JADAVPUR UNIVERSITY

Faculty of Engineering & Technology

...CSE/PC/B/S/322 Compiler Design Lab...Engg. Laboratory

Class....CSE, UG3... Sec.....A1.....
Date of Experiment.....
Date of Submission.....
Marks Obtained.....
Signature of Examiner.....

NAME	Roll
......Jeetesh Abrol.....	.....002210501021.....
.....	.....
.....	.....
.....	.....
.....	.....
.....	.....

Title.... Generation of Lexical Analyzer using Lex

Commence at..... **Completed at.....**

Name of Teacher concerned: Prof. Nandini Mukherjee

Q1: Write a lex file to check the validity of a binary message that starts with 1 and ends with 101 and contains any number of dots in between the parts of the messages.

Code Implementation:

```
%{
#include<stdio.h>
%}

%%
1[.]*101        {printf("%s is a valid string",yytext);}
.*              {printf("%s is an invalid string",yytext);}
%%

int main()
{
        printf("Enter the string: ");
        yylex();
        return 0;
}

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

Output:

```
jeet@jeet-vivobook:~/basicCodes/compilerDesignAss2AndAss3/ass3/ques1$ ./q1
Enter the string: 1101
1101 is a valid string
1..101
1..101 is a valid string
1.......101
1.......101 is a valid string
1....1....101
1....1....101 is an invalid string
```

Q2: Write a lex program to recognize a string that starts with a capital letter which is followed by any small letter or decimal digits and ends with a special character of your choice.

Code Implementation:

```
%{
#include<stdio.h>
%}

%%
[A-Z]+([a-z]+|[0-9]*[.][0-9]+)#        {printf("%s is a valid string\n",yytext);}
.*        {printf("%s is an invalid string\n",yytext);}
%%
int main(){
        printf("Enter the string: ");
        yylex();
        return 0;
}
int yywrap(){
        return 1;
}
```

Output:

```
jeet@jeet-vivobook:~/basicCodes/compilerDesignAss2AndAss3/ass3/ques2$ ls
lex.yy.c  q2  q2.l
jeet@jeet-vivobook:~/basicCodes/compilerDesignAss2AndAss3/ass3/ques2$ ./q2
Enter the string: Db#
Db# is a valid string

Db$
Db$ is an invalid string

R.7#
R.7# is a valid string

T9.01#
T9.01# is a valid string

R9.#
R9.# is an invalid string
```

Q3: Write a lex program that will function as a calculator and perform the operations like addition, subtraction, multiplication, division, modulo and power.

Code Implementation:

```
%{
   #include<stdio.h>
   #include<math.h>

   #define NUM     1
   #define ADD     2
   #define SUB     3
   #define MUL     4
   #define DIV     5
   #define MOD     6
   #define POW     7
   #define EXIT    8
%}

%%
[0-9]+              {return NUM;}
"+"                 {return ADD;}
"-"                 {return SUB;}
"*"                 {return MUL;}
"/"                 {return DIV;}
"%"                 {return MOD;}
"^"                 {return POW;}
[eE][xX][iI][tT]    {return EXIT;}
[ \t\n]             ;
.               {printf("%s is invalid input.\n",yytext);
                    return 0;}
%%
```

```
int main(){
    int n1,n2;
    char op;
    printf("\nSimple Calculator:\n\n");
    printf("------------------------------------------------\n");
    printf("|    Enter the expression like 2+5 or 3*6       |\n");
    printf("------------------------------------------------\n");
    printf("|     To exit enter exit(case-insensitive)      |\n");
    printf("------------------------------------------------\n");
    printf("|   add(+), sub(-), mul(*), div(/), mod(%%) and  |\n|           pow(^) are supported
|\n");
    printf("------------------------------------------------\n\n");

    while(1){
        printf("$   ");

        int token=yylex();
        if(token==EXIT){
            printf("\nExiting...\n");
            break;
        }
        if(token!=NUM){
            printf("Error:Expected a number.\n");
            return 1;
        }
        n1=atoi(yytext);

        op=yylex();

        if(yylex()!=NUM){
            printf("Error:Expected a number.\n");
            return 1;
```

```
}
n2=atoi(yytext);

switch(op){
   case ADD:
      printf("%d + %d = %d\n",n1,n2,n1+n2);
      break;
   case SUB:
      printf("%d - %d = %d\n",n1,n2,n1-n2);
      break;
   case MUL:
      printf("%d * %d = %d\n",n1,n2,n1*n2);
      break;
   case DIV:
      if(n2==0){
         printf("Error: Encountered Division by Zero.\n");
      }
      else{
         printf("%d / %d = %d\n",n1,n2,n1/n2);
      }
      break;
   case MOD:
      if (n2==0){
         printf("Error: Modulo by zero.\n");
      }
      else {
         printf("%d %% %d = %d\n", n1, n2, n1 % n2);
      }
      break;

   case POW:
      if(n1==0 && n2==0){
```

```
            printf("Error: 0^0 is invalid\n");
        }
        else{
            printf("%d ^ %d = %f\n", n1, n2, pow(n1, n2));
        }
        break;
      default:
        printf("Error:Invalid Operator.\n");
        break;
    }
  }
  return 0;
}

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

Output:

```
jeet@jeet-vivobook:~/basicCodes/compilerDesignAss2An

Simple Calculator:

-------------------------------------------------
|    Enter the expression like 2+5 or 3*6       |
-------------------------------------------------
|     To exit enter exit(case-insensitive)      |
-------------------------------------------------
|   add(+), sub(-), mul(*), div(/), mod(%) and  |
|           pow(^) are supported                |
-------------------------------------------------

$   4+6
4 + 6 = 10
$   7 + 8
7 + 8 = 15
$   3^7
3 ^ 7 = 2187.000000
$   3&6
& is invalid input.
Error:Invalid Operator.
$   4/2
4 / 2 = 2
$
```