Laboratory 8

Language specification:

%{

#include <stdio.h>

#include <string.h>

int lines = 0;

int correct=1;

int badLine=0;

%}

%option noyywrap

%option caseless

NUMBER [+-]?[1-9][0-9]\*|0

STRING \"[a-zA-Z0-9]\*\"

CONST {NUMBER}|{STRING}

ID [a-zA-Z]+[a-zA-Z0-9\_]\*

%%

start {printf("Reserved word: %s\n", yytext);}

stop {printf("Reserved word: %s\n", yytext);}

get {printf("Reserved word: %s\n", yytext);}

give {printf("Reserved word: %s\n", yytext);}

int {printf("Reserved word: %s\n", yytext);}

string {printf("Reserved word: %s\n", yytext);}

bool {printf("Reserved word: %s\n", yytext);}

if {printf("Reserved word: %s\n", yytext);}

else {printf("Reserved word: %s\n", yytext);}

while {printf("Reserved word: %s\n", yytext);}

true {printf("Reserved word: %s\n", yytext);}

false {printf("Reserved word: %s\n", yytext);}

"main" {printf("Reserved word: %s\n", yytext);}

"<" {printf("Separator: %s\n", yytext);}

">" {printf("Separator: %s\n", yytext);}

"(" {printf("Separator: %s\n", yytext);}

")" {printf("Separator: %s\n", yytext);}

"[" {printf("Separator: %s\n", yytext);}

"]" {printf("Separator: %s\n", yytext);}

";" {printf("Separator: %s\n", yytext);}

" " {printf("Separator: %s\n", yytext);}

Plus {printf( "Operator: %s\n", yytext );}

Minus {printf( "Operator: %s\n", yytext );}

Multiply {printf( "Operator: %s\n", yytext );}

Div {printf( "Operator: %s\n", yytext );}

Mod {printf( "Operator: %s\n", yytext );}

Equals {printf( "Operator: %s\n", yytext );}

Greater {printf( "Operator: %s\n", yytext );}

GreaterOrEqual {printf( "Operator: %s\n", yytext );}

Smaller {printf( "Operator: %s\n", yytext );}

SmallerOrEqual {printf( "Operator: %s\n", yytext );}

EqualEqual {printf( "Operator: %s\n", yytext );}

Different {printf( "Operator: %s\n", yytext );}

And {printf( "Operator: %s\n", yytext );}

Or {printf( "Operator: %s\n", yytext );}

Not {printf( "Operator: %s\n", yytext );}

{ID} {printf( "Identifier: %s\n", yytext );}

{CONST} {printf( "Constant: %s\n", yytext );}

[ \t]+ {}

[\n]+ {lines++;}

[0-9][0-9]\*{ID} {correct=0; badLine= lines; printf("Incorect:%s\n",yytext);}

. {correct=0; badLine= lines; printf("Incorect:%s\n",yytext);}

%%

void main(int argc, char\*\* argv){

if (argc > 1){

FILE \*file;

file = fopen(argv[1],"r");

if(!file){

fprintf(stderr,"Could not open: %s\n",argv[1]);

exit(1);

}

yyin = file;

}

yylex();

if(correct==1){

printf("Correct program");

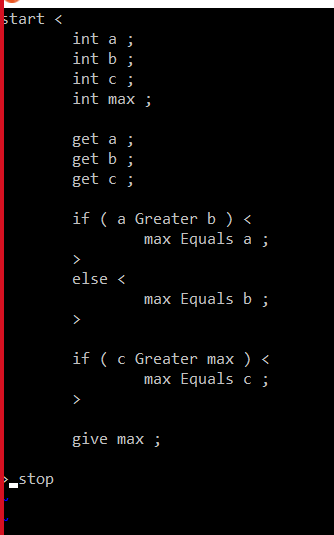
}

else{

printf("Incorrect at line %d\n",badLine);

}

}

Problem 1:

OUTPUT:

Reserved word: start

Separator:

Separator: <

Reserved word: int

Separator:

Identifier: a

Separator:

Separator: ;

Reserved word: int

Separator:

Identifier: b

Separator:

Separator: ;

Reserved word: int

Separator:

Identifier: c

Separator:

Separator: ;

Reserved word: int

Separator:

Identifier: max

Separator:

Separator: ;

Reserved word: get

Separator:

Identifier: a

Separator:

Separator: ;

Reserved word: get

Separator:

Identifier: b

Separator:

Separator: ;

Reserved word: get

Separator:

Identifier: c

Separator:

Separator: ;

Reserved word: if

Separator:

Separator: (

Separator:

Identifier: a

Separator:

Operator: Greater

Separator:

Identifier: b

Separator:

Separator: )

Separator:

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: a

Separator:

Separator: ;

Separator: >

Reserved word: else

Separator:

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: b

Separator:

Separator: ;

Separator: >

Reserved word: if

Separator:

Separator: (

Separator:

Identifier: c

Separator:

Operator: Greater

Separator:

Identifier: max

Separator:

Separator: )

Separator:

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: c

Separator:

Separator: ;

Separator: >

Reserved word: give

Separator:

Identifier: max

Separator:

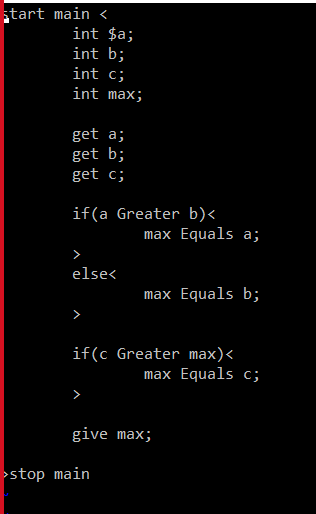
Separator: ;

Separator: >

Separator:

Reserved word: stop

Correct program

Problem with error:

Reserved word: start

Separator:

Reserved word: main

Separator:

Separator: <

Reserved word: int

Separator:

Incorect:$

Identifier: a

Separator: ;

Reserved word: int

Separator:

Identifier: b

Separator: ;

Reserved word: int

Separator:

Identifier: c

Separator: ;

Reserved word: int

Separator:

Identifier: max

Separator: ;

Reserved word: get

Separator:

Identifier: a

Separator: ;

Reserved word: get

Separator:

Identifier: b

Separator: ;

Reserved word: get

Separator:

Identifier: c

Separator: ;

Reserved word: if

Separator: (

Identifier: a

Separator:

Operator: Greater

Separator:

Identifier: b

Separator: )

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: a

Separator: ;

Separator: >

Reserved word: else

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: b

Separator: ;

Separator: >

Reserved word: if

Separator: (

Identifier: c

Separator:

Operator: Greater

Separator:

Identifier: max

Separator: )

Separator: <

Identifier: max

Separator:

Operator: Equals

Separator:

Identifier: c

Separator: ;

Separator: >

Reserved word: give

Separator:

Identifier: max

Separator: ;

Separator: >

Reserved word: stop

Separator:

Reserved word: main

Incorrect at line 1