Stack:

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

class Stack{
    private:
        int top, ar_size;
        int arr[100];

        public:
            Stack(int n){
                top = -1;
                ar_size = n;
            }

        void push(){
            if( top == (ar_size-1) ){
                cout << "Stack is Full ";
                return;
            }

            cout << "Enter the element :\t";
            cin >> arr[++top];
        }
        void pop(){
            if(top == -1){
                cout << "Stack is Empty";
                return;
            }
            cout << "Deleted element is : \t" << arr[top--];
        }

        void Display(){
            if (top == -1){
                cout << "Stack is Empty";
                return;
            }
            for (int i = 0; i <= top;i++){
                cout << arr[i] << "\t";
            }
            cout << endl;
        }
};

int main(){
    int n,ch;
    cout << "Enter the size of array";
    cin >> n;
    Stack st(n);
    do{
        cout << "\n 1.PUSH \n 2.POP \n 3.DISPLAY \n 4.EXIT";
        cout << "\nEnter the Choice";
        cin >> ch;
        switch (ch){

            case 1:
                st.push();
                break;
            case 2:
```

```
                st.pop();
                break;
            case 3:
                st.Display();
                break;
        }
        cout << endl;
    } while (ch < 4);
    return 0;
}
```

Infix to Postfix with ()

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

int prec(char c){

    if (c == '^')
        return 3;

    else if (c == '/' || c == '*')
        return 2;

    else if (c == '+' || c == '-' )
        return 1;

    else
        return -1;
}


string InfixToPostfix(string s){
    stack<char> st;
    string res;

    for (int i = 0; i < s.length();i++){

        if ( (s[i] >= 'a' && s[i] <= 'z') || ( s[i] >='A' && s[i] <='Z') )
            res += s[i];

        else if( s[i] == '(')
            st.push(s[i]);

        else if(  s[i] ==')' ){
            while( !st.empty() && st.top()!='('){
                res += st.top();
                st.pop();
            }
            if(!st.empty())
                st.pop();

        }
        else{
            while( !st.empty() && prec( st.top()) > prec(s[i]) ){
                res += st.top();
                st.pop();
            }
            st.push(s[i]);
        }
    }

    while(!st.empty()){
        res += st.top();
        st.pop();
    }
    return res;
}
int main(){
    cout<<"\n\n(a-b/c)*(a/k-l)\n\n"<<endl;
```

```
    cout << InfixToPostfix("(a-b/c)*(a/k-l)")<<endl;
    cout<<"\n\n";

    return 0;
}
```

```
PS E:\Academic\Sem1\DSA\Test CPP> .\eval

(a-b/c)*(a/k-l)

abc/-ak/l-*
```

Infix to Postfix without ()

```cpp
// Infix to Postfix without ()

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

int prec(char c){
    if (c == '^')
        return 3;
    else if (c == '*' || c == '/')
        return 2;
    else if (c == '+' || c == '-')
        return 1;
    else
        return -1;
}

string InfixtoPostfix(string s){
    stack<char> st;
    string res;

    for (int i = 0; i < s.length();i++){
        if ((s[i] >= 'a' && s[i] <= 'z') || (s[i] >= 'A' && s[i] <= s[i]))
            res += s[i];
        else{
            while( !st.empty() && prec(st.top()) >= prec(s[i])){
                res += st.top();
                st.pop();
            }
            st.push(s[i]);
        }
    }
    while(!st.empty()){
        res += st.top();
        st.pop();
    }
    return res;
}

int main(){
    cout << "\n\n";
    cout << "a+b/c-g"<<endl;
    cout << InfixtoPostfix("a+b/c-g") << endl;
    cout << "\n\n";


    cout << "----------Example 2--------------------";
    cout << "\n\n";
    cout << "a+b*c/g"<<endl;
    cout << InfixtoPostfix("a+b*c/g") << endl;
    cout << "\n\n";
    return 0;
}
```

```
PS E:\Academic\Sem1\DSA\Test CPP> g++ .\eva2.cpp -o .\eva2
PS E:\Academic\Sem1\DSA\Test CPP> .\eva2


a+b/c-g
abc/+g-


----------Example 2------------------

a+b*c/g
abc*g/+
```

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

int Evaluation(string s){
    stack<char> st;

    for (int i = 0; i < s.length();i++){
        if( s[i]>='0' && s[i]<='9'){
            st.push(s[i] - '0');
        }
        else{
            int op2 = st.top();
            st.pop();

            int op1 = st.top();
            st.pop();

            switch(s[i]){
                case '+':
                    st.push(op1 + op2);
                    break;
                case '-':
                    st.push(op1 - op2);
                    break;
                case '/' :
                    st.push(op1 / op2);
                    break;
                case '*':
                    st.push(op1 * op2);
                    break;
                case '^':
                    st.push(pow(op1, op2));
                    break;
            }
        }
    }
    return st.top();
}

int main(){
    cout << "\n\n\n";
    cout << "46+2/5*7+" << endl;
    cout << "THe Result is \t" << Evaluation("46+2/5*7+") << endl;
    cout << "\n\n";

    return 0;
}
```

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

int prec(char c)
{
    if (c == '^')
        return 3;
    else if (c == '/' || c == '*')
        return 2;
    else if (c == '+' || c == '-')
        return 1;
    else
        return -1;
}

string itof(string s)
{
    stack<char> st;
    string res;

    for (int i = 0; i < s.length(); i++)
    {
        if ((s[i] >= 'a' && s[i] <= 'z') || (s[i] >= 'A' && s[i] <= 'Z'))
            res += s[i];
        else{
            while (!st.empty() && prec(st.top()) >= prec(s[i]))
            {
                res += st.top();
                st.pop();
            }
            st.push(s[i]);
        }
    }
    while (!st.empty())
    {
        res += st.top();
        st.pop();
    }
    return res;
}

string rev(string sr)
{
    string s1;
    for (int i = sr.length() - 1; i >= 0; i--)
    {
        s1 += sr[i];
    }
    return s1;
}

int main()
{
    cout << "\n\n";

    string w = rev("a+b/c-g");
    string ans = itof(w);
    ans = rev(ans);
    cout << ans << endl;
```

```
    cout << "\n\n";
    return 0;
}
```