

Main.cpp

3zryqyg8y

NEW

```
1 // C++ program to evaluate a prefix expression.
2 #include <bits/stdc++.h>
3 using namespace std;
4
5 bool isOperand(char c)
6 {
7     // If the character is a digit then it must
8     // be an operand
9     return isdigit(c);
10 }
11
12 double evaluatePrefix(string exprsn)
13 {
14     stack<double> stack;
15
16     for (int j = exprsn.size() - 1; j >= 0; j--) {
17
18         // Push operand to Stack
19         // To convert exprsn[j] to digit subtract
20         // '0' from exprsn[j].
21         if (isOperand(exprsn[j]))
22             stack.push(exprsn[j] - '0');
23     }
```

STDIN

Input for the p

Output:

postfix evaluati



main.cpp

3zryqyg8y

```
16 for (int j = exprsn.size() - 1; j >= 0; j--) {
17
18     // Push operand to stack
19     // To convert exprsn[j] to digit subtract
20     // '0' from exprsn[j].
21     if (isOperand(exprsn[j]))
22         Stack.push(exprsn[j] - '0');
23
24     else {
25
26         // Operator encountered
27         // Pop two elements from stack
28         double o1 = Stack.top();
29         Stack.pop();
30         double o2 = Stack.top();
31         Stack.pop();
32
33         // Use switch case to operate on o1
34         // and o2 and perform o1 Or o2.
35         switch (exprsn[j]) {
36             case '+':
37                 Stack.push(o1 + o2);
38             break;
```

NEW

CPP

STDIN

Input for the pro

Output:

postfix evaluation



Main.cpp

3zryqyg8y

NEW

CPP ✓

STDIN

Input for the program

Output:

postfix evaluation: 4

```
35 switch (exprsn[j]) {
36     case '+':
37         stack.push(o1 + o2);
38         break;
39     case '-':
40         stack.push(o1 - o2);
41         break;
42     case '*':
43         stack.push(o1 * o2);
44         break;
45     case '/':
46         stack.push(o1 / o2);
47         break;
48 }
49 }
50 }
51
52 return stack.top();
53 }
54
55 // Driver code
56 int main()
57 {
```

