**11. In a Grade 3 Mathematics class, the teacher asked the students for the acronym PEMDAS. All of them thought for a while, and a smart kid, Kishore, responded that PEMDAS stands for Parentheses, Exponentiation, Multiplication, Division, Addition, and Subtraction. Can you write a C Program to help students understand operator precedence parsing? The program should evaluate an expression containing multiple operators, ensuring that the order of evaluation follows the order of operations (PEMDAS).**

**Code:**

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <ctype.h>**

**int applyOperator(int a, int b, char op) {**

**switch (op) {**

**case '+': return a + b;**

**case '-': return a - b;**

**case '\*': return a \* b;**

**case '/': return a / b;**

**case '^': {**

**int result = 1;**

**for (int i = 0; i < b; i++) result \*= a;**

**return result;**

**}**

**default: return 0;**

**}**

**}**

**int precedence(char op) {**

**if (op == '+' || op == '-') return 1;**

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

**if (op == '^') return 3;**

**return 0;**

**}**

**int evaluateExpression(const char \*expr) {**

**int values[100], valTop = -1;**

**char ops[100];**

**int opsTop = -1;**

**for (int i = 0; expr[i] != '\0'; i++) {**

**if (isdigit(expr[i])) {**

**int val = 0;**

**while (isdigit(expr[i])) {**

**val = (val \* 10) + (expr[i] - '0');**

**i++;**

**}**

**i--;**

**values[++valTop] = val;**

**} else if (expr[i] == '(') {**

**ops[++opsTop] = expr[i];**

**} else if (expr[i] == ')') {**

**while (opsTop >= 0 && ops[opsTop] != '(') {**

**int b = values[valTop--];**

**int a = values[valTop--];**

**char op = ops[opsTop--];**

**values[++valTop] = applyOperator(a, b, op);**

**}**

**opsTop--;**

**} else {**

**while (opsTop >= 0 && precedence(ops[opsTop]) >= precedence(expr[i])) {**

**int b = values[valTop--];**

**int a = values[valTop--];**

**char op = ops[opsTop--];**

**values[++valTop] = applyOperator(a, b, op);**

**}**

**ops[++opsTop] = expr[i];**

**}**

**}**

**while (opsTop >= 0) {**

**int b = values[valTop--];**

**int a = values[valTop--];**

**char op = ops[opsTop--];**

**values[++valTop] = applyOperator(a, b, op);**

**}**

**return values[valTop];**

**}**

**int main() {**

**char expr[100];**

**printf("Enter a mathematical expression: ");**

**scanf("%s", expr);**

**printf("Result: %d\n", evaluateExpression(expr));**

**return 0;**

**}**

