Name: Tushar Panchal

En.No: 21162101014

Sub: CD(Compiler Design)

Branch: CBA

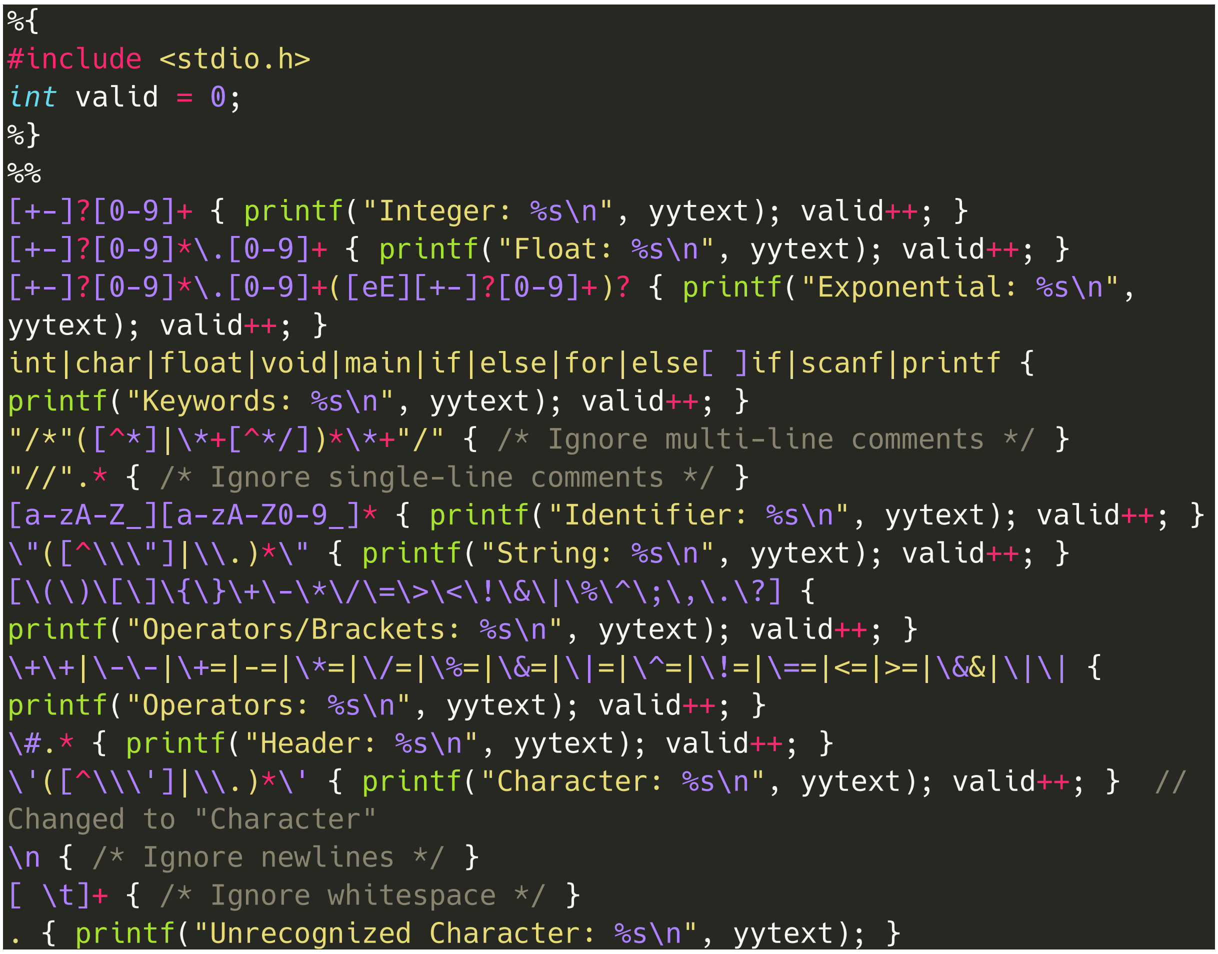
Batch:71

----------------------------PRACTICAL 05----------------------------

To implement Lexical Analyzer in LEX Tool for the input file.

Question: Tokenize the C file

✓ Word.l :



```
%{
#include <stdio.h>
int valid = 0;
%}
%%
[+-]?[0-9]+ { printf("Integer: %s\n", yytext); valid++; }
[+-]?[0-9]*\.[0-9]+ { printf("Float: %s\n", yytext); valid++; }
[+-]?[0-9]*\.[0-9]+([eE][+-]?[0-9]+)? { printf("Exponential: %s\n",
yytext); valid++; }
int|char|float|void|main|if|else|for|else[ ]if|scanf|printf {
printf("Keywords: %s\n", yytext); valid++; }
"/*"([^*]|\*+[^*/])*\*+"/" { /* Ignore multi-line comments */ }
"//".* { /* Ignore single-line comments */ }
[a-zA-Z_][a-zA-Z0-9_]* { printf("Identifier: %s\n", yytext); valid++; }
\"([^\\\"]|\\.)*\" { printf("String: %s\n", yytext); valid++; }
[\(\)\[\]\{\}\+\-\*\/\=\>\<\!\&\|\%\^\;\,\.\?] {
printf("Operators/Brackets: %s\n", yytext); valid++; }
\+\+|\-\-|\+=|-=|\*=|\/=|\%=|\&=|\|=|\^=|\!=|\==|<=|>=|\&&|\|\| {
printf("Operators: %s\n", yytext); valid++; }
\#.* { printf("Header: %s\n", yytext); valid++; }
\'([^\\\']|\\.)*\' { printf("Character: %s\n", yytext); valid++; }  //
Changed to "Character"
\n { /* Ignore newlines */ }
[ \t]+ { /* Ignore whitespace */ }
. { printf("Unrecognized Character: %s\n", yytext); }
```

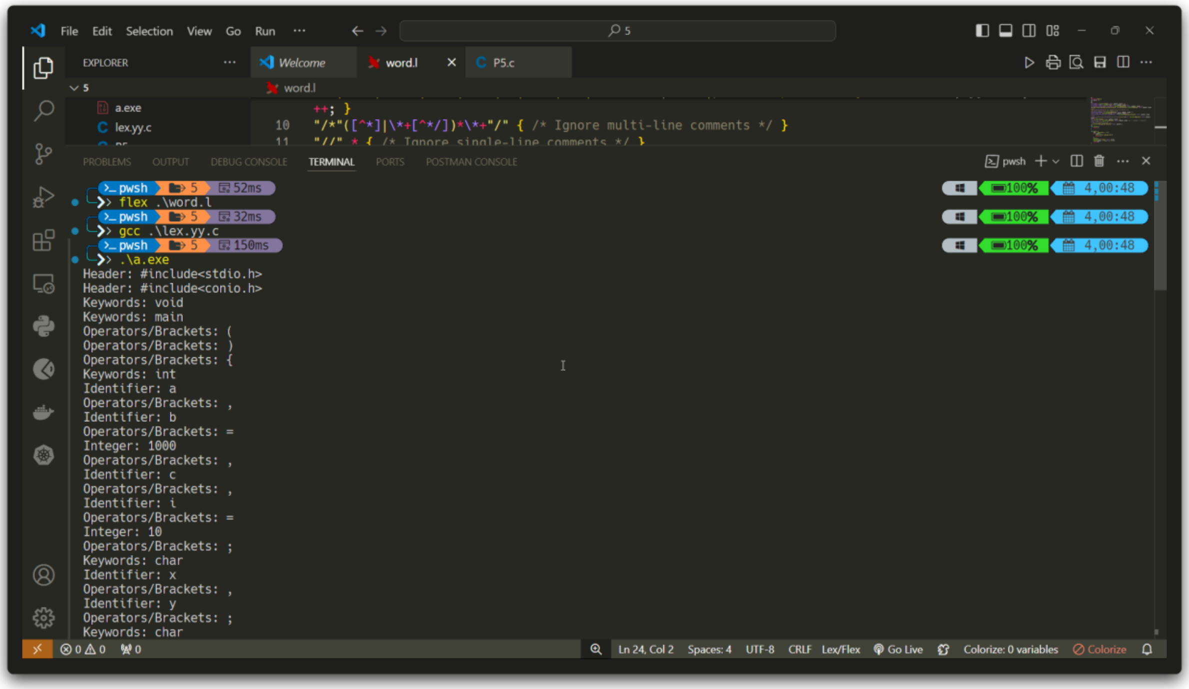
```
%%
int yywrap() {
    return 1;
}

int main() {
    yyin = fopen("p5.c", "r");
    if (!yyin) {
        perror("Error opening file");
        return 1;
    }
    yylex();
    printf("Valid Tokens: %d\n", valid);
    fclose(yyin);  // Added file closure
    return 0;
}
```

✓ **P5.c :**

```
#include<stdio.h>
#include<conio.h>

void main() {
    // Declaration of variable
    int a, b = 1000, c, i = 10; // Integer declarations
    char x, y;                 // Character declarations
    char a = 'x';              // Incorrect: redeclaration of 'a'
    float p = 10.2, q = 20.5;  // Float declarations

    scanf("%d %d", &a, &b);   // Input two integers

    /*
    Addition of Two number
    */
    c = a + b;                 // Addition of a and b
    printf("Sum: %d", c);      // Print sum

    // Comment1
    if (a > b) {               // If statement to check max
        printf("a is max");
    } else {
        printf("b is max");
    }

    a = b++ + c++;             // Post-increment a and b
    a += b;                    // Add b to a
    b = c && a;                // Logical AND assignment

    // Print 1 to 100
    for (i = 1; i < 100; i++) { // Loop to print numbers
```

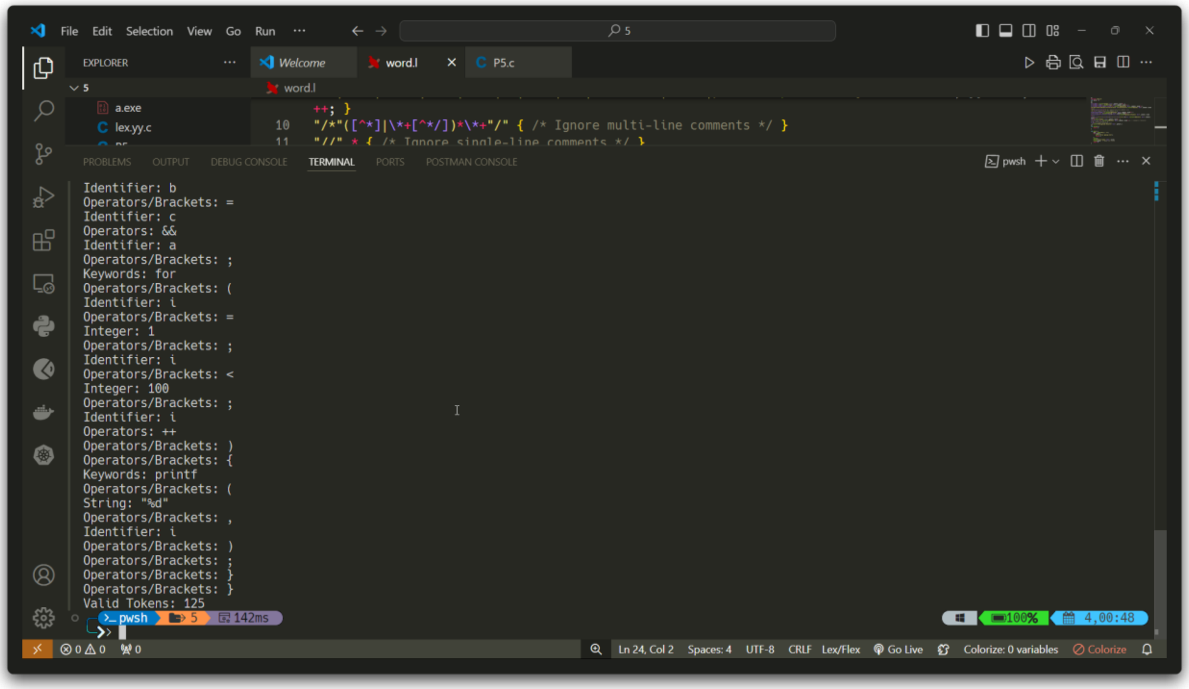
```
        printf("%d", i);
    }
}
```

✓ Output :



```
++; }
"/*"([^*]|\*+[^*/])*\*+"/" { /* Ignore multi-line comments */ }
"//" .* { /* Ignore single-line comments */ }
```

```
pwsh  5  52ms
>> flex .\word.l
pwsh  5  32ms
>> gcc .\lex.yy.c
pwsh  5  150ms
>> .\a.exe
Header: #include<stdio.h>
Header: #include<conio.h>
Keywords: void
Keywords: main
Operators/Brackets: (
Operators/Brackets: )
Operators/Brackets: {
Keywords: int
Identifier: a
Operators/Brackets: ,
Identifier: b
Operators/Brackets: =
Integer: 1000
Operators/Brackets: ,
Identifier: c
Operators/Brackets: ,
Identifier: i
Operators/Brackets: =
Integer: 10
Operators/Brackets: ;
Keywords: char
Identifier: x
Operators/Brackets: ,
Identifier: y
Operators/Brackets: ;
Keywords: char
```



```
Identifier: b
Operators/Brackets: =
Identifier: c
Operators: &&
Identifier: a
Operators/Brackets: ;
Keywords: for
Operators/Brackets: (
Identifier: i
Operators/Brackets: =
Integer: 1
Operators/Brackets: ;
Identifier: i
Operators/Brackets: <
Integer: 100
Operators/Brackets: ;
Identifier: i
Operators: ++
Operators/Brackets: )
Operators/Brackets: {
Keywords: printf
Operators/Brackets: (
String: "%d"
Operators/Brackets: ,
Identifier: i
Operators/Brackets: )
Operators/Brackets: ;
Operators/Brackets: }
Operators/Brackets: }
Valid Tokens: 125
pwsh  5  142ms
>>
```