```
%{
    #include<stdio.h>
    #include<stdlib.h>
    #include<math.h>
    int data[60];
%}

/* bison declarations */

%token NUM VAR IF ELSE MAIN INT FLOAT CHAR START END SWITCH CASE DEFAULT BREAK FOR PF SIN COS TAN LOG BIGMOD LOG10
%nonassoc IFX
%nonassoc ELSE
%nonassoc SWITCH
%nonassoc CASE
%nonassoc DEFAULT
%left '<' '>'
%left '+' '-'
%left '*' '/'

/* Grammar rules and actions follow.  */

%%

program: MAIN ':' START cstatement END
     ;

cstatement: /* NULL */

    | cstatement statement
    ;

statement: ';'
    | declaration ';'       { printf("Declaration\n"); }

    | expression ';'            {   printf("value of expression: %d\n", $1); $$=$1;}

    | VAR '=' expression ';' {
                            data[$1] = $3;
                            printf("Value of the variable: %d\t\n",$3);
                            $$=$3;
                        }

    | FOR '(' NUM '<' NUM ')' START statement END {
                                    int i;
                                    for(i=$3 ; i<$5 ; i++) {printf("value of the loop: %d expression value: %d\n", i,$8);}

                               }
    | SWITCH '(' VAR ')' START B  END

    | IF '(' expression ')' START expression ';' END %prec IFX {
                                if($3){
                                    printf("\nvalue of expression in IF: %d\n",$6);
                                }
                                else{
                                    printf("condition value zero in IF block\n");
                                }
                            }

    | IF '(' expression ')' START expression ';' END ELSE START expression ';' END {
```

```
                                if($3){
                                    printf("value of expression in IF: %d\n",$6);
                                }
                                else{
                                    printf("value of expression in ELSE: %d\n",$11);
                                }
                            }
      | PF '(' expression ')' ';' {printf("Print Expression %d\n",$3);}
      ;

B     : C
      | C D
      ;
C     : C '+' C
      | CASE NUM ':' expression ';' BREAK ';' {}
      ;
D     : DEFAULT ':' expression ';' BREAK ';' {}

declaration : TYPE ID1
             ;


TYPE : INT
     | FLOAT
     | CHAR
     ;



ID1  : ID1 ',' VAR
     |VAR
     ;

expression: NUM                 { $$ = $1;  }

      | VAR                     { $$ = data[$1]; }

      | expression '+' expression { $$ = $1 + $3; }

      | expression '-' expression { $$ = $1 - $3; }

      | expression '*' expression { $$ = $1 * $3; }

      | expression '/' expression { if($3){
                                        $$ = $1 / $3;
                                    }
                                    else{
                                        $$ = 0;
                                        printf("\ndivision by zero\t");
                                    }
                                }
      | expression '%' expression { if($3){
                                        $$ = $1 % $3;
                                    }
                                    else{
                                        $$ = 0;
                                        printf("\nMOD by zero\t");
                                    }
                                }
      | expression '^' expression { $$ = pow($1 , $3);}
      | expression '<' expression { $$ = $1 < $3; }
```

```

    | expression '>' expression { $$ = $1 > $3; }

    | '(' expression ')'        { $$ = $2;  }
    | SIN expression            {printf("Value of Sin(%d) is %lf\n",$2,sin($2*3.1416/180)); $$=sin($2*3.1416/180);}

    | COS expression            {printf("Value of Cos(%d) is %lf\n",$2,cos($2*3.1416/180)); $$=cos($2*3.1416/180);}

    | TAN expression            {printf("Value of Tan(%d) is %lf\n",$2,tan($2*3.1416/180)); $$=tan($2*3.1416/180);}

    | LOG10 expression          {printf("Value of Log10(%d) is %lf\n",$2,(log($2*1.0)/log(10.0))); $$=(log($2*1.0)/log(10.0));}
    | LOG expression            {printf("Value of Log(%d) is %lf\n",$2,(log($2))); $$=(log($2));}
    |BIGMOD '(' expression ',' expression ',' expression ')' {
            long long res = 1;
            long long x = $3;
            long long p= $5;
            long long m=$7;
            while ( p )
            {
                if (p%2== 1) //p is odd
                {
                    res = ( res * x ) % m;
                }
                x = ( x * x ) % m;
                p = p / 2;
            }
            printf("\nBigmod of %d ^ %d MOD %d is = %lld\n",$3,$5,$7,res);
            $$=res;
        }
    ;
%%


yyerror(char *s){
    printf( "%s\n", s);
}

```