**AIM: EVALUATION OF INFLIX,PREFIX,POSTFIX EXPRESSION**

**CODE:**

//C Program to convert a given infix expression to postfix expression and evaluate

#define SIZE 50 /\* Size of Stack \*/

#include <ctype.h>

#include <stdio.h>

char s[SIZE];

int top = -1; /\* Global declarations \*/

/\* Function to remove spaces from given string \*/

void RemoveSpaces(char\* source) {

char\* i = source;

char\* j = source;

while(\*j != 0) {

\*i = \*j++;

if(\*i != ' ')

i++;

}

\*i = 0;

}

/\* Function for PUSH operation \*/

void push(char elem) {

s[++top] = elem;

}

/\* Function for POP operation \*/

char pop() {

return (s[top--]);

}

/\* Function for precedence \*/

int pr(char elem) {

switch (elem) {

case '#':

return 0;

case '(':

return 1;

case '+':

case '-':

return 2;

case '\*':

case '/':

return 3;

}

}

/\*

\* Function to convert from infix to postfix expression

\*/

void infix\_to\_postfix(char \*infix, char \*postfix) {

char ch, elem;

int i = 0, k = 0;

RemoveSpaces(infix);

push('#');

while ((ch = infix[i++]) != '\n') {

if (ch == '(')

push(ch);

else if (isalnum(ch))

postfix[k++] = ch;

else if (ch == ')') {

while (s[top] != '(')

postfix[k++] = pop();

elem = pop(); /\* Remove ( \*/

} else { /\* Operator \*/

while (pr(s[top]) >= pr(ch))

postfix[k++] = pop();

push(ch);

}

}

while (s[top] != '#') /\* Pop from stack till empty \*/

postfix[k++] = pop();

postfix[k] = 0; /\* Make postfix as valid string \*/

}

/\*

\* Function to evaluate a postfix expression

\*/

int eval\_postfix(char \*postfix) {

char ch;

int i = 0, op1, op2;

while((ch = postfix[i++]) != 0) {

if(isdigit(ch))

push(ch-'0'); /\* Push the operand \*/

else { /\* Operator,pop two operands \*/

op2 = pop();

op1 = pop();

switch(ch) {

case '+' : push(op1+op2);

break;

case '-' : push(op1-op2);

break;

case '\*' : push(op1\*op2);

break;

case '/' : push(op1/op2);

break;

}

}

}

return s[top];

}

void main() { /\* Main Program \*/

char infx[50], pofx[50];

printf("\nInput the infix expression: ");

fgets(infx, 50, stdin);

infix\_to\_postfix(infx, pofx);

printf("\nGiven Infix Expression: %sPostfix Expression: %s", infx, pofx);

top = -1;

printf("\nResult of evaluation of postfix expression : %d", eval\_postfix(pofx));

}

**OUTPUT:**

