

## **//CODE:**

```
#include <iostream>

#include <stack>

#include <cctype>

#include <string>

using namespace std;

int precedence(char op) {

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

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

    return 0;

}

string infixToPostfix(const string &infix) {

    stack<char> operators;

    string postfix;

    for (char ch : infix) {

        if (isdigit(ch)) {

            postfix += ch;

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

            operators.push(ch);

        } else if (ch == ')') { // Right parenthesis

            while (!operators.empty() && operators.top() != '(') {

                postfix += operators.top();

                operators.pop();

            }

            operators.pop();

        } else {

            while (!operators.empty() && precedence(operators.top()) >= precedence(ch)) {

                postfix += operators.top();

                operators.pop();

            }

            operators.push(ch);

        }

    }

    return postfix;

}
```

```

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

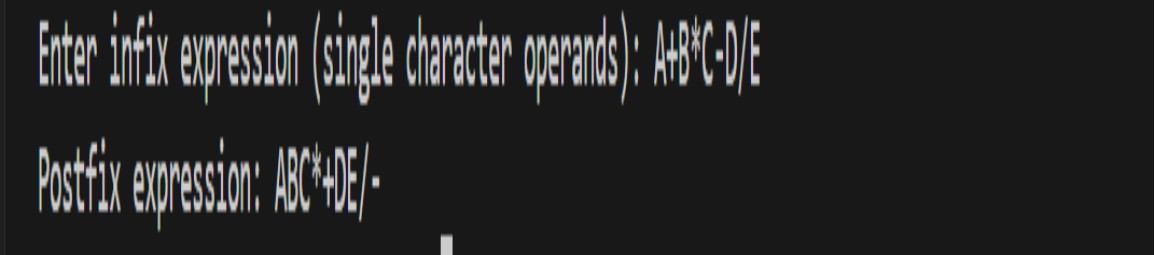
int evaluatePostfix(const string &postfix) {
    stack<int> operands;
    for (char ch : postfix) {
        if (isdigit(ch)) {
            operands.push(ch - '0');
        } else {
            int right = operands.top(); operands.pop();
            int left = operands.top(); operands.pop();
            switch (ch) {
                case '+': operands.push(left + right); break;
                case '-': operands.push(left - right); break;
                case '*': operands.push(left * right); break;
                case '/': operands.push(left / right); break;
            }
        }
    }
    return operands.top(); // Final result
}

int main() {
    string infix;
    cout << "Enter infix expression (single character operands): ";
    cin >> infix;
    string postfix = infixToPostfix(infix);

```

```
cout << "Postfix expression: " << postfix << endl;  
int result = evaluatePostfix(postfix);  
cout << "Evaluation result: " << result << endl;  
return 0;  
}
```

### **//OUTPUT:**



```
Enter infix expression (single character operands): A+B*C-D/E  
Postfix expression: ABC*+DE/-
```