## LAB 3 - INAx to Postax

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression should consists of single character operands and the binary operators + (plus), -(minus), \*(multiply) and / (divide).

#include Estdio.n.) #include Lstring.h > int A(char symbol)

switch (syribol)

care (+':

case (-': rehm 2;

case (\*1:

case //: return 4;

case 'N':

case 1 21: rebry 5;

case (c's rehmo;

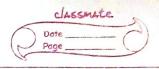
case '#' rehrn - L;

default : rehrn 8;

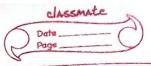
int B (char symbol)

switch (symbol)

case (+1: case 1-1: return 1;



```
case 1 x1:
        case '/': rehrn 3;
        case 's'
        case 'k': rebrn 6;
         case 'C': rehrn 9;
        case ')': Tehra O;
        default : rehrn 7;
void infix postfix (char infix[], char postfix[])
       int hop, i, i;
       char st30], symbol;
        S[++hop] = '#';
       for (i=0; i / strlen (infix); i++)
        symbol = infix [:];
wile (A(s [top]) > B (symbol))
         postfix[j] = sttop - - ];
        3 3++9
     if (A(S[DD])! = B (symbol))
      St++ top ] = Symbol;
      10p -- 9
```



while (strop]!=\frac{1}{2};

postfix(j++) = strop--];

postfix(j] = 100;

boid main()

char infix [25];

char infix [25];

pridf("Enter one infix expression: ");

sant ("0/8", 9nlix);

print ("he postfix expression is: ");

print ("he postfix expression is: ");

,

Capt Jak

nor 12= Lilailt

Cloumus d= 1C1

governé à Co

7