# include <iostream>

# include <stdlib.h>

# define MAX 10

using namespace std;

template <typename p>

class stack {

private:

p op[MAX]; // stack of operators

p top;

public:

stack() {

top = -1;

}

void push (int ele) {

if (!isFull())

op[++top] = ele;

else

cout << endl << "Stack Overflow!";

}

p pop () {

if (!isEmpty()) {

--top;

}

else

cout << endl << "Stack Underflow!";

}

p topel () {

if (top != -1)

return op[top];

}

p returnTop () {

return top;

}

p returnEle(int val) {

return op[val];

}

bool isEmpty () {

if (top == -1)

return true;

else

return false;

}

bool isFull () {

if (top == MAX-1)

return true;

else

return false;

}

};

int main () {

stack<char> s1;

char array[MAX];

char chr, ele;

do {

int top = 0;

cout << endl << "Enter your inflix expression: ";

for (int i = 0; i < MAX; ++i)

cin >> array[i];

for (int i = 0; i < MAX; ++i) {

if (((array[i] >= 'a') & (array[i] <= 'z')) | ((array[i] >= 'A') & (array[i] <= 'Z'))) // if found a character

cout << ele;

else {

if (s1.isEmpty())

s1.push (array[i]);

else {

char topp = s1.topel();

if ((ele == '+') | (ele == '-') & ((topp == '+') | (topp == '-'))) { // if same precedence

cout << s1.topel();

s1.pop();

s1.push(array[i]);

}

else if (array[i] == '\0') { // end encountered

top = s1.returnTop();

while (top != -1) {

cout << s1.returnEle(top) << endl;

--top;

}

}

else { // if more

s1.push(array[i]);

}

}

}

}

cout << endl << "Do you want to enter more? (Y/N): ";

cin >> chr;

} while (chr == 'Y' | chr == 'y');

return 0;

}