Program :-

```
#include <stdio.h>
#include <string.h>
# include <ctype.h>
# define N 100.
char stack[N];
int top = -1;
int prec (char a) {
    if (a == '^') {
        return 3;
    }
    if (a == '*' || a == '/') {
        return 2;
    }
    if (a == '+' || a == '-') {
        return 1;
    }.
    return 0;
}.
int isRightAssociative (char op) {
    return op == '^';
}
Void push (char x) {
    if (top < N-1) {
        stack [++top] = x;
}.  }
```

```
char p
    if
    e
}.
char
    if
    e
}.
Void i
```

```
char pop() {
    if (top == -1) {
        return '\0';
    else {
        return stack[top--];
}.
char peek() {
    if (top == -1) {
        return '\0';
    else {
        return stack[top];
}.
Void infixtopostfix (char exp[], char res[]) {
    int j=0;
    for(int i=0; i< strlen(exp); i++) {
        char p= exp[i];
        if (isalnum(p)){
            res[j++]=p;
        }
        else if (p==')') {
            while (top != -1 && peek() != '(') {
                res[j++] = pop();
            }.
            if (top != -1) {
                pop();
            }.
            else {
                while (top != -1 && peek() != '(' &&
                    prec(peek()) > prec(p) ||
                prec(peek()) == prec(p) &&
                !isRightAssociative(p))) {
```

```
        res[j++] = pop();
    }
    push(p);
  }
}
while (top != -1) {
    res[j++] = pop();
  }
  res[j] = '\0';
}
int main() {
  char exp[N];
  char res[N];
  printf("Enter the expression : \n");
  scanf("%s", exp);
  infix to postfix(exp, res);
  printf("The postfix is : %s\n", res);
  return 0;
}
```

Output :-

Enter the expression :

(A+B)*C+(D^M)+(E-F).

The postfix is : AB+C*DM^+EF-+.

14/12/22