15)Write a C Program to implement the operator precedence parsing.  
**Program:**

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>

#define MAX 100

typedef enum { false, true } bool;

char input[MAX];

char stack[MAX];

int top = -1;

void push(char c) {

stack[++top] = c;

}

char pop() {

return stack[top--];

}

char peek() {

return stack[top];

}

bool is\_operator(char c) {

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

}

int precedence(char c) {

switch (c) {

case '+':

return 1;

case '\*':

return 2;

default:

return 0;

}

}

bool has\_higher\_precedence(char op1, char op2) {

return precedence(op1) >= precedence(op2);

}

void handle\_operator(char c) {

while (top != -1 && has\_higher\_precedence(peek(), c) && peek() != '(') {

printf("%c ", pop());

}

push(c);

}

void parse(char\* expr) {

int len = strlen(expr);

for (int i = 0; i < len; ++i) {

char c = expr[i];

if (isdigit(c)) {

printf("%c ", c);

} else if (is\_operator(c)) {

if (c == '(') {

push(c);

} else if (c == ')') {

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

printf("%c ", pop());

}

pop(); // Remove '(' from stack

} else {

handle\_operator(c);

}

}

}

while (top != -1) {

printf("%c ", pop());

}

}

int main() {

printf("Enter an expression: ");

fgets(input, MAX, stdin);

input[strcspn(input, "\n")] = '\0'; // Remove trailing newline

printf("Postfix expression: ");

parse(input);

printf("\n");

return 0;

}

**Output:**

