FILE STRUCTURE / CODE STRUCTURE

SRC/ this folder contains compiler.y ,compiler.l,out.c (this file will contain abstract syntax tree as comments and resultant c code also)
INCLUDE/ this folder contains absyntree.h, which has all the structures required for implementations of nodes,lists.
BIN/ has the final executable compiler, also the executable outputcode that is the output executable of generated c code.
MAKEFILE/ contains make file code , if you do make test it will run compiler on testcase.txt then output abstract syntax tree ,c code is also outputed,syntax tree is printed as comment and c code is printed in out.c file in src folder, make clean will clean Bin , build folders and remove unnecessary files.
COMPILER.L/ contains what tokens will be returned according to the grammar.
COMPILER.Y/
Starts with function declaration that are used for tree construction , printing , managing symbol table.
The creation of syntax tree happens bottom up in grammar, in int main() first yyparse is called then ,
First printing of abstract syntax tree is done, then c code is printed in file out.c.
Testcase.txt contains my own created testcase , you can see it for reference.
REPORT.PDF contains report of this project.

FEATURES

Array and function declaration of bool and int, variable declaration of bool and int , but only single dimensional arrays are allowed.

Function calls,For loops,While loops , if else statements are allowed

Also you can use the write function to print value of a expression,variable , or value at a array index.

ADDITIONAL FEATURES

Integer main and bool main supported .

If you write command write(" i am a string \n"); then " i am string " will be outputed on terminal and \n will give new line.

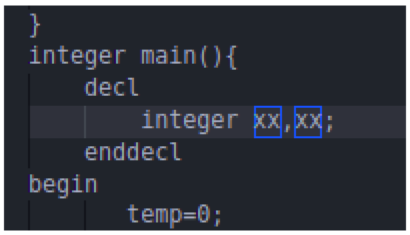
Recursive function calls are supported.

Nested if else is supported ., nested while loop, nested For loop are supported .

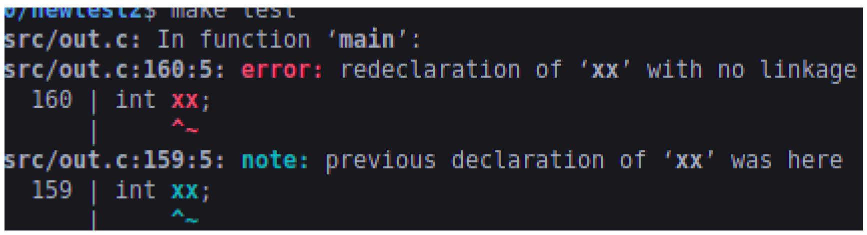
Boolean declarations are also supported.

ERROR DETECTION

I declared a variable twice and this is what error is found.

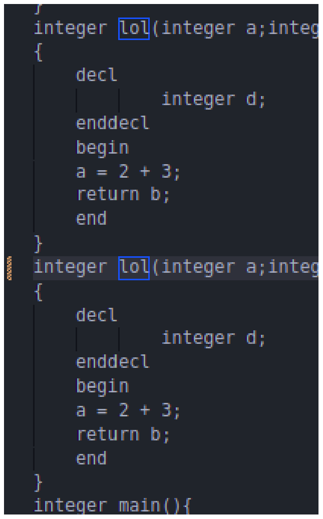


```
}
integer main(){
    decl
        integer xx,xx;
    enddecl
begin
      temp=0;
```

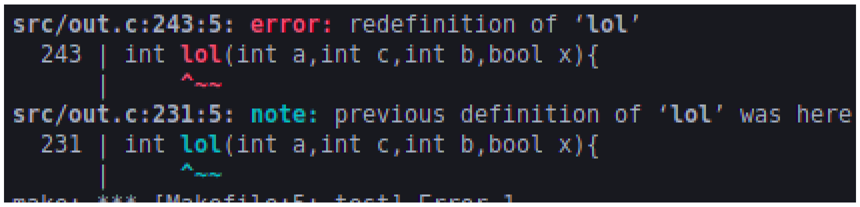


```
src/out.c: In function 'main':
src/out.c:160:5: error: redeclaration of 'xx' with no linkage
  160 | int xx;
      |     ^~
src/out.c:159:5: note: previous declaration of 'xx' was here
  159 | int xx;
      |     ^~
```

Then i declare a function twice

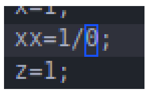


```
}
integer lol(integer a;integ
{
    decl
            integer d;
    enddecl
    begin
    a = 2 + 3;
    return b;
    end
}
integer lol(integer a;integ
{
    decl
            integer d;
    enddecl
    begin
    a = 2 + 3;
    return b;
    end
}
integer main(){
```

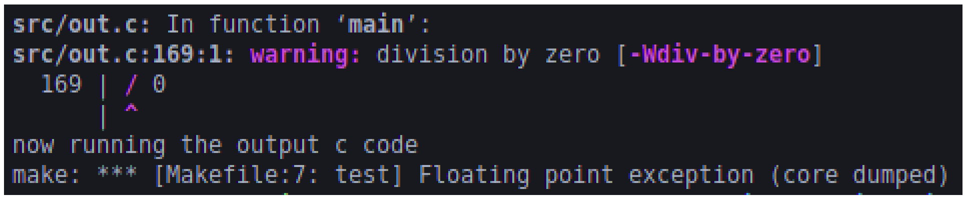


```
src/out.c:243:5: error: redefinition of ‘lol’
  243 | int lol(int a,int c,int b,bool x){
      |     ^~~
src/out.c:231:5: note: previous definition of ‘lol’ was here
  231 | int lol(int a,int c,int b,bool x){
      |     ^~~
make: *** [Makefile:5: test] Error 1
```

I do divide by zero



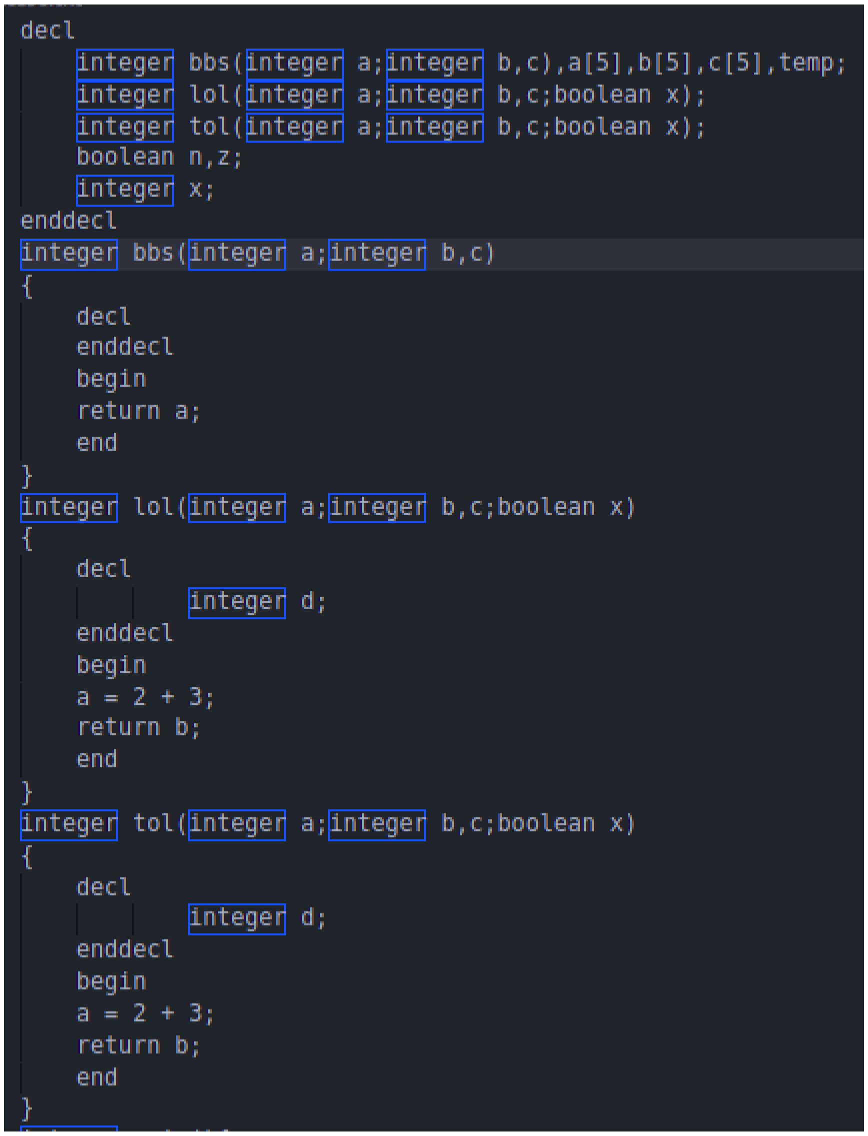
```
x=1;
xx=1/0;
z=1;
```



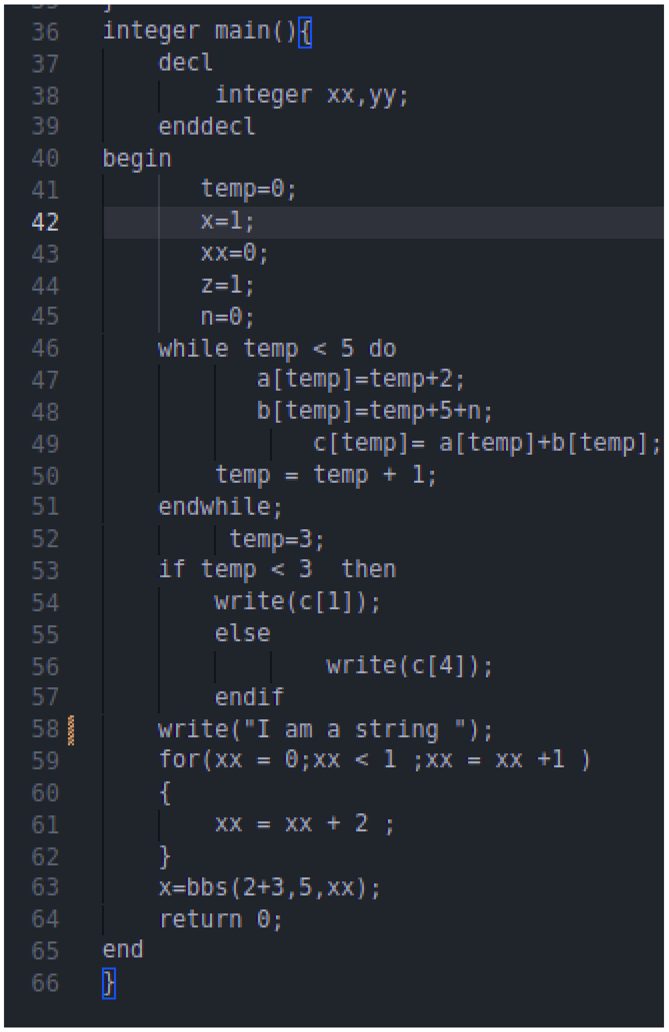
```
src/out.c: In function ‘main’:
src/out.c:169:1: warning: division by zero [-Wdiv-by-zero]
  169 | / 0
      | ^
now running the output c code
make: *** [Makefile:7: test] Floating point exception (core dumped)
```

It also handles semantic errors.

EXAMPLE CODE AND OUTPUT .



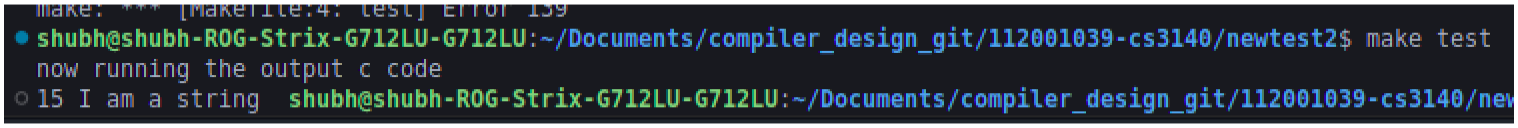
```
decl
    integer bbs(integer a;integer b,c),a[5],b[5],c[5],temp;
    integer lol(integer a;integer b,c;boolean x);
    integer tol(integer a;integer b,c;boolean x);
    boolean n,z;
    integer x;
enddecl
integer bbs(integer a;integer b,c)
{
    decl
    enddecl
    begin
    return a;
    end
}
integer lol(integer a;integer b,c;boolean x)
{
    decl
            integer d;
    enddecl
    begin
    a = 2 + 3;
    return b;
    end
}
integer tol(integer a;integer b,c;boolean x)
{
    decl
            integer d;
    enddecl
    begin
    a = 2 + 3;
    return b;
    end
}
```



```
integer main(){
    decl
        integer xx,yy;
    enddecl
begin
       temp=0;
       x=1;
       xx=0;
       z=1;
       n=0;
    while temp < 5 do
           a[temp]=temp+2;
           b[temp]=temp+5+n;
               c[temp]= a[temp]+b[temp];
        temp = temp + 1;
    endwhile;
         temp=3;
    if temp < 3  then
        write(c[1]);
        else
                write(c[4]);
        endif
    write("I am a string ");
    for(xx = 0;xx < 1 ;xx = xx +1 )
    {
        xx = xx + 2 ;
    }
    x=bbs(2+3,5,xx);
    return 0;
end
}
```

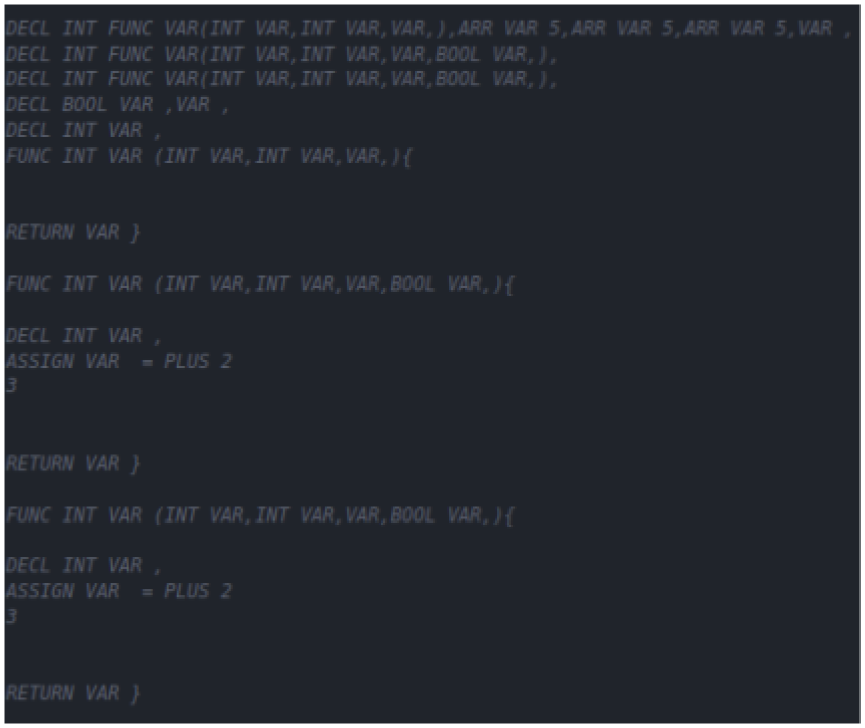
THE OUTPUT OF THE CODE

AS , YOU CAN SEE THE WRITE STATEMENTS ARE PRINTED ON TERMINAL.

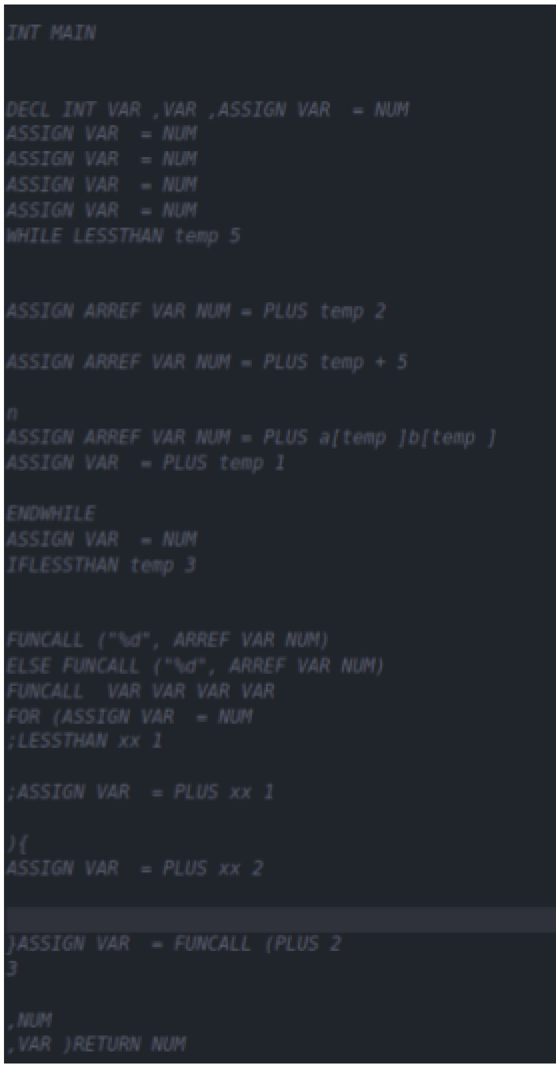


```
 make: *** [Makefile:4: test] Error 139
● shubh@shubh-ROG-Strix-G712LU-G712LU:~/Documents/compiler_design_git/112001039-cs3140/newtest2$ make test
 now running the output c code
○ 15 I am a string  shubh@shubh-ROG-Strix-G712LU-G712LU:~/Documents/compiler_design_git/112001039-cs3140/new
```

SYNTAX TREE PRINTED AS COMMENT IN OUT.C

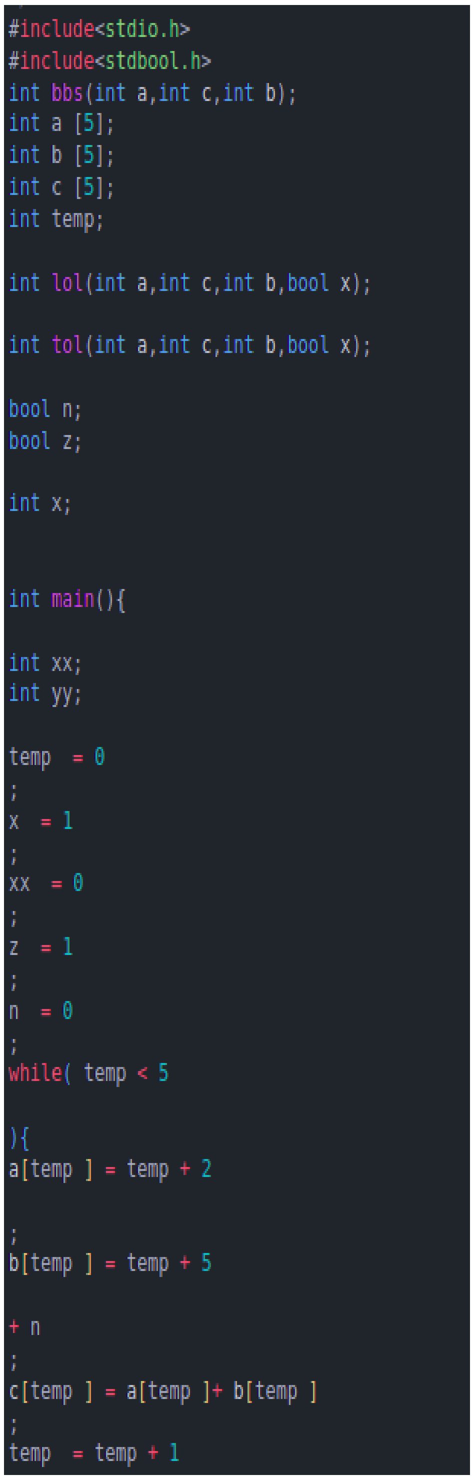


```
DECL INT FUNC VAR(INT VAR,INT VAR,VAR,),ARR VAR 5,ARR VAR 5,ARR VAR 5,VAR ,
DECL INT FUNC VAR(INT VAR,INT VAR,VAR,BOOL VAR,),
DECL INT FUNC VAR(INT VAR,INT VAR,VAR,BOOL VAR,),
DECL BOOL VAR ,VAR ,
DECL INT VAR ,
FUNC INT VAR (INT VAR,INT VAR,VAR,){


RETURN VAR }

FUNC INT VAR (INT VAR,INT VAR,VAR,BOOL VAR,){

DECL INT VAR ,
ASSIGN VAR  = PLUS 2
3


RETURN VAR }

FUNC INT VAR (INT VAR,INT VAR,VAR,BOOL VAR,){

DECL INT VAR ,
ASSIGN VAR  = PLUS 2
3


RETURN VAR }
```

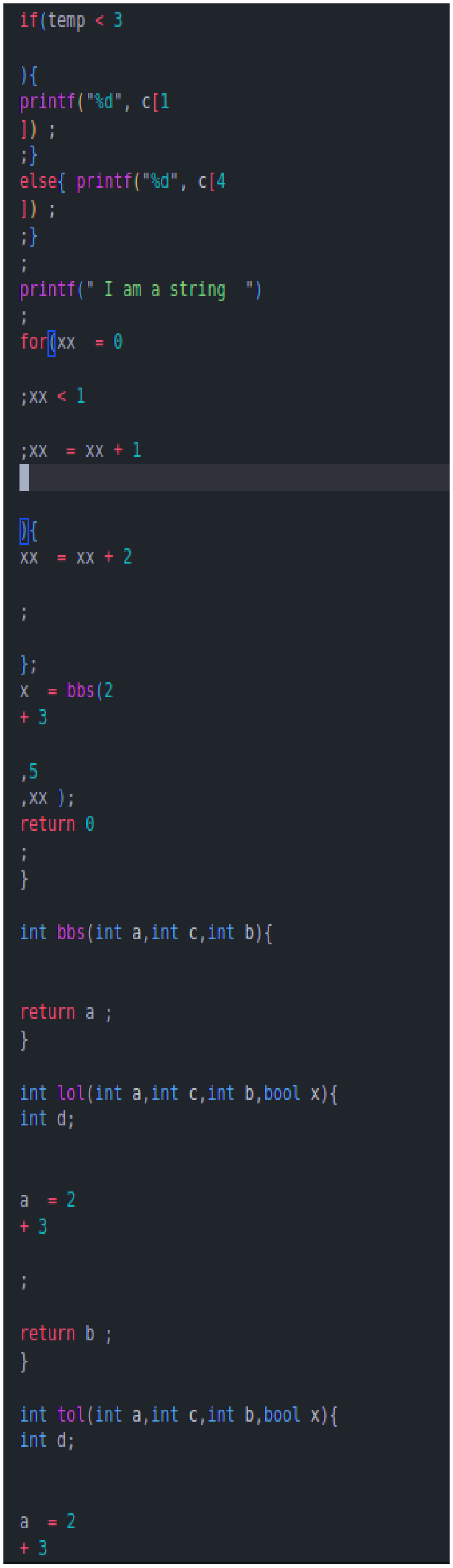


```
INT MAIN


DECL INT VAR ,VAR ,ASSIGN VAR  = NUM
ASSIGN VAR  = NUM
ASSIGN VAR  = NUM
ASSIGN VAR  = NUM
ASSIGN VAR  = NUM
WHILE LESSTHAN temp 5


ASSIGN ARREF VAR NUM = PLUS temp 2

ASSIGN ARREF VAR NUM = PLUS temp + 5

n
ASSIGN ARREF VAR NUM = PLUS a[temp ]b[temp ]
ASSIGN VAR  = PLUS temp 1

ENDWHILE
ASSIGN VAR  = NUM
IFLESSTHAN temp 3


FUNCALL ("%d", ARREF VAR NUM)
ELSE FUNCALL ("%d", ARREF VAR NUM)
FUNCALL  VAR VAR VAR VAR
FOR (ASSIGN VAR  = NUM
;LESSTHAN xx 1

;ASSIGN VAR  = PLUS xx 1

){
ASSIGN VAR  = PLUS xx 2


}ASSIGN VAR  = FUNCALL (PLUS 2
3

,NUM
,VAR )RETURN NUM
```

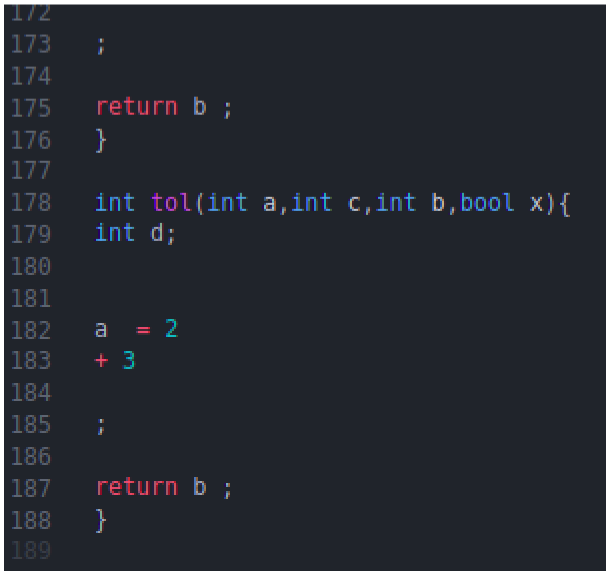
As we can see the input code had for loop while loop , function calls everything is executed and correct output is given in terminal , also equivalent c code is outputed in out.c file .



```
#include<stdio.h>
#include<stdbool.h>
int bbs(int a,int c,int b);
int a [5];
int b [5];
int c [5];
int temp;

int lol(int a,int c,int b,bool x);

int tol(int a,int c,int b,bool x);

bool n;
bool z;

int x;


int main(){

int xx;
int yy;

temp  = 0
;
x  = 1
;
xx  = 0
;
z  = 1
;
n  = 0
;
while( temp < 5

){
a[temp ] = temp + 2


;
b[temp ] = temp + 5

+ n
;
c[temp ] = a[temp ]+ b[temp ]
;
temp  = temp + 1
```



```
if(temp < 3

){
printf("%d", c[1
]) ;
;}
else{ printf("%d", c[4
]) ;
;}
;
printf(" I am a string  ")
;
for(xx  = 0

;xx < 1

;xx  = xx + 1


){
xx  = xx + 2


;


};
x  = bbs(2
+ 3


,5
,xx );
return 0
;
}

int bbs(int a,int c,int b){


return a ;
}

int lol(int a,int c,int b,bool x){
int d;


a  = 2
+ 3


;

return b ;
}

int tol(int a,int c,int b,bool x){
int d;


a  = 2
+ 3
```



```

;

return b ;
}

int tol(int a,int c,int b,bool x){
int d;


a  = 2
+ 3

;

return b ;
}

```