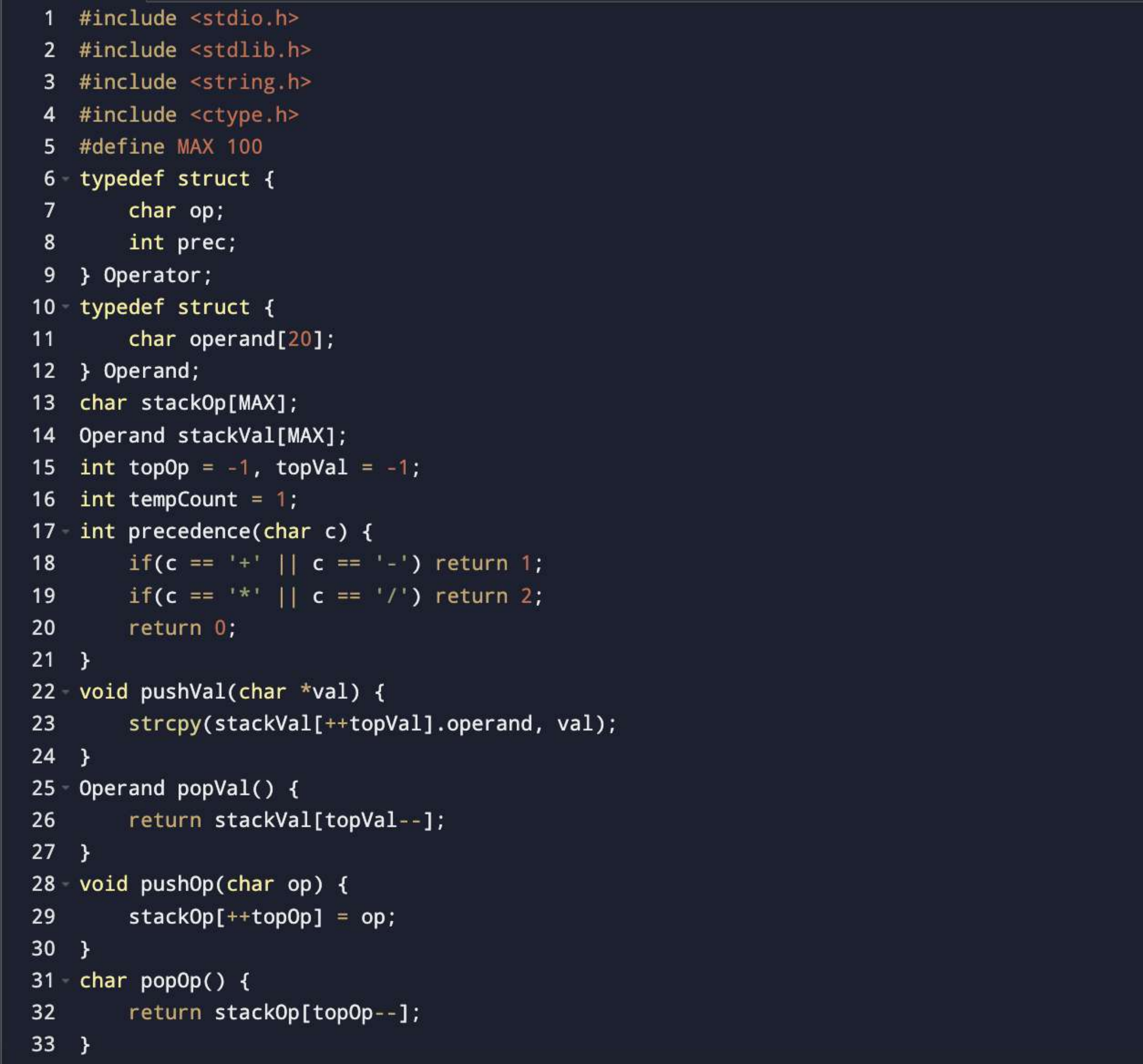
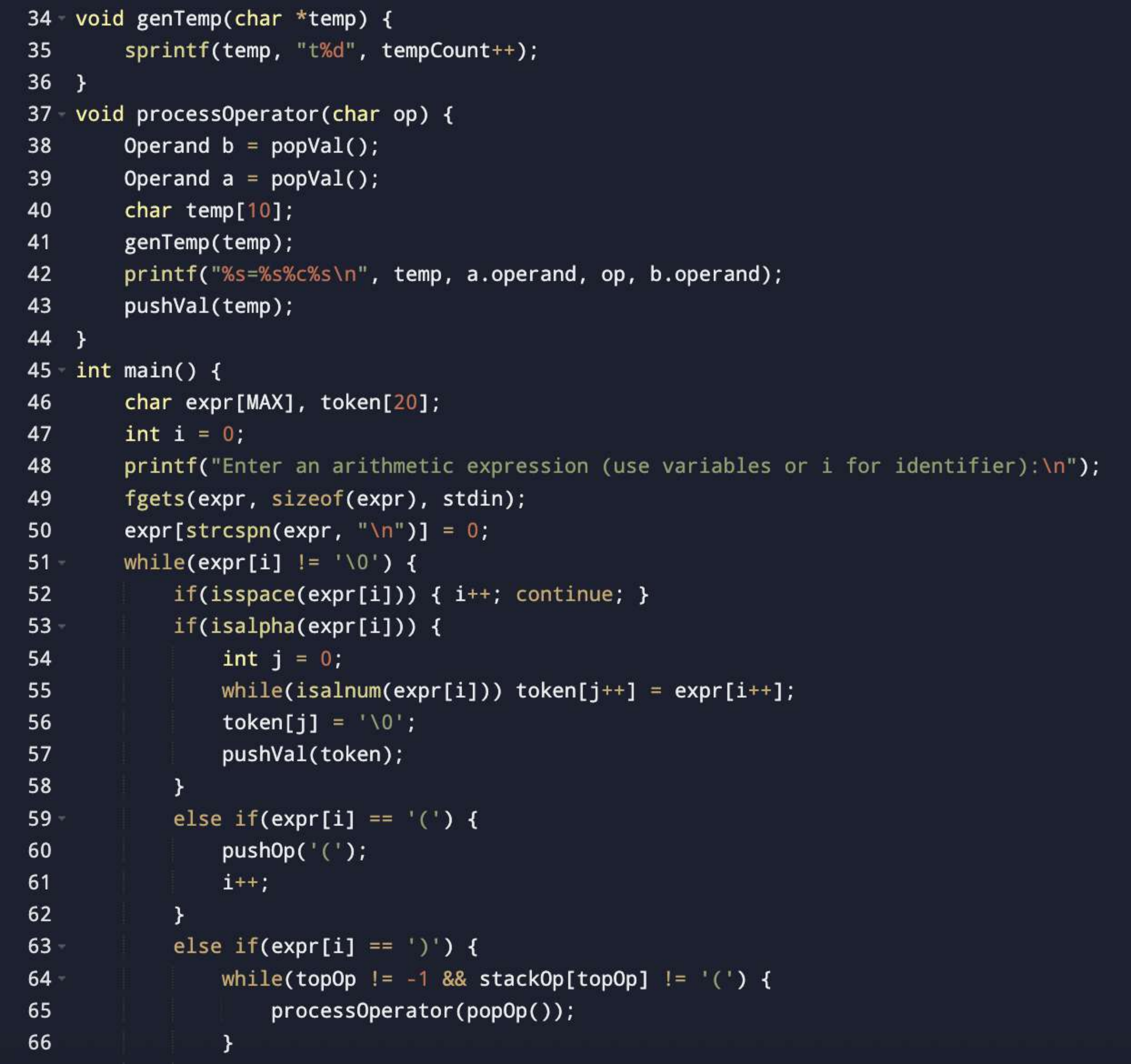
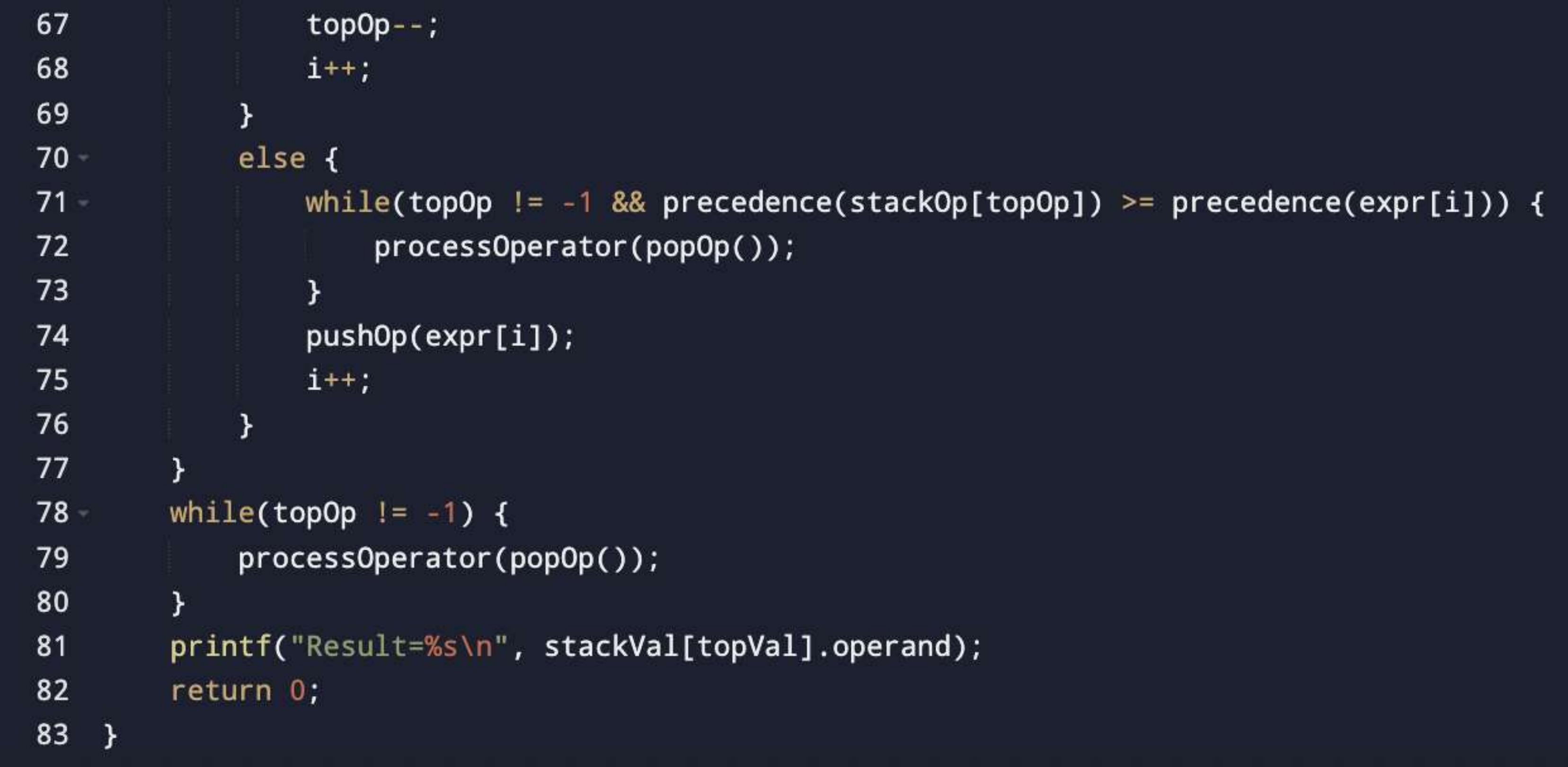


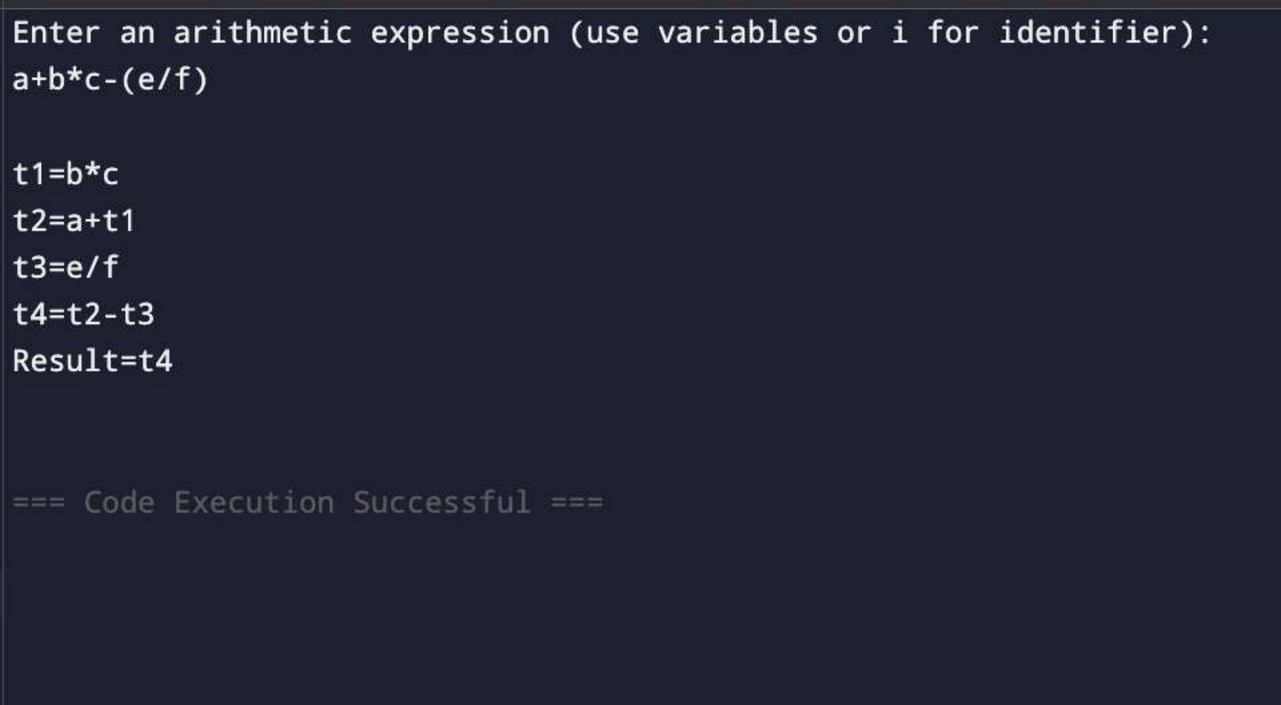
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#define MAX 100
typedef struct {
    char op;
    int prec;
} Operator;
typedef struct {
    char operand[20];
} Operand;
char stackOp[MAX];
Operand stackVal[MAX];
int topOp = -1, topVal = -1;
int tempCount = 1;
int precedence(char c) {
    if(c == '+' || c == '-') return 1;
    if(c == '*' || c == '/') return 2;
    return 0;
}
void pushVal(char *val) {
    strcpy(stackVal[++topVal].operand, val);
}
Operand popVal() {
    return stackVal[topVal--];
}
void pushOp(char op) {
    stackOp[++topOp] = op;
}
char popOp() {
    return stackOp[topOp--];
}
```



```c
void genTemp(char *temp) {
    sprintf(temp, "t%d", tempCount++);
}
void processOperator(char op) {
    Operand b = popVal();
    Operand a = popVal();
    char temp[10];
    genTemp(temp);
    printf("%s=%s%c%s\n", temp, a.operand, op, b.operand);
    pushVal(temp);
}
int main() {
    char expr[MAX], token[20];
    int i = 0;
    printf("Enter an arithmetic expression (use variables or i for identifier):\n");
    fgets(expr, sizeof(expr), stdin);
    expr[strcspn(expr, "\n")] = 0;
    while(expr[i] != '\0') {
        if(isspace(expr[i])) { i++; continue; }
        if(isalpha(expr[i])) {
            int j = 0;
            while(isalnum(expr[i])) token[j++] = expr[i++];
            token[j] = '\0';
            pushVal(token);
        }
        else if(expr[i] == '(') {
            pushOp('(');
            i++;
        }
        else if(expr[i] == ')') {
            while(topOp != -1 && stackOp[topOp] != '(') {
                processOperator(popOp());
            }
```



```c
            topOp--;
            i++;
        }
        else {
            while(topOp != -1 && precedence(stackOp[topOp]) >= precedence(expr[i])) {
                processOperator(popOp());
            }
            pushOp(expr[i]);
            i++;
        }
    }
    while(topOp != -1) {
        processOperator(popOp());
    }
    printf("Result=%s\n", stackVal[topVal].operand);
    return 0;
}
```



```
Enter an arithmetic expression (use variables or i for identifier):
a+b*c-(e/f)


t1=b*c
t2=a+t1
t3=e/f
t4=t2-t3
Result=t4



=== Code Execution Successful ===
```