## Question 1

Design a LEX Code to count the number of lines, space, tab-meta character, and rest of characters in each Input pattern.

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>

    int noLines = 0;
    int noSpace = 0;
    int noTabs = 0;
    int noCharacter = 0;
%}

%%
\n noLines++;
\t noTabs++;
[ ] noSpace++;
. noCharacter++;
%%

int main(){
    yylex();
    printf("noLines: %d\n", noLines);
    printf("noTabs: %d\n", noTabs);
    printf("noSpaces: %d\n", noSpace);
    printf("noWords: %d\n", noCharacter);
    return 0;
}
```

### Output:

```
hi this is _    my place
i live here *
noLines: 2
noTabs: 1
noSpaces: 7
noWords: 26
```

## Question 2

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

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

%%
^[a-zA-Z_][a-zA-Z0-9_]* printf("Valid Identifier: %s\n", yytext);
.* printf("Invalid Identifier: %s\n", yytext);
%%

int main(){
    yylex();
    return 0;
}
```

### Output:

```
2hi
Invalid Identifier
shi
Valid Identifier
_shi
Valid Identifier
_23sdh
Valid Identifier
```

## Question 3

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

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

%%
[0-9]*"."[0-9]* printf("Float value: %s\n", yytext);
[0-9]+ printf("Integer Value: %s\n", yytext);
.|\n {/* Ignore all other characters. */}
%%

int main(){
    yylex();
    return 0;
}
```

### Output:

```
231
Integer Value
123.32
Float value
.32
Float value
123.
Float value
213.1
Float value
2
Integer Value
```

## Question 4

Design a LEX Code for Tokenizing (Identify and print OPERATORS, SEPARATORS, KEYWORDS, IDENTIFIERS) from 'in.c' file.

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

KEYWORD int|float|if|else|while|main|return
SEPARATOR [,;(){}]
OPERATOR ==|<=|>=|--|"++"|[*+-/%=<>!~]
ID [a-zA-Z_][a-zA-Z0-9_]*

%%
{KEYWORD} { printf("Keyword: %s\n", yytext); }
{OPERATOR} { printf("Operator: %s\n", yytext); }
{SEPARATOR} { printf("Seperator: %s\n", yytext); }
{ID} { printf("Identifier: %s\n", yytext); }
.|\n {/* Ignore all other characters. */}
%%

int main(){
    extern FILE *yyin;
    yyin = fopen("in.c", "r");
    yylex();
    return 0;
}
```

## Output:

**Input File:**

```
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++;  }
```

**Output:**

```
Keyword: int
Identifier: p
Operator: =
Operator: ,
Identifier: d
Operator: =
Operator: ,
Identifier: r
Operator: =
Seperator: ;
Keyword: float
Identifier: m
Operator: =
Operator: .
Operator: ,
Identifier: n
Operator: =
Operator: .
Keyword: while
Seperator: (
Identifier: p
Operator: <=
Seperator: )
Seperator: {
Keyword: if
Seperator: (
Identifier: d
Operator: ==
Seperator: )
Seperator: {
Identifier: m
Operator: =
Identifier: m
Operator: +
Identifier: n
Operator: *
Identifier: r
Operator: +
Operator: .
Seperator: ;
Identifier: d
Operator: ++
Seperator: ;
Seperator: }
Keyword: else
Seperator: {
Identifier: r
Operator: ++
Seperator: ;
Identifier: m
Operator: =
Identifier: m
Operator: +
Identifier: r
Operator: +
Operator: .
Seperator: ;
Seperator: }
Identifier: p
Operator: ++
Seperator: ;
Seperator: }
```

## Question 5

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

### Code:

```
%{
    // c code
    #include <stdio.h>

    int noCharacters = 0;
    int noWords = 0;
    int noSpace = 0;
%}

%%
[ ] {noSpace++; printf("space: \"%s\"", yytext);}
[^ \n\t]+ {noWords++, noCharacters=noCharacters+yyleng; printf("words: \"%s\""
    , yytext);}
\n {noCharacters++; printf("char: \"%s\"", yytext);}
%%

int main(){
    extern FILE *yyin;
    yyin = fopen("input.txt", "r");
    FILE *fp = fopen("output.txt", "w");
    yylex();
    fprintf(fp, "noWords: %d\n", noWords);
    fprintf(fp, "noSpaces: %d\n", noSpace);
    fprintf(fp, "noCharacters: %d\n", noCharacters);
    fclose(fp);
    return 0;
}
```

### Output:

**Input File:**

```
hii total no of
words in this file are
10
```

**Output:**

```
noWords: 10
noSpaces: 7
noCharacters: 34
```

## Question 6

Design a LEX Code to replace white spaces of Input.txt file by a single blank character into Output.txt file.

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

%%
[ \t\n]+ fprintf(yyout, " ");
. fprintf(yyout, "%s", yytext);
%%

int main(){
    extern FILE *yyin, *yyout;
    yyin = fopen("input.txt", "r");
    yyout = fopen("output.txt", "w");
    yylex();
    return 0;
}
```

### Output:

**Input File:**

```
hi my   name is ram
and i am in ds section
```

**Output File:**

```
hi my name is ram and i am in ds section
```

## Question 7

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

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

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

int main(){
    extern FILE *yyin, *yyout;
    yyin = fopen("input.c", "r");
    yyout = fopen("out.c", "w");
    yylex();
    return 0;
}
```

## Output:

**Input File:**

```
#include <stdio.h>

// main function
int main()
{
    /* code */
    printf("hello world!");
    return 0;
}
```

**Output File:**

```
#include <stdio.h>


int main()
{

    printf("hello world!");
    return 0;
}
```

## Question 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.

### Code:

```
%{
    // c code
    #include <stdio.h>
    #include <stdlib.h>
%}

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

int main(){
    extern FILE *yyin, *yyout;
    char in[100], out[100];
    printf("Enter the input filename: ");
    scanf("%s", in);
    printf("Enter the ouput filename: ");
    scanf("%s", out);
    yyin = fopen(in, "r");
    yyout = fopen(out, "w");
    yylex();
    return 0;
}
```

## Output:

**Input File:**

```
<!DOCTYPE html>
<html>

<head>
    <title>Document</title>
</head>

<body>
    <p>
        This is a html file.
    </p>
</body>

</html>
```

**Output File:**

```
<!DOCTYPE html>
<html>
<head>
<title>
</title>
</head>
<body>
<p>
</p>
</body>
</html>
```