1. Write a program for the evaluation of a Postfix expression.

#include<iostream>

#include<cstring>

#include<cmath>

using namespace std;

#define MAX 100

class Stack {

private:

int arr[MAX];

int top;

public:

Stack() {

top = -1;

}

bool isEmpty() {

return top == -1;

}

bool isFull() {

return top == MAX - 1;

}

void push(int x) {

if (isFull()) {

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

return;

}

arr[++top] = x;

}

int pop() {

if (isEmpty()) {

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

return -1;

}

return arr[top--];

}

int peek() {

if (isEmpty()) {

return -1;

}

return arr[top];

}

};

bool isOperator(char c) {

return (c == '+' || c == '-' || c == '\*' || c == '/' || c == '%' || c == '^');

}

bool isDigit(char c) {

return (c >= '0' && c <= '9');

}

int performOperation(int operand1, int operand2, char op) {

switch (op) {

case '+':

return operand1 + operand2;

case '-':

return operand1 - operand2;

case '\*':

return operand1 \* operand2;

case '/':

if (operand2 == 0) {

cout << "Error: Division by zero!" << endl;

return 0;

}

return operand1 / operand2;

case '%':

if (operand2 == 0) {

cout << "Error: Division by zero!" << endl;

return 0;

}

return operand1 % operand2;

case '^':

return pow(operand1, operand2);

default:

cout << "Error: Invalid operator!" << endl;

return 0;

}

}

int evaluatePostfix(string postfix) {

Stack stack;

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

char c = postfix[i];

if (c == ' ') {

continue;

}

if (isDigit(c)) {

stack.push(c - '0');

}

else if (isOperator(c)) {

if (stack.isEmpty()) {

cout << "Error: Invalid postfix expression!" << endl;

return -1;

}

int operand2 = stack.pop();

if (stack.isEmpty()) {

cout << "Error: Invalid postfix expression!" << endl;

return -1;

}

int operand1 = stack.pop();

int result = performOperation(operand1, operand2, c);

stack.push(result);

}

else {

cout << "Error: Invalid character in expression: " << c << endl;

return -1;

}

}

if (stack.isEmpty()) {

cout << "Error: Invalid postfix expression!" << endl;

return -1;

}

int finalResult = stack.pop();

if (!stack.isEmpty()) {

cout << "Error: Invalid postfix expression!" << endl;

return -1;

}

return finalResult;

}

int main() {

string postfix;

cout << "Enter postfix expression (single digit operands only): ";

getline(cin, postfix);

cout << "Postfix expression: " << postfix << endl;

int result = evaluatePostfix(postfix);

if (result != -1) {

cout << "Result: " << result << endl;

}

return 0;

}

OUTPUT :

