp(trimmed, "/\*", 2) == 0) {

comment\_lines++;

if (!strstr(trimmed, "\*/")) {

in\_multiline\_comment = 1;

}

}

}

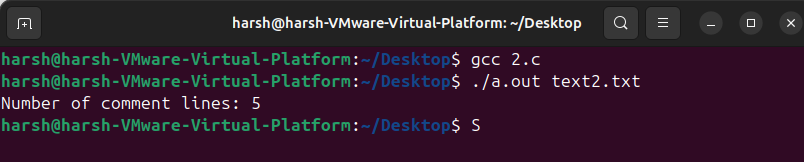
fclose(file);

printf("Number of comment lines: %d\n", comment\_lines);

return 0;

}

Output-



Lex code-

%{

#include <stdio.h>

int comment\_lines = 0;

int in\_multiline\_comment = 0;

%}

%%

\/\/[^\n]\* { comment\_lines++; } // Single-line comments

\/\[^]\+([^/][^]\+)\*\/ { comment\_lines++; } // Multi-line comments (single line)

\/\[^]\* { comment\_lines++; in\_multiline\_comment = 1; } // Start of multi-line comment

[^]\\*+\/ { comment\_lines++; in\_multiline\_comment = 0; } // End of multi-line comment

\n { if (in\_multiline\_comment) comment\_lines++; } // New line in multi-line comment

. { } // Match all other characters

%%

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

if (argc < 2) {

printf("Usage: %s <c\_program\_file>\n", argv[0]);

return 1;

}

FILE \*file = fopen(argv[1], "r");

if (!file) {

printf("Could not open file %s\n", argv[1]);

return 1;

}

yyin = file;

yylex();

fclose(file);

printf("Number of comment lines: %d\n", comment\_lines);

return 0;

}

Output-

