PRACTICAL 1

AIM: Write a program to categorize each word as keyword or identifier

CODE:

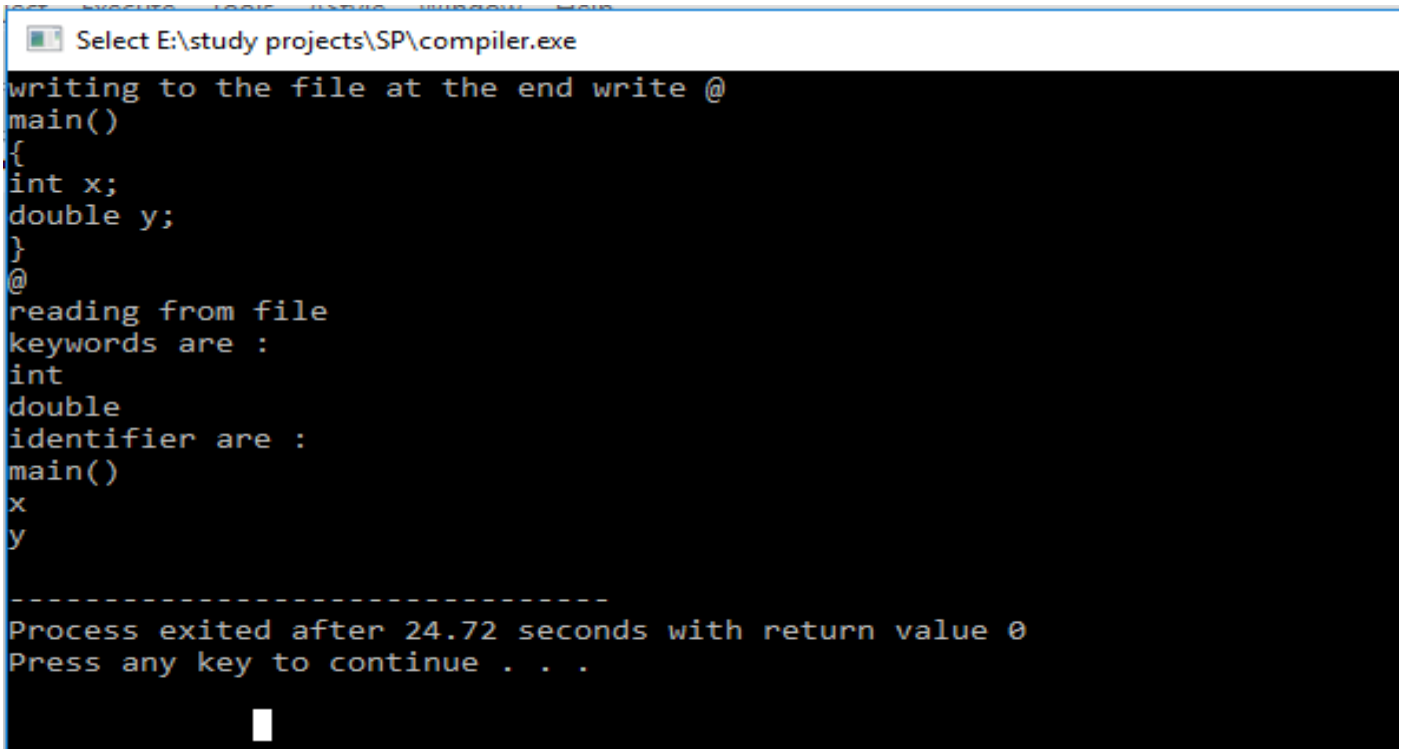
```
1: #include <iostream>
2: #include <fstream>
4: #include <string.h>
6: using namespace std;
8: int main()
9: {
10: ofstream mayank;
11: char data[50];
12: char char1;
13: string filedata;
14: string spaceless;
15: int i;
16: char keywords[5][10] ={"double", "int" , "void" , "return", "float"
17: char store[10][10];
18: char identifier[10][10];
22: mayank.open("cfile.c");
23: cout<<"writing to the file at the end write @ "<<endl;
25: while(data[0] != '@' )
26: {
27: cin.getline(data,100);
28: if(data[0] != '@')
29: mayank<<data<<endl;
30: }
31: mayank.close();
33: ifstream mayank1;
34: mayank1.open("cfile.c");
36: cout<<"reading from file"<<endl;
37: filedata=" ";
38: while(!mayank1.eof())
39: {
40: mayank1.getline(data,50);
42: filedata = filedata +" "+ data;
44: }
47: int flag;
```

```
48: for(i=2;i<filedata.length();i++)
49: {
50: if(filedata[i] == ' ')
51: { flag=i;
52: while(filedata[i] == ' ')
53: {
54: i++;
55: }
56: if(flag!=2)
57: {
58: char1=' ';
59: spaceless.append(sizeof(char1),char1);
60: }
61: char1=filedata[i];
62: spaceless.append(sizeof(char1),char1);
64: }
66: else{
67: char1=filedata[i];
68: spaceless.append(sizeof(char1),char1);
69: }
70: }
73: //comparisoion
74: string temp;
75: int x=1,j=0,k=0,flag2;
76: int identi=0;
79: for(i=0;i<spaceless.length()-1;i++)
80: {flag2=0;
81: temp.clear();
82:
83: if(i==0)
84: {
86: while(spaceless[i] != ' ') //copy word from one end to second
87: {
88: char1=spaceless[i];
89: temp.append(sizeof(char1),char1);
90: i++;
92: }
```

```
96: if(spaceless[i] == ' ') //comparisions
97: {
99: for(k=0;k<5;k++) //for keywords
100: {
101: x=temp.compare(keywords[k]);
103: if(x==0 && flag2==0)
104: {
105: flag2=1; // if comparisoion true
106: for(int z=0;z<temp.length();z++)
107: {
109: char1=temp[z];
110: store[j][z]=char1;
111: }
113: temp.clear();
114: j++; // for store array row change
116: }
117: }
119: if(flag2==0 && temp[0]!='{' && temp[0] !='}')
120: {
122: for(int z=0;z<temp.length() && temp[z]!=';' ;z++)
123: {
124: char1=temp[z];
125: identifier[identi][z]=char1;
126: }
128: temp.clear();
129: identi++;
131: }
134: }
136: }
140: else if(spaceless[i] !=' ')
141: {
142: while(spaceless[i] != ' ') //copy word from one end to 143: {
144: char1=spaceless[i];
145: temp.append(sizeof(char1),char1);
146: i++;
148: }
150: if(spaceless[i] == ' ') //comparisions
```

```
151: {
153: for(k=0;k<5;k++) //for keywords
154: {
155: x=temp.compare(keywords[k]);
157: if(x==0)
158: {
159: flag2=1; // if comparisoion true
161: for(int z=0; ((z<temp.length()) && (temp[z] !=';'
162: {
163: char1=temp[z];
165: store[j][z]=char1;
166: }
168: temp.clear();
169: j++; // for store array row change
171: }
173: }
175: if(flag2==0 && temp[0]!='{' && temp[0] !='}')
176: {
178: for(int z=0;((z<temp.length()) && (temp[z] !=';' ) );z
179: {
180: char1=temp[z];
181: identifier[identi][z]=char1;
182: }
184: temp.clear();
185: identi++;
188: }
190: }
193: }
195: }
198: cout<<"keywords are :"<<endl;
199: for(int z=0;z<j;z++)
200: {
201: cout<<store[z]<<endl;
202: }
204: cout<<"identifier are :"<<endl;
205: for(int z=0;z<identi;z++)
206: {
```

```
208: cout<<identifier[z]<<endl;
209: }
210: mayank1.close();
212: return 0;
213: }
```



PRACTICAL 2

AIM: Write a program check whether parentheses are balanced or not.

CODE:

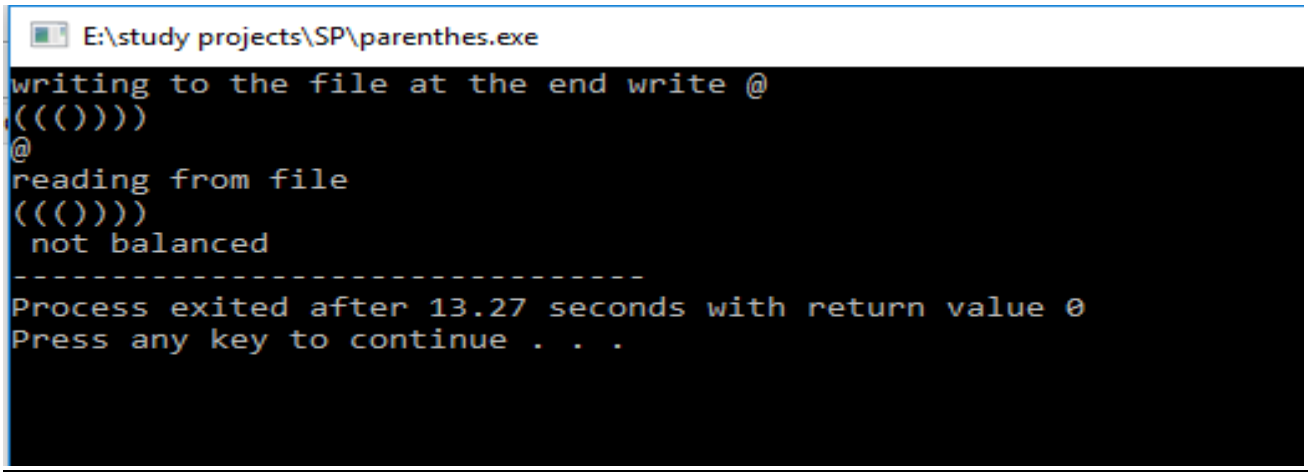
```
1: // Parenthesis
2: #include<iostream>
3: #include<string.h>
4: #include<fstream>

6: using namespace std;
7: char stack[100];
8: int TOP=-1;
10: class s{
11: public:
12: char pop()
13: {char temp;
14: if(TOP !=-1) { temp=stack[TOP]; TOP--;}
15: else
16: cout<<"stack is empty";
17: return temp;
18: }
20: char push(char x ) {
22: if(TOP <100 )
23: {TOP++;
24: stack[TOP]=x; } }
28: char peep()
29: {return stack[TOP];}  };
33: int main()
34: {
35: ofstream mayank;
36: char data[50];
37: char char1;
```

```
38: string filedata;
39: string spaceless;
40: int i;
41: s stack;
42: mayank.open("parenthes.txt");
43: cout<<"writing to the file at the end write @ "<<endl;
44: while(data[0] != '@' )
45: {
46: cin.getline(data,100);
47: if(data[0] != '@')
48: mayank<<data<<endl;
49: }
50: mayank.close();
51: ifstream mayank1;
52: mayank1.open("parenthes.txt");
53: cout<<"reading from file"<<endl;
54: filedata=" ";
55: while(!mayank1.eof())
56: {
57: mayank1.getline(data,50);
58: filedata = filedata +" "+ data;
59: }
60: //data saved in file data with extra space now i have to re
61: int flag;
62: for(i=2;i<filedata.length();i++)
63: {
64: if(filedata[i] == ' ')
65: { flag=i;
66: while(filedata[i] == ' ')
68: i++;
70: if(flag!=2)
71: {
72: char1=' ';
73: spaceless.append(sizeof(char1),char1);
74: }
```

```
75: char1=filedata[i];
76: spaceless.append(sizeof(char1),char1);
77: }
78: else{
79: char1=filedata[i];
80: spaceless.append(sizeof(char1),char1);
81: } }
83: cout<<spaceless;
84: for(i=0;i<spaceless.length()-1;i++) {
86: if(spaceless[i]=='('
      stack.push('(');
90: else if(spaceless[i]==')')
91: { if(stack.peep()=='('
92: stack.pop();
94: else
95: {stack.push(')');
96: cout<<"\n not balanced";
97: goto label;
98: } } }
101: if(TOP==-1)
102: cout<<"\n is balanced";
104: else
105: cout<<"\n not balanced";
107: label:
108: return 0;  }
```

OUTPUT:



PRACTICAL 3

AIM: Write a program to implement symbol table.

CODE:

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
#include<ctype.h>
FILE *fp;
char delim[18]={' ','\t','\n',',',';','(',')','[',']','{','}','#','+','-','*','/','%','=','!'};
char key[21][12]={"int","float","char","double","bool","void","extern","auto",
"bool","goto","static","class","struct","for","if","else","return","register","long","w
hile","do"};
char ctype[12];
char avoid[5][12]={"include","define","getch","printf","scanf"};
struct symtab
{ char id[20];
  char type[20];
}p[30];
int in=0;
void construct();
int isdelim(char);
void check(char[]);
int checkkey(char[]);
void showtable();
void main()
{
        char fname[12];
        clrscr();
        printf("\nEnter the filename : ");
        scanf("%s",fname);
        fp=fopen(fname,"r");
        if(fp==NULL)
        printf("\nThe file doesn't exist.");
        else{construct();
        showtable();}
        fclose(fp);
        getch();
```

```
}
void construct()
{char c,ch,token[12];
int f=0,j=0,kf=0;
strcpy(ctype,"NULL");
while(!feof(fp))
{c=getc(fp);
    if(c==';'||c=='(')
        {if(f==1)
        {token[j]='\0';j=0;f=0;
         kf=checkkey(token);
                    if(kf==0)
                        check(token);
                    }
                    strcpy(ctype,"NULL");
                }
                else if(c=='"')
                {
                    while((c=getc(fp))!='"');
                }
                else if(c=='<')
                {
                    while((c=getc(fp))!='>');
                }
                else if(isdelim(c))
                {
                    if(f==1)
                    {
                        token[j]='\0';j=0;f=0;
                        kf=checkkey(token);
                        if(kf==0)
                    check(token);}
                }
                else if(isalpha(c)||c=='_')
                {
                    token[j++]=c;
                    f=1;
                }
        }
}
```

```
int isdelim(char c)
{
	int i;
	for(i=0;i<18;i++)
	{
		if(c==delim[i])
			return 1;
	}
	return 0;
}
int checkkey(char t[])
{int i;
for(i=0;i<5;i++)
		if(strcmp(avoid[i],t)==0)
			return 1;
	for(i=0;i<21;i++)
		if(strcmp(key[i],t)==0)
		{strcpy(ctype,key[i]);
			return 1;}
	return 0;}
void check(char t[])
{
	int i;
	for(i=0;i<in;i++)
	{		if(((strcmp(t,p[i].id))==0)&&((strcmp(ctype,p[i].type))==0))
		{
			printf("\nRedeclaration for '%s'",t);
			return 0;
		}
		else
		if(((strcmp(t,p[i].id))==0)&&((strcmp(ctype,p[i].type))!=0)&&((strcm
		p(ctype,"NULL")!=0)))
		{
			printf("\nMultiple declaration for %s",t);
			return 1;
		}
	}
	if(strcmp(ctype,"NULL")==0)
	{for(i=0;i<in;i++)
		{if(strcmp(t,p[i].id)==0)
```

```
                        return;
                }
                printf("\n%s Undeclared ",t);
                return 0;
        }
        strcpy(p[in].id,t);
        strcpy(p[in].type,ctype);
        in++;
}
void showtable()
{
        int i;
        if(in==0)
        {printf("\nSymbol table is empty.");
                return;}
        printf("\nSymbol table");
        printf("\n------------");
        printf("\nIdentifier\tType\tAddress");
        printf("\n----------\t----\t-------");
        for(i=0;i<in;i++)
                printf("\n%s\t\t%s\t%u",p[i].id,p[i].type,&p[i]);
}
```

OUTPUT:

```
Enter the filename : Symtext.c

Symbol table
------------
Identifier      Type    Address
----------      ----    -------
main            void    1838
a               int     1878
b               int     1918
c               int     1958
m               float   1998
```

PRACTICAL 4

AIM: Write a program to find whether leftrecursion in program and if it is remove it.

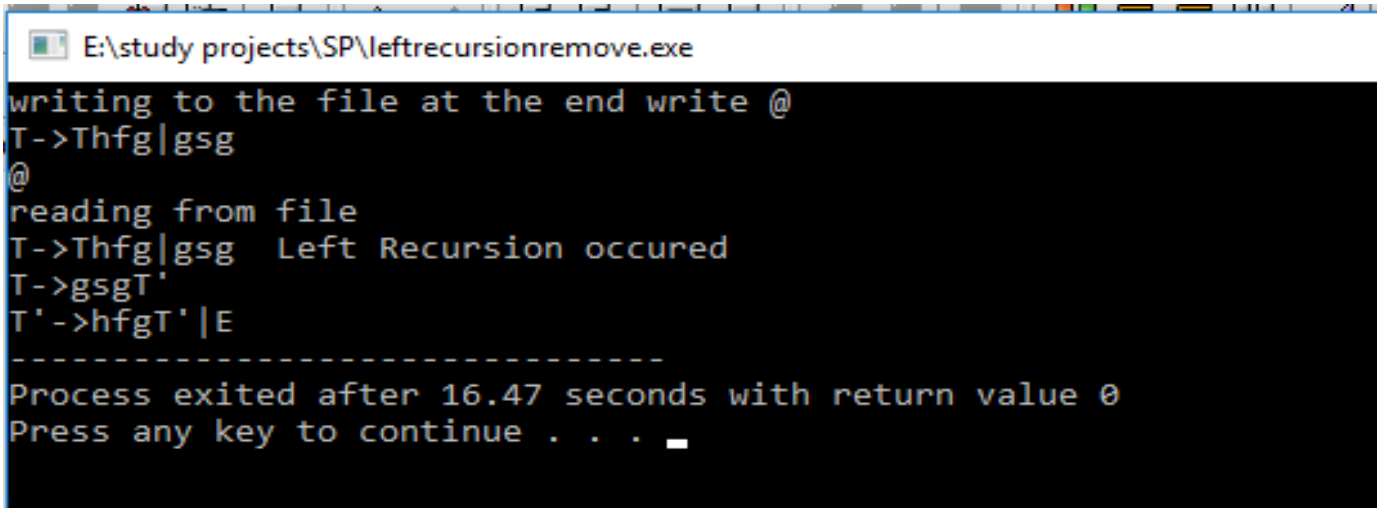
CODE:

```
1: //Left recursion Remove
2: #include<iostream>
3: #include<string.h>
4: #include<fstream>
6: using namespace std;
7: int main()
8: {
9: ofstream mayank;
10: int recursion,i;
11: char data[50],char1,d;
12: string filedata,spaceless,leftdata,rightdata;
14: mayank.open("string.txt");
15: cout<<"writing to the file at the end write @ "<<endl;
17: while(data[0] != '@' )
18: {
19: cin.getline(data,100);
20: if(data[0] != '@')
21: mayank<<data<<endl;
22: }
23: mayank.close();
```

```
25: ifstream mayank1;
26: mayank1.open("string.txt");
28: cout<<"reading from file"<<endl;
29: filedata=" ";
30: while(!mayank1.eof())
31: {
32: mayank1.getline(data,50);
34: filedata = filedata +" "+ data;
36: }
39: //data saved in file data with extra space now i have to re
40: int flag;
41: for(i=2;i<filedata.length();i++)
42: {
43: if(filedata[i] == ' ')
44: { flag=i;
45: while(filedata[i] == ' ')
46: {
47: i++;
48: }
49: if(flag!=2)  {
51: char1=' ';
52: spaceless.append(sizeof(char1),char1);  }
54: char1=filedata[i];
55: spaceless.append(sizeof(char1),char1); }
59: else{
60: char1=filedata[i];
61: spaceless.append(sizeof(char1),char1); }}
64: recursion=0;
65: cout<<spaceless;
66: for(i=0;i<spaceless.length()-1;i++)
67: { if(spaceless[i]=='-' && spaceless[i+1]=='>')
69: {  if(spaceless[i-1]==spaceless[i+2])
71: { d=spaceless[i-1];
72: recursion=1;
73: cout<<"Left Recursion occured";
74: break; } }  }
79: i=i+3;
```

```
80: if(recursion==1){
82: while(spaceless[i]!='|')
83: {
84: char1=spaceless[i];
85: leftdata.append(sizeof(char1),char1);
86: i++; }
88: i+=1;
90: while(i<spaceless.length()-2)
91: { char1=spaceless[i];
93: rightdata.append(sizeof(char1),char1);
94: i++; }
96: cout<<endl;
97: cout<<d<<"->"<<rightdata<<d<<"'"<<endl;
98: cout<<d<<"'"<<"->"<<leftdata<<d<<"'"<<"|E";  } }
```

OUTPUT:

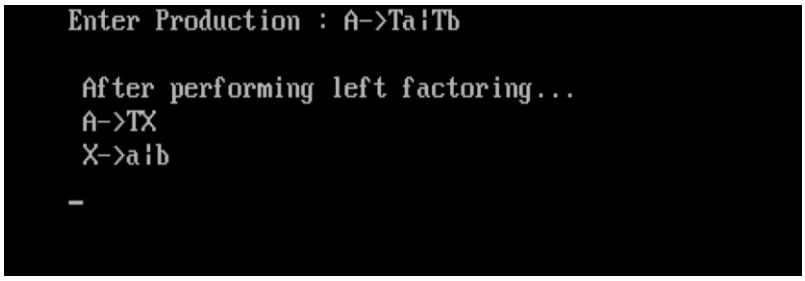


PRACTICAL 5

AIM: Write a program to perform left factoring on a grammar.

CODE:

```
#include<stdio.h>
#include<string.h>
void main()
{
char gram[20],part1[20],part2[20],modifiedGram[20],newGram[20],
tempGram[20];
      int i,j=0,k=0,pos;
      printf("Enter Production : A->");
      gets(gram);
     for(i=0;gram[i]!='|';i++,j++)
            part1[j]=gram[i];
      part1[j]='\0';
      for(j=++i,i=0;gram[j]!='\0';j++,i++)
           part2[i]=gram[j];
    part2[i]='\0';
      for(i=0;i<strlen(part1)||i<strlen(part2);i++){
 if(part1[i]==part2[i])
  {modifiedGram[k]=part1[i];  k++;   pos=i+1;}}
      for(i=pos,j=0;part1[i]!='\0';i++,j++)
     {newGram[j]=part1[i];}
      newGram[j++]='|';
      for(i=pos;part2[i]!='\0';i++,j++){
      newGram[j]=part2[i];}
      modifiedGram[k]='X';
      modifiedGram[++k]='\0';
      newGram[j]='\0';
      printf("\n After performing left factoring...");
      printf("\n A->%s",modifiedGram);
      printf("\n X->%s\n",newGram);
      getch();
 }
```



PRACTICAL 6

AIM: Write a program to convert infix to postfix

CODE:

```
import java.io.*;
	class Stack
	  {
	  char a[]=new char[100];
	  int top=-1;
	  void push(char c)
	  {
	  try
	  {
	  a[++top]= c;
	  }
	  catch(StringIndexOutOfBoundsException e)
	  {
	  System.out.println("Stack full , no room to push , size=100");
	  System.exit(0);
	  }
	  }
	char pop()
	  {
	  return a[top--];
	  }
```

```
  boolean isEmpty()
  {
  return (top==-1)?true:false;
  }
 char peek()
  {
  return a[top];
  }
 }
 public class JavaApplication43 {
static Stack operators = new Stack();
 public static void main(String[] args) throws IOException {
   String infix;
  BufferedReader keyboard = new BufferedReader (new
  InputStreamReader(System.in));
        System.out.print("\nEnter the algebraic expression in infix: ");
        infix = keyboard.readLine();
        System.out.println("The expression in postfix is:" + toPostfix(infix));
  }

  private static String toPostfix(String infix)
  {
        char symbol;
        String postfix = "";
            for(int i=0;i<infix.length();++i)
```

```
{
        symbol = infix.charAt(i);
        //if it's an operand, add it to the string
      if (Character.isLetter(symbol))
              postfix = postfix + symbol;
        else if (symbol=='(')
        //push (
        {


              operators.push(symbol);
        }
        else if (symbol==')')
        //push everything back to (
        {
              while (operators.peek() != '(')
              {
                    postfix = postfix + operators.pop();
              }
              operators.pop();              //remove '('
        }
        else
        //print operators occurring before it that have greater precedence
        {
           while (!operators.isEmpty() && !(operators.peek()=='(') &&
prec(symbol) <= prec(operators.peek()))
                    postfix = postfix + operators.pop();
```

```
                    operators.push(symbol);
                }
            }
            while (!operators.isEmpty())
                postfix = postfix + operators.pop();
            return postfix;
        }

        static int prec(char x)
        {
            if (x == '+' || x == '-')
                return 1;
            if (x == '*' || x == '/' || x == '%')
                return 2;
            return 0;
        }

}
```

OUTPUT:

Enter the algebraic expression in infix: a+b+c

The expression in postfix is: ab+c+

PRACTICAL 7

AIM:Write a program to build triplet for given equation

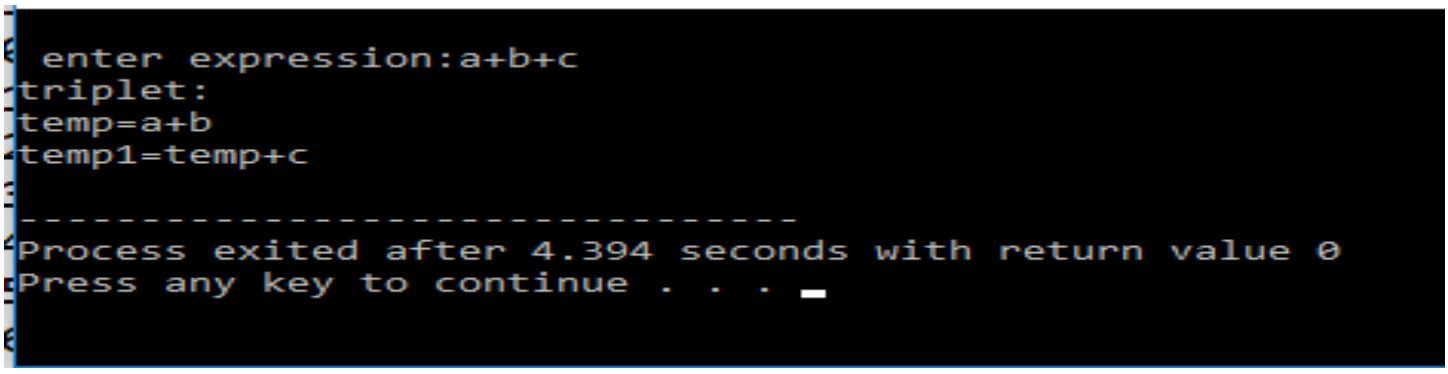
CODE:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void pm();
void add();
void div();
int i,j,l,address=100;
char ex[10],exp[10],exp1[10],exp2[10],id1[5],op[5],id2[5];
int main()
{
        printf("\n enter expression:");
        scanf("%s",ex);
        strcpy(exp,ex);
        l=strlen(exp);
        exp1[0]='\0';

        for(i=0;i<l;i++)
        {
                if(exp[i] == '+' || exp[i] == '-')
                {if(exp[i+2] == '/' || exp[i+2] == '*' )
                {pm();break;}
                else{add();
```

```
                break;}}
else if(exp[i] == '/' || exp[i] == '*')
{div();
break;
}}}
void pm()
{strrev(exp);
j=l-i-1;
strncat(exp1,exp,j);
strrev(exp1);
printf("triplet:\ntemp=%s\ntemp1=%c%ctemp\n",exp1,exp[j+1],exp[j]);}
void div()
{strncat(exp1,exp,i+2);
printf("triplet:\ntemp=%s\ntemp1=temp%c%c\n",exp1,exp[i+2],exp[i+3]);}
void add()
{strncat(exp1,exp,i+2);
printf("triplet:\ntemp=%s\ntemp1=temp%c%c\n",exp1,exp[i+2],exp[i+3]);}
```

OUTPUT:



PRACTICAL 8

AIM : Write a C program to input an assembly program and prepare mnemonic and symbol table.

CODE:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
FILE *f1,*f2,*f3,*f4;
int lc,sa,l,op1,o,len;
char m1[20],la[20],op[20],otp[20],s,s1;
clrscr();

f1=fopen("input.txt","r");
f3=fopen("symtab.txt","w");
fscanf(f1,"%s %s %d",la,m1,&op1);
if(strcmp(m1,"START")==0)
{
sa=op1;
lc=sa;
printf("\t%s\t%s\t%d\n",la,m1,op1);
 }
else
lc=0;
fscanf(f1,"%s %s",la,m1);
while(!feof(f1))
{
fscanf(f1,"%s",op);
printf("\n%d\t%s\t%s\t%s\n",lc,la,m1,op);
if(strcmp(la,"-")!=0)
 {
fprintf(f3,"\n%d\t%s\n",lc,la);
 }
 f2=fopen("optab.txt","r");
fscanf(f2,"%s %d",otp,&o);
while(!feof(f2))
 {
```

```
if(strcmp(m1,otp)==0)
  {
lc=lc+3;
break;
  }
fscanf(f2,"%s %d",otp,&o);
  }
fclose(f2);
if(strcmp(m1,"WORD")==0)

    {
lc=lc+3;
   }
else if(strcmp(m1,"RESW")==0)
   {
    op1=atoi(op);
lc=lc+(3*op1);
    }
else if(strcmp(m1,"BYTE")==0)
    {
if(op[0]=='X')
lc=lc+1;
else
      {
len=strlen(op)-2;
lc=lc+len;}
    }
else if(strcmp(m1,"RESB")==0)
    {
     op1=atoi(op);
lc=lc+op1;
     }
fscanf(f1,"%s%s",la,m1);
    }
if(strcmp(m1,"END")==0)
    {
printf("Program length =\n%d",lc-sa);
    }
fclose(f1);
fclose(f3);
```

```
printf("\n----Symbol table----");
   f3=fopen("symtab.txt","r");
do{
   s=getc(f3);
printf("%c",s);
}while(s!=EOF);
fclose(f3);
printf("\n----Mnemonic table----\n");
   f2=fopen("optab.txt","r");
do{
   s1=getc(f2);
printf("%c",s1);
}while(s1!=EOF);
fclose(f2);
getch();
   }
```

OUTPUT:

PRACTICAL 9

AIM: Write a program to find unused variables.

CODE:

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class JavaApplication45 {
	public static void main(String args[]) throws IOException {
		int i, j;
		File f = new File("C:\\file.txt");
		FileReader fr = new FileReader(f);
		BufferedReader b = new BufferedReader(fr);
		String temp = "";
		String line;
		while ((line = b.readLine()) != null) {
			temp = temp.concat(line).concat("\n");
		}
		System.out.println(temp);
		String p[] = temp.split("\n");
		String p1[] = p[2].split(" ");
		String p2[] = p1[1].split("[\\,\\;]");
		String p3[] = p[3].split("[\\=\\+\\;]");
		for (i = 0; i < p2.length; i++)
		{
			System.out.println(p2[i]);
		}
		for (i = 0; i < p3.length; i++)
		{
			System.out.println(p3[i]);
		}


		int flag;
		for (i = 0; i < p2.length; i++) {
			flag = 0;
			for (j = 0; j < p3.length; j++) {
```

```
                        if (p2[i].equals(p3[j])) {
                            flag = 1;
                            break;
                        }

                    }
                    if (flag == 0)
                        System.out.println(p2[i] + " is not used");
                }
            }
}
```

OUTPUT:

```
int main
{
int x,y,z;
y=5;
}
```

```
x
y
z
y
5
x is not used
z is not used
```

PRACTICAL 10

AIM: Write a program to check following syntax error.

1. Semicolon missing
2. Operator missing
3. Variable not declared.

CODE:

```
import java.util.*;
public class Test
{
        public static void main(String args[]){
String input=”void main()\n{\nint a,b,c;\nprintf(\”hello world”\)\na+b\n}”;
System.out.println(input+”\n”);
For(int i=0;i<sarr.length;i++){
 If(!(sarr[i].contains(“()”))){
If(sarr[i].equals(“{“) ||sarr[i].equals(“}”))
{
}
else
{
 If(!sarr[i].rndWith(“;”)))
{
        System.out.println(“Semicolon missing error “);
} } }
int flag=1;
String subinput = sarr[2].substring(4,sarr[2].indexOf(“;”));
```

```
String[] subarr = subinput.split(“,”);
String[] s1arr = sarr[4].split(“=”);
for(int i=0;i<subarr.length();i++)
{
        for(int j=0;j<s1arr.length();j++)
        {
          If(!(subarr[i].equals(s1arr[j]))){ flag=i;}
}
}if(flag!=-1){
System.out.println(“Operator error”);
}
}
```

OUTPUT:

```
 Void main()
{
int a,b,c;
printf(“hello world”)
a+b
}
Semicolon missing error
Semicolon missing error
```