```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define MAX_SIZE 100

// Function to check if a character is an operator
int isOperator(char ch) {
   return (ch == '+' || ch == '-' || ch == '*' || ch == '/' || ch == '^');
}

// Function to get the precedence of an operator
int getPrecedence(char op) {
   if (op == '^')
      return 4;
   else if (op == '*' || op == '/')
      return 3;
   else if (op == '+' || op == '-')
      return 2;
   else
      return 1; // Default for '('
}

// Function to convert infix to postfix
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 (isalnum(infix[i])) {
         postfix[j++] = infix[i];
      } else if (infix[i] == '(') {
         stack[++top] = infix[i];
      } else if (infix[i] == ')') {
         while (top != -1 && stack[top] != '(') {
            postfix[j++] = stack[top--];
         }
         top--; // Pop '('
      } else if (isOperator(infix[i])) {
         while (top != -1 && getPrecedence(stack[top]) >= getPrecedence(infix[i])) {
            postfix[j++] = stack[top--];
         }
         stack[++top] = infix[i];
      }
   }
```

```
    while (top != -1) {
        postfix[j++] = stack[top--];
    }

    postfix[j] = '\0';
}

int main() {
    char infix[MAX_SIZE], postfix[MAX_SIZE];

    printf("Enter infix expression: ");
    scanf("%s", infix);

    infixToPostfix(infix, postfix);

    printf("Infix Expression: %s\n", infix);
    printf("Postfix Expression: %s\n", postfix);
    printf("Shreyash Sinha 1BM22CS273");
    return 0;
}
```

```
Enter infix expression: A-(B/C+(D%E+F)/G)*H
Infix Expression: A-(B/C+(D%E+F)/G)*H
Postfix Expression: ABC/DEF+G/+H*-
Shreyash Sinha 1BM22CS273
```