**Program:**

#include <limits.h>

#include <stdio.h>

#include <stdlib.h>

#define MAX 20

char stk[20];

int top = -1;

int isEmpty()

{

    return top == -1;

}

int isFull()

{

    return top == MAX - 1;

}

char peek()

{

    return stk[top];

}

char pop()

{

    if (isEmpty())

        return -1;

    char ch = stk[top];

    top--;

    return (ch);

}

void push(char oper)

{

    if (isFull())

        printf("Stack Full!!!!");

    else

    {

        top++;

        stk[top] = oper;

    }

}

int checkIfOperand(char ch)

{

    return (ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z');

}

int precedence(char ch)

{

    switch (ch)

    {

    case '+':

    case '-':

        return 1;

    case '\*':

    case '/':

        return 2;

    case '^':

        return 3;

    }

    return -1;

}

int covertInfixToPostfix(char \*expression)

{

    int i, j;

    for (i = 0, j = -1; expression[i]; ++i)

    {

        if (checkIfOperand(expression[i]))

            expression[++j] = expression[i];

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

            push(expression[i]);

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

        {

            while (!isEmpty() && peek() != '(')

                expression[++j] = pop();

            if (!isEmpty() && peek() != '(')

                return -1;

            else

                pop();

        }

        else

        {

            while (!isEmpty() && precedence(expression[i]) <= precedence(peek()))

                expression[++j] = pop();

            push(expression[i]);

        }

    }

    while (!isEmpty())

        expression[++j] = pop();

    expression[++j] = '\0';

    printf("%s", expression);

}

int main()

{

    char expression[] = "(a\*b)+(c\*d)";

    covertInfixToPostfix(expression);

    return 0;

}

**Output screenshots:**

