EX.NO:4B CONVERSION OF INFIX EXPRESSION TO POSTFIX NOTATION

AIM:

To write program in C to convert given infix expression to postfix notation

ALGORITHM:

1: Get an infix expression.

2: Scan the expression from left to right.

3: If any operands come display it.

4: If the incoming symbol in a operator and has more priority then the symbol into the stack. 5: If the incoming operator has less priority than the stack symbol then copy the symbol at the top of the stack and then print until the condition becomes false and push the following operator on the stack.

6: If the symbol is ')' then copy operators from top of the stack. Deletion opening parenthesis is from top of the stack.

7: Stop the process.

DESCRIPTION:

Infix expression: The expression of the form a op b. When an operator is in-between every pair of operands.
Postfix expression: The expression of the form a b op. When an operator is followed for every pair of operands.

PROGRAM:

```
#include<std
io.h> char
stack[20];
int top = -
1; void
push(char x)
{
    stack[++top] = x;
}   char
pop() {
if(top == -
1)
```

```
return -1;
else
        return stack[top--];
}
int priority(char x)
{
    if(x == '(')
return 0;     if(x ==
'+' || x == '-')
return 1;     if(x ==
'*' || x == '/')
```



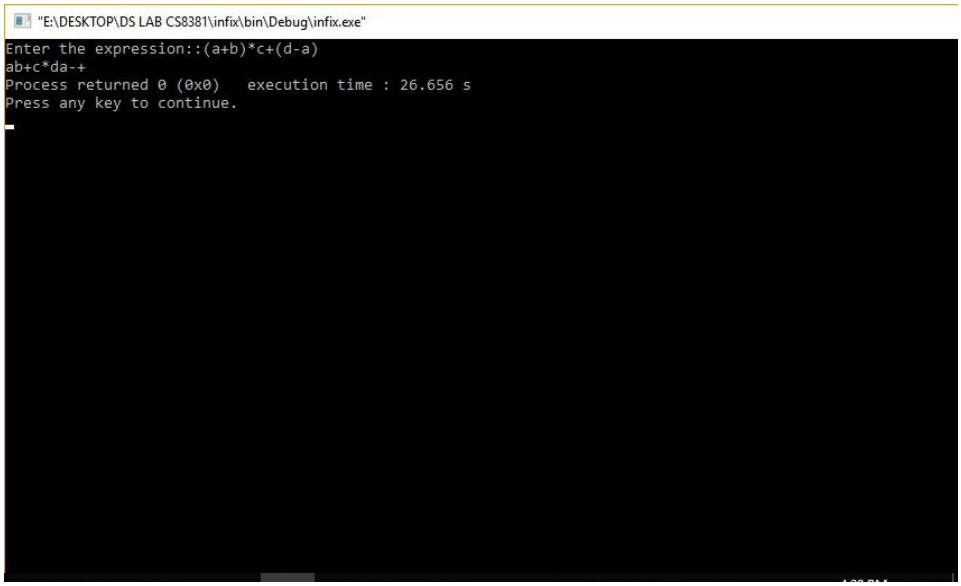
```
return 2; }
main() {
char
exp[20];
char *e, x;
    printf("Enter the expression
:: ");     scanf("%s",exp);     e
= exp;     while(*e != '\0')
    {

if(isalnum(*e))
printf("%c",*e);
else if(*e == '(')
push(*e);         else
if(*e == ')')
        {
            while((x = pop()) !=
'(')                  printf("%c",
x);

}
else
{
            while(priority(stack[top]) >=
priority(*e))                 printf("%c",pop());
push(*e);

}
e++;
}
    while(top != -1)
    {
        printf("%c",pop());
    }
}
```

OUTPUT



RESULT:

Thus the program in C to convert given infix expression to postfix notation