**PROJCT NO : 6**

**AIM:** **Write an appropriate language description for a layman language which can do mathematical operations using English like sentences.**

**EXAMPLE:**

Add 100 ,200,300,400 . Sub 250 from result.

Mul 400 to it . Div the answer by 2. Show me

the answer.

Lab4: Implement scanner phase of compiler using flex

**Group Members:**

1. Kathiriya Darshak (IT057)
2. Limbani Nihal (IT064)
3. Karia Stuti (IT055)

**F2) Flex program for your project allotted.**

* flex is a tool for generating scanners: programs which recognize lexical patterns in text.
* flex reads the given input files (or its standard input if no file names are given) for a description of the scanner to generate.

|  |  |
| --- | --- |
| Regular Defination | Examples: |
| Keywords | From, from, Show, show, by, to, it, number, and, result, the, me, Answer, answer |
| Operation | Add, Sub, Mul, Div |
| Digit(Number) | [0-9] |
| Que. Mark | “?”//end of line |
| White Space | (Tab | Newline)**+** |
| Letter | [A-Za-z] |

**Regular Expression Related to Regular Language**

|  |  |
| --- | --- |
| **Int** | {Digit}+- Atleast one or more Digit |
| **Float** | {Digit}+(\.{Digit}+)?(E[+\-]?{Digit}+)? – Means Digit followed by digit or exponent of 10(digit) |
| **Space** | {white space}+ |

**Code:**

%{

#include<stdio.h>

int totaltk=0;

%}

Keywords "From"|"from"|"show"|"Show"|"by"|"to"|"it"|"the"|"me"|"number"|"and"|"Result"|"result"|"answer"|"Answer"

Operator "Add"|"add"|"Sub"|"sub"|"Mul"|"mul"|"Div"|"div"

Digit [0-9]

QM "?"

WS [\t\n]

Int {Digit}+

Float {Digit}+(\.{Digit}+)?(E[+\-]?{Digit}+)?

Space {WS}+

%%

{Keywords} {printf("Keyword : %s\n",yytext);totaltk++;}

{Operator} {printf("operator is: %s\n",yytext);totaltk++;}

{Int} {printf("Integer : %s\n",yytext);totaltk++;}

{Float} {printf("Float No : %s\n",yytext);totaltk++;}

{QM} {printf("\n");totaltk++;}

{Space} {}

. {}

%%

int yywrap()

{

return 1;

}

int main()

{

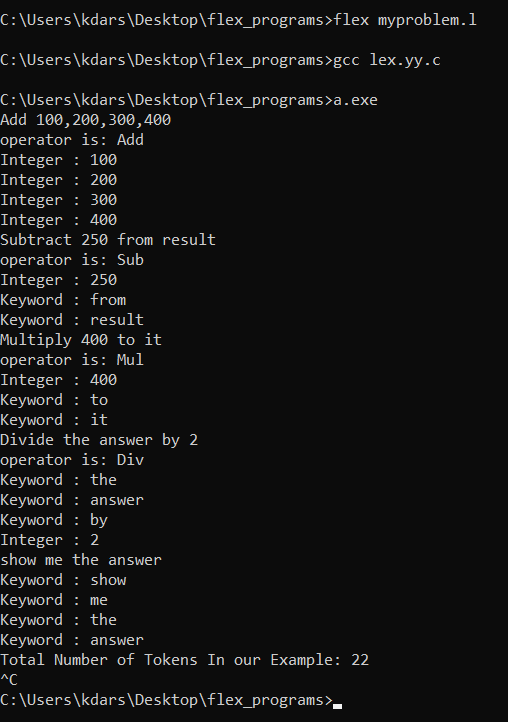
yylex();

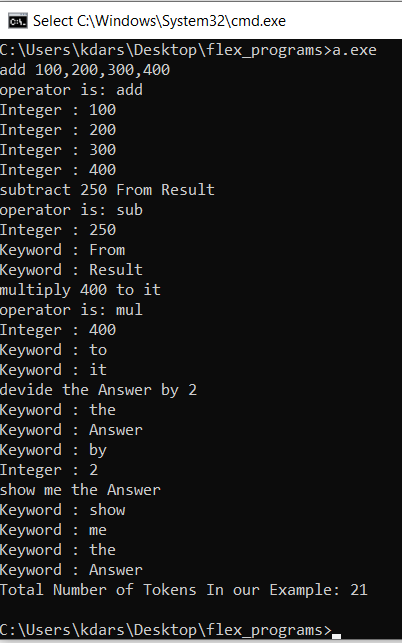
printf("Total Number of Tokens In our Example: %d\n",totaltk);

return 0;

}

**Output:**

****

****

**In second ss there is devide instead of divide. =>Show that one less token identify.**

**All see that case sensitive validation in this ss.**