

## 1. Design and Implementation of a Simple Calculator using Flex and Bison

**Description:** This project presents the design and implementation of a simple arithmetic expression compiler using **Flex** and **Bison**. The compiler takes basic mathematical expressions as input (e.g.,  $5 + 3$ ;  $10 / 2$ ;) and evaluates them by performing lexical analysis and syntax parsing. **Flex** is used to tokenize the input, while **Bison** is used to parse the tokens based on defined grammar rules. The compiler supports operations like **addition**, **subtraction**, **multiplication**, and **division**, and prints the result after successful parsing. This implementation helps in understanding the fundamental concepts of compiler construction, including token generation, grammar parsing, and syntax-directed translation.

### Bison code:

```
%{
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
int yylex(void);
void yyerror(const char *s);
}%
%union
{int num;};

%token <num> NUMBER

%%
start:
|start expr
;
expr:
NUMBER '+' NUMBER ';' {printf("sum:%d\n", $1+$3); }
|NUMBER '-' NUMBER ';' {printf("sub:%d\n", $1-$3); }
|NUMBER '/' NUMBER ';' {printf("div:%d\n", $1/$3); }
|NUMBER '*' NUMBER ';' {printf("mul:%d\n", $1*$3); }
%%

void yyerror(const char *s)
{printf("syntax error: %s\n", s);}

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

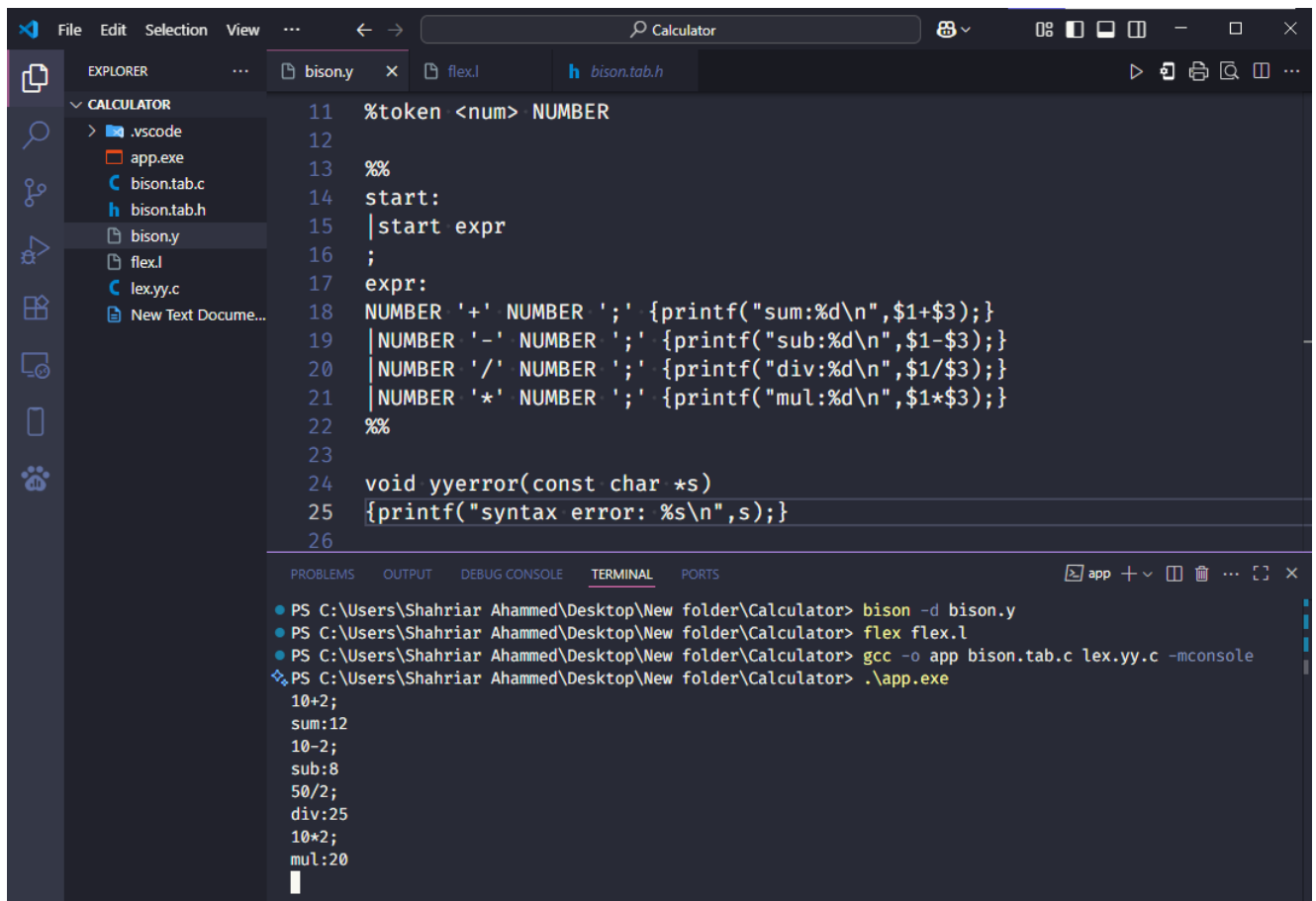
## Flex code:

```
%{
#include "cal.tab.h"
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
}%

%%
[0-9]+ {yyval.num=atoi(yytext); return NUMBER;}
[ \n\t] ;
. {return yytext[0];}
%%

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

## Output :



The screenshot shows a VS Code editor window with the file `bison.y` open. The file contains the following code:

```
11 %token <num> NUMBER
12
13 %%
14 start:
15 |start expr
16 ;
17 expr:
18 NUMBER '+' NUMBER ';' {printf("sum:%d\n",$1+$3);}
19 |NUMBER '-' NUMBER ';' {printf("sub:%d\n",$1-$3);}
20 |NUMBER '/' NUMBER ';' {printf("div:%d\n",$1/$3);}
21 |NUMBER '*' NUMBER ';' {printf("mul:%d\n",$1*$3);}
22 %%
23
24 void yyerror(const char *s)
25 {printf("syntax error: %s\n",s);}
26
```

Below the editor, the TERMINAL window shows the following commands and output:

```
PS C:\Users\Shahriar Ahammed\Desktop\New folder\Calculator> bison -d bison.y
PS C:\Users\Shahriar Ahammed\Desktop\New folder\Calculator> flex flex.l
PS C:\Users\Shahriar Ahammed\Desktop\New folder\Calculator> gcc -o app bison.tab.c lex.yy.c -mconsole
PS C:\Users\Shahriar Ahammed\Desktop\New folder\Calculator> .\app.exe
10+2;
sum:12
10-2;
sub:8
50/2;
div:25
10*2;
mul:20
```