

```
// Lösungsskizze zu Aufgabe 5
```

```
#include <iostream>
```

```
#include <cstdlib> // wg. exit
```

```
using namespace std;
```

```
/*
```

```
    Ausdruck ::= "+" V | "-" V | V
    V        ::= V "+" S | V "-" S | S
    S        ::= S "*" F | S "/" F | F
    F        ::= Zahl | (Ausdruck)
```

```
*/
```

```
/*
```

```
Konventionen:
```

```
    next enthält nächstes Zeichen,
    nnext enthält übernächstes Zeichen,
    zahl enthält die nächste Zahl,
    scan liest nächstes Zeichen nach nnext,
    Abschluß mit ##,
    # ist Endesymbol.
```

```
*/
```

```
char next = 'D';
```

```
char nnext = 'B';
```

```
int zahl = 0;
```

```
bool ziffer (char x) {
```

```
    if ((x=='0')||(x=='1')||(x=='2')||(x=='3')||(x=='4')
        || (x=='5')||(x=='6')||(x=='7')||(x=='8')||(x=='9'))
```

```
        return true;
```

```
    else
```

```
        return false;
```

```
}//ziffer
```

```
void scan () {
```

```
    next = nnext;
```

```
    cin >> nnext;
```

```
    if (((next=='+')||(next=='-')||(next=='*')||(next=='/'))
        && ((nnext=='+')||(nnext=='-')
            ||(nnext=='*')||(nnext=='/')) {
```

```
        cout << "Fehler: Zwei Operatoren folgen "
                "aufeinander! Abbruch!"
```

```
        << endl;
```

```
        exit (1);
```

```
    }
```

```
    if (ziffer (next)) {
```

```
        // Zahl wird erwartet
```

```
        // nur vorzeichenlose Zahlen
```

```
        zahl = next - '0';
```

```
        while (ziffer (nnext)) {
```

```
            zahl = 10*zahl + nnext - '0';
```

```
            cin >> nnext;
```

```
        }
```

```
        next = 'Z';
```

```
    }
```

```
}//scan
```

```
int V ();
```

```
int S ();
```

```
int F ();
```

```

int Ausdruck () {
    // Sonderbehandlung bei Ausdrucksbeginn
    // Operator wird an V gebunden
    if (next == '+') {
        scan ();
        return V ();
    } else if (next == '-') {
        scan ();
        if (next == 'Z') {
            zahl = -zahl;
            return V ();
        } else
            return - V ();
    } else
        return V ();
}//Ausdruck

```

```

int V () {
    int x = S ();
    while ((next == '+')||(next == '-'))
    if (next == '+') {
        scan ();
        x = x + S ();
    } else if (next == '-') {
        scan ();
        x = x - S ();
    }
    return x;
}//V

```

```

int S () {
    int x = F ();
    while ((next == '*') || (next == '/'))
    if (next == '*') {
        scan ();
        x = x * F ();
    } else if (next == '/') {
        scan ();
        x = x / F ();
    }
    return x;
}//S

```

```

int F () {
    if (next == '(') {
        scan ();
        int x = Ausdruck ();
        scan (); // ')' überlesen
        return x;
    } else if (next == 'Z') {
        int x = zahl;
        scan ();
        return x;
    } else {
        cout << "Zahl oder '(' erwartet. Abbruch"
        << endl;
        exit (1);
    }
}//F

```

```
int main () {
    cout << "Bitte Eingabe: ";
    scan ();
    scan ();
    int erg = Ausdruck ();
    cout << "Wert = " << erg << endl;
    if (next == '#')
        cout << "Korrektes Ende" << endl;
    else
        cout << "Fehler !!";

    return 0;
} //main
```

```
/*
Bitte Eingabe: -12 - 10 - 8 ##
Wert = -30
Korrektes Ende
```

```
Bitte Eingabe: - -8 + 9 ##
Fehler: Zwei Operatoren folgen aufeinander! Abbruch!
```

```
Bitte Eingabe: 3 * (-12 + (2 * 8)) ##
Wert = 12
Korrektes Ende
```

```
Bitte Eingabe: ((3+12)*()) ##
Zahl oder '(' erwartet. Abbruch
```

```
Bitte Eingabe: -(900 / 12 / 8) * (((4+20))) ##
Wert = -216
Korrektes Ende
*/
```