COURSE NO: CSE 3212

Project Name: Compiler design using flex and Bison

SUBMITTED BY

MD MOSHARAF HOSSAN

ROLL:1307053

INPUT FILE :

```
main()
{

	integer x eq 3 #
	integer y eq 4 #
	integer z eq 7 #

	z eq x * 5 + y * 3 #
	if(2<3) x eq 6 #
	if (1<2)  z eq z + 2 #
	else  z eq 10 #


switch ( z) {

		case 5 :
		  y eq y + 5 #
		case 9  :
		  y eq y + 7 #
		case 10 :
		 y eq y + 10 #
		default :
		  y eq y + 4 #
	}

	integer m eq 10 #
}
```

FLEX FILE :

```
%{
	#include<stdio.h>
	#include "main.tab.h"
```

```
        #include<stdlib.h>
        extern int yylval;
%}



%%

[0-9]+  {
                        yylval = atoi(yytext);
                        return NUM;
                }

[a-z]   {
                        yylval = *yytext - 'a';
                        return  VAR;
                }

"if"    {
                        return IF;
                }
"else"  {
                        return ELSE;
                }


[<>,(){}:]      {
                                yylval = yytext[0];
                                return *yytext;
                        }


"#"             { return ';'            ;}
"/"             { return '/'            ;}
"-"             { return '-'            ;}
"*"             { return '*'            ;}
"+"             { return '+'            ;}
"eq"    { return '='            ;}

"main"    { return(VOIDMAIN)    ;}

"print"         { return PRINT                  ;}

"integer"       { return(INT)           ;}

"float"         { return(FLOAT)                 ;}

"character"     { return(CHAR)                  ;}
```

```
"case"          { return CASE           ;}

"default"       {return DEFAULT     ;}

"switch"        { return SWITCH     ;}

[ \t\n]* ;

.       {

                yyerror("Unknown Character.\n");
        }
%%

main(){
        yyin = freopen("in.txt","r",stdin);
        //yyout = freopen("out.txt","w",stdout);
        yyparse();
}
```

BISON FILE :

```
/* C Declarations */

%{
        #include<stdio.h>
```

```
	#include <math.h>
	#define YYSTYPE int
	int varinfo1[100],varinfo2[100],varinfo3[100],opara[100] ;
	int p = 0 ,s= 0;

	int symbolara[26];
	int freq[26];
	int if_flag = 1, if_else_flag = 1, check = 1;
	int value ;

	int f1 = 0 ;
		int casevalue[100] ;
	int casestatement[100];



	int switchfunc(int i)
	{

			if(opara[i]==1)
			symbolara[varinfo1[i]] = symbolara[varinfo2[i]]
+varinfo3[i];

			if (opara[i]==2)
			symbolara[varinfo1[i]] = symbolara[varinfo2[i]]
-varinfo3[i];



			if (opara[i]==3)
			symbolara[varinfo1[i]] = symbolara[varinfo2[i]]
*varinfo3[i];

			if (opara[i]==4)
			{
				symbolara[varinfo1[i]] = symbolara[varinfo2[i]] /
varinfo3[i];
			}
```

```
                return symbolara[varinfo1[i]] ;
        }




%}

/* bison declarations */

%token NUM VAR IF ELSE VOIDMAIN INT FLOAT CHAR ID PRINT
LOOP  CASE DEFAULT SWITCH

%nonassoc IFX

%nonassoc ELSE

%left '<' '>'

%left '+' '-'

%left '*' '/'

/* Grammar rules and actions follow.  */

%%

program: VOIDMAIN '(' ')' '{' bstatement '}' //{printf("void main function");}
        ;




bstatement: /* empty */                    //{printf("start\n");}
```

```
          | bstatement statement                //{printf("b s \n");}
          ;




statement: ';'                                  //{printf("sem\n");}

          | declaration ';'               //{printf("d sem\n");}

          | expression ';'                {
                                                /*if(check == 0)
                                                {
                                                        printf("value of expression:
%d\n", $1);
                                                }*/
                                          }
;

          |  SWITCH '(' expression ')'  '{'  caseinstructions  '}'        {

                                                        value = $3 ;

                                                        int v =0 ;
                                                        int f2 = 1 ;



                                                        for ( v=0;v<p;v++)
                                                        {

                                                                if (value == casevalue[v] )
                                                                {

                                                                        printf("result of
evaluation is :  %d\n",switchfunc(v) );
```

```
                                                f2 =0 ;

                                        }

                                }

                                    if (f2==1) {

                                        printf("default value is :
%d\n",switchfunc(p));

                                    }
                                }
;


    | IF '(' expression ')' statement %prec IFX  {
                                    if($3)
                                    {
                                        //printf("\nonly if true and
value: %d",$3);
                                        printf("\nvalue of
expression in if: %d\n",$5);
                                        //if_else_flag = 0;
                                        if_flag = 1;
                                        check = 1;
                                    }

                                    else
                                    {
                                        if(if_flag == 1)
                                        {
                                            printf("condition
value zero in IF block\n");
                                            if_flag = 0;
```

```
                                                if_else_flag = 0;
                                                check = 1;
                                            }
                                        }

                                    }



          | IF '(' expression ')' statement ELSE statement {
                                            if($3  )
                                            {
                                                if_flag = 0;
                                                if_else_flag = 0;
                                                //printf("\nonly else if
true and value: %d",$3);
                                                printf("\nvalue of
expression in if: %d\n",$5);
                                                check = 1;
                                            }
                                            else
                                            {
                                                if(if_else_flag == 1)
                                                {
                                                    check = 1;
                                                    if_flag = 0;
                                                    if_else_flag = 0;
                                                    //printf("\nonly
else else true");
                                                    printf("\nvalue
of expression in else: %d\n",$7);
                                                }
                                            }
                                        }

;
```

```
        | '{' statement_list '}'   { $$ = $2; }
        ;




dowork :    VAR '=' VAR '+' NUM ';'  {
                                      varinfo1[s] = $1         ;
                                      varinfo2[s] = $3         ;
                                      varinfo3[s] = $5;
                                      opara[s] = 1    ;
                                      s++ ;
                                }


        | VAR '=' VAR '-' NUM ';' {

                                      varinfo1[s] = $1         ;
                                      varinfo2[s] = $3         ;
                                      varinfo3[s] = $5;
                                      opara[s] = 2    ;
                                      s++ ;

                                }


        | VAR '=' VAR '*' NUM  ';'{

                                      varinfo1[s] = $1         ;
                                      varinfo2[s] = $3         ;
                                      varinfo3[s] = $5;
                                      opara[s] = 3    ;
```

```
                                        s++ ;
                                  }


                | VAR '=' VAR '/' NUM ';'{

                                        varinfo1[s] = $1         ;
                                        varinfo2[s] = $3         ;
                                        varinfo3[s] = $5;
                                        opara[s] = 4    ;
                                        s++ ;
                                  }

;







declaration: TYPE ID1
       ;



TYPE : INT                                                  //{printf("int\n");}
   | FLOAT                                                  //{printf("flt\n");}
   | CHAR                                                   //{printf("char\n");}
   ;

ID1:  ID1 ',' expression {
                if (freq[$3]==0) freq[$3]++;
                else printf("this was declared before ");
                  }

      | expression   {
             if (freq[$1]==0) freq[$1]++;
```

```
                else printf("this was declared before ");

                  }
    ;


statement_list: statement_list statement              //{printf("inside if or else
up\n");}

                |statement
                            { $$ = $1;   // printf("inside if or else down\n");
                             }
                ;


expression:      NUM                               { $$ = $1; }

                | VAR                          { $$ = symbolara[$1]; //printf("e:var
%d \n",$1);
                                         }

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

                | 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   { $$ = $1 < $3  ;}

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

                | '(' expression ')'        { $$ = $2   ;}
                ;


caseinstructions : steps caseinstructions                   {
                                                        }
        | DEFAULT ':' dowork {

                        //printf("in default\n");

        }
        ;
steps : CASE NUM ':'  dowork  {
                                casevalue[p] = $2 ;

                                p++ ;
                        }
                ;

%%

int yywrap()
{
```

```
return 1;
}



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

OUTPUT FILE :

```
var = exp Value of the variable: 3 hash
var = exp Value of the variable: 4 hash
var = exp Value of the variable: 7 hash
var = exp Value of the variable: 27 hash
var = exp Value of the variable: 6 hash

value of expression in if: 6
var = exp Value of the variable: 29 hash
var = exp Value of the variable: 10 hash

value of expression in if: 29
result of evaluation is :  14
var = exp Value of the variable: 10 hash
```