ASSIGNMENT NO. 6

NAME: Payal Bhagvan Charvande

CLASS: S.Y. **DIV:** IT A

PRN NO: 125B2F002

A)

CODE:-

```
#include <iostream>
#include <stack>
using namespace std;

int precedence(char c) {
    if (c == '^') return 3;
    if (c == '*' || c == '/') return 2;
    if (c == '+' || c == '-') return 1;
    return 0;
}
int main() {
    string infix, postfix = "", prefix = "";
    stack<char> s, s2;

    cout << "Enter infix : ";
    cin >> infix;

    // --- Infix to Postfix ---
    for (int i = 0; i < infix.size(); i++) {
        char c = infix[i];

        if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z') || (c >= '0' && c <= '9')) {
```

```
        postfix += c;
    }
    else if (c == '(') {
        s.push(c);
    }
    else if (c == ')') {
        while (!s.empty() && s.top() != '(') {
            postfix += s.top();
            s.pop();
        }
        s.pop(); // '('remove it
    }
    else {
        while (!s.empty() && precedence(s.top()) >= precedence(c)) {
            postfix += s.top();
            s.pop();
        }
        s.push(c);
    }
}
while (!s.empty()) {
    postfix += s.top();
    s.pop();
}
// --- Infix to Prefix ---
// Step 1: Reverse infix
string rev_infix = "";
for (int i = infix.size() - 1; i >= 0; i--) {
    char c = infix[i];
```

```
    if (c == '(') c = ')';
    else if (c == ')') c = '(';
    rev_infix += c;
}
// Step 2: Convert reversed infix to postfix
string rev_postfix = "";
for (int i = 0; i < rev_infix.size(); i++) {
    char c = rev_infix[i];
    if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z') || (c >= '0' && c <= '9')) {
        rev_postfix += c;
    }
    else if (c == '(') {
        s2.push(c);
    }
    else if (c == ')') {
        while (!s2.empty() && s2.top() != '(') {
            rev_postfix += s2.top();
            s2.pop();
        }
        s2.pop();
    }
    else {
        while (!s2.empty() && precedence(s2.top()) >= precedence(c)) {
            rev_postfix += s2.top();
            s2.pop();
        }
        s2.push(c);
    }
}
```

```
    while (!s2.empty()) {
        rev_postfix += s2.top();
        s2.pop();
    }
    // Step 3: Reverse the postfix to get prefix
    for (int i = rev_postfix.size() - 1; i >= 0; i--) {
        prefix += rev_postfix[i];
    }
    // --- Print all ---
    cout << "Infix: " << infix << endl;
    cout << "Postfix: " << postfix << endl;
    cout << "Prefix: " << prefix << endl;
    return 0;
}
```

OUTPUT :-