1. Hello world program

**hello.l ( Lex code )**

%{

#include "hello.tab.h"

int yyerror(char \*errormsg);

%}

%%

("hi"|"oi")"\n" { return HI; }

("adios"|"bye")"\n" { return BYE; }

. { yyerror("Unknown char"); }

%%

int main(void){

yyparse();

return 0;}

int yywrap(void){

return 0;}

int yyerror(char \*errormsg){

fprintf(stderr, "%s\n", errormsg);

exit(1);}

**hello.y ( Yacc Code )**

%{

#include <stdio.h>

#include <stdlib.h>

int yylex(void);

int yyerror(const char \*s);

%}

%token HI BYE

%%

program:

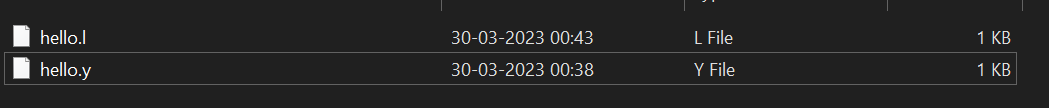
hi bye

;

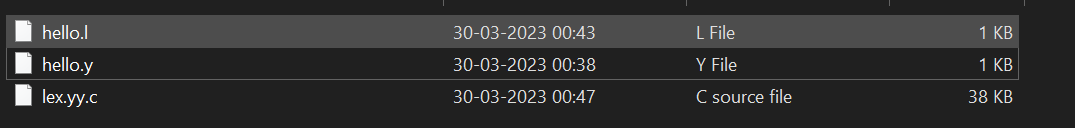
hi: HI { printf("Hello World\n"); }

;

bye: BYE { printf("Bye World\n"); exit(0); }

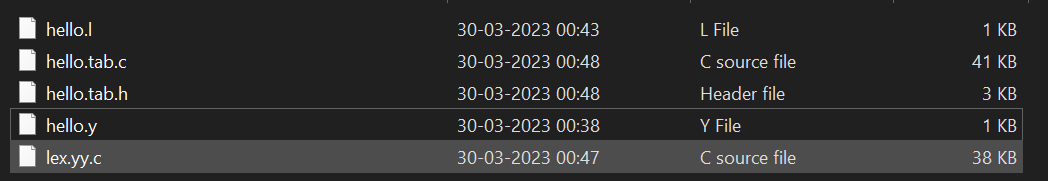


Cmd: flex hello.l generates lex.yy.c



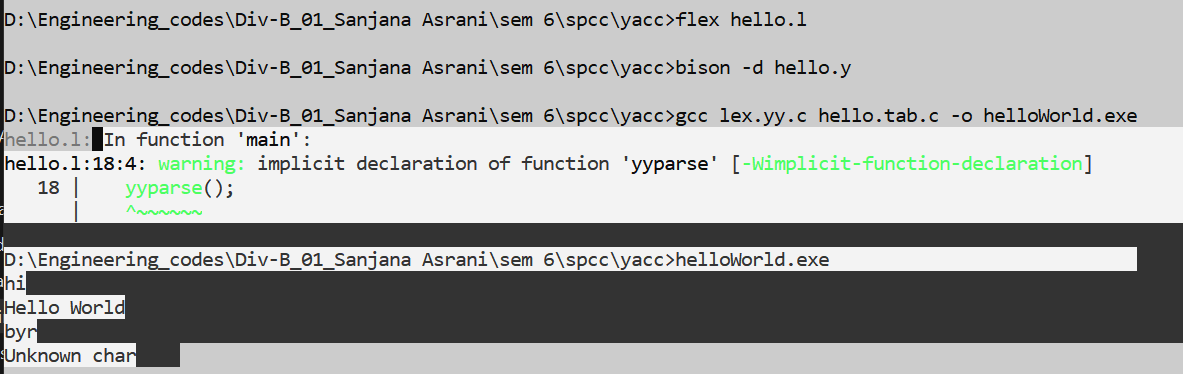
Cmd: bison -d hello.y generates hello.tab.h & hello.tab.c

(filenameOfDotYFile.tab.h)

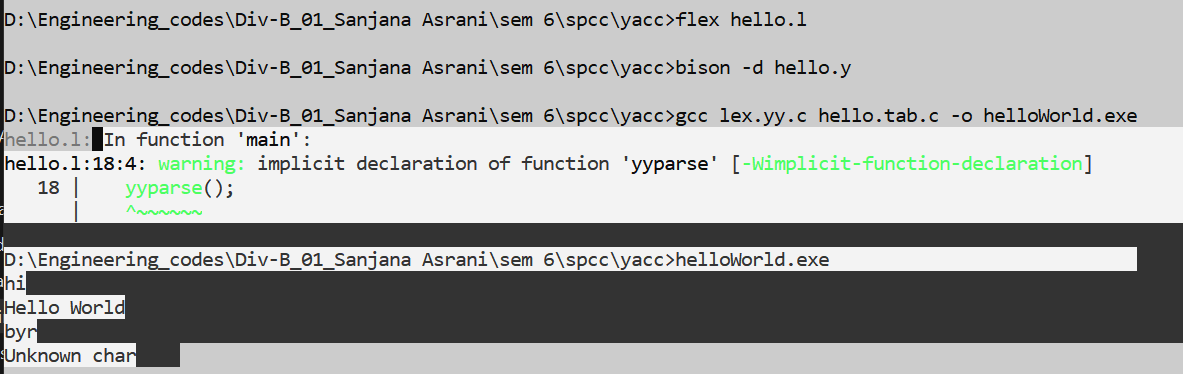


Cmd: gcc lex.yy.c hello.tab.c -o helloWorld.exe generates helloWorld.exe application file.

(if cmd is gcc lex.yy.c hello.tab.c , to run it *a.exe* can be done)







….…………………………………..cmds:

flex first.l

bison -d sanj.y

gcc lex.yy.c sanj.tab.c -o firstPgm.exe

1. Simple calculator

Yacc Code:

%{

#include<stdio.h>

int flag=0;

%}

%token NUMBER

%left '+' '-'

%left '\*' '/' '%'

%left '(' ')'

%%

ArithmeticExpression: E{

printf("\nResult=%d\n",$$);

return 0;

};

E:E'+'E {$$=$1+$3;}

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

|E'\*'E {$$=$1\*$3;}

|E'/'E {$$=$1/$3;}

|E'%'E {$$=$1%$3;}

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

| NUMBER {$$=$1;}

;

%%

int main()

{

printf("\nEnter Any Arithmetic Expression which can have operations

Addition, Subtraction, Multiplication, Divison, Modulus and Round

brackets:\n");

yyparse();

if(flag==0)

printf("\nEntered arithmetic expression is Valid\n\n");

}

int yyerror(void)

{

printf("error");

flag=1;

}

**Lex Code:**

%{

#include<stdio.h>

#include "calculator\_yacc.tab.h"

extern int yylval;

%}

%%

[0-9]+ {

yylval=atoi(yytext);

return NUMBER;}

[\t] ;

[\n] return 0;

. return yytext[0];

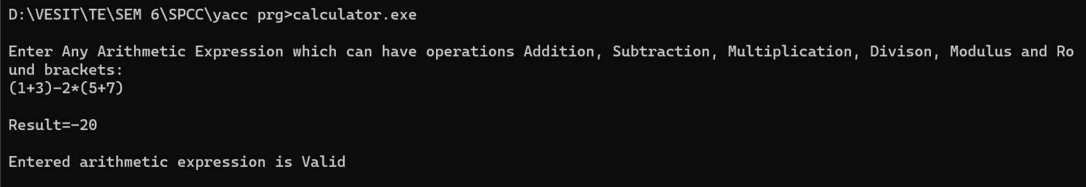
%%

int yywrap()

{

return 1;

}



1. **Recognize nested IF stmt and display levels**

**Yacc Code:**

%{

#include<stdio.h>

#include<stdlib.h>

int count=0;

int level=0;

%}

%token IF RELOP S NUMBER ID NL

%%

stmt: if\_stmt NL {printf("No. of nested if statements=%d\n",count);

exit(0);}

;

if\_stmt : IF'('cond')''{'if\_stmt'}'

{count++;level++;printf("Level=%d\n",count);}

|S

;

cond: x RELOP x

;

x:ID | NUMBER

;

%%

int yyerror(char \*msg) {

printf("the statement is invalid\n");

exit(0); }

main() {

printf("enter the statement\n");

yyparse();}

**Lex Code:**

%option noyywrap

%{

#include "ifelse.tab.h"

%}

%%

"if" {return IF;}

[sS][0-9]\* {return S;}

"<"|">"|"=="|"<="|">="|"!=" {return RELOP;}

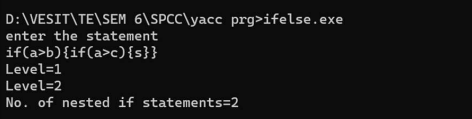
[0-9]+ {return NUMBER;}

[a-z][a-zA-Z0-9\_]\* {return ID;}

\n {return NL;}

. {return yytext[0];}

%%



1. **recognize a valid variable**

**Yacc Code:**

%{

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

%}

%union {

char \*strval;

}

%token IDENTIFIER

%%

program**:**

| program line

;

Line:

IDENTIFIER '\n' { printf("Valid variable: %s\n", $1);free($1);

printf("Enter variable name: ");}

| '\n'

;

%%

int main(int argc, char \*argv[]) {

printf("Enter variable name: ");

yyparse();

return 0;}

int yyerror(char \*s) {

printf("Error: %s\n", s);

return 0;

}

**Lex Code:**

%{

#include "validVariable.tab.h"

%}

%%

[a-zA-Z\_][a-zA-Z0-9\_]\* { yylval.strval = strdup(yytext); return IDENTIFIER;

}

.|\n

{ return yytext[0]; }

%%

int yywrap() {

return 1;

}

