```
//Name: 2. Stack

#include <iostream>
#include <stack>
#include <string>
#include <algorithm>
#include <cmath>
using namespace std;
bool isOperator(char c) {
  return (c == '+' || c == '-' || c == '*' || c == '/' || c == '^');
}
int precedence(char op) {
  if (op == '+' || op == '-') return 1;
  if (op == '*' || op == '/') return 2;
  if (op == '^') return 3;
  return 0;
}
string infixToPos ix(string infix) {
  stack<char> s;
  string pos ix = "";
  for (int i = 0; i < infix.length(); i++) {
      char c = infix[i];

      if (isdigit(c)) {
```

```
        pos ix += c;
    }

    else if (c == '(') {
        s.push(c);
    }

    else if (c == ')') {
        while (!s.empty() && s.top() != '(') {
            pos ix += s.top();
            s.pop();
        }
        s.pop();
    }

    else if (isOperator(c)) {
        while (!s.empty() && precedence(c) <= precedence(s.top())) {
            pos ix += s.top();
            s.pop();
        }
        s.push(c);
    }
}
```

```
  while (!s.empty()) {
     pos ix += s.top();
     s.pop();
  }
  return pos ix;
}
string infixToPrefix(string infix) {

  reverse(infix.begin(), infix.end());

  for (int i = 0; i < infix.length(); i++) {
     if (infix[i] == '(') infix[i] = ')';
     else if (infix[i] == ')') infix[i] = '(';
  }

  string pos ix = infixToPos ix(infix);

  reverse(pos ix.begin(), pos ix.end());
  return pos ix;
}
int evaluatePos ix(string pos ix) {
  stack<int> s;
  for (char c : pos ix) {
```

```
      if (isdigit(c)) {
         s.push(c - '0');
      }

      else {
         int val2 = s.top(); s.pop();
         int val1 = s.top(); s.pop();
         switch (c) {
            case '+': s.push(val1 + val2); break;
            case '-': s.push(val1 - val2); break;
            case '*': s.push(val1 * val2); break;
            case '/': s.push(val1 / val2); break;
         }
      }
   }
   return s.top();
}
int evaluatePrefix(string prefix) {
   stack<int> s;

   for (int i = prefix.length() - 1; i >= 0; i--) {
      char c = prefix[i];

      if (isdigit(c)) {
```

```
            s.push(c - '0');
        }

        else {
            int val1 = s.top(); s.pop();
            int val2 = s.top(); s.pop();
            switch (c) {
                case '+': s.push(val1 + val2); break;
                case '-': s.push(val1 - val2); break;
                case '*': s.push(val1 * val2); break;
                case '/': s.push(val1 / val2); break;
            }
        }
    }
    return s.top();
}
 int main() {
    string infix = "(3+4)*5/2";
    string pos ix = infixToPos ix(infix);
cout << "Pos ix: " << pos ix << endl;
string prefix = infixToPrefix(infix);
cout << "Prefix: " << prefix << endl;
int pos ixResult = evaluatePos ix(pos ix);
cout << "Pos ix Evalua on: " << pos ixResult << endl;
```

```
int prefixResult = evaluatePrefix(prefix);
cout << "Prefix Evalua on: " << prefixResult << endl;
return 0;
}
```