Week: 02 (Infix to postfix) in C

```
#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define N 100

int top = -1;
char stack[N];

int prec(char a) {
    if (a == '^') return 3;
    if (a == '*' || a == '/') return 2;
    if (a == '+' || a == '-') return 1;
    return 0;
}
void push(char a) {
    if (top < N-1)
        stack[++top] = a;
}
char pop() {
    if (top == -1) return '\0';
    return stack[top--];
}
char peek() {
    if (top == -1) return '\0';
    return stack[top];
}
```

```
void infixtopostfix(char exp[], char res[]){
    int j = 0;
    for(int i = 0; i < strlen(exp); i++){
        char c = exp[i];

        if (isalnum(c)) {
            res[j++] = c;
        }
        else if (c == '(') {
            push(c);
        }
        else if (c == ')') {
            while (peek() != '(' && top != -1){
                res[j++] = pop();
            }
            pop(); // discard '('
        }
        else { // operator
            while (top != -1 && prec(peek()) >= prec(c)) {
                res[j++] = pop();
            }
            push(c);
        }
    }
    while (top != -1) {
        res[j++] = pop();
    }
}
```

```
    res[j] = '\0';
}
int main() {
    char exp[N], result[N];
    printf("Enter infix expression: ");
    scanf("%s", exp);
    infixtopostfix(exp, result);
    printf("postfix expression: %s\n", result);
    return 0;
}
```

O/P :-

Enter a valid parenthesized infix expression

(A+B) * C^1 2

postfix expression :

A B + C * 1 2 ^