Expt. No. 4 Page No. 11

Expt. Name. Use LEX & YACC tool to implement Desktop Calculator Date :

Aim:

To write the program using LEX and YACC to implement ~~parser or~~ a Desktop Calculator

Algorithm:

File.l

step 1: start
step 2: Include the necessary header files and declare the necessary variables
step 3: Define the keywords and the identifiers with the constant and operator
step 4: Get the input for analysis from user
step 5: check each and every element in the statement with number
step 6: check each and every element in the statement with small alphabet
step 7: check each and every element for the operator
step 8: Else print Invalid Token
step 9: return the value
step 10: stop

File.y

step 1: start
step 2: Include the necessary header files and declare the necessary variables
step 3: Define the keywords and the identifiers with the constant and operator.
step 4: Take the value which was taken from user and implement the respective operator.
step 5: return the value and print it.

Expt. No. _____ Page No. 12

Expt. Name. _____ Date : _____

Step 6: Stop

Program:

File.l

```
%{
    #include <stdlib.h>
    #include "y.tab.h"
    void yyerror(char *s);
    extern int yylval;
%}
%%
[0-9]+      {yylval = atoi(yytext); return INT;}
[a-z]       {yylval = toascii (*yytext)-97;} return ID;}
[A-Z]       {yylval = to ascii (*yytext)-65; return ID;}
[-+*=\n]    {return *yytext;}
'('         {return *yytext;}
')'         {return *yytext;}
[\t] ;
.           {yyerror ("Invalid Token!!");}
%%
int yywrap()
{
return 1;
}
```

File.y

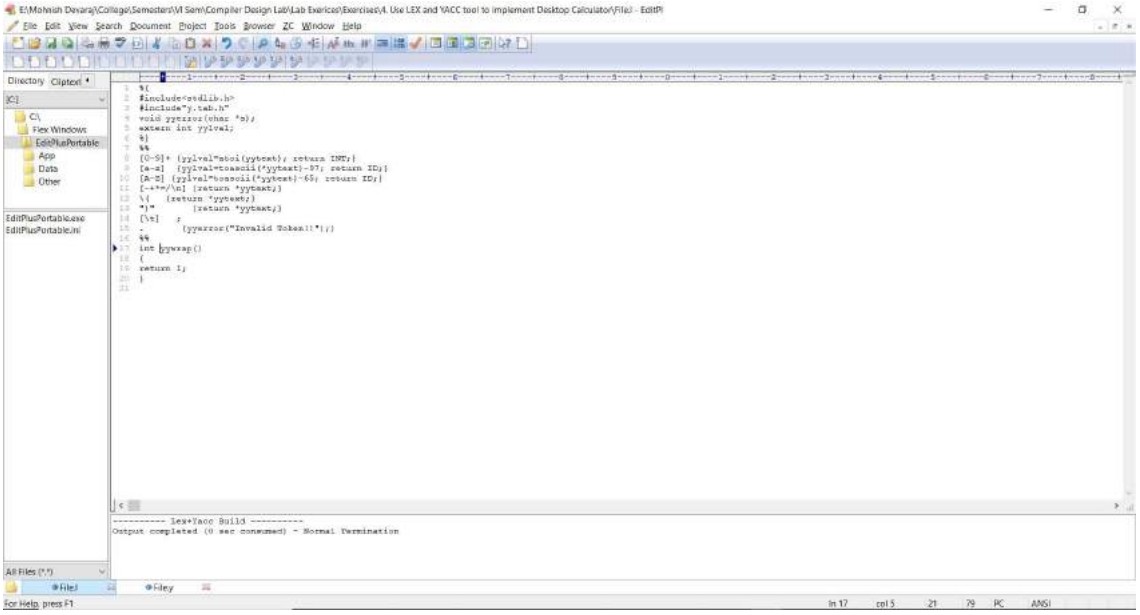
```
%{
#include <stdio.h>
extern int yylex(void);
void yyerror(char *);
```

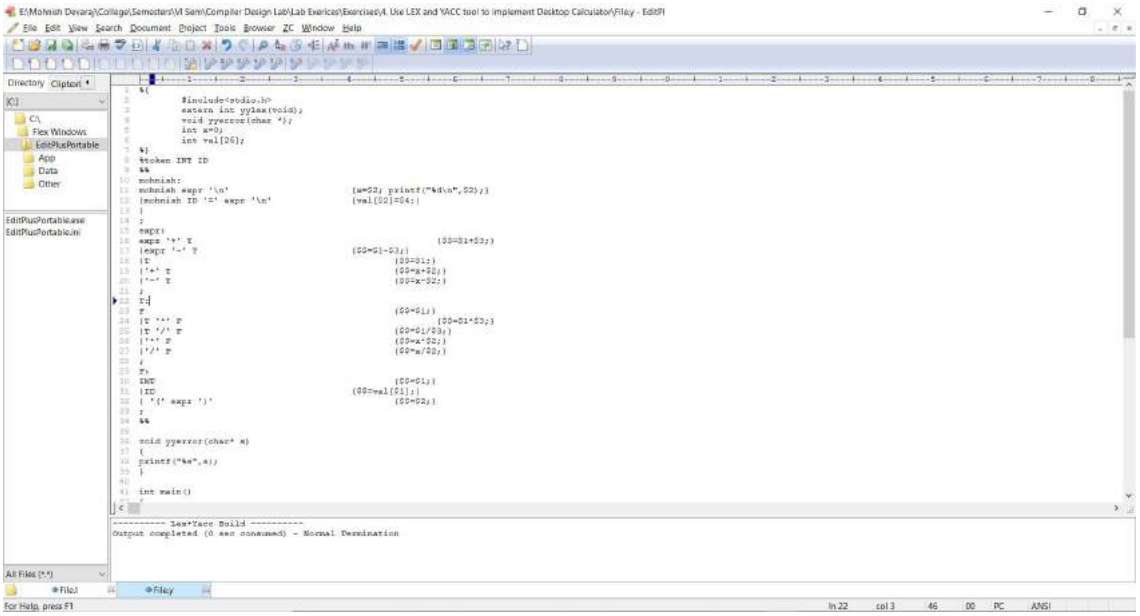
```
int x=0
int val[26];
%}
% token INT ID
%%
Mohnish:
 mohnish expr '\n'          { x=$2; printf("%d\n", $2);}
|mohnish ID '=' expr '\n'    { val[$2]=$4;}
|
;
 expr:
 expr '+' T                 {$$ = $1 + $3;}
|expr '-' T                 {$$= $1 - $3;}
| T                         {$$ = $1;}
| '+' T                     {$$ = x + $2;}
| '-' T                     {$$ = x - $2;}
;
T:
 F                          {$ $= $1;}
|T '*' F                    {$$ = $1 * $3;}
|T '/' F                    {$$ = $1 / $3;}
| '*' F                     {$$ = x * $2;}
| '/' F                     {$$ = x / $2;}
;
 INT                        {$$ = $1;}
|ID                         {$$ = val[$1];}
| '(' expr ')'              {$$ = $2;}
 ;
%%
```

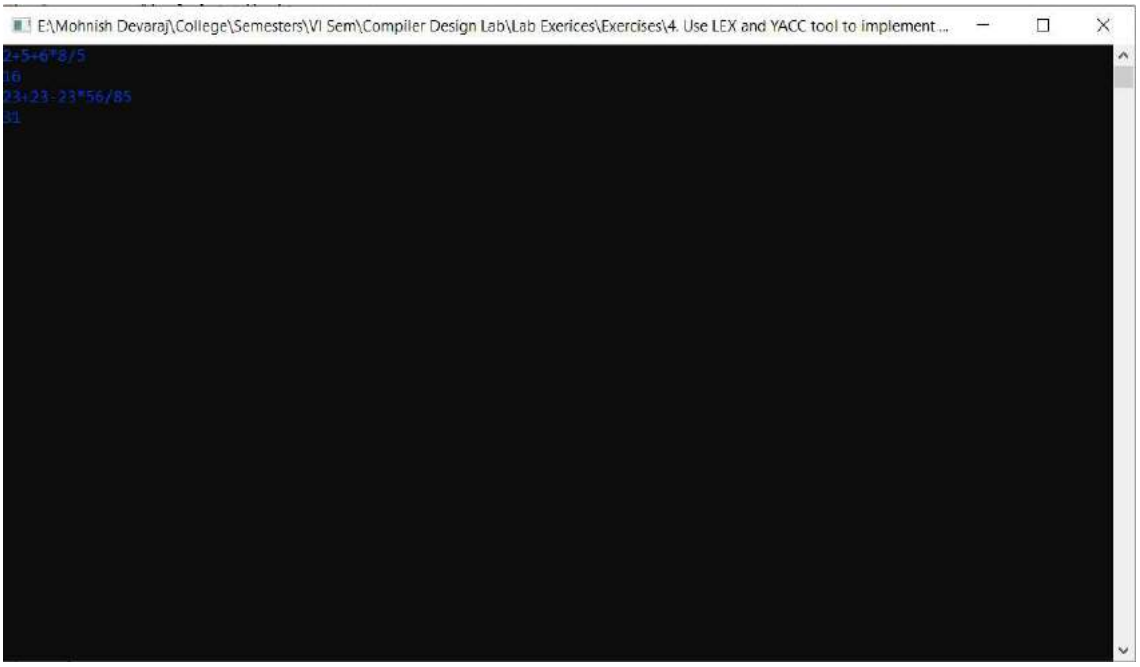
```
void yyerror (char* s)
{
printf ("%s", s);
}

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

Result:

Use LEX and YACC to implement ~~parsers for~~ Desktop Calculator is executed successfully.


E:\Mohnish Devaraj\College\Semesters\VI Sem\Compiler Design Lab\Lab Exercises\Exercises\4. Use LEX and YACC tool to implement Desktop Calculator\File1 - EditPl
File Edit View Search Document Project Tools Browser ZC Window Help
Directory Cliptext
C:\
Flex Windows
EditPlusPortable
App
Data
Other
EditPlusPortable.exe
EditPlusPortable.ini
%{
#include<stdlib.h>
#include"y.tab.h"
void yyerror(char *s);
extern int yylval;
%}
%%
[0-9]+ {yylval=atoi(yytext); return INT;}
[a-z] {yylval=tolower(*yytext)-97; return ID;}
[A-Z] {yylval=toupper(*yytext)-65; return ID;}
[-+*/\n] {return *yytext;}
\({return *yytext;}
"\)" {return *yytext;}
[\t] ;
. {yyerror("Invalid Token!!");}
%%
int yywrap()
{
return 1;
}
---------- Lex+Yacc Build ----------
Output completed (0 sec consumed) - Normal Termination
All Files (*.*)
File1
File2
For Help, press F1
ln 17 col 5 21 79 PC ANSI


E:\Mohnish Devaraj\College\Semesters\VI Sem\Compiler Design Lab\Lab Exercises\Exercises\4. Use LEX and YACC tool to implement Desktop Calculator\File2 - EditPl
File Edit View Search Document Project Tools Browser ZC Window Help
Directory Cliptext
C:\
Flex Windows
EditPlusPortable
App
Data
Other
EditPlusPortable.exe
EditPlusPortable.ini
%{
#include<stdio.h>
extern int yylex(void);
void yyerror(char *);
int x=0;
int val[26];
%}
%token INT ID
%%
mohnish:
mohnish expr '\n' {x=$2; printf("%d\n",$2);}
|mohnish ID '=' expr '\n' {val[$2]=$4;}
|
;
expr:
expr '+' T {$$=$1+$3;}
|expr '-' T {$$=$1-$3;}
|T {$$=$1;}
|'+' T {$$=x+$2;}
|'-' T {$$=x-$2;}
;
T:
F {$$=$1;}
|T '*' F {$$=$1*$3;}
|T '/' F {$$=$1/$3;}
|'*' F {$$=x*$2;}
|'/' F {$$=x/$2;}
;
F:
INT {$$=$1;}
|ID {$$=val[$1];}
| '(' expr ')' {$$=$2;}
;
%%
void yyerror(char* s)
{
printf("%s",s);
}
int main()
{
---------- Lex+Yacc Build ----------
Output completed (0 sec consumed) - Normal Termination
All Files (*.*)
File1
File2
For Help, press F1
ln 22 col 3 46 00 PC ANSI


E:\Mohnish Devaraj\College\Semesters\VI Sem\Compiler Design Lab\Lab Exerices\Exercises\4. Use LEX and YACC tool to implement ...
2+5+6*8/5
16
23+23-23*56/85
31