**C PROGRAM FOR INFIX TO POSTFIX NOTATIONS**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_SIZE 100

int isOperator(char c) {

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

}

int getPrecedence(char op) {

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

return 1;

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

return 2;

else

return 0;

}

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

char stack[MAX\_SIZE];

int top = -1;

int i, j;

for (i = 0, j = 0; infix[i] != '\0'; i++) {

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

stack[++top] = '(';

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

while (top != -1 && stack[top] != '(') {

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

}

top--;

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

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

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

}

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

} else {

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

}

}

while (top != -1) {

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

}

postfix[j] = '\0';

}

int main() {

char infix[MAX\_SIZE];

char postfix[MAX\_SIZE];

printf("Enter the infix expression: ");

fgets(infix, MAX\_SIZE, stdin);

infix[strlen(infix) - 1] = '\0';

infixToPostfix(infix, postfix);

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

return 0;

}