Kapitel 7 Beispiel 2

```
 1  #include "C_Bruch.h"
 2  #include "stdio.h"
 3  // Implementierung der Klassen-Methoden
 4  // Standardkonstruktor
 5  C_Bruch::C_Bruch()
 6  {
 7  }
 8  // Überladener Konstruktor
 9  C_Bruch::C_Bruch(int nr)
10  {
11     printf("\n\tBitte Zaehler des %i.Bruchs angeben: ",nr);
12     scanf("%i",&zaehler);
13     do
14     {
15        printf("\tBitte Nenner  des %i.Bruchs angeben: ",nr);
16        scanf("%i",&nenner);
17        if (nenner == 0)
18        {
19           printf("\n\n\tNenner duerfen nicht 0 sein!!\n");
20           printf("\tEingabe bitte korrigieren!!\n\n");
21        }
22     } while(nenner == 0);
23  }
24  // Überladener Konstruktor
25  C_Bruch::C_Bruch(int neuZaehler, int neuNenner)
26  {
27     zaehler = neuZaehler;
28     nenner  = neuNenner;
29  }
30  // Destruktor
31  C_Bruch::~C_Bruch()
32  {
33  }
34  // Zuweisung Zähler
35  void C_Bruch::setZaehler(int neuZaehler)
36  {
37     zaehler = neuZaehler;
38  }
39  // Zuweisung Nenner
40  void C_Bruch::setNenner(int neuNenner)
41  {
42     nenner  = neuNenner;
43  }
44  // Rückgabe Zähler
45  int C_Bruch::getZaehler()
46  {
47     return zaehler;
48  }
49  // Rückgabe Nenner
50  int C_Bruch::getNenner()
51  {
52     return nenner;
53  }
54  // Addition von Brüchen - Rückgabe Bruch
55  C_Bruch C_Bruch::bruchAddition(C_Bruch summand)
56  {
57     C_Bruch result;
58     result.zaehler = zaehler * summand.nenner +
                                summand.zaehler * nenner;
59     result.nenner  = nenner * summand.nenner;
60     return result;
61  }
62  // Subtraktion von Brüchen - Rückgabe Bruch
63  C_Bruch C_Bruch::bruchSubtraktion(C_Bruch subtrahend)
64  {
65     C_Bruch result;
66     result.zaehler = zaehler * subtrahend.nenner -
                                subtrahend.zaehler * nenner;
67     result.nenner  = nenner * subtrahend.nenner;
68     return result;
69  }
70  // Multiplikation von Brüchen - Rückgabe Bruch
71  C_Bruch C_Bruch::bruchMultiplikation(C_Bruch faktor)
72  {
73     C_Bruch result;
74     result.zaehler = zaehler * faktor.zaehler;
75     result.nenner  = nenner * faktor.nenner;
76     return result;
77  }
78  // Division von Brüchen - Rückgabe Bruch
79  C_Bruch C_Bruch::bruchDivision(C_Bruch divisor)
```

```
 80  {
 81     C_Bruch result;
 82     result.zaehler = zaehler * divisor.nenner;
 83     result.nenner  = nenner * divisor.zaehler;
 84     return result;
 85  }
 86  // Kürzen von Brüchen - Rückgabe Bruch
 87  C_Bruch C_Bruch::bruchKuerzen()
 88  {
 89     C_Bruch result;
 90     int sign = 1;
 91     if (zaehler < 0)
 92     {
 93        sign = -sign;
 94        zaehler = -zaehler;
 95     }
 96     if (nenner < 0)
 97     {
 98        sign = -sign;
 99        nenner = -nenner;
100     }
101     int teiler = ggT(zaehler,nenner);
102     result.zaehler = sign * zaehler / teiler;
103     result.nenner  = nenner / teiler;
104     return result;
105  }
106  // Ermittlung des größten gemeinsamen Teilers zweier Zahlen
107  // (wird beim kürzen gebraucht)
108  int C_Bruch::ggT(int z1,int z2)
109  {
110     int rest;
111     while(z2 > 0)
112     {
113        rest = z1 % z2;
114        z1 = z2;
115        z2 = rest;
116     }
117     return z1;
118  }
119  // Dokumentation (Ausgabe) von Brüchen
120  void C_Bruch::bruchDoku()
121  {
122     if (zaehler != 0)
123     {
124        printf("\tBruch = %i / %i\n",zaehler,nenner);
125     }
126     else
127     {
128        printf("\tBruch = 0\n");
129     }
130  }
```