**Q.1) Design a LEX Code to count the number of lines, space, tab-meta character and rest of characters in a given Input pattern.**

**Program:**

%{

#include<stdio.h>

int lines=0,spaces=0,tabs=0,chars;

%}

%%

[\n] { lines++;}

[\t] { tabs++;}

[" "] { spaces++;}

[^\t" "] {chars++;}

%%

int yywrap()

{

return 1;

}

int main()

{

printf("Enter the input:\n");

yylex();

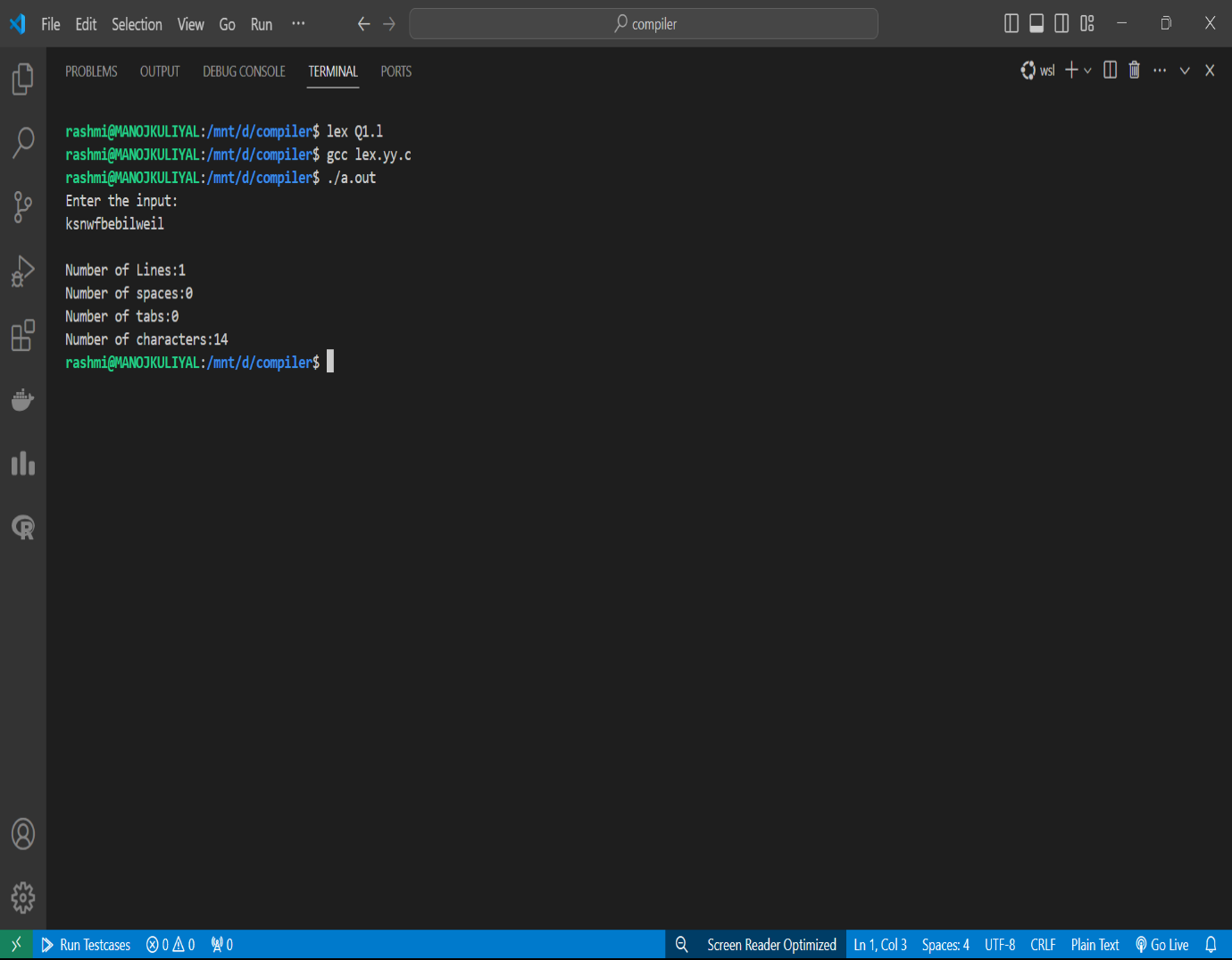
printf("\nNumber of Lines:%d\n",lines);

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

printf("Number of tabs:%d\n",tabs);

printf("Number of characters:%d\n",chars);

}

**OUTPUT**

**Q.2) Design a LEX Code to identify and print valid Identifier of C/C++ in given Input pattern.**

**Program:**

%{

#include<stdio.h>

%}

%%

[a-zA-z\_][a-zA-Z0-9\_]\*$ {printf("valid Identifier\n");}

^[^a-zA-Z\_][a-zA-Z0-9]\* {printf("Invalid Identifier\n");}

. {}

%%

int yywrap()

{

return 1;

}

int main()

{

printf("Enter any character:\n");

yylex();

return 1;

}

**OUTPUT**

A screenshot of a computer

Description automatically generated

**Q.3 Design a LEX Code to identify and print integer and float value in given Input pattern.**

**Program:**

%{

#include<stdio.h>

%}

%%

-?[0-9]+ {printf("Integer\n");}

-?[0-9]\*"."[0-9]+ {printf("Float\n");}

. {printf("Not a valid Number\n");}

%%

int yywrap()

{

return 1;

}

int main()

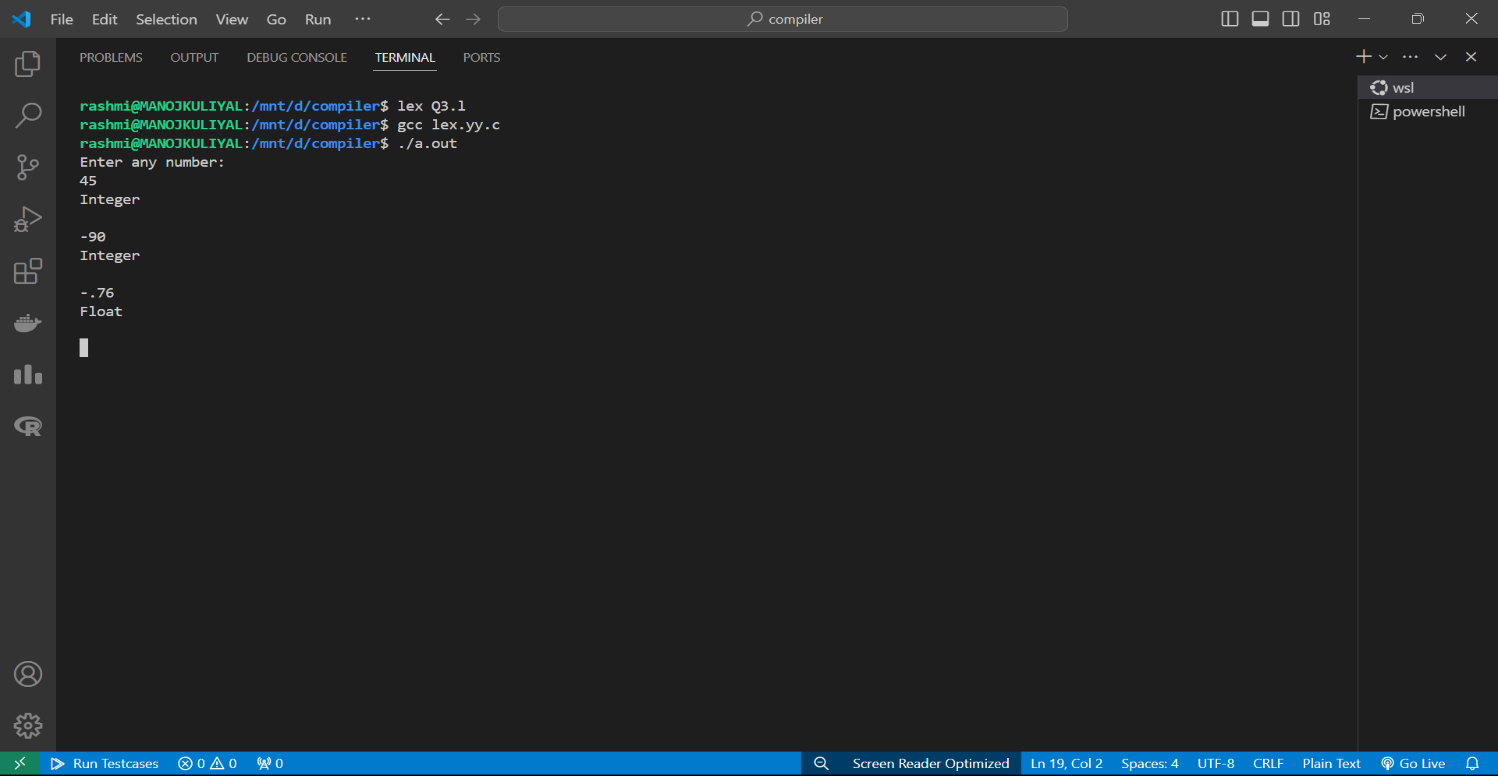
{

printf("Enter any number: \n");

yylex();

return 0;

}

**OUTPUT**

**Q.4** Design a LEX Code for Tokenizing (Identify and print OPERATORS, SEPERATORS, KEYWORDS, IDENTIFERS) the following C-fragment:

int p=1,d=0,r=4,

float m=0.0, n=200.0,

while (p <= 3)

{ if(d==0)

{ m= m+n\*r+4.5, d++, }

else

{ r++, m=m+r+1000.0, }

p++, }

**Program:**

%{

#include<stdio.h>

%}

%%

[{},;() ] {printf("Seprators:%s\n",yytext);}

[+|-|\*|/|%|=|==|++|<] {printf("operators:%s\n",yytext);}

[while|if|else|int|float] {printf("Keywords:%s\n",yytext);}

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

. {}

%%

int yywrap()

{

return 1;

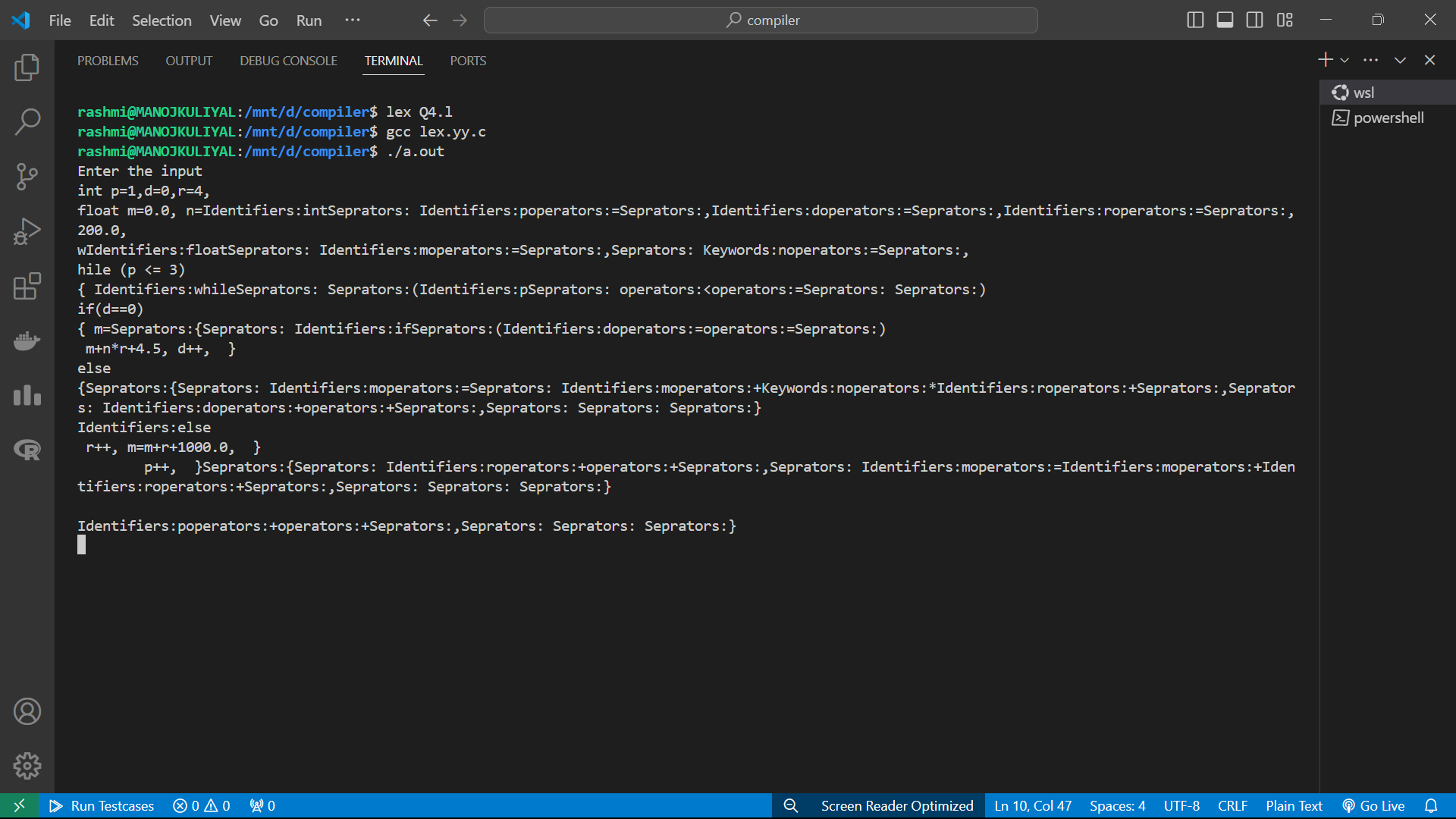
}

int main()

{

printf("Enter the input\n");

yylex();}

**OUTPUT**

**Q.5 Design a LEX Code to count and print the number of total characters, words, white spaces in given ‘Input.txt’ file.**

**Program:**

wap to count and print the number of total characters ,words,white spaces

%{

#include<stdio.h>

int chars=0,spaces=0,words=0;

%}

%%

[\t " "]+ {chars++;spaces++;}

([a-zA-Z0-9]\*) {words++;}

.|\n {chars++;}

%%

int yywrap()

{

return 1;

}

int main()

{

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

yylex();

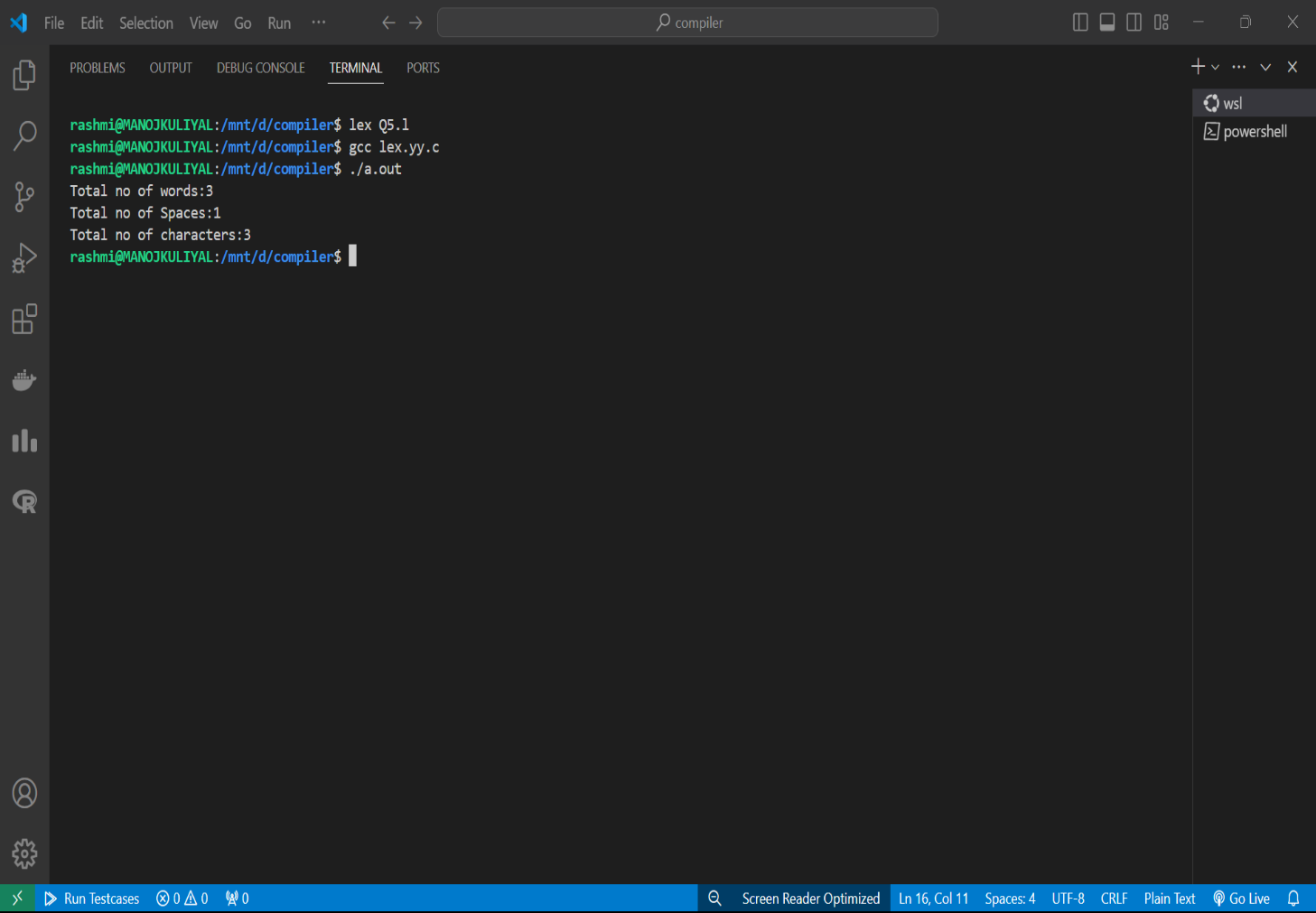
printf("Total no of words:%d\n",words);

printf("Total no of Spaces:%d\n",spaces);

printf("Total no of characters:%d\n",chars);

return 0;

}

**OUTPUT**

**Q.6** **Design a LEX Code to replace white spaces of ‘Input.txt’ file by a single blank character into ‘Output.txt’ file.**

**Program:**

%{

#include<stdio.h>

%}

%%

[\t " "]+ {fprintf(yyout," ");}

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

%%

int yywrap()

{

return 1;

}

int main()

{

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

yyout=fopen("output.txt","w");

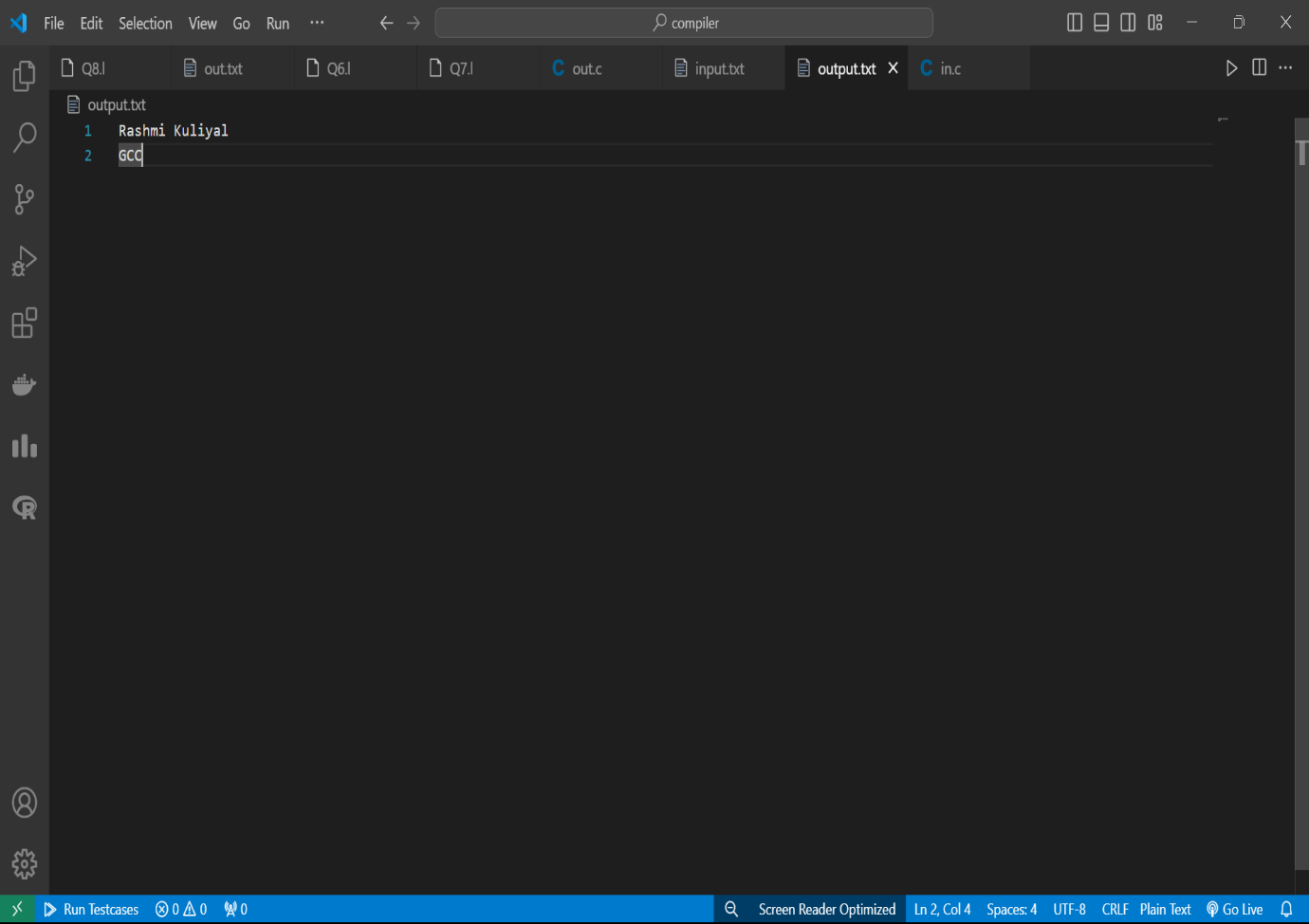
yylex();

return 0;

}

A screenshot of a computer

Description automatically generated**OUTPUT**



**Q.7** Design a LEX Code to remove the comments from any C-Program given at run-time and store into ‘out.c’ file.

**Program:**

%{

#include<stdio.h>

%}

%%

\/\/(.\*) {};

\/\\*(.\*\n)\*.\*\\*\/ {};

%%

int yywrap()

{

return 1;

}

int main()

{

printf("Enter code with comment line:");

yyout=fopen("out.c","w");

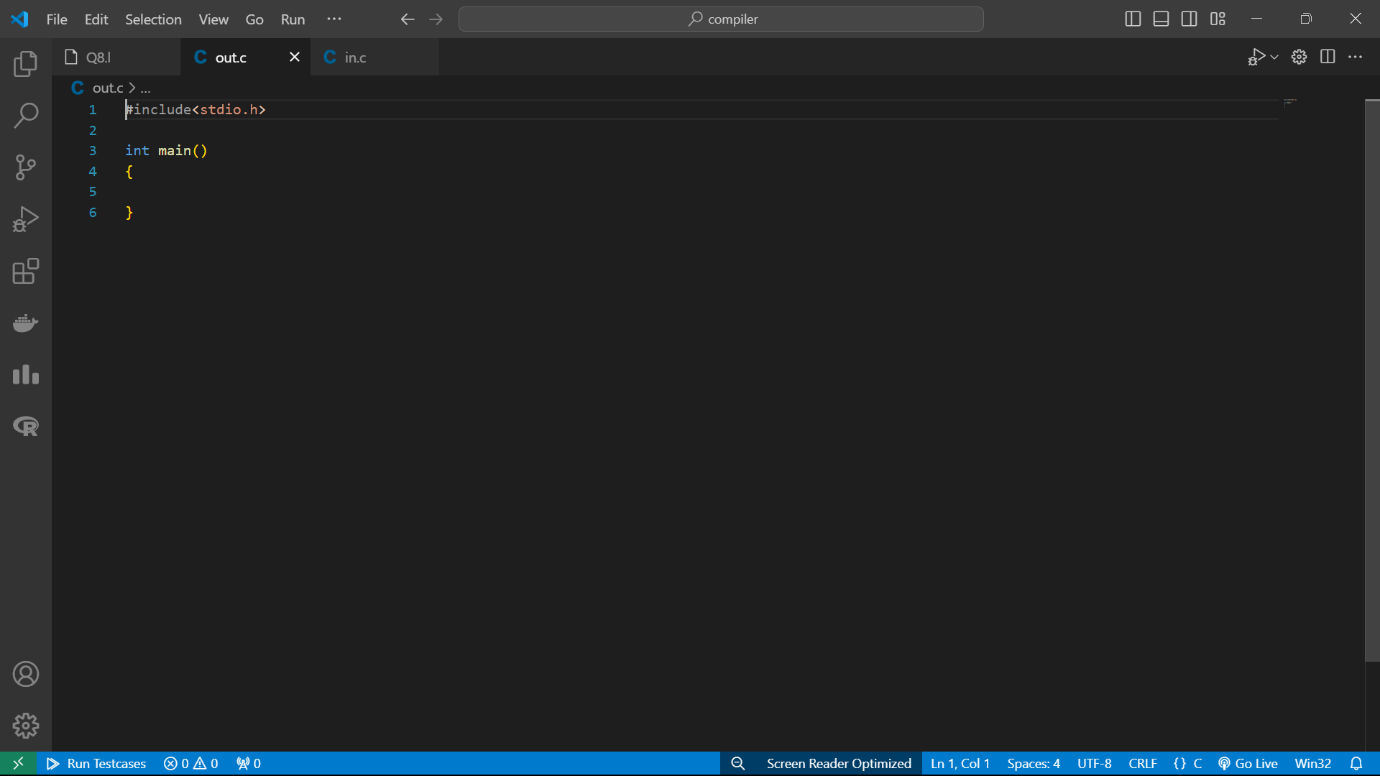
yylex();

}

**OUTPUT**

A computer screen shot of a black screen

Description automatically generated



**Q.8** Design a LEX Code to extract all html tags in the given HTML file at run time and store into Text file given at run time.

**Program:**

wap to extract all html tags at run time and store into text file given at run time

%{

#include<stdio.h>

%}

%%

\<[^>]\*\> {fprintf(yyout,"%s\n",yytext);}

.|\n {}

%%

int yywrap()

{

return 1;

}

int main()

{

printf("Enter html code:");

yyout=fopen("out.txt","w");

yylex();

}

**OUTPUT**

