

PRACTICAL-2

AIM:

Implement a lexical analyzer for identification of numbers.

IMPLEMENTATION:

- lex <filename with .l extension>
- gcc <newly created .c file> -o <file name for exe file>
- <filename of exe file>

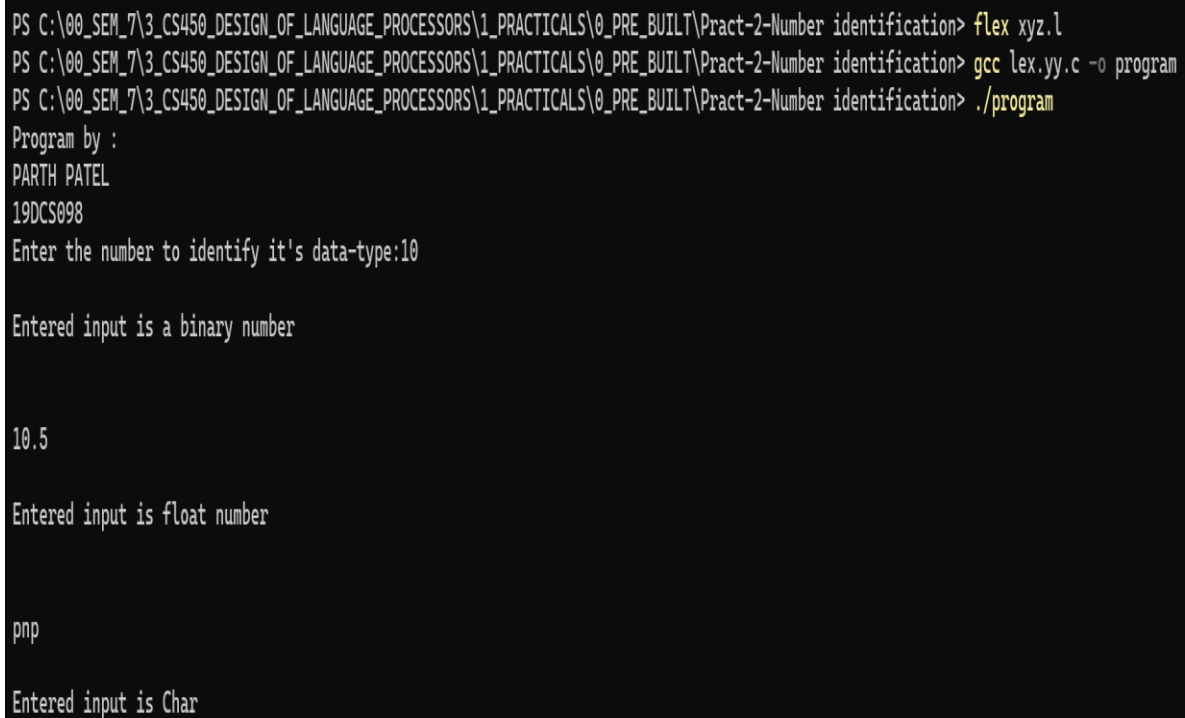
In this case, create an extra text file named abc.txt which will contain some C code to work as input for lexical analysis.

PROGRAM CODE:

```
bin (0|1)*
oct  [0-7]
char [A-Za-z]*
dec  [0-9]*
digit [0-9]
float {digit}+("."{digit}+)?
expo {digit}+("."{digit}+)?("E"("+|-")?{digit}+)?
hex  [0-9a-fA-f]+
%%
{bin} printf("\nEntered input is a binary number");
{oct} printf("\nEntered input is a Octal number");
{char} printf("\nEntered input is Char");
{dec} printf("\nEntered input is decimal number");
{float} printf("\nEntered input is float number");
{expo} printf("\nEntered input is expo. number");
{hex} printf("\nEntered input is hex number");
%%
int yywrap()
{
```

```
    return 1;
}
int main()
{
    printf("Program by : \nPARTH PATEL\n19DCS098\n");
    printf("Enter the number to identify it's data-type:");
    yylex();
    return 0;
}
```

OUTPUT:



```
PS C:\00_SEM_7\3_CS450_DESIGN_OF_LANGUAGE_PROCESSORS\1_PRACTICALS\0_PRE_BUILT\Pract-2-Number identification> flex xyz.l
PS C:\00_SEM_7\3_CS450_DESIGN_OF_LANGUAGE_PROCESSORS\1_PRACTICALS\0_PRE_BUILT\Pract-2-Number identification> gcc lex.yy.c -o program
PS C:\00_SEM_7\3_CS450_DESIGN_OF_LANGUAGE_PROCESSORS\1_PRACTICALS\0_PRE_BUILT\Pract-2-Number identification> ./program
Program by :
PARTH PATEL
19DCS098
Enter the number to identify it's data-type:10

Entered input is a binary number

10.5

Entered input is float number

pnp

Entered input is Char
```