**LAB ASSIGNMENT #8**

**STATEMENT:** WRITE A PROGRAM TO CONVERT GIVEN INFIX EXPRESSION INTO POSTFIX EXPRESSION USING STACKS:

1. OPERATOR, PARENTHESIS
2. OPERANDS

**SOURCE CODE:**

#include<stdio.h>

#include<conio.h>

#include<math.h>

#include<string.h>

#include<ctype.h>

int precedency(char);

void main()

{

clrscr();

int i,otos=-1,ptos=-1,l, l1;

char infix[100],poststack[100],opstack[100];

printf("Enter a valid infix\n");

gets(infix);

l=strlen(infix);

l1=l;

for(i=0;i<=l-1;i++)

{

if(infix[i]=='(')

{

opstack[++otos]=infix[i];

l1++;

}

else if(isalpha(infix[i]))

{

poststack[++ptos]=infix[i];

}

else if (infix[i]==')')

{

l1++;

while(opstack[otos]!='(')

{

poststack[++ptos]=opstack[otos];

otos--;

}

otos--;

}

else //operators

{

if(precedency(opstack[otos])>precedency(infix[i]))

{

poststack[++ptos]=opstack[otos--];

opstack[++otos]=infix[i];

}

opstack[++otos]=infix[i];

}

}

while(otos!=-1)

{

poststack[++ptos]=opstack[otos];

otos--;

}

/\*\*\*\*\*\*\*\*for displaying\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

for(i=0;i<l1;i++)

{

printf("%c",poststack[i]);

}

getch();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*precedency function\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int precedency(char ch)

{

switch(ch)

{

case '$':

return(4);

// break;

case'\*':

case'/':

return(3);

// break;

case'+':

case'-':

return(2);

// break;

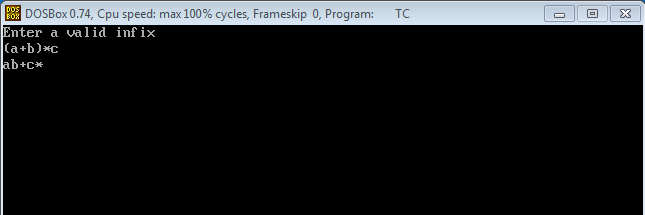
default:

return(1);

}

}

**OUTPUT:**

****

**CONCLUSION:**

Hence, the program was successful, and given infix expression was converted into postfix expression using the stacks: operator, parenthesis and operands.