

```

1  %{
2      #include<stdio.h>
3      #include<stdlib.h>
4      #include<math.h>
5      int data[60];
6  %}
7
8  /* bison declarations */
9
10 %token NUM VAR IF ELSE MAIN INT FLOAT CHAR START END SWITCH CASE
    DEFAULT BREAK FOR PF SIN COS TAN LOG BIGMOD LOG10
11 %nonassoc IFX
12 %nonassoc ELSE
13 %nonassoc SWITCH
14 %nonassoc CASE
15 %nonassoc DEFAULT
16 %left '<' '>'
17 %left '+' '-'
18 %left '*' '/'
19
20 /* Grammar rules and actions follow. */
21
22 %%
23
24 program: MAIN ':' START cstatement END
25         ;
26
27 cstatement: /* NULL */
28             | cstatement statement
29             ;
30
31 statement: ';'
32             | declaration ';' { printf("Declaration\n"); }
33             | expression ';' { printf("value of expression:
34 %d\n", $1); $$=$1;}
35             | VAR '=' expression ';' {
36                                     data[$1] = $3;
37                                     printf("Value of the variable:
38 %d\t\n", $3);
39                                     $$=$3;
40                                 }
41             | FOR '(' NUM '<' NUM ')' START statement END {
42                                     int i;
43                                     for(i=$3 ; i<$5 ; i++) {
44                                         printf("value of the loop: %d expression value: %d\n", i,$8);
45                                     }
46             }
47             | SWITCH '(' VAR ')' START B END
48             | IF '(' expression ')' START expression ';' END %prec IFX {
49                                     if($3){
50                                         printf("\nvalue of
51 expression in IF: %d\n", $6);
52                                     }
53                                     else{
54                                         printf("condition value zero
55 in IF block\n");
56                                     }
57             }
58             | IF '(' expression ')' START expression ';' END ELSE START
    expression ';' END {

```

```

59                                     if($3){
60                                     printf("value of expression
in IF: %d\n", $6);
61                                     }
62                                     else{
63                                     printf("value of expression
in ELSE: %d\n", $11);
64                                     }
65                                     }
66         | PF '(' expression ')' ';' {printf("Print Expression %d\n",
$3);};
67         ;
68
69     B      : C
70           | C D
71           ;
72     C      : C '+' C
73           | CASE NUM ':' expression ';' BREAK ';' {}
74           ;
75     D      : DEFAULT ':' expression ';' BREAK ';' {}
76
77     declaration : TYPE ID1
78                 ;
79
80
81     TYPE      : INT
82               | FLOAT
83               | CHAR
84               ;
85
86
87
88     ID1      : ID1 ',' VAR
89               | VAR
90               ;
91
92     expression: NUM { $$ = $1; }
93
94           | VAR { $$ = data[$1]; }
95
96           | expression '+' expression { $$ = $1 + $3; }
97
98           | expression '-' expression { $$ = $1 - $3; }
99
100          | expression '*' expression { $$ = $1 * $3; }
101
102          | expression '/' expression { if($3){
103                                         $$ = $1 / $3;
104                                         }
105                                         else{
106                                         $$ = 0;
107                                         printf("\ndivision by
zero\t");
108                                         }
109          }
110          | expression '%' expression { if($3){
111                                         $$ = $1 % $3;
112                                         }
113                                         else{
114                                         $$ = 0;
115                                         printf("\nMOD by zero\t"
);
116                                         }
117          }
118          | expression '^' expression { $$ = pow($1, $3); }
119          | expression '<' expression { $$ = $1 < $3; }

```

```

120
121     | expression '>' expression { $$ = $1 > $3; }
122
123     | '(' expression ')'          { $$ = $2; }
124     | SIN expression              { printf("Value of Sin(%d) is
125 %lf\n", $2, sin($2*3.1416/180)); $$=sin($2*3.1416/180); }
126     | COS expression              { printf("Value of Cos(%d) is
127 %lf\n", $2, cos($2*3.1416/180)); $$=cos($2*3.1416/180); }
128     | TAN expression              { printf("Value of Tan(%d) is
129 %lf\n", $2, tan($2*3.1416/180)); $$=tan($2*3.1416/180); }
130     | LOG10 expression            { printf("Value of Log10(%d) is
131 %lf\n", $2, (log($2*1.0)/log(10.0))); $$=(log($2*1.0)/log(10.0)); }
132     | LOG expression              { printf("Value of Log(%d) is
133 %lf\n", $2, (log($2))); $$=(log($2)); }
134     | BIGMOD '(' expression ',' expression ')' {
135         long long res = 1;
136         long long x = $3;
137         long long p = $5;
138         long long m = $7;
139         while ( p )
140         {
141             if (p%2== 1) //p is odd
142             {
143                 res = ( res * x ) % m;
144             }
145             x = ( x * x ) % m;
146             p = p / 2;
147         }
148         printf("\nBigmod of %d ^ %d MOD %d is = %lld\n", $3,
149 $5, $7, res);
150         $$=res;
151     }
152
153 yyerror(char *s){
154     printf( "%s\n", s);
155 }
156

```