**CD Practical 13:**

**Write a C program to implement operator precedence parsing**

#include <iostream.h>

#include <conio.h>

#include <stdlib.h>

struct node

{

char symbol;

struct node \*left;

struct node \*right;

};

typedefstruct node node;

struct stack

{

char op;

node \*op\_pointer;

};

typedefstruct stack stack;

int TOS=-1;

stack s[10];

/\* This is the operator precedence table stored as it is in the

form of matrix and operators are assigned values as follows:

NULL = 0, Equal (=) = 1, less than (<) = 2, greater than (>) = 3. \*/

// +,\*,(,),<>,-,/,^

int m[8][8]={ {3,2,2,3,3,3,2,2}, //+

{3,3,2,3,3,3,2,2}, //\*

{2,2,2,1,0,2,2,2}, //(

{3,3,0,3,3,3,3,3}, //)

{2,2,2,0,1,2,2,2}, //><

{3,2,2,3,3,3,2,2}, //-

{3,3,3,3,3,3,3,2}, ///

{3,3,3,3,3,3,3,2}}; //^

void main()

{

clrscr();

int comp(char );

void push(char);

void pop();

void display(node \*);

node \*temp;

charstr[20];

int i=0;

cout<< "Enter the string : ";

cin>>str;

push('<');

i++;

while(str[i]!='\0')

{

if((str[i]>='a'&&str[i]<='z'))

{

temp = new node;

temp->symbol=str[i];

temp->left=NULL;

temp->right=NULL;

s[TOS].op\_pointer = temp;

}

else

{

intp\_index,q\_index;

p\_index = comp(s[TOS].op);

q\_index = comp(str[i]);

while(m[p\_index][q\_index]==3)

{

temp = new node;

temp->symbol=s[TOS].op;

temp->left=s[TOS-1].op\_pointer;

temp->right=s[TOS].op\_pointer;

pop();

s[TOS].op\_pointer=temp;

p\_index = comp(s[TOS].op);

q\_index = comp(str[i]);

}

if(m[p\_index][q\_index]==2)

{

push(str[i]);

}

if(m[p\_index][q\_index]==1)

{

if(comp(s[TOS].op)==4)

break;

if(comp(s[TOS].op)==2)

{

temp=s[TOS].op\_pointer;

pop();

s[TOS].op\_pointer=temp;

}

}

if(m[p\_index][q\_index]==0)

{

cout<< "Invalid string.";

getch();

exit(1);

}

}

i++;

}

display(s[TOS].op\_pointer);

getch();

}

// function to get the index of the operator that comes in the TOS

// and also for the current operator that come in the string.

int comp(char x)

{

int y;

switch(x)

{

case '+':y=0;

break;

case '\*':y=1;

break;

case '(':y=2;

break;

case ')':y=3;

break;

case '>':

case '<':y=4;

break;

case '-':y=5;

break;

case '/':y=6;

break;

case '^':y=7;

break;

}

return y;

}

void push(char x)

{

TOS++;

s[TOS].op=x;

}

void pop()

{

TOS--;

}

void display(node \*s)

{

if(s==NULL)

return;

if(s->left!=NULL)

display(s->left);

if(s->right!=NULL)

display(s->right);

cout<< s->symbol;

}

.

**Program output:**

