**Write a C program to perform infix to postfix conversion using stack.**

**Algorithm:**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define MAX\_SIZE 100

int isOperator(char ch) {

return (ch == '+' || ch == '-' || ch == '\*' || ch == '/');

}

int precedence(char ch) {

if (ch == '+' || ch == '-')

return 1;

else if (ch == '\*' || ch == '/')

return 2;

else

return 0;

}

void infixToPostfix(char \*infix, char \*postfix) {

int i = 0, j = 0;

char stack[MAX\_SIZE];

int top = -1;

while (infix[i] != '\0') {

if (isalnum(infix[i]))

postfix[j++] = infix[i++];

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

stack[++top] = infix[i++];

else if (infix[i] == ')') {

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

postfix[j++] = stack[top--];

top--;

i++;

} else if (isOperator(infix[i])) {

while (top != -1 && precedence(stack[top]) >= precedence(infix[i]))

postfix[j++] = stack[top--];

stack[++top] = infix[i++];

}

}

while (top != -1)

postfix[j++] = stack[top--];

postfix[j] = '\0';

}

int main() {

char infix[MAX\_SIZE], postfix[MAX\_SIZE];

printf("Enter infix expression: ");

scanf("%s", infix);

infixToPostfix(infix, postfix);

printf("Postfix expression: %s\n", postfix);

return 0;

}

**OUTPUT:**

**Enter infix expression: a+b\*c**

**Postfix expression: abc\*+**