Experiment No. 5

Lex Program(lex.l):

```
%{
#include "y.tab.h"
extern int yylval;
%}
%%
[0-9]+  { yylval = atoi(yytext); return NUM; }
"="     { return '='; }
"+"     { return '+'; }
\n      { return 0; }
.       { return yytext[0]; }
%%
int yywrap() {
   return 1;
}
```

Yacc Program(x1.y):

```
%{
#include <stdio.h>
extern int yylex(void);  // Explicit declaration for yylex
void yyerror(const char *s);  // Proper prototype for yyerror
extern FILE *yyin;
%}
%token NUM
%%
start:
   expr '=' expr { printf("\nResult = %d\n", $3); }
   | expr       { printf("\nResult = %d\n", $1); } ;
```

```
expr:
    expr '+' NUM { $$ = $1 + $3; }
    | NUM       { $$ = $1; }
    ;
%%
int main() {
    yyin = stdin;
    do {
        yyparse();
    } while (!feof(yyin));
    return 0;
}
void yyerror(const char *s) {  // Match prototype with const
    fprintf(stderr, "Error: %s\n", s);
}
```

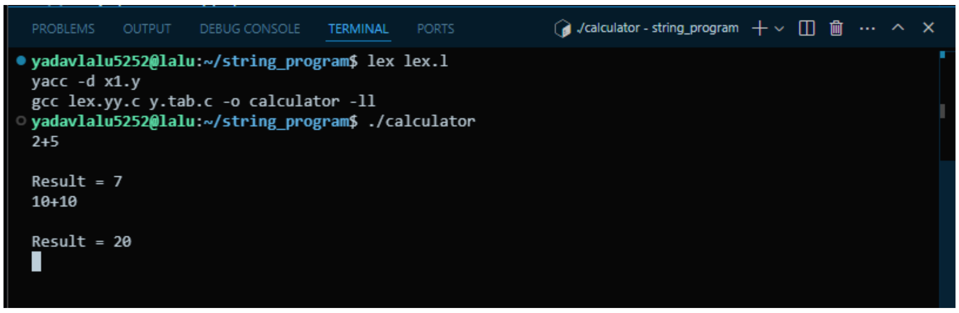
Run command:

```
lex lex.l

yacc -d x1.y

gcc lex.yy.c y.tab.c -o calculator -ll
```

Output:



```
yadavlalu5252@lalu:~/string_program$ lex lex.l
yacc -d x1.y
gcc lex.yy.c y.tab.c -o calculator -ll
yadavlalu5252@lalu:~/string_program$ ./calculator
2+5

Result = 7
10+10

Result = 20
```