#include <stdio.h>

#include <string.h>

#include <math.h>

#define MAX 10

int top = -1;

int stack[MAX];

void push(int value) {

if(top >= (MAX-1)) {

printf("Stack Overflow\n");

}

else {

stack[++top] = value;

}

}

int pop() {

if(top < 0) {

printf("Stack Underflow\n");

return 0;

}

else {

return stack[top--];

}

}

int is\_operator(char c) {

switch(c) {

case '+':

case '-':

case '\*':

case '/':

case '%':

case '^': return 1;

23

default: return 0;

}

}

int evaluate\_postfix(char \*postfix) {

int i, op1, op2;

char c;

for(i=0; postfix[i]!= '\0'; i++) {

c = postfix[i];

if(c >= '0' && c <= '9') {

push(c - '0');

}

else if(is\_operator(c)) {

op2 = pop();

op1 = pop();

switch(c) {

case '+': push(op1 + op2); break;

case '-': push(op1 - op2); break;

case '\*': push(op1 \* op2); break;

case '/':

if(op2 == 0) {

printf("Error: Division by zero is not allowed.\n");

return -1;

}

else {

push(op1 / op2);

}

break;

case '%':

if(op2 == 0) {

printf("Error: Modulus by zero is not allowed.\n");

return -1;

}

else {

push((int)fmod(op1, op2));

}

break;

case '^': push((int)pow(op1, op2)); break;

}

24

}

}

return pop();

}

int main() {

char postfix[MAX];

int result;

printf("Enter a postfix expression: ");

scanf("%s", postfix);

result = evaluate\_postfix(postfix);

if(result != -1)

printf("Result: %d\n", result);

return 0;

}