***NAME : JANHAVI GATTANI***

***BATCH: 2***

***PRN : 12311291***

***ROLL NO : 37***

***LAB ASSIGNMENT 8***

**QUESTION:** **WAP to convert given Infix expression into its Equivalent Prefix and Postfix form using Stack.**

**CODE:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define MAX 100

char stack[MAX];

int top = -1;

void push(char c) {

stack[++top] = c;

}

char pop() {

return stack[top--];

}

char peek() {

return stack[top];

}

int isEmpty() {

return top == -1;

}

int precedence(char c) {

if (c == '^') return 3;

if (c == '\*' || c == '/') return 2;

if (c == '+' || c == '-') return 1;

return 0;

}

int isOperator(char c) {

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

}

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

int i, k = 0;

char ch;

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

ch = infix[i];

if (ch == ' ') continue;

if (isalnum(ch)) {

postfix[k++] = ch;

}

else if (ch == '(') {

push(ch);

}

else if (ch == ')') {

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

postfix[k++] = pop();

}

pop(); // pop '('

}

else if (isOperator(ch)) {

while (!isEmpty() && precedence(peek()) >= precedence(ch)) {

postfix[k++] = pop();

}

push(ch);

}

}

while (!isEmpty()) {

postfix[k++] = pop();

}

postfix[k] = '\0';

}

void reverse(char\* str) {

int i, j;

char temp;

for (i = 0, j = strlen(str) - 1; i < j; i++, j--) {

temp = str[i];

str[i] = str[j];

str[j] = temp;

}

}

void replaceBrackets(char\* expr) {

for (int i = 0; expr[i]; i++) {

if (expr[i] == '(')

expr[i] = ')';

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

expr[i] = '(';

}

}

void infixToPrefix(char\* infix, char\* prefix) {

char revInfix[MAX], temp[MAX];

strcpy(revInfix, infix);

reverse(revInfix);

replaceBrackets(revInfix);

top = -1; // Reset stack

infixToPostfix(revInfix, temp);

reverse(temp);

strcpy(prefix, temp);

}

int main() {

char infix[MAX], postfix[MAX], prefix[MAX];

printf("Enter Infix Expression: ");

gets(infix);

infixToPostfix(infix, postfix);

infixToPrefix(infix, prefix);

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

printf("Prefix: %s\n", prefix);

return 0;

}

**OUTPUT:**

