**Practical 7**

**Compiler Construction**

2CS701

**Mistry Unnat**

20BCE515



Department of Computer Science and Engineering

Institute of Technology

Nirma University

Ahmedabad

**Aim :**

#### **To implement grammar rules for control statements, and Loop control.**

# **lab\_7.y**

%{

#include<stdio.h> int yylex();

void yyerror(char \*msg); void yywrap();

int count\_for=0,count\_if=0,count\_while=0,count\_switch=0,count\_do\_while=0; extern int line;

%}

%token FOR IF ELSE WHILE DO SWITCH CASE DEFAULT BREAK REOP DT AROP1 ID INT\_CONST BOOL\_CONST CHAR\_CONST STR\_CONST

%%

SS : SS S

| S

;

S : FOR\_STMT

| IF\_STMT

| WHILE\_STMT

| DO\_WHILE\_STMT

| SWITCH\_STMT

| BLOCK

| ';'

;

FOR\_STMT : FOR '(' ASMT ';' COND ';' INCR ')' S {count\_for++;} ; WHILE\_STMT : WHILE '(' COND ')' S {count\_while++;};

DO\_WHILE\_STMT : DO '{' S '}' WHILE '(' COND ')' ';' {count\_do\_while++;}; IF\_STMT : IF '(' COND ')' S {count\_if++;}

| IF '(' COND ')' S ELSE S {count\_if++;}

;

SWITCH\_STMT : SWITCH '(' ID ')' '{' CASE\_STMTS DEFAULT\_STMT '}' {count\_switch++;}

;

CASE\_STMTS : CASE\_STMT CASE\_STMTS

| CASE\_STMT

;

CASE\_STMT : CASE CONST ':' SS BREAK ';'

;

DEFAULT\_STMT : DEFAULT ':' SS

;

BLOCK : '{' SS '}'

;

ASMT : DT ID '=' INT\_CONST

| DT ID '=' ID

| ID '=' INT\_CONST

;

COND : ID REOP ID

| ID REOP CONST

| CONST REOP ID

| CONST REOP CONST

;

INCR : ID AROP1

;

CONST : INT\_CONST

| CHAR\_CONST

| BOOL\_CONST

| STR\_CONST

;

%%

int main()

{

yyparse(); return 0;

}

void yyerror(char \*msg)

{

printf("\nError Message :- %s statements are not following grammer rules at %d",msg,line);

}

void yywrap()

{

printf("\nNumber of for loop is %d",count\_for); printf("\nNumber of while loop is %d",count\_while); printf("\nNumber of do while loop is %d",count\_do\_while); printf("\nNumber of if statements is %d",count\_if); printf("\nNumber of switch statements is %d",count\_switch);

}

# **lab\_7.l**

%{

#include "y.tab.h" int line=1;

%}

%%

"\n" {printf("");line++;}

"\t"|" " {printf("");}

"++"|"--" {return AROP1;} "int"|"float"|"double"|"char" {return DT;} "=="|"!="|"<"|">"|"<="|">=" {return REOP;}

[();{}:=] {return yytext[0];} [0-9]+ {return INT\_CONST;}

['].['] {return CHAR\_CONST;}

"true"|"false" {return BOOL\_CONST;} ["].\*["] {return STR\_CONST;}

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

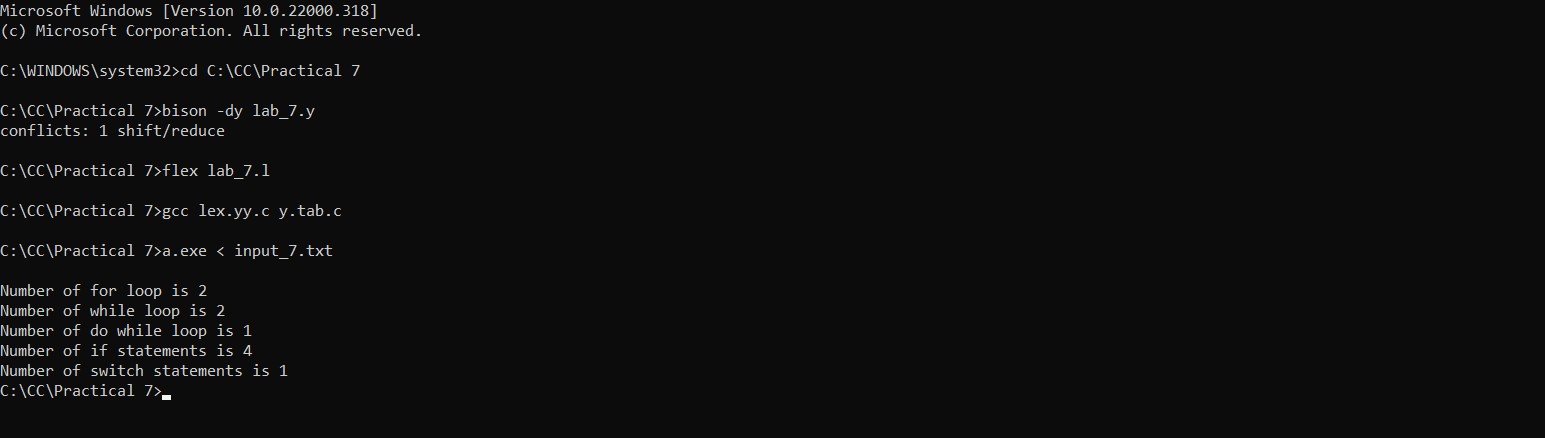
"while" {return WHILE;} "do" {return DO;} "switch" {return SWITCH;} "case" {return CASE;}

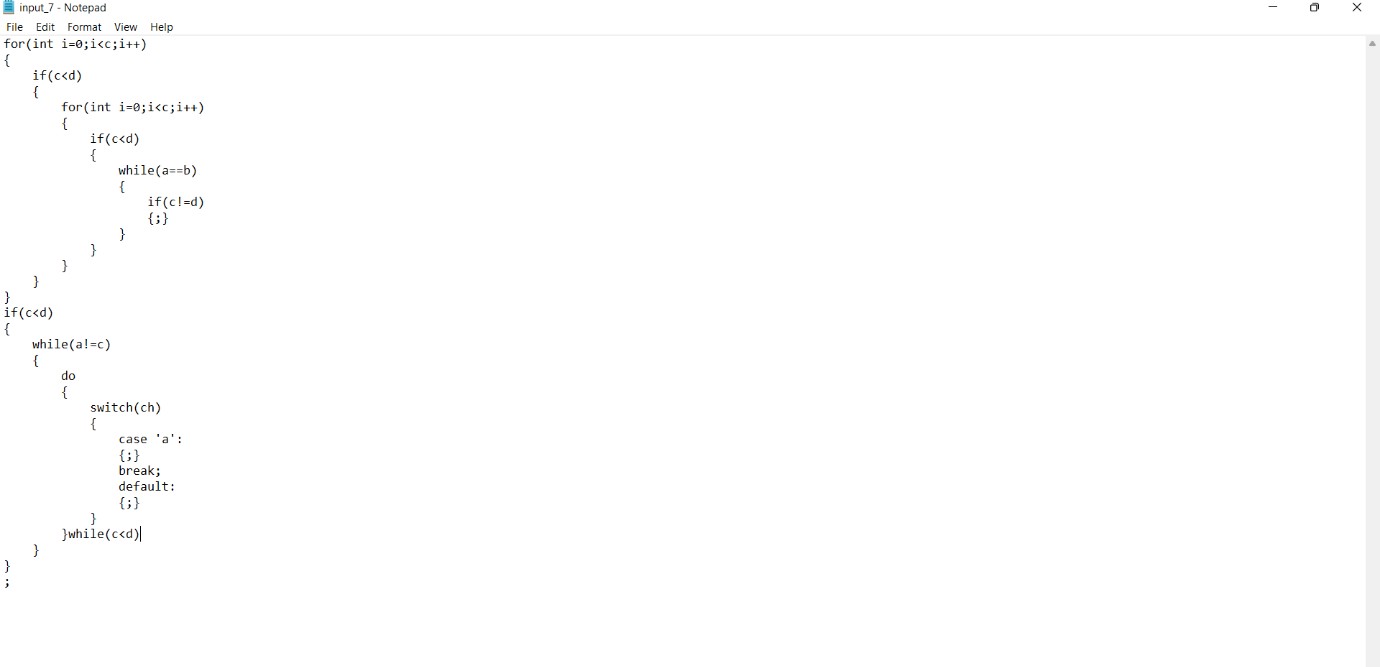
"default" {return DEFAULT;} "break" {return BREAK;}

([\_a-zA-Z](0-9)\*)+ {return ID;}

%%

# **Screenshots**







END