```cpp
#include <iostream>
#include <stack>
#include <string>

int precedence(char op) {
    if (op == '+' || op == '-')
        return 1;
    if (op == '*' || op == '/')
        return 2;
    return 0;
}

std::string infixToPostfix(const std::string& infixExpression) {
    std::string postfix;
    std::stack<char> operatorStack;

    for (char ch : infixExpression) {
        if (isalnum(ch)) {

            postfix += ch;
        } else if (ch == '(') {

            operatorStack.push(ch);
        } else if (ch == ')') {

            while (!operatorStack.empty() && operatorStack.top() != '(') {
                postfix += operatorStack.top();
                operatorStack.pop();
            }
            operatorStack.pop();
        } else {

            while (!operatorStack.empty() && precedence(operatorStack.top
              ()) >= precedence(ch)) {
                postfix += operatorStack.top();
                operatorStack.pop();
            }
            operatorStack.push(ch);
        }
    }


    while (!operatorStack.empty()) {
        postfix += operatorStack.top();
        operatorStack.pop();
    }

    return postfix;
}
```

```cpp

int main() {
    std::string infixExpression;
    std::cout << "Enter an infix expression: ";
    std::getline(std::cin, infixExpression);
    std::string postfixExpression = infixToPostfix(infixExpression);
    std::cout << "Postfix expression: " << postfixExpression << std::endl;

    return 0;
}

```