Final Assignment

CSE-0302 Summer 2021

Prachurja Kanti Banrman Praggo
Department of Computer Science and Engineering
State University of Bangladesh (SUB)
Dhaka, Bangladesh
prachurjapraggo@gmail.com

***Abstract*—Final Assignment**

***Index Terms*—code in c/c++**

I. Introduction

This assignment is given by Compiler design course. The assignment is done with c and c++ code.

II. Codes

Assignment=1 include¡stdio.h¿ include¡string.h¿

int i,j,k,l,m,n=0,o,p,nv,z=0,t,x=0; char str[10],temp[20],temp2[20],temp3[20];

struct prod char lhs[10],rhs[10][10]; int n; pro[10];

void findter() for(k=0;k¡n;k++) if(temp[i]==pro[k].lhs[0]) for(t=0;t¡pro[k].n;t++) for(l=0;l¡20;l++) temp2[l]="; for(l=i+1;l¡strlen(temp);l++) temp2[l-i-1]=temp[l]; for(l=i;l¡20;l++) temp[l]="; for(l=0;l¡strlen(pro[k].rhs[t]);l++) temp[i+l]=pro[k].rhs[t][l]; strcat(temp,temp2); if(str[i]==temp[i]) return; else if(str[i]!=temp[i] temp[i]¿=65 temp[i]¡=90) break; break; if(temp[i]¿=65 temp[i]¡=90) findter();

int main() FILE *f; // clrscr();

for(i=0;i¡10;i++) pro[i].n=0;

f=fopen("in.txt","r"); while(!feof(f)) fscanf(f,"if(n¿0) if(strcmp(pro[n].lhs,pro[n-1].lhs) == 0) pro[n].lhs[0]="; fscanf(f,"pro[n-1].n++; continue; fscanf(f,"pro[n].n++; n++; n–;

printf("GRAMMAR IS AS FOLLOWS"); for(i=0;i¡n;i++) for(j=0;j¡pro[i].n;j++) printf("

while(1) for(l=0;l¡10;l++) str[0]=NULL;

printf("ANY STRING (0 for EXIT) : "); scanf("if(str[0]=='0') break;

for(j=0;j¡pro[0].n;j++) for(l=0;l¡20;l++) temp[l]=NULL; strcpy(temp,pro[0].rhs[j]);

m=0; for(i=0;i¡strlen(str);i++) if(str[i]==temp[i]) m++; else if(str[i]!=temp[i] temp[i]¿=65 temp[i]¡=90) findter(); if(str[i]==temp[i]) m++; else if(str[i]!=temp[i] (temp[i]¡65 —— temp[i]¿90)) break;

if(m==strlen(str) strlen(str)==strlen(temp)) printf("STRING can be PARSED !!!"); break;

if(j==pro[0].n) printf("STRING can NOT be PARSED !!!");

// cin.ignore(numeric$_limits < streamsize >:: max(),' ');$

III. Codes

include ¡stdio.h¿ include ¡stdlib.h¿

FILE *fp , *fp2;

void check$_comment(chara)charx;$

if(a == '/') //checking if the character starts with '/', it will be a comment if((x=fgetc(fp))=='*') check$_block_comment();$

else if(x == '/') // else if the next character '/', it is the beginning of single line comment check$_single_comment();$

else // when both the cases fail then it is not a comment fputc(a,fp2); fputc(x,fp2);

// when all the conditions are false, add the character as it is in the new file. else fputc(a,fp2);

// function for block comments void check$_block_comment()$

char x,y;

while((x=fgetc(fp))!=EOF) // the block comment has started

if(x=='*') y=fgetc(fp); // check if it ends

if(y=='/') return;

// function for single line comments void check$_single_comment()charx,y;$

while((x=fgetc(fp))!=EOF)

if(x==") return; // if the comment ends return from the function

int main(void) char c;

fp = fopen ("testfile.txt","r") ; // first file in read mode fp2 = fopen ("solved.txt","w") ; // second file in write mode

while((c=fgetc(fp))!=EOF) check$_comment(c); //checkingforthebeginningofacomment$

// closing both files fclose(fp); fclose(fp2);

return 0;

IV. Conclusion and Future Work

None

Acknowledgment

I would like to thank my honourable**Khan Md. Hasib Sir** for his time, generosity and critical insights into this project.

References

[1] Wilhelm, R., Maurer, D. (1995). Compiler design. Reading: Addison-Wesley Publishing Company.
[2] Grune, D., Van Reeuwijk, K., Bal, H. E., Jacobs, C. J., Langendoen, K. (2012). Modern compiler design. Springer Science Business Media.

[3] Muchnick, S. (1997). Advanced compiler design implementation. Morgan kaufmann.

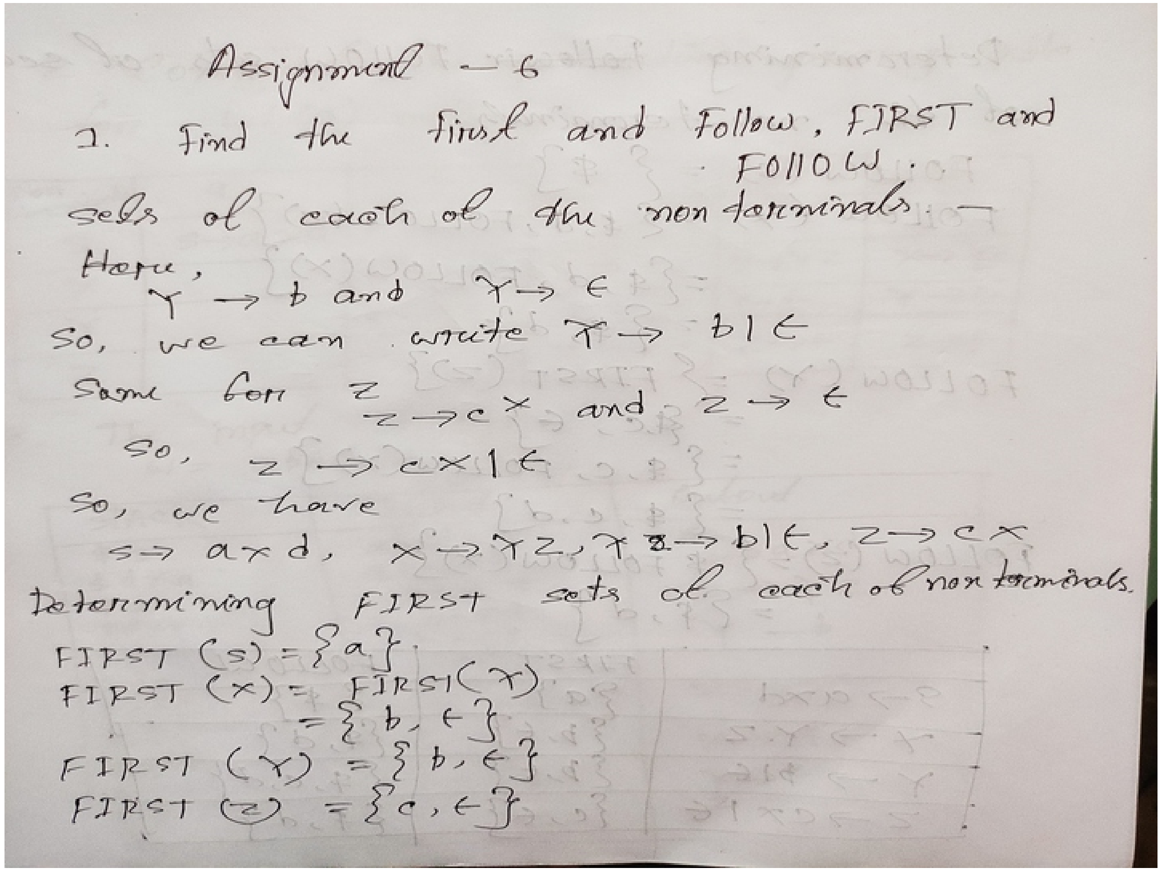
[4] Hoare, C. A. R., Jifeng, H., Sampaio, A. (1993). Normal form approach to compiler design. Acta informatica, 30(8), 701-739.

[5] Hoare, C. A. R., He Jifeng, and Augusto Sampaio. ”Normal form approach to compiler design.” Acta informatica 30.8 (1993): 701-739.

[6] Bozkus, Z., Choudhary, A., Fox, G., Haupt, T., Ranka, S. (1993, November). Fortran 90D/HPF compiler for distributed memory MIMD computers: Design, implementation, and performance results. In Supercomputing’93: Proceedings of the 1993 ACM/IEEE Conference on Supercomputing (pp. 351-360). IEEE.

[7] Bozkus, Zeki, et al. ”Fortran 90D/HPF compiler for distributed memory MIMD computers: Design, implementation, and performance results.” Supercomputing’93: Proceedings of the 1993 ACM/IEEE Conference on Supercomputing. IEEE, 1993.

[8] Millstein, R. E. (1971). Compiler Design for the ILLIAC 4. MASSACHUSETTS COMPUTER ASSOCITAES INC WAKEFIELD.



Assignment – 6

1. Find the first and Follow, FIRST and FOLLOW sets of each of the non terminals: –

Here,
$Y \rightarrow b$ and $Y \rightarrow \epsilon$

So, we can write $Y \rightarrow b \mid \epsilon$

Same for Z
$Z \rightarrow cX$ and $Z \rightarrow \epsilon$

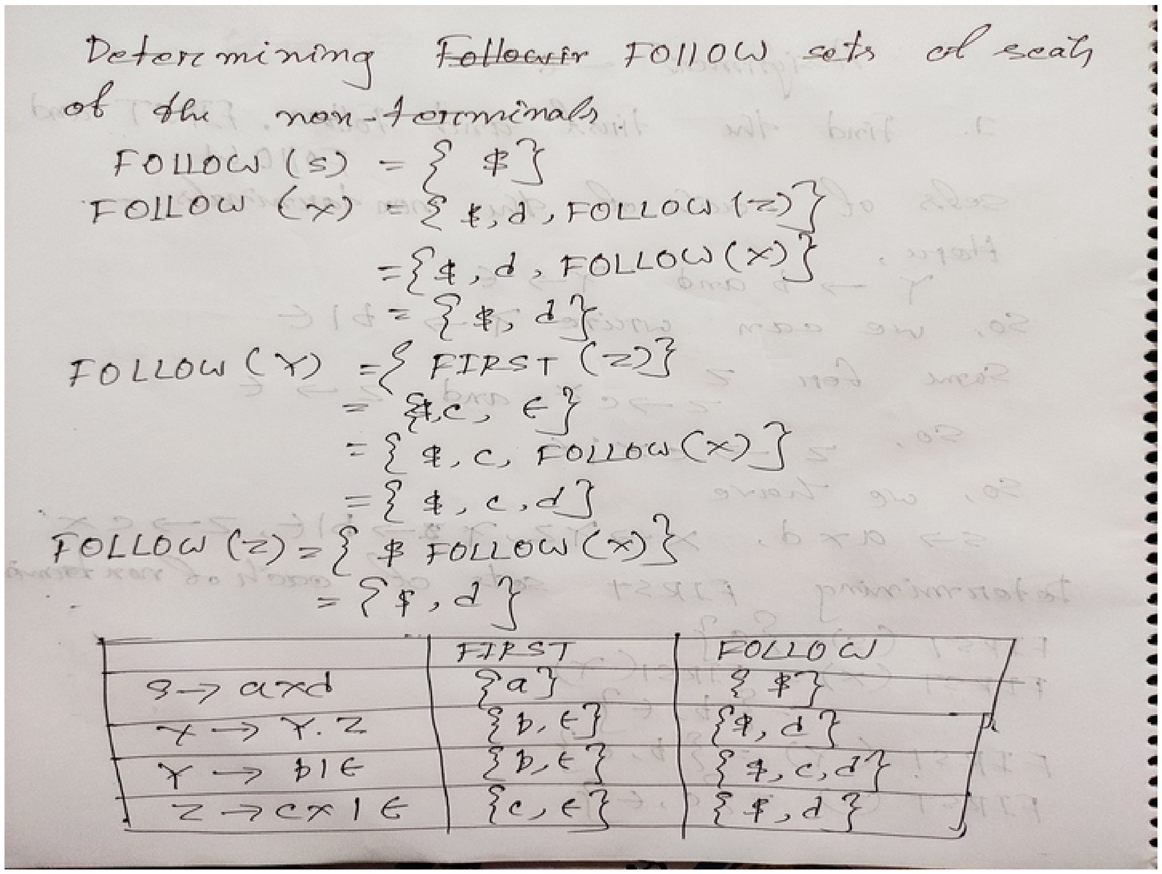
So, $Z \rightarrow cX \mid \epsilon$

So, we have
$S \rightarrow aXd$, $X \rightarrow YZ$, $Y \rightarrow b \mid \epsilon$, $Z \rightarrow cX$

Determining FIRST sets of each of non terminals.

$FIRST(S) = \{a\}$
$FIRST(X) = FIRST(Y)$
$= \{b, \epsilon\}$
$FIRST(Y) = \{b, \epsilon\}$
$FIRST(Z) = \{c, \epsilon\}$

Fig. 1. Code



Determining ~~Followfir~~ FOLLOW sets of each of the non-terminals.

FOLLOW (S) = { \$ }

FOLLOW (X) = { \$, d, FOLLOW (Z) }

= { \$, d, FOLLOW (X) }

= { \$, d }

FOLLOW (Y) = { FIRST (Z) }

= { \$, c, ϵ }

= { \$, c, FOLLOW (X) }

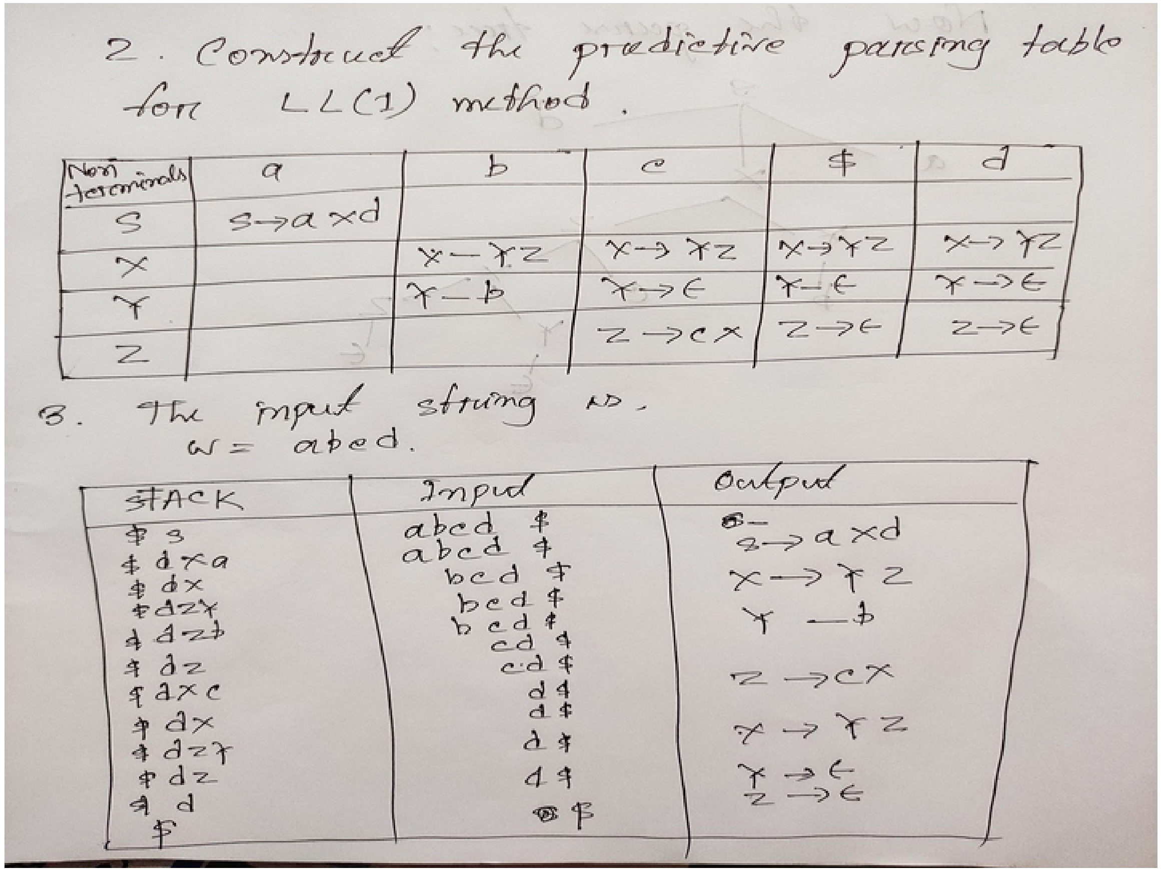
= { \$, c, d }

FOLLOW (Z) = { \$ FOLLOW (X) }

= { \$, d }

	FIRST	FOLLOW
S → aXd	{a}	{ \$ }
X → Y.Z	{b, ϵ}	{\$, d}
Y → b | ϵ	{b, ϵ}	{\$, c, d}
Z → cX | ϵ	{c, ϵ}	{\$, d}

Fig. 2. Code



2. Construct the predictive parsing table for LL(1) method.

Non terminals	a	b	c	$	d
S	S→axd				
X		X→YZ	X→YZ	X→YZ	X→YZ
Y		Y→b	Y→ε	Y→ε	Y→ε
Z			Z→cX	Z→ε	Z→ε

3. The input string is.
w = abcd.

STACK	Input	Output
$ S	abcd $	S→axd
$ d X a	abcd $	
$ d X	bcd $	X→YZ
$ d Z Y	bcd $	
$ d Z b	bcd $	Y→b
$ d Z	cd $	
$ d X c	cd $	Z→cX
$ d X	d $	
$ d Z Y	d $	X→YZ
$ d Z	d $	Y→ε
$ d	d $	Z→ε
$	$	

Fig. 3. Code

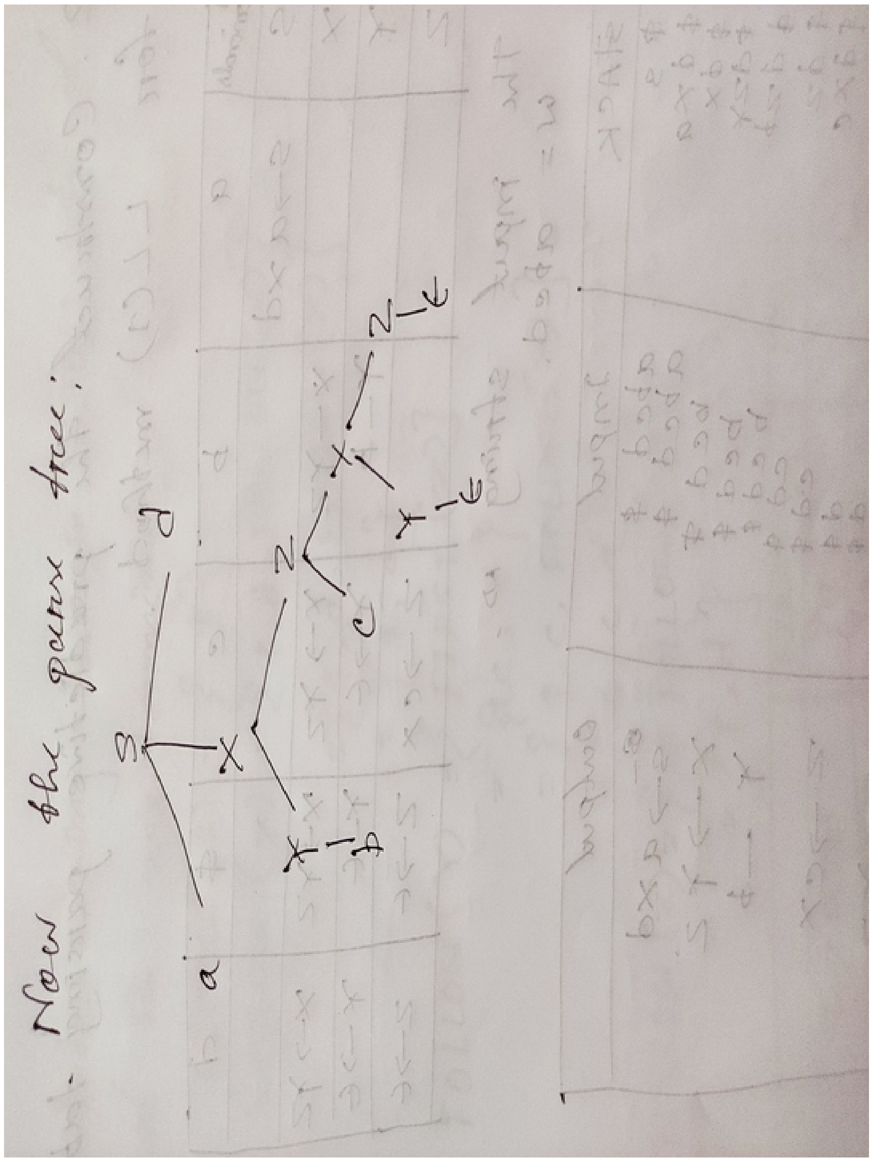
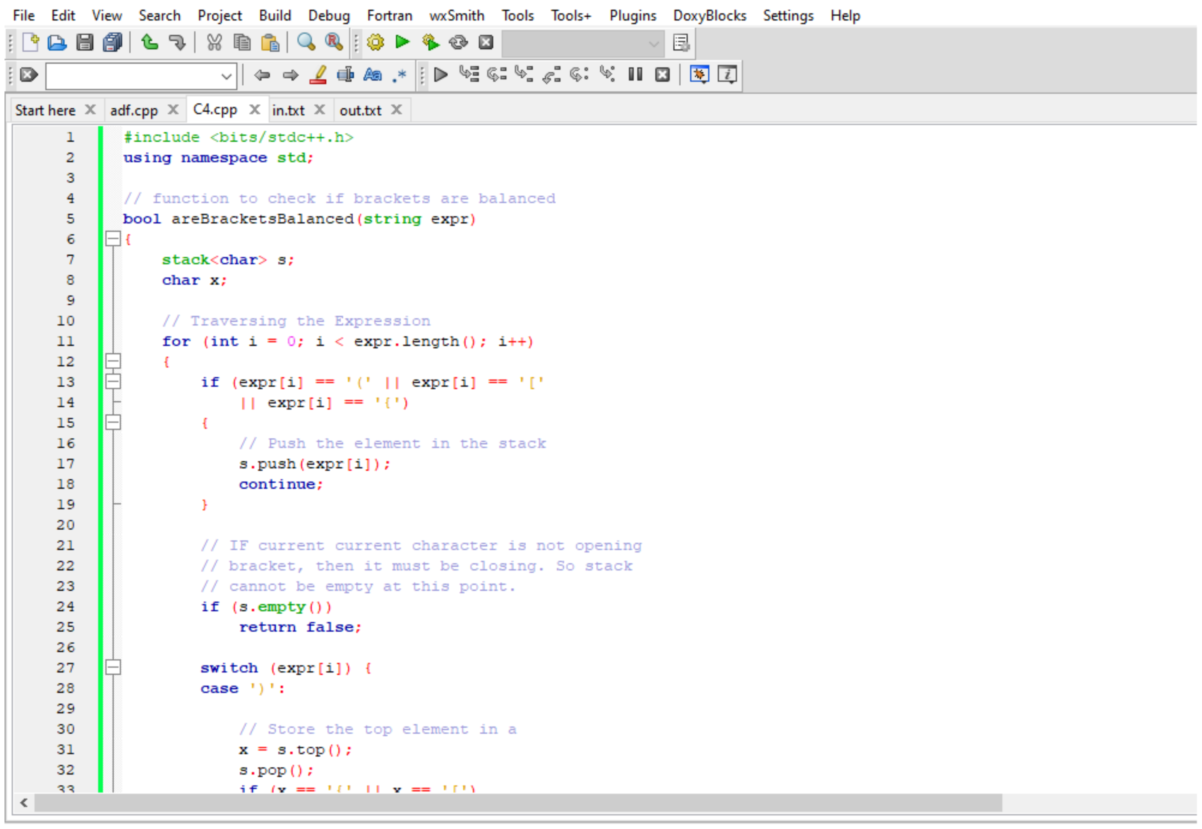
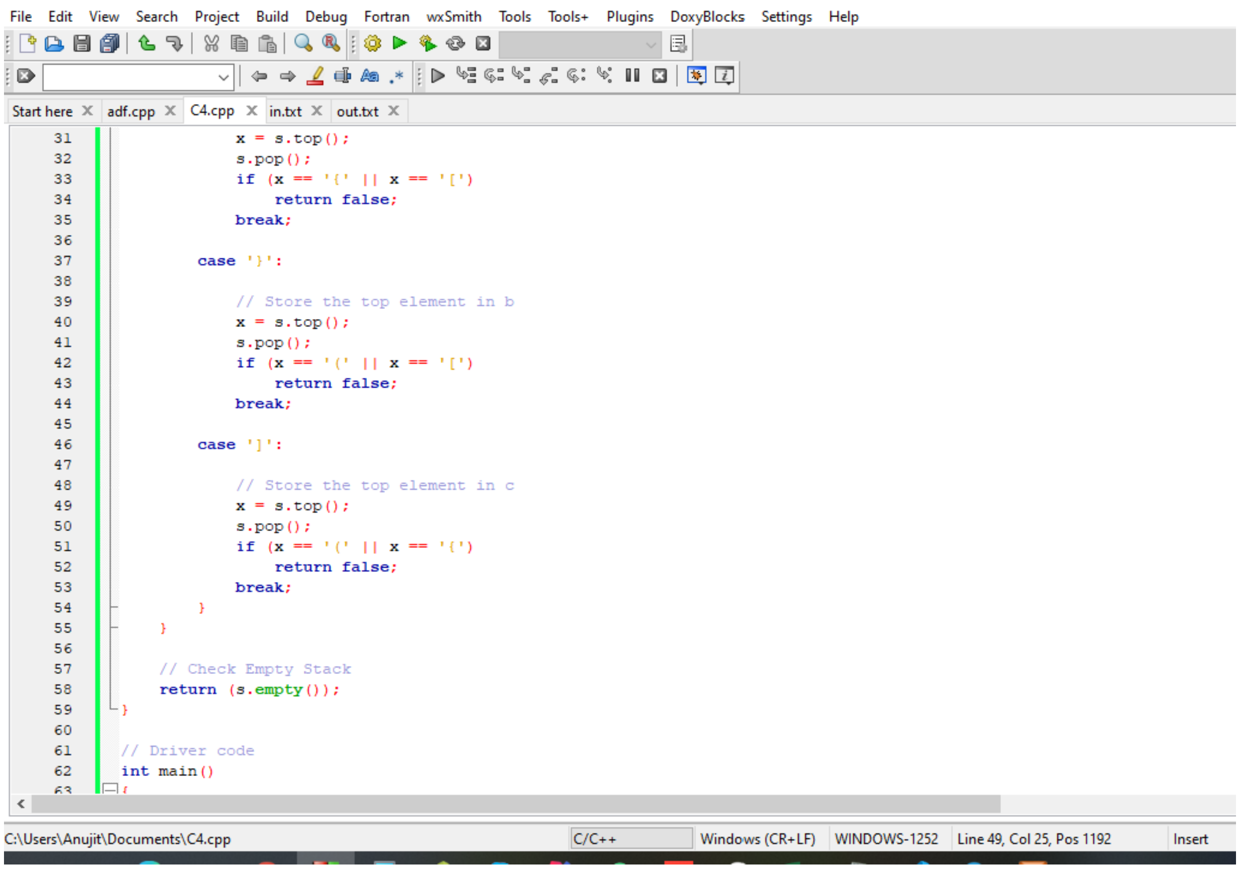


Fig. 4. Code



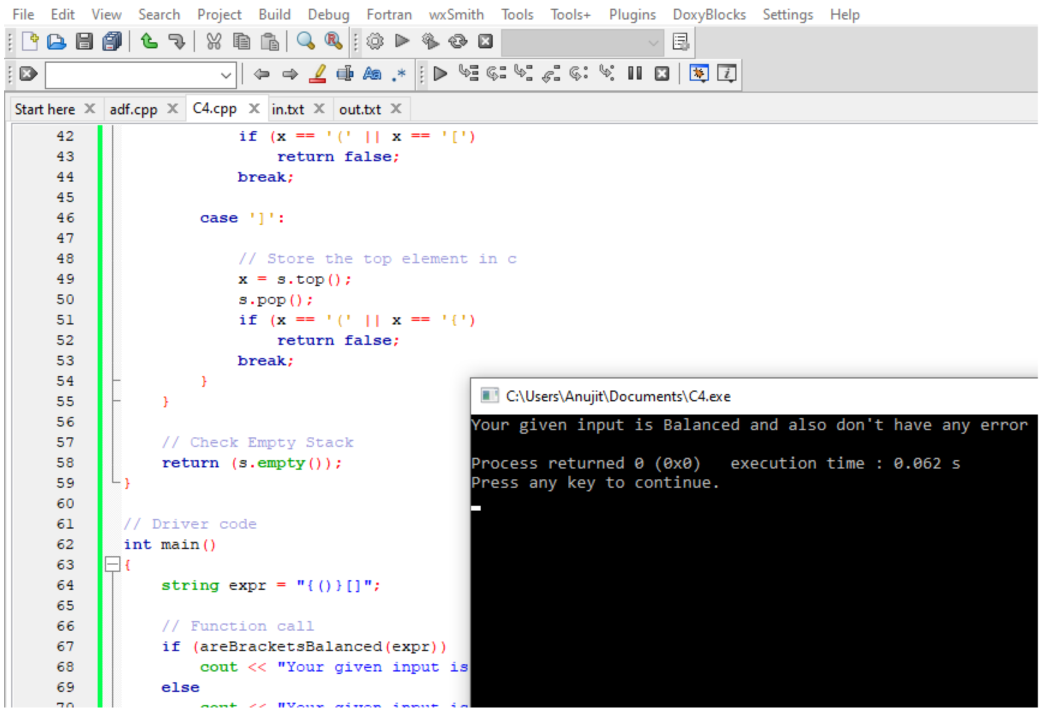
```cpp
#include <bits/stdc++.h>
using namespace std;

// function to check if brackets are balanced
bool areBracketsBalanced(string expr)
{
    stack<char> s;
    char x;

    // Traversing the Expression
    for (int i = 0; i < expr.length(); i++)
    {
        if (expr[i] == '(' || expr[i] == '['
            || expr[i] == '{')
        {
            // Push the element in the stack
            s.push(expr[i]);
            continue;
        }

        // IF current current character is not opening
        // bracket, then it must be closing. So stack
        // cannot be empty at this point.
        if (s.empty())
            return false;

        switch (expr[i]) {
        case ')':

            // Store the top element in a
            x = s.top();
            s.pop();
            if (x == '{' || x == '[')
```

Fig. 5. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt
C:\Users\Anujit\Documents\C4.cpp
C/C++
Windows (CR+LF)
WINDOWS-1252
Line 49, Col 25, Pos 1192
Insert

```
            x = s.top();
            s.pop();
            if (x == '{' || x == '[')
                return false;
            break;

        case '}':

            // Store the top element in b
            x = s.top();
            s.pop();
            if (x == '(' || x == '[')
                return false;
            break;

        case ']':

            // Store the top element in c
            x = s.top();
            s.pop();
            if (x == '(' || x == '{')
                return false;
            break;
        }
    }

    // Check Empty Stack
    return (s.empty());
}

// Driver code
int main()
{
```

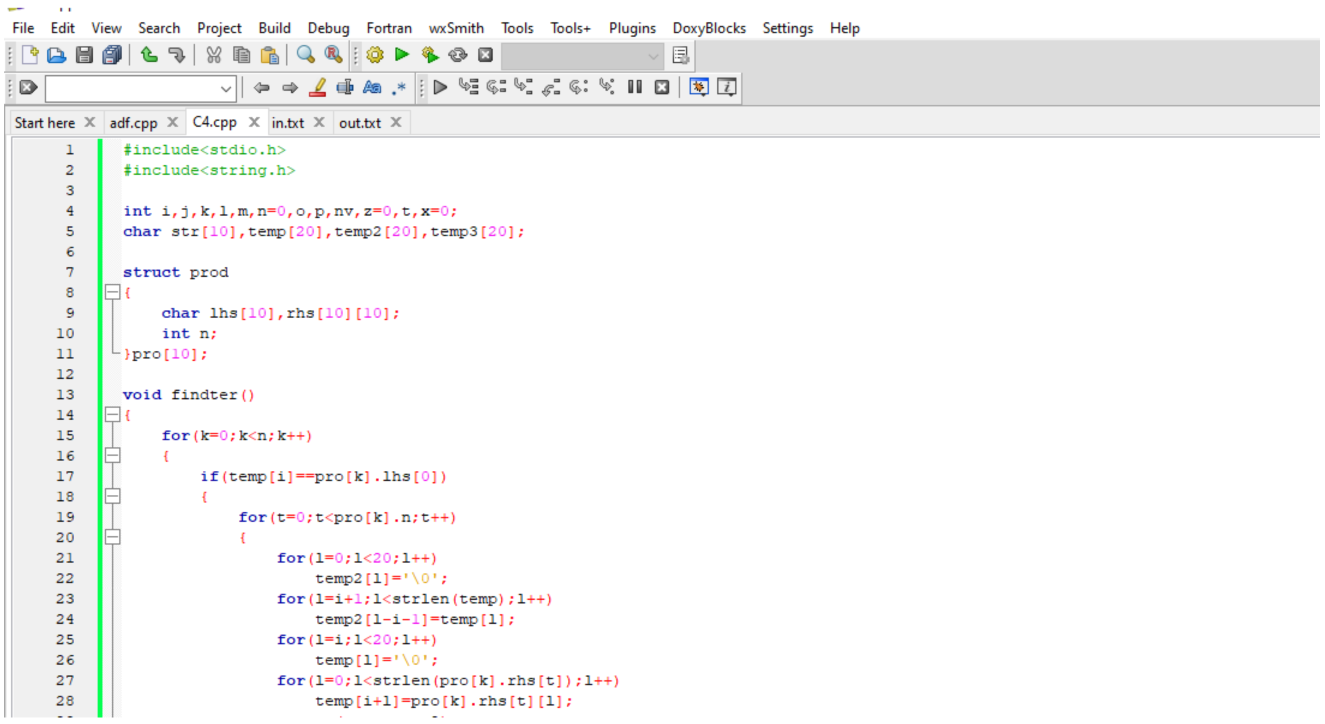
Fig. 6. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here adf.cpp C4.cpp in.txt out.txt
C:\Users\Anujit\Documents\C4.exe

```
                if (x == '(' || x == '[')
                    return false;
                break;

            case ']':

                // Store the top element in c
                x = s.top();
                s.pop();
                if (x == '(' || x == '{')
                    return false;
                break;
            }
        }

        // Check Empty Stack
        return (s.empty());
}

// Driver code
int main()
{
    string expr = "{()}[]";

    // Function call
    if (areBracketsBalanced(expr))
        cout << "Your given input is
    else
```

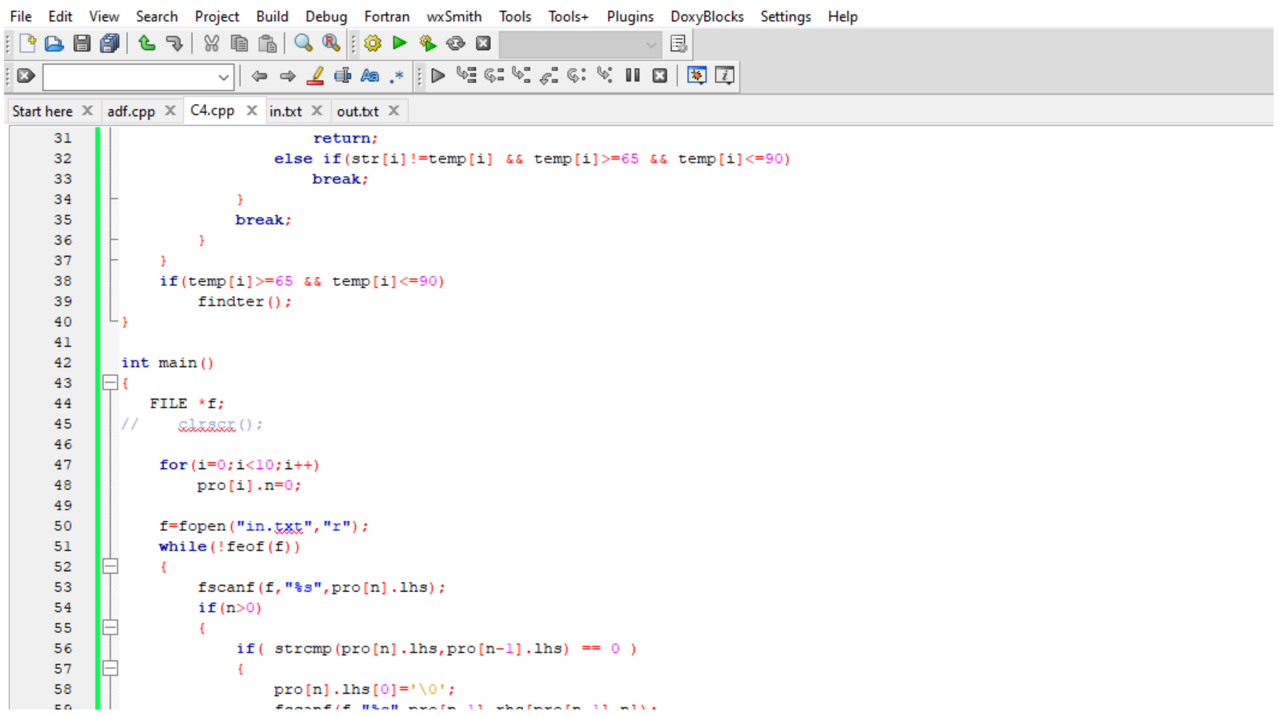
```
Your given input is Balanced and also don't have any error

Process returned 0 (0x0)   execution time : 0.062 s
Press any key to continue.
```

Fig. 7. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

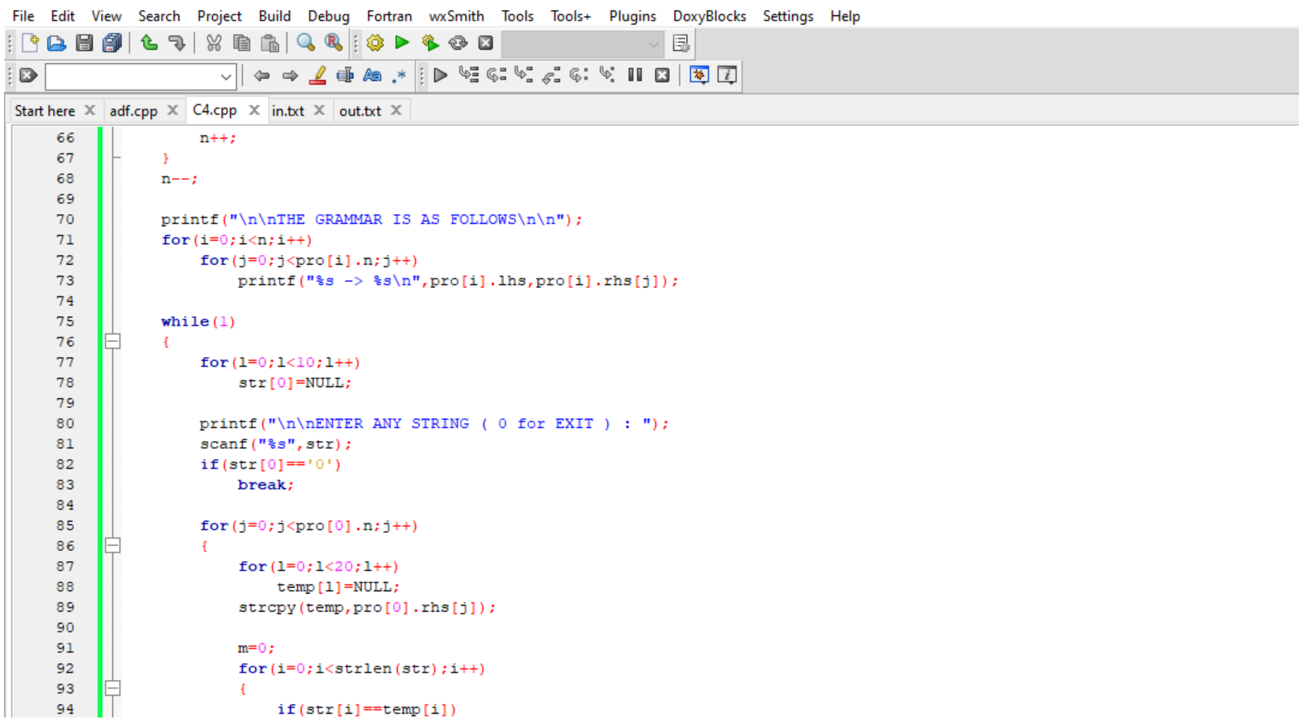
```
#include<stdio.h>
#include<string.h>

int i,j,k,l,m,n=0,o,p,nv,z=0,t,x=0;
char str[10],temp[20],temp2[20],temp3[20];

struct prod
{
    char lhs[10],rhs[10][10];
    int n;
}pro[10];

void findter()
{
    for(k=0;k<n;k++)
    {
        if(temp[i]==pro[k].lhs[0])
        {
            for(t=0;t<pro[k].n;t++)
            {
                for(l=0;l<20;l++)
                    temp2[l]='\0';
                for(l=i+1;l<strlen(temp);l++)
                    temp2[l-i-1]=temp[l];
                for(l=i;l<20;l++)
                    temp[l]='\0';
                for(l=0;l<strlen(pro[k].rhs[t]);l++)
                    temp[i+l]=pro[k].rhs[t][l];
```

Fig. 8. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

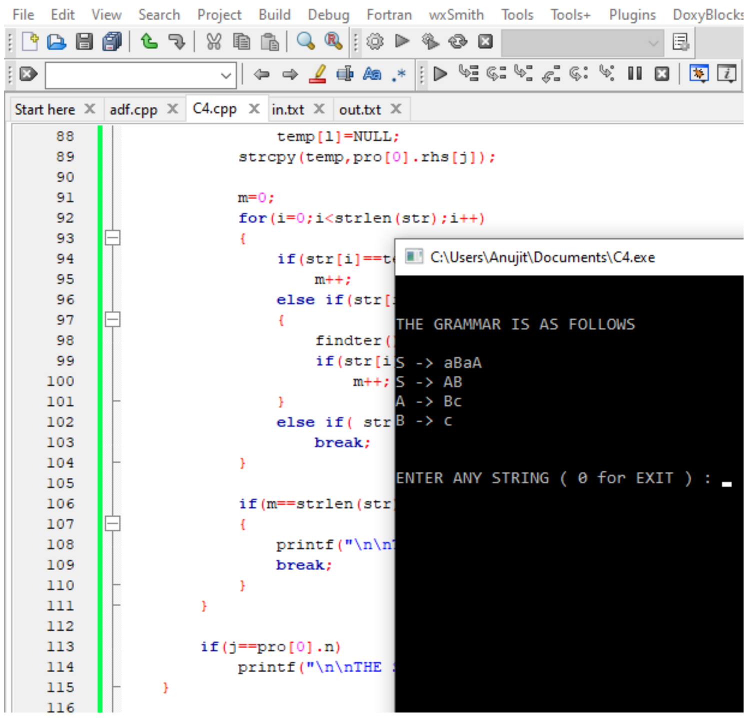
```
                    return;
                else if(str[i]!=temp[i] && temp[i]>=65 && temp[i]<=90)
                    break;
            }
            break;
        }
    }
    if(temp[i]>=65 && temp[i]<=90)
        findter();
}

int main()
{
   FILE *f;
//    clrscr();

    for(i=0;i<10;i++)
        pro[i].n=0;

    f=fopen("in.txt","r");
    while(!feof(f))
    {
        fscanf(f,"%s",pro[n].lhs);
        if(n>0)
        {
            if( strcmp(pro[n].lhs,pro[n-1].lhs) == 0 )
            {
                pro[n].lhs[0]='\0';
                fscanf(f,"%s",pro[n-1].rhs[pro[n-1].n]);
```

Fig. 9. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

```
        n++;
    }
    n--;

    printf("\n\nTHE GRAMMAR IS AS FOLLOWS\n\n");
    for(i=0;i<n;i++)
        for(j=0;j<pro[i].n;j++)
            printf("%s -> %s\n",pro[i].lhs,pro[i].rhs[j]);

    while(1)
    {
        for(l=0;l<10;l++)
            str[0]=NULL;

        printf("\n\nENTER ANY STRING ( 0 for EXIT ) : ");
        scanf("%s",str);
        if(str[0]=='0')
            break;

        for(j=0;j<pro[0].n;j++)
        {
            for(l=0;l<20;l++)
                temp[l]=NULL;
            strcpy(temp,pro[0].rhs[j]);

            m=0;
            for(i=0;i<strlen(str);i++)
            {
                if(str[i]==temp[i])
```

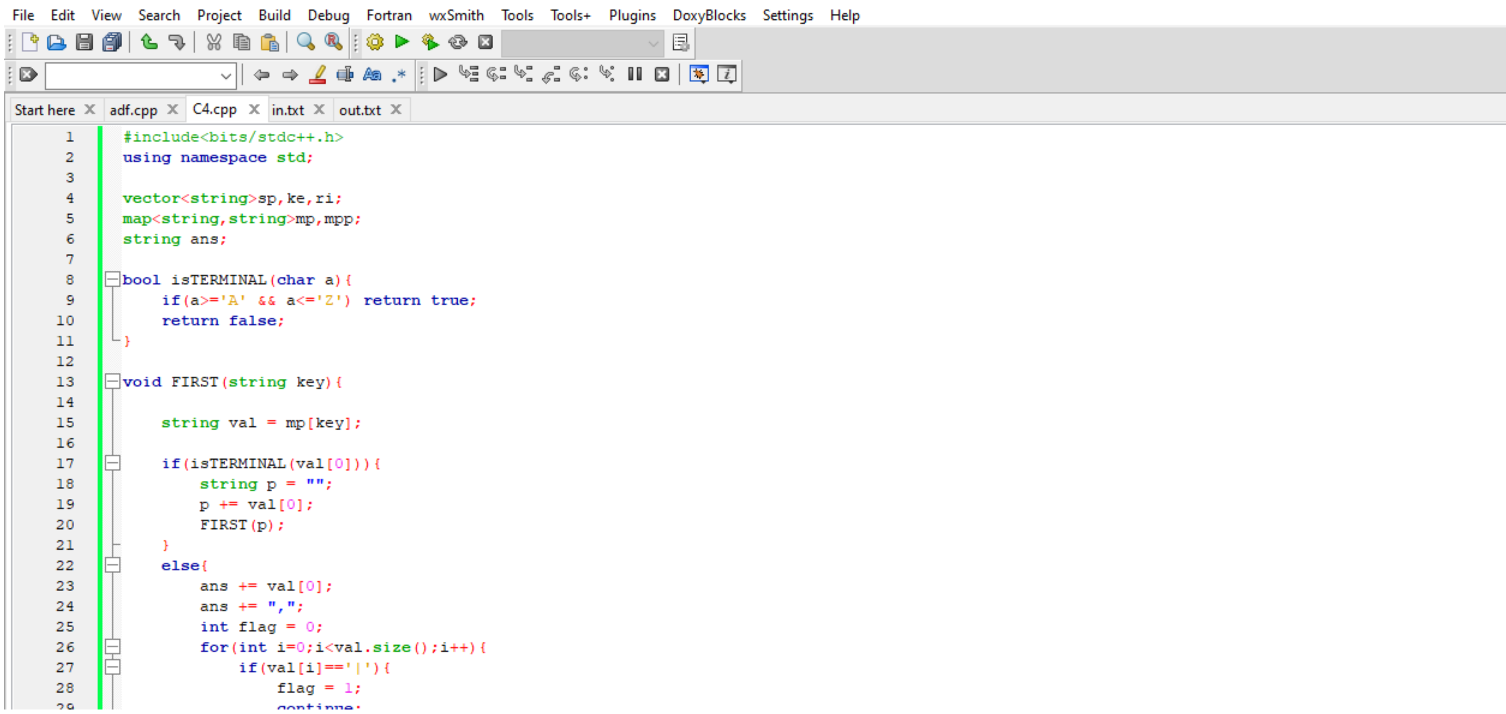
Fig. 10. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks
Start here adf.cpp C4.cpp in.txt out.txt
C:\Users\Anujit\Documents\C4.exe

```
            temp[1]=NULL;
        strcpy(temp,pro[0].rhs[j]);

        m=0;
        for(i=0;i<strlen(str);i++)
        {
            if(str[i]==t
                m++;
            else if(str[
            {
                findter(
                if(str[i
                    m++;
            }
            else if( str
                break;
        }

        if(m==strlen(str
        {
            printf("\n\n
            break;
        }
    }

    if(j==pro[0].n)
        printf("\n\nTHE
}
```

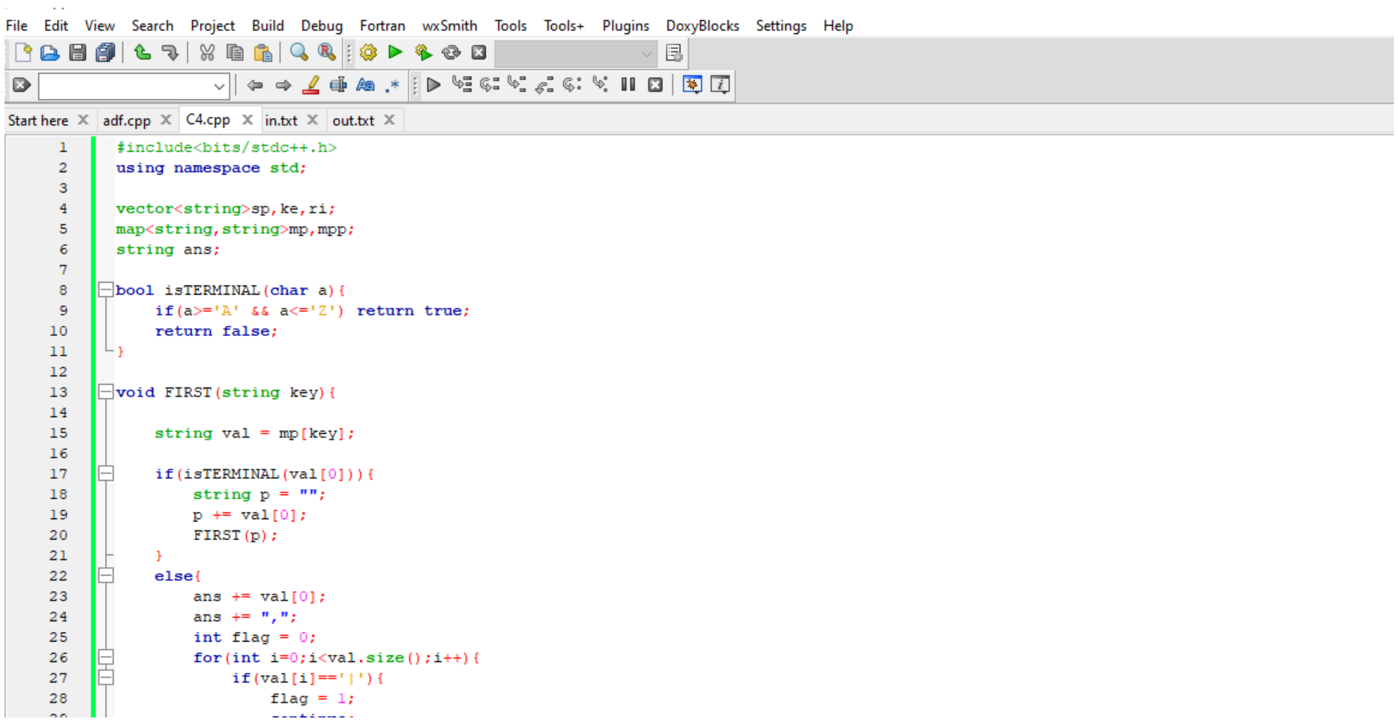
```
THE GRAMMAR IS AS FOLLOWS

S -> aBaA
S -> AB
A -> Bc
B -> c


ENTER ANY STRING ( 0 for EXIT ) :
```

Fig. 11. Code



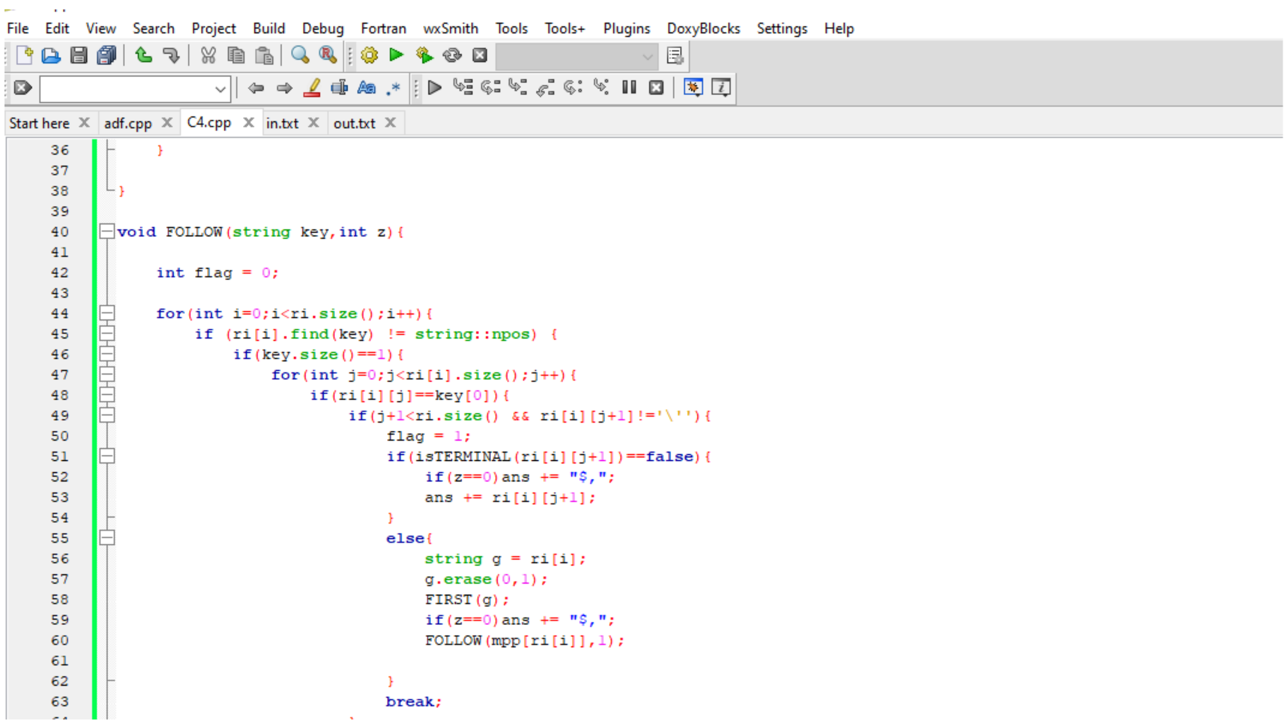
```cpp
#include<bits/stdc++.h>
using namespace std;

vector<string>sp,ke,ri;
map<string,string>mp,mpp;
string ans;

bool isTERMINAL(char a){
    if(a>='A' && a<='Z') return true;
    return false;
}

void FIRST(string key){

    string val = mp[key];

    if(isTERMINAL(val[0])){
        string p = "";
        p += val[0];
        FIRST(p);
    }
    else{
        ans += val[0];
        ans += ",";
        int flag = 0;
        for(int i=0;i<val.size();i++){
            if(val[i]=='|'){
                flag = 1;
                continue;
```

Fig. 12. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

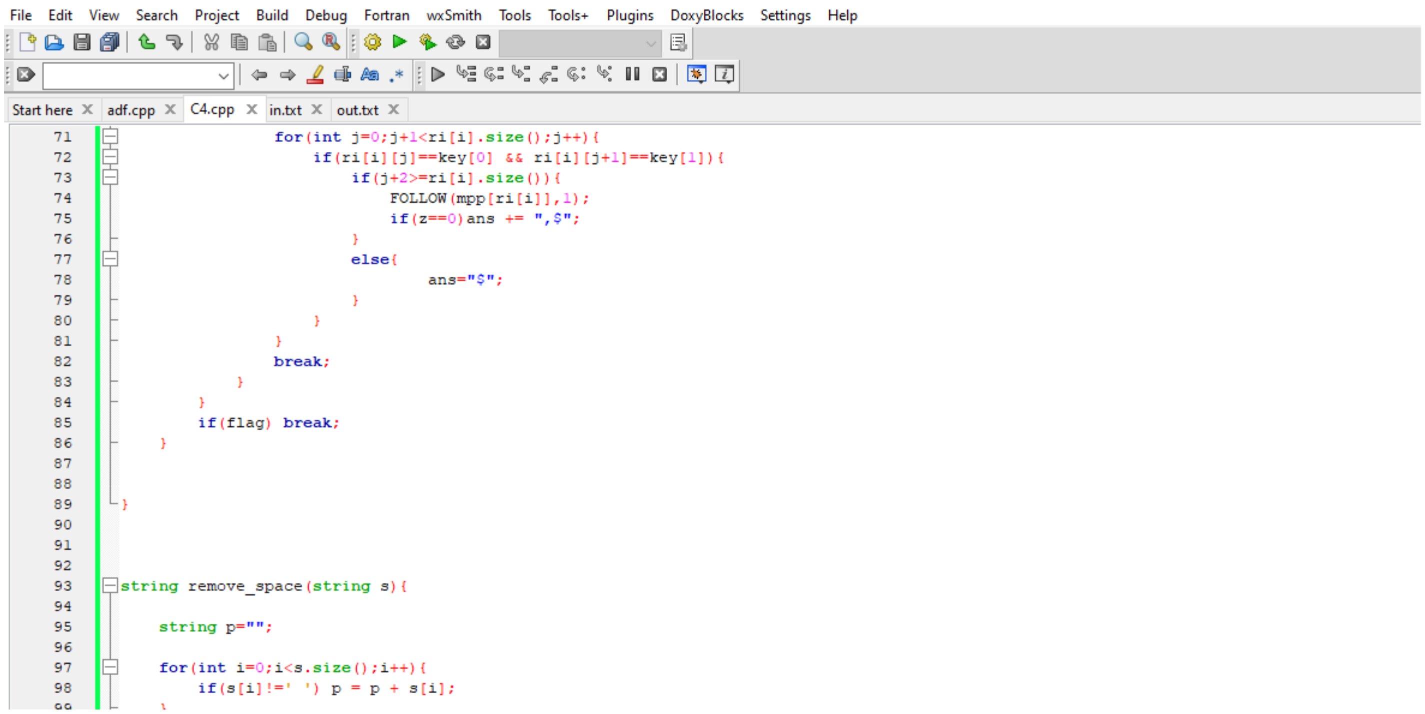
```
#include<bits/stdc++.h>
using namespace std;

vector<string>sp,ke,ri;
map<string,string>mp,mpp;
string ans;

bool isTERMINAL(char a){
    if(a>='A' && a<='Z') return true;
    return false;
}

void FIRST(string key){

    string val = mp[key];

    if(isTERMINAL(val[0])){
        string p = "";
        p += val[0];
        FIRST(p);
    }
    else{
        ans += val[0];
        ans += ",";
        int flag = 0;
        for(int i=0;i<val.size();i++){
            if(val[i]=='|'){
                flag = 1;
```

Fig. 13. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

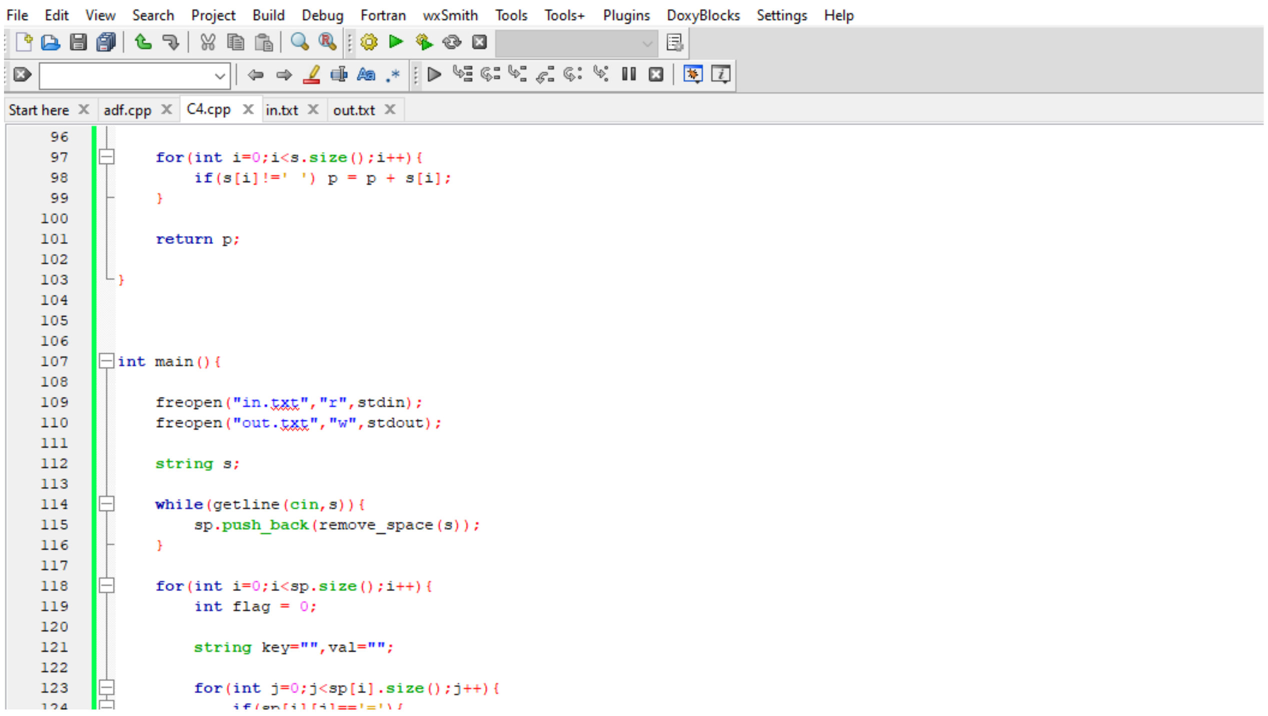
```
    }

}

void FOLLOW(string key,int z){

    int flag = 0;

    for(int i=0;i<ri.size();i++){
        if (ri[i].find(key) != string::npos) {
            if(key.size()==1){
                for(int j=0;j<ri[i].size();j++){
                    if(ri[i][j]==key[0]){
                        if(j+1<ri.size() && ri[i][j+1]!='\''){
                            flag = 1;
                            if(isTERMINAL(ri[i][j+1])==false){
                                if(z==0)ans += "$,";
                                ans += ri[i][j+1];
                            }
                            else{
                                string g = ri[i];
                                g.erase(0,1);
                                FIRST(g);
                                if(z==0)ans += "$,";
                                FOLLOW(mpp[ri[i]],1);

                            }
                            break;
```

Fig. 14. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

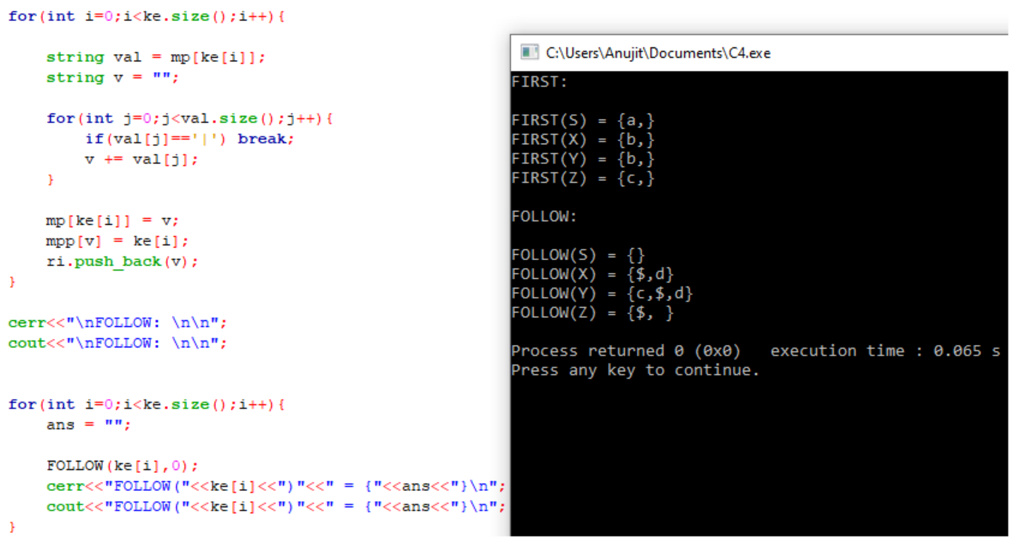
```
                for(int j=0;j+1<ri[i].size();j++){
                    if(ri[i][j]==key[0] && ri[i][j+1]==key[1]){
                        if(j+2>=ri[i].size()){
                            FOLLOW(mpp[ri[i]],1);
                            if(z==0)ans += ",$";
                        }
                        else{
                                ans="$";
                        }
                    }
                }
                break;
            }
        }
        if(flag) break;
    }


}



string remove_space(string s){

    string p="";

    for(int i=0;i<s.size();i++){
        if(s[i]!=' ') p = p + s[i];
```

Fig. 15. Code


File Edit View Search Project Build Debug Fortran wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Start here
adf.cpp
C4.cpp
in.txt
out.txt

```

    for(int i=0;i<s.size();i++){
        if(s[i]!=' ') p = p + s[i];
    }

    return p;

}



int main(){

    freopen("in.txt","r",stdin);
    freopen("out.txt","w",stdout);

    string s;

    while(getline(cin,s)){
        sp.push_back(remove_space(s));
    }

    for(int i=0;i<sp.size();i++){
        int flag = 0;

        string key="",val="";

        for(int j=0;j<sp[i].size();j++){
```

Fig. 16. Code



```
for(int i=0;i<ke.size();i++){

    string val = mp[ke[i]];
    string v = "";

    for(int j=0;j<val.size();j++){
        if(val[j]=='|') break;
        v += val[j];
    }

    mp[ke[i]] = v;
    mpp[v] = ke[i];
    ri.push_back(v);
}

cerr<<"\nFOLLOW: \n\n";
cout<<"\nFOLLOW: \n\n";


for(int i=0;i<ke.size();i++){
    ans = "";

    FOLLOW(ke[i],0);
    cerr<<"FOLLOW("<<ke[i]<<")"<<" = {"<<ans<<"}\n";
    cout<<"FOLLOW("<<ke[i]<<")"<<" = {"<<ans<<"}\n";
}
```

C:\Users\Anujit\Documents\C4.exe

```
FIRST:

FIRST(S) = {a,}
FIRST(X) = {b,}
FIRST(Y) = {b,}
FIRST(Z) = {c,}

FOLLOW:

FOLLOW(S) = {}
FOLLOW(X) = {$,d}
FOLLOW(Y) = {c,$,d}
FOLLOW(Z) = {$, }

Process returned 0 (0x0)   execution time : 0.065 s
Press any key to continue.
```

Fig. 17.   Code