



Jatiya Kabi Kazi Nazrul Islam University

Trishal, Mymensingh

Lab Report

Course Code: CSE-324

Compiler Design Lab

Submitted To

Dr AHM Kamal

Professor,

Department of Computer Science & Engineering

Submitted By

Md Hasibur Rahman

Roll:18102009

Session:2017-18

Department of Computer Science & Engineering

INDEX

Problem Number.	Problem Name	Pages
1.	Write a Program to separate delimiter from a statement.	1-2
2.	Write a Program to separate comments from a statement.	2-4
3.	Write a Program to separate all keywords from a statement.	4-5
4.	Write a Program to separate all arithmetic operator from a statement.	6-7
5.	Write a Program to separate all identifier from a statement.	7-9
6.	Write a Program to separate a number from a statement.	9-11
7.	Write a Program to separate all relational operator from a statement.	12-13
8.	Write a Program to Perform leftmost derivation for a given set of productions.	14-16
9.	Write a Program to simulate a string in a given DFA.	17-19
10.	Write a Program to find first from a given set of production.	19-21
11.	Write a Program to given a set of instructions. Now generate three address code.	22-25
12.	Write a Program to generate assembly language code for a given three address code.	26-29

Problem No: 01

Problem Name: Write a Program to separate delimiter from a given statement.

Objective: Main objective is to separate delimiter from a given statement and simulate the process of parser.

Algorithm:


1. Start
2. Read the input statement and set i:=0
3. Repeat process 4 and 5 until i<inputSize
4. if input[i] == ' ' continue else scan until next space
5. If scanned string is a delimiter print delimiter and set i:=i+1
6. End

Code :

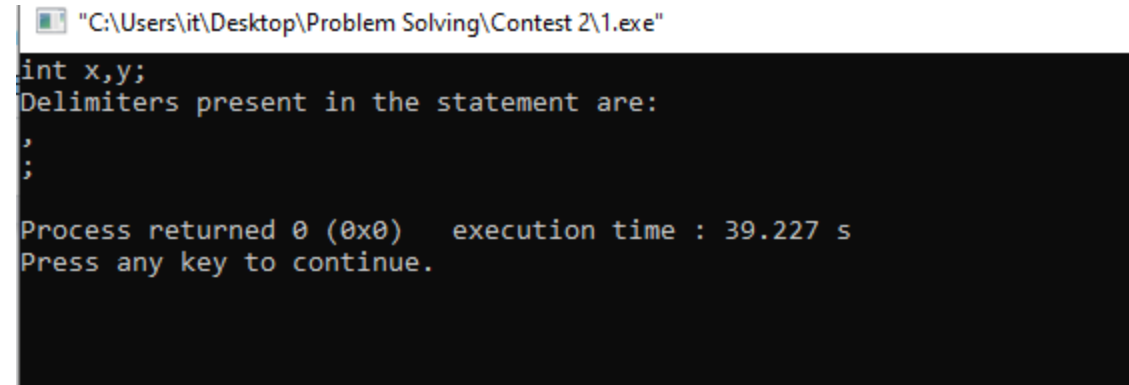
```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    set<string> st = {"\\", ",", "\\;", "\"", "\\'", "\\{", "\\}", "\\|", "\\\\", "\\./"};
    string s;
    getline(cin, s);
    int l = s.size(), i=0;
    cout<<"Delimiters present in the statement are: \n";
    while(i<l){
        string tmp="";
        tmp+=s[i];
        if(st.find(tmp)!=st.end())
            cout<<s[i]<<"\n";
        i++;
    }
    return 0;
}
```

Input:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\1.exe"  
int x,y;
```

Output:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\1.exe"  
int x,y;  
Delimiters present in the statement are:  
,  
;  
Process returned 0 (0x0)   execution time : 39.227 s  
Press any key to continue.
```

Discussion: From this experiment we could be able to separate all the delimiter from a statement.

Problem No: 02

Problem Name: Write a Program to separate comments from a statement.

Objective: Main objective is to separate comment from a given statement and simulate the process of parser.

Algorithm:

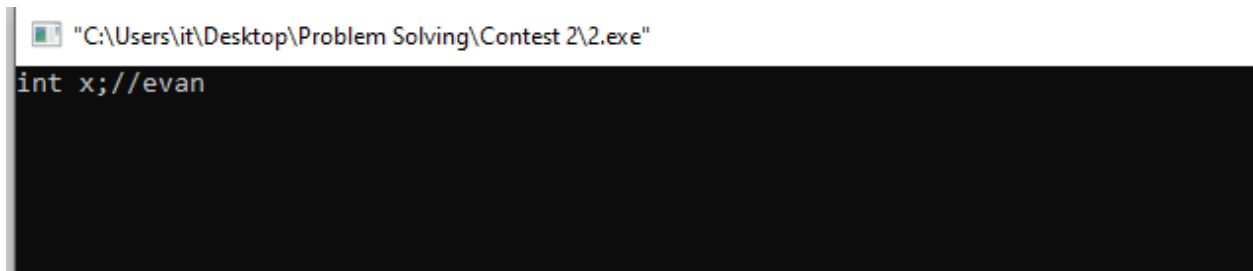
- 1.Start
2. Read the input statement.
3. Check whether input is comment.
- 4.If comment then print the comment.
- 5.End

Code :

```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    string s;
    getline(cin,s);
    int l = s.size();
    cout<<"comment present in the statement is: \n";
    for(int i=0;i<l;i++){
        if(s[i]=='/'){
            i++;
            while(i<l){
                if(s[i]=='*' || s[i]=='/'){
                    i++;
                    continue;
                }
                cout<<s[i++];
            }
        }
    }
    return 0;
}
```

Input:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\2.exe"
int x;//evan
```

Output:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\2.exe"
int x;//evan
comment present in the statement is:
evan
Process returned 0 (0x0) execution time : 30.940 s
Press any key to continue.
```

Discussion: From this experiment we could be able to separate comments from a statement.

Problem No: 03

Problem Name: Write a Program to separate all keywords from a statement.

Objective: To Write a Program that separate all keywords from a statement.

Algorithm:

- 1.Start
2. Read the input statement.
3. Check whether input is alphabet.
- 4.If alphabet then check is it keyword.
- 5.Then print the string is keyword
- 6.End

Code :

```
#include<bits/stdc++.h>

using namespace std;

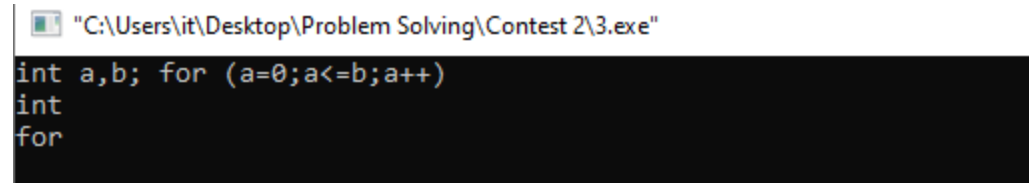
int main()
{
    set<string>keywords =
{"for","while","int","float","string","char","return"};
    string s;
```

```

getline(cin,s);
int i = 0, l = s.size();
while(i<l){
    string tmp="";
    while(s[i]!=' ' && i<l) tmp+=s[i++];
    if(keywords.find(tmp)!=keywords.end())
        cout<<tmp<<"\n";
    if(s[i]==' ') i++;
}
return 0;
}

```

Input:

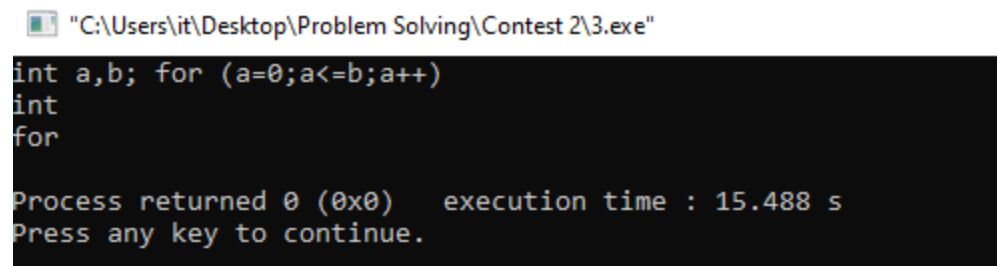


```

"C:\Users\it\Desktop\Problem Solving\Contest 2\3.exe"
int a,b; for (a=0;a<=b;a++)
int
for

```

Output:



```

"C:\Users\it\Desktop\Problem Solving\Contest 2\3.exe"
int a,b; for (a=0;a<=b;a++)
int
for

Process returned 0 (0x0)   execution time : 15.488 s
Press any key to continue.

```

Discussion: From this experiment we could be able to separate all keywords from a statement.

Problem No: 04

Problem Name: Write a Program to separate all arithmetic operator from a statement.

Objective: To Write a Program that separate all arithmetic operator from a statement.

Algorithm:

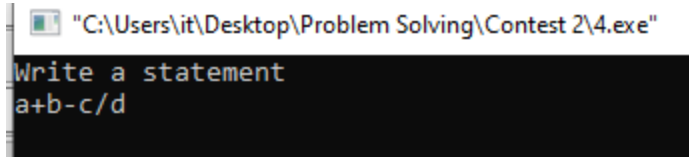
- 1.Start
2. Read the input statement.
3. Check the input if there is any arithmetic operator.
4. Then print it is arithmetic operator.
5. End

Code :

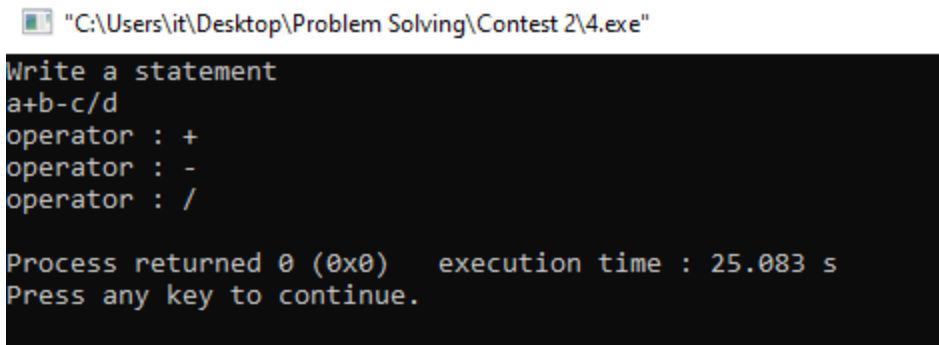
```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    set<string>operators = {"+", "-", "/", "*"};
    string s;
    cout<<"Write a statement\n";
    getline(cin,s);
    int l = s.size();
    for(int i=0;i<l;i++){
        string tmp = "";tmp+=s[i];
        if(operators.find(tmp)!=operators.end())cout<<"operator : 
"<<tmp<<"\n";
    }
    return 0;
}
```


Input:

"C:\Users\it\Desktop\Problem Solving\Contest 2\4.exe"
Write a statement
a+b-c/d

Output:

"C:\Users\it\Desktop\Problem Solving\Contest 2\4.exe"
Write a statement
a+b-c/d
operator : +
operator : -
operator : /

Process returned 0 (0x0) execution time : 25.083 s
Press any key to continue.

Discussion: From this experiment we could be able to separate all arithmetic operator from a statement.

Problem No: 05

Problem Name: Write a Program to separate all identifier from a statement.

Objective: To Write a Program that separate all identifier from a statement.

Algorithm:

- 1.Start
2. Read the input statement.
3. Check whether the input is keyword.
4. If the input is not a keyword then check whether the input is identifier
5. Then print the input is identifier
- 6.End

Code :

```

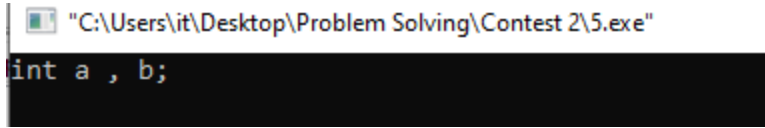
#include<bits/stdc++.h>

using namespace std;

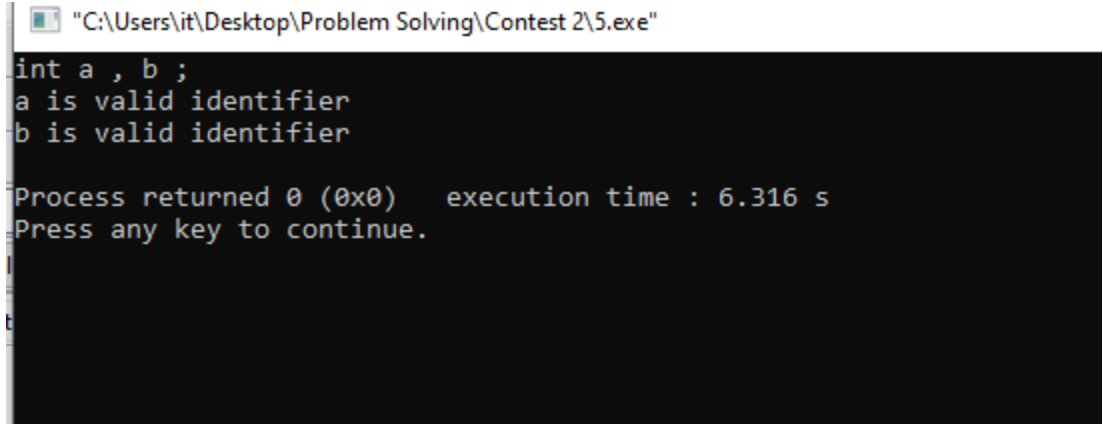
bool isValidIdentifier(string s){
    if(!(isalpha(s[0]) || s[0]=='_')){
        return false;
    }
    for(int i=0;i<s.size();i++){
        if(isalpha(s[i]) || isdigit(s[i]) || s[i]=='_')
            {
                continue;
            }
        return false;
    }
    return true;
}

int main()
{
    set<string>keywords =
{"for","while","int","float","string","char","return"};
    string s;
    getline(cin,s);
    int i = 0, l = s.size();
    while(i<l){
        string tmp="";
        while(s[i]!=' ' && i<l)tmp+=s[i++];
        if(keywords.find(tmp)==keywords.end())
        {
            if(isValidIdentifier(tmp)){
                cout<<tmp<<" is valid identifier \n";
            }
        }
        if(s[i]==' ')i++;
    }
    return 0;
}

```

Input:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\5.exe"  
int a , b;
```

Output:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\5.exe"  
int a , b ;  
a is valid identifier  
b is valid identifier  
  
Process returned 0 (0x0) execution time : 6.316 s  
Press any key to continue.
```

Discussion: From this experiment we could be able to separate all identifier from a statement.

Problem No:06

Problem Name: Write a Program to separate number from a input statement.

Objective: To Write a Program that separate number from a input statement.

Algorithm:

- 1.Start
2. Read the input statement.
3. Declare a string variable 'token'.
- 4.Check whether the input is digit.
5. Then store the digit in 'token'.
6. Check whether there is any delimiter.

7.Then print the input is number.


8.End

Code :


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

bool isdelimiter(char a)
{
    char delimiter[5] ={' ',' ',' ',' ','('};
    for(int i=0;i<5;i++)
    {
        if(a==delimiter[i])
            return true;
    }
}

int main()
{
    string inp,token;
    token = "";
    printf("give input statement:");
    getline(cin, inp);
    for(int i=0;i<inp.length();i++)
    {
        if(isdigit(inp[i]))
        {
            token = token + inp[i];
        }
        else if(isdelimiter(inp[i]))
        {
            if(token != "")
            {
                cout << token << " is a number \n";
            }
            token = "";
        }
    }
}
```

Input: "C:\Users\it\Desktop\Problem Solving\Contest 2\6.exe"

```
give input statement:int a=6,b=7;
```

Output: "C:\Users\it\Desktop\Problem Solving\Contest 2\6.exe"

```
give input statement:int a=6,b=7;
```

```
6 is a number
```

```
7 is a number
```

```
Process returned 0 (0x0)   execution time : 29.152 s
```

```
Press any key to continue.
```

Discussion: From this experiment we could be able to separate number from a input statement.

Problem No: 07

Problem Name: Write a Program to separate all relational operator from a statement.

Objective: To Write a Program that separate all relational operator from a statement.

Algorithm:

- 1.Start
2. Read the input statement.
3. Check whether the input is relational operator.
4. Then print all the relational operator in the input statement.
5. End

Code :

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


int main()
{
    string inp,token;
    token = "";
    printf("give input statement:");
    getline(cin, inp);
    for(int i=0;i<inp.length();i++)
    {
        if(inp[i] == '>')
        {
            if(inp[i+1] == '=')
            {
                cout << ">= is a relational operator\n";
            }
            else
            {
                cout << "> is a relational operator\n";
            }
        }
        else if(inp[i] == '<')
        {
            if(inp[i+1] == '=')
            {
```

```

        cout << "<= is a relational operator\n";
    }
    else
    {
        cout << "< is a relational operator\n";
    }
}
else if(inp[i]=='!' && inp[i+1]=='=')
{
    cout << "!= is a relational operator\n";
}
else if(inp[i]=='=' && inp[i+1]=='=')
{
    cout << "== is a relational operator\n";
}
}
}


```

Input:

 "C:\Users\it\Desktop\Problem Solving\Contest 2\7.exe"

give input statement: int x; if(x>=10) print("ok");

Output:

 "C:\Users\it\Desktop\Problem Solving\Contest 2\7.exe"

give input statement: int x; if(x>=10) print("ok");
 >= is a relational operator

Process returned 0 (0x0) execution time : 35.713 s
 Press any key to continue.

Discussion: From this experiment we could be able to separate all relational operator from a statement.

Problem No: 08

Problem Name: Write a Program to Perform leftmost derivation for a given set of productions.

Objective: To Write a Program to Perform leftmost derivation for a given set of productions.

Algorithm:

- 1.Start
2. Read the terminal and non-terminal
3. Read the input production and start symbol
- 4.Read the input string
- 5.Compare start symbol with the left side of the production
- 6.If match found then check the right side of the production
- 7.Check if the letter of right side of the production is capital letter then check this letter with left side of the production.
- 8.If match then replace capital letter with the right side of the production and check right side of the production if it is capital then repeat step 6
- 9.compare the input string with the right side of the production of last production
- 10.If matched then print string parsed successfully
- 11.Otherwise print string cannot be parsed

Code :

```
#include<bits/stdc++.h>

using namespace std;

map<char, string>mp;

bool isTerminal(char ch){
    return ch>='A' && ch<='Z';
}

int main()
{
    cout<<"Enter number of productions: ";
```



```

int numberOfProduction;
cin>>numberOfProduction;
while(numberOfProduction--){
    char terminal;
    string production;
    cout<<"Enter terminal: ";
    cin>>terminal;
    cout<<"Enter production : ";
    cin>>production;
    mp[terminal] = production;
}
cout<<"Enter target production: ";
string target;
cin>>target;
string initialString = mp['S'];
for(int i=0;i<initialString.size();i++){
    if(!isTerminal(initialString[i]))continue;
    cout<<"Substituting for "<<initialString[i]<<" || ";
    string
firstHalf,SubstitutedString,RestofTarget,tmp=initialString;
    firstHalf = initialString.substr(0,i);
    SubstitutedString = mp[initialString[i]];
    RestofTarget = tmp.substr(i+1,tmp.size()-i);
    cout<<"First = "<<firstHalf<<" || mid = "<<SubstitutedString<<"
last = "<<RestofTarget<<endl;
    initialString = firstHalf+SubstitutedString+RestofTarget;
    cout<<"Current String "<<initialString<<"\n";
    i--;
}
target==initialString?cout<<"String Parsable\n":cout<<"String not
parsable\n";
return 0;
}

```

Input:

```

"C:\Users\it\Desktop\Problem Solving\Contest 2\9.exe"
Enter number of productions: 4
Enter terminal: S
Enter production : aBs
Enter terminal: A
Enter production : aB
Enter terminal: B
Enter production : eF
Enter terminal: F
Enter production : d
Enter target production: aed

```

Output:

```

"C:\Users\it\Desktop\Problem Solving\Contest 2\9.exe"
Enter number of productions: 4
Enter terminal: S
Enter production : aBs
Enter terminal: A
Enter production : aB
Enter terminal: B
Enter production : eF
Enter terminal: F
Enter production : d
Enter target production: aed
Substituting for B || First = a || mid = eF last = s
Current String aeFs
Substituting for F || First = ae || mid = d last = s
Current String aeds
String not parsable

Process returned 0 (0x0)   execution time : 48.953 s
Press any key to continue.

```

Discussion: From this experiment we could be able to perform leftmost derivation for a given set of productions.

Problem No: 09

Problem Name: Write a Program to simulate a string in a given DFA.

Objective: To Write a Program that simulate a string in a given DFA.

Algorithm:

- 1.Start
- 2.Check weather the input string can parsed;
- 3.Then print 'yes'.
- 4.Otherwise print 'no'.
- 5.End

Code :

```

/* The DFA is:
{W|W is a string of a and b ending withe the substring "bb".}
states,Q= {1,2,3}
start State,S=1;
Final state,F=3;
Input Symbols={a,b};
Transitions Are:
T( 1,a)=1;   T(1,b)=2;   T(2,a)=1;   T(2,b)=3;   T(3,a)=1;   T(3,b)=3;

*/
#include<bits/stdc++.h>
using namespace std;
int states(int s,char c)
{
    if(s==1)
    {
        if(c=='a')
            return 1;
        else
            return 2;
    }
    else if(s==2)
    {
        if(c=='a')
            return 1;
        else
            return 3;
    }
}

```

```

    }
    else if(s==3)
    {
        if(c=='a')
            return 1;
        else
            return 3;
    }

}


int main()
{
    int state,i,j,k,next;
    char inp[100];

    gets(inp);

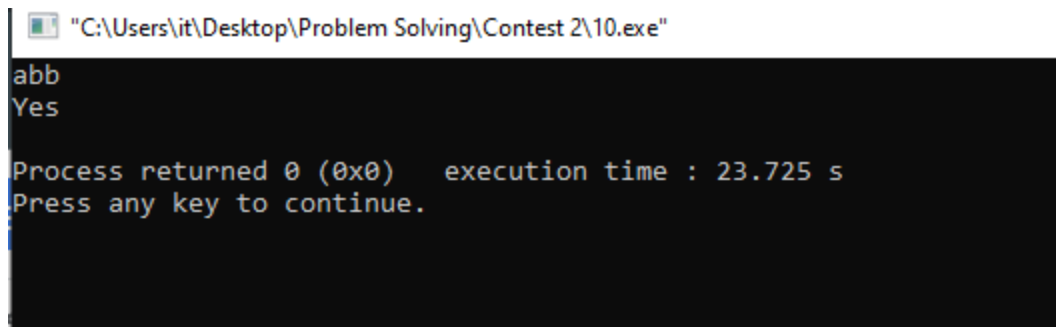
    i=0;
    state=1;
    while(inp[i]!='\0')
    {
        next=inp[i];
        state=states(state,next);
        i++;
    }
    if(state==3)
        printf("Yes\n");
    else
        printf("No\n");
    return 0;
}

```

Input:

 "C:\Users\it\Desktop\Problem Solving\Contest 2\10.exe"

abb

Output:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\10.exe"  
abb  
Yes  
  
Process returned 0 (0x0)   execution time : 23.725 s  
Press any key to continue.
```

Discussion: From this experiment we could be able to simulate a string in a given DFA.

Problem No: 10

Problem Name: Write a Program to Find first from a given set of production.

Objective: To Write a Program that Find first from a given set of production.

Algorithm:

- 1.Start
- 2.Read the input production
- 3.Read the element to get FIRST of it
- 4.Check weather the element is uppercase letter
- 5.If the element is not uppercase letter then store this element for the FIRST of the element
- 6.Otherwise check if the element is match with the left side of any production.
 - 7.Then check if first element of the right side of the production is '\$' then store this as the FIRST of the element.
 - 8.Else if check if first element of the right side of the production is lower case letter then store this first element as the FIRST of the element.
 9. Else goto step 4
- 10.End

Code :

```

#include<stdio.h>
#include<ctype.h>

void FIRST(char );
int count,n=0;
char prodn[10][10], first[10];

main()
{
    int i,choice;
    char c,ch;
    printf("How many productions ? :");
    scanf("%d",&count);
    printf("Enter %d productions epsilon= $ :\\n\\n",count);
    for(i=0; i<count; i++)
        scanf("%s%c",prodn[i],&ch);
    do
    {
        n=0;
        printf("Element :");
        scanf("%c",&c);
        FIRST(c);
        printf("\\n FIRST(%c)= { ",c);
        for(i=0; i<n; i++)
            printf("%c ",first[i]);
        printf("}\\n");

        printf("press 1 to continue : ");
        scanf("%d%c",&choice,&ch);
    }
    while(choice==1);
}

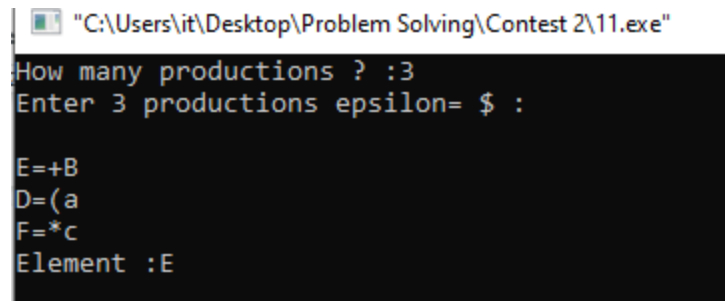
void FIRST(char c)
{
    int j;
    if(!(isupper(c)))
        first[n++]=c;
    for(j=0; j<count; j++)
    {
        if(prodn[j][0]==c)
        {
            if(prodn[j][2]=='$')
                first[n++]='$';

```

```

        else if (islower(prodn[j][2]))
            first[n++] = prodn[j][2];
        else
            FIRST(prodn[j][2]);
    }
}

```

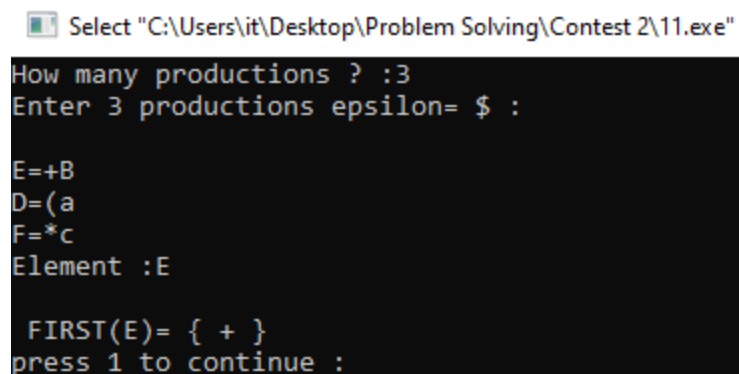
Input:


```

"C:\Users\it\Desktop\Problem Solving\Contest 2\11.exe"
How many productions ? :3
Enter 3 productions epsilon= $ :

E=+B
D=(a
F=*c
Element :E

```

Output:


```

Select "C:\Users\it\Desktop\Problem Solving\Contest 2\11.exe"
How many productions ? :3
Enter 3 productions epsilon= $ :

E=+B
D=(a
F=*c
Element :E

FIRST(E)= { + }
press 1 to continue :

```

Discussion: From this experiment we could be able to Find first from a given set of production

Problem No:11

Problem Name: Write a Program to given a set of instructions. Now generate three address code.

Objective: To Write a Program that generate three address code for given a set of instructions.

Algorithm:

1.Start

2.Read the input statement

For i=0 to i<length of the string

3.First perform the operation of multiplication and division

4.Check if it is the first operator then print left element of operator,operator and right side of the operator showing equal to a variable t0.

5.Otherwise if both left and right side of the operator is 0 then print previous variable of ti ,operator and next variable of ti showing equal to a variable ti.

6.Else if left side of the operator is 0 then print previous variable of t i,operator and right side of the operator showing equal to a variable ti.

7. Else if right side of the operator is 0 then print left side of the operator,operator and next variable of ti showing equal to a variable ti.

8.Else print left element of operator,operator and right side of the operator showing equal to a variable ti, i++;

9.For i=0;i<length of string

10. perform the operation of addition and subtraction.

11.Repeat the same process as step 4 to step 8 ,i++

12.End

Code :

```

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

int main()
{
    string str;
    printf("enter the statement:");
    cin >> str;

    int count =1;
    for(int i=0;i<str.length();i++)
    {
        if(str[i]=='*' || str[i]=='/')
        {

            if(count==1)
            {
                cout << "t" << count << "=" << str[i-1] << str[i] <<
str[i+1] << "\n";
            }
            else
            {
                if (str[i+1]=='0' && str[i-1]=='0')
                {

                    cout << "t" << count << "=" << "t" << count-1 <<
str[i] << "t" << count-2 << "\n";
                }
                else if(str[i-1]=='0')
                {
                    cout << "t" << count << "=" << "t" << count-1 <<
str[i] << str[i+1] << "\n";
                }
                else if(str[i+1]=='0')
                {
                    cout << "t" << count << "=" << "t" << count-1 <<
str[i] << str[i-1] << "\n";
                }

                else
                {

```

```

        cout << "t" << count << "=" << str[i-1] << str[i]
<< str[i+1] << "\n";
    }
    }
    count++;
    for(int j=i-1;j<=i+1;j++)
    {
        str[j] = '0';
    }
    i=0;
}

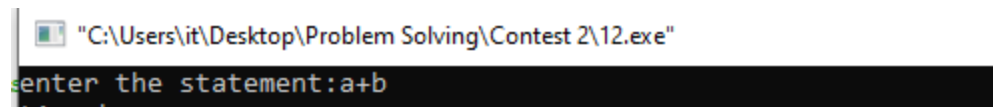
for(int i=0;i<str.length();i++)
{
    if(str[i]=='+' || str[i]=='-')
    {
        if(count==1)
        {
            cout << "t" << count << "=" << str[i-1] << str[i] <<
str[i+1] << "\n";
        }
        else
        {
            if (str[i+1]=='0' && str[i-1]=='0')
            {
                cout << "t" << count << "=" << "t" << count-1 <<
str[i] << "t" << count-2 << "\n";
            }
            else if(str[i-1]=='0')
            {
                cout << "t" << count << "=" << "t" << count-1 <<
str[i] << str[i+1] << "\n";
            }
            else if(str[i+1]=='0')
            {
                cout << "t" << count << "=" << "t" << count-1 <<
str[i] << str[i-1] << "\n";
            }
            else
            {
                cout << "t" << count << "=" << str[i-1] << str[i]
<< str[i+1] << "\n";
            }
        }
    }
}

```

```

        count++;
        for(int j=i-1;j<=i+1;j++)
        {
            str[j] = '0';
        }
        i=0;
    }
}
cout << str[0] << "=" << "t" << count-1;
}

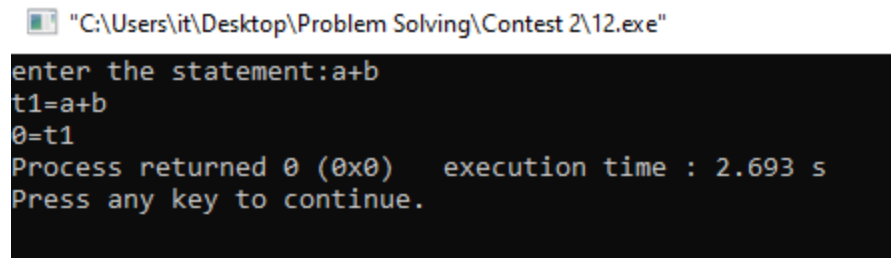
```

Input:


```

"C:\Users\it\Desktop\Problem Solving\Contest 2\12.exe"
enter the statement:a+b

```

Output:


```

"C:\Users\it\Desktop\Problem Solving\Contest 2\12.exe"
enter the statement:a+b
t1=a+b
0=t1
Process returned 0 (0x0)   execution time : 2.693 s
Press any key to continue.

```

Discussion: From this experiment we could be able to generate three address code for Given a set of instructions.

Problem No: 12

Problem Name: Write a Program to generate assembly language code for a given three address code.

Objective: To Write a Program that generate assembly language code for a given three address code.

Code :

```
#include<bits/stdc++.h>

using namespace std;

int main()
{
    while(true){
        cout<<"Input three address code : ";
        string s;
        getline(cin,s);
        string firstRegister = "";
        int i = 0;
        while(s[i]!='='){
            firstRegister+=s[i++];
        }
        vector<char>v;
        while(i<s.size()){
            v.push_back(s[i++]);
        }
        if(find(v.begin(),v.end(),'+')!=v.end()){
            cout<<"ADD "<<firstRegister<<" ";
            string op1 = "" , op2 = "";
```

```

    int i = firstRegister.length();
    while(s[i]!='+'){
        if(s[i]!='=')op1+=s[i];
        i++;
    }
    i++;
    while(i<s.size())op2+=s[i++];
    cout<<op1<<" , "<<op2<<"\n";
}

else if(find(v.begin(),v.end(),'*')!=v.end()){
    cout<<"MUL "<<firstRegister<<" ";
    string op1 = "" , op2 = "";
    int i = firstRegister.length();
    while(s[i]!='*'){
        if(s[i]!='=')op1+=s[i];
        i++;
    }
    i++;
    while(i<s.size())op2+=s[i++];
    cout<<op1<<" , "<<op2<<"\n";
}

else if(find(v.begin(),v.end(),'/')!=v.end()){
    cout<<"DIV "<<firstRegister<<" ";
    string op1 = "" , op2 = "";
    int i = firstRegister.length();
    while(s[i]!='/'){
        if(s[i]!='=')op1+=s[i];
        i++;
    }

```

```

    }

    i++;

    while(i<s.size())op2+=s[i++];

    cout<<op1<<" , "<<op2<<"\n";

}

else if(find(v.begin(),v.end(),'-')!=v.end()){

    cout<<"SUB "<<firstRegister<<" ";

    string op1 = "" , op2 = "";

    int i = firstRegister.length();

    while(s[i]!='-'){

        if(s[i]!='=')op1+=s[i];

        i++;

    }

    i++;

    while(i<s.size())op2+=s[i++];

    cout<<op1<<" , "<<op2<<"\n";

}

else{

    cout<<"MOV "<<firstRegister<<" , ";

    string op1 = "" , op2 = "";

    int i = firstRegister.length();

    while(i<s.size()){

        if(s[i]!='=')op1+=s[i];

        i++;

    }

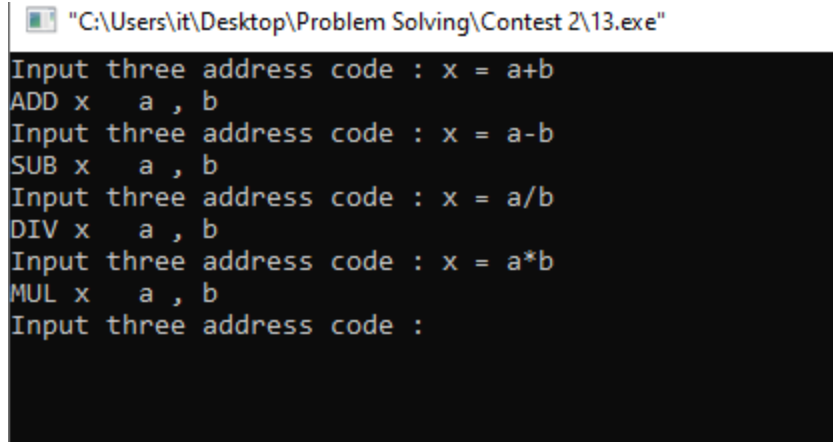
    cout<<op1<<"\n";

}

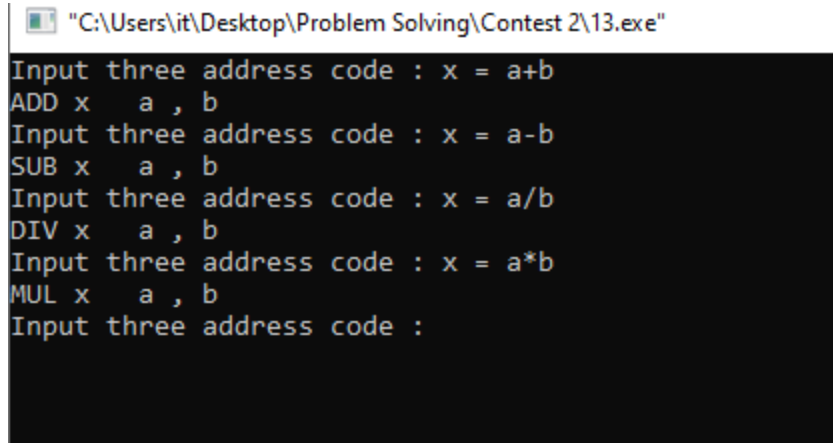
}

```

```
    return 0;  
}
```

Input:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\13.exe"  
Input three address code : x = a+b  
ADD x  a , b  
Input three address code : x = a-b  
SUB x  a , b  
Input three address code : x = a/b  
DIV x  a , b  
Input three address code : x = a*b  
MUL x  a , b  
Input three address code :
```

Output:

```
"C:\Users\it\Desktop\Problem Solving\Contest 2\13.exe"  
Input three address code : x = a+b  
ADD x  a , b  
Input three address code : x = a-b  
SUB x  a , b  
Input three address code : x = a/b  
DIV x  a , b  
Input three address code : x = a*b  
MUL x  a , b  
Input three address code :
```

Discussion: From this experiment we could be able to generate assembly language code for a given three address code.