Name: novojit das
ID: CSE 066 07773

Ans No 1)

```
%{
int tokencount = 0;
%}

%%
((April)[ ](1[2-9]|[1-2][0-9]|3[0-1])[ ]\,(1971)))|
((May|July|August|October|December)[ ](0[1-9]|[1-2][0-9]|3[0-1])[ ]\,(1971))|
((June|September|November )[ ](1[0-9]|[2][0-9]|3[0])[ ]\,(1971))|
((January|March|May|)[ ](0[1-9]|[1-2][0-9]|3[0-1])[ ]\,(2021))|
((February|April|June)[ ](0[1-9]|[1-2][0-9]|3[0])[ ]\,(2021))|
((July)[ ](0[1-9]|[1][0-9]|2[0-6])[ ]\,(2021))|
((January|March|May|July|August|October|December)[ ](0[1-9]|[1-2][0-9]|3[0-1])[ ]\,( 197[2-9]|198[0-9]|199[0-9]|200[0-9]|201[0-9]|2020))|
((February|April|June|September|November )[ ](0[1-9]|[1-2][0-9]|3[0])[ ]\,(197[2-9]|198[0-9]|199[0-9]|200[0-9]|201[0-9]|2020))|
{printf("%s Accepted\n",yytext);tokencount++;}
(\(\x\/[0-9]+\)\^\2[ ]\-[ ]\(\y\/[0-9]+\)\^\2[ ]\=[ ]\1) {printf("%s Accepted\n",yytext);tokencount++;}
ip {printf("%s Accepted\n",yytext);tokencount++;}
(ing)$  {printf("%s Accepted\n",yytext);tokencount++;}
.  {printf("Not Accepted\n");}

%%
```

```
int main(){

        yylex();
        printf("number of tokens is : %d\n", tokencount);
        }
```

Ans No 2)

```
%{
int a = 0;
int s = 0;
int n = 0;
int o = 0;
int t = 0;
int w = 0;
int alw = 0;
int alm = 0;

%}

%%
(actually)[ ] {printf("%s token\n",yytext);a++;}
(seriously)[ ] {printf("%s token\n",yytext);s++;}
(never)[ ] {printf("%s token\n",yytext);n++;}
(only)[ ] {printf("%s token\n",yytext);o++;}
(truly)[ ] {printf("%s token\n",yytext);t++;}
```

```
(well)[ ] {printf("%s token\n",yytext);w++;}
(always)[ ] {printf("%s token\n",yytext);alw++;}
(almost)[ ] {printf("%s token\n",yytext);alm++;}


int main(){
        FILE *file;
file = fopen("code.c", "r") ;
if (!file) {
printf("couldnot open file");
exit (1);
}
else {
yyin = file;
}
        yylex();
        printf("number of actually: %d\n", a);
    printf("number of seriously: %d\n", s);
    printf("number of never: %d\n", n);
        printf("number of only: %d\n", o);
        printf("number of truly: %d\n", t);
        printf("number of well: %d\n", w);
        printf("number of always: %d\n", alw);
        printf("number of almost: %d\n", alm);

        }
```

Ans no 3)

Lex.l file

```
%{
#include "y.tab.h"
%}

%%
[0-9]+ { yylval.dval = atof(yytext); return NUMBER;}
[0-9]+\.[0-9]+ { yylval.dval = atof(yytext); return NUMBER;}
[ \t]+ {} /* ignore whitespace */
\n {return 0;} /* logical EOF */
(ceil) {return CEIL;}
(tan) {return TAN;}
(log) {return LOG;}

. {return yytext[0];}
%%
```

YAC.Y FILE

```
%{
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
yylex();
%}

%union{
        double dval;
```

```
	int vblno;
}

%token <dval> NUMBER



%token CEIL TAN LOG

%type <dval> statements
%type <dval> expressions

%%

statements: expressions { printf("= %lf\n",$1); }
	;
expressions: NUMBER '*' expressions '-' expressions { $$ = $1 * $3 - $5; }
	|CEIL '(' expressions ')' {$$ = ceil($3);}
	|TAN '(' expressions ')' {$$ = tan($3);}
	|LOG '(' expressions ')' {$$ = log($3);}
	| NUMBER { $$ = $1; }
	;
%%

int main(){
	yyparse();
}
```