PB 38 Vedant Karle B3
CHIT P
SSC PRAC EXAM

3] Implement a parser for validation of If-Then anbd While control structure using lex and yacc

->

.l file

```
%{
#include "y.tab.h"
extern int yyerror(char *str);
extern int yyparse();
%}

%%

"while" return WH;
"if" return IF;
"then" return THN;
"(" return OP;
")" return CP;
"{" return OCB;
"}" return CCB;
"<" |
">" |
"<=" |
">=" |
"==" |
"!=" return CMP;
"+" |
"-" |
"*" |
"/" return OPR;
"=" return ASG;
([a-zA-Z])("_"|[a-zA-Z0-9])* return ID;
[0-9]+ return NUM;
";" return SC;
"," return COMMA;

%%

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

```
%{
#include<stdio.h>
extern int yylex();
extern int yywrap();
extern int yyparse();
%}

%token WH IF THN OP CP OCB CCB CMP SC ASG ID NUM COMMA OPR

%%

start:    swh | mwh | ifth ;
swh: WH OP cmpn CP stmt {printf("VALID SINGLE WHILE STATEMENT\n");};
mwh: WH OP cmpn CP OCB stmt CCB {printf("VALID MULTIPLE WHILE STATEMENT\n");};
cmpn:  ID CMP ID | ID CMP NUM
ifth:    IF OP cmpn CP THN OCB stmt CCB                    {printf("VALID IF THEN
STATEMENT\n");};
stmt: ID ASG ID OPR ID SC | ID ASG ID OPR NUM SC | ID ASG NUM OPR ID SC | ID ASG
NUM OPR NUM SC | ID ASG ID SC | ID ASG NUM SC
       | start                             {printf("NESTED INSIDE A ");}
       ;

%%
int yyerror(char *str)
{
       printf("%s", str);
}

main()
{
       yyparse();
}
```

main()
{
    yyparse();
}

OUTPUT:

```
computer@c04l0117:~$ flex lexparser.l
computer@c04l0117:~$ byacc -d lexparser.y
computer@c04l0117:~$ gcc lex.yy.c y.tab.c
lexparser.y:27:1: warning: return type defaults to 'int' [-Wimplicit-int]
 main()
 ^
computer@c04l0117:~$ ./a.out
if(a=b)then{a=b+c};
syntax errorcomputer@c04l0117:~$ ./a.out
if(a==b)then{a=b+c}
syntax errorcomputer@c04l0117:~$ ./a.out
if (a==b) then {c=d+e;}
    VALID IF THEN STATEMENT

^C
computer@c04l0117:~$ ./a.out
while (a<b) v=d+x;
    VALID SINGLE WHILE STATEMENT

^C
computer@c04l0117:~$ ./a.out
while (a>b) {g=h-j;}
    VALID MULTIPLE WHILE STATEMENT
```

OUTPUT:

```
if(a=b)then{a=b+c};
syntax errorcomputer@c04l0117:~$ ./a.out
if(a==b)then{a=b+c}
syntax errorcomputer@c04l0117:~$ ./a.out
if (a==b) then {c=d+e;}
    VALID IF THEN STATEMENT

^C
computer@c04l0117:~$ ./a.out
while (a<b) v=d+x;
    VALID SINGLE WHILE STATEMENT

^C
computer@c04l0117:~$ ./a.out
while (a>b) {g=h-j;}
    VALID MULTIPLE WHILE STATEMENT

^C
computer@c04l0117:~$ ./a.out
while (a>b) { while(c>d) z=k-l;}
    VALID SINGLE WHILE STATEMENT
NESTED INSIDE A VALID MULTIPLE WHILE STATEMENT
```