* **C program to evaluate postfix expression**

**Code:**

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#define MAX\_SIZE 100

struct Stack {

int top;

int items[MAX\_SIZE];

};

void initialize(struct Stack \*s) {

s->top = -1;

}

int isEmpty(struct Stack \*s) {

return s->top == -1;

}

void push(struct Stack \*s, int value) {

if (s->top == MAX\_SIZE - 1) {

printf("Stack Overflow\n");

exit(1);

}

s->items[++s->top] = value;

}

int pop(struct Stack \*s) {

if (isEmpty(s)) {

printf("Stack Underflow\n");

exit(1);

}

return s->items[s->top--];

}

int evaluatePostfix(char \*expression) {

struct Stack stack;

initialize(&stack);

for (int i = 0; expression[i]; i++) {

if (isdigit(expression[i])) {

push(&stack, expression[i] - '0');

} else {

int operand2 = pop(&stack);

int operand1 = pop(&stack);

switch (expression[i]) {

case '+':

push(&stack, operand1 + operand2);

break;

case '-':

push(&stack, operand1 - operand2);

break;

case '\*':

push(&stack, operand1 \* operand2);

break;

case '/':

if (operand2 == 0) {

printf("Division by zero\n");

exit(1);

}

push(&stack, operand1 / operand2);

break;

default:

printf("Invalid operator: %c\n", expression[i]);

exit(1);

}

}

}

if (isEmpty(&stack)) {

printf("Invalid expression\n");

exit(1);

}

return pop(&stack);

}

int main() {

char expression[MAX\_SIZE];

printf("Enter a postfix expression: ");

scanf("%s", expression);

int result = evaluatePostfix(expression);

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

return 0;

}

**Output:**

