Kapitel 9 Beispiel 5

Funktion dualtabelle():

```
 1  #include "stdafx.h"
 2  #include <stdio.h>
 3  #include <stdlib.h>
 4  #include <conio.h>
 5  #include <math.h>
 6  // "dual" ist Zeiger auf short int-Zeiger
 7  // daher kann mit "normalen" Indizes auf die Zeilen
 8  // und Spalten des zweidimensionalen Arrays "gezeigt"
 9  // werden (also wie bei einer zweidimensionalen Matrix)
10  // Die Felder werden in Zeilenrichtung von 1 bis 2^n
11  // und in Spaltenrichtung von 1 bis n belegt um später
12  // keine Indextransformation durchführen zu müssen!
13  void dualtabelle(int anz, short int** dual)
14  {
15     int anz_zeile;  // Anzahl der Zeilen im Array
16     int i, j, k;    // Hilfsvariablen
17     anz_zeile  = (int)pow(2.,anz);
18     float s = 0.5;
19     i = 0;
20     for (j = anz; j >= 1; j--) // Schleife über Spalten
21     {
22        s*=2.;
23        do                      // Schleife über Zeilen
24        {
25           for (k = 1; k <=s; k++)
26           {
27              i++;
28              dual[i][j] = 0;
29           }
30           for (k = 1; k <= s; k++)
31           {
32              i++;
33              dual[i][j] = 1;
34           }
35        } while (i < anz_zeile);
36        i = 0;
37     }
38  }
```

Hauptprogramm zum Test der Funktion dualtabelle():

```
 1  // Programm933.cpp: Hauptprojektdatei.
 2  // Erstellung einer Tabelle mit Dualzahlen
 3  // Eingabe: Anzahl der Zustände
 4  // Ausgabe: Tabelle der Dualzahlen
 5  // Autor: Heiderich / Meyer
 6  // ----------------------------------------
 7  #include "stdafx.h"
 8  #include <stdio.h>
 9  #include <stdlib.h>
10  #include <conio.h>
11  #include <math.h>
12  #include "dualtabelle.h"
13  using namespace System;
14  void main()
15  {
16     int anz;        // Anzahl der Eingänge
17     int anz_zeile;  // Anzahl der Zeilen im Array
18     int anz_spalte; // Anzahl der Spalten im Array
19     int i, j;       // Hilfsvariablen
20     // Begrüßung
21     printf("\n\n\tProgramm zur Erstellung einer Tabelle\n");
22     printf("\t   von Dualzahlen f%cr \"n\"
                Eing%cnge\n",char(129),char(132));
23     printf("\t-------------------------------------\n");
24     // Eingabe der Anzahl der Eingänge
25     printf("\n\tBitte geben Sie die Anzahl der Eing%cnge an: ",
              char(132));
26     fflush(stdin);
27     scanf("%i",&anz);
28     // dynamisches Allocieren eines zweidimensionalen Arrays
29     // mit 2^n Zeilen
30     // und n   Spalten
31     // Vorsicht: so erhält die erste Zeile den Index "0"
32     //           und die erste Spalte ebenfalls den Index "0"
33     // besser hier:
34     // dynamisches Allocieren eines zweidimensionalen Arrays
```

```
35     // mit 2^n + 1 Zeilen
36     // und n + 1   Spalten
37     anz_zeile  = (int)pow(2.,anz) + 1;
38     anz_spalte = anz + 1;
39     // "dual" ist Zeiger auf short int-Zeiger
40     short int ** dual;
41     // Speicher reservieren für die short int-Zeiger (=zeile)
42     dual = (short int **)malloc(anz_zeile * sizeof(short int *));
43     if (NULL == dual)
44     {
45        printf("Kein Virtueller RAM mehr vorhanden ... !");
46        getch();
47        exit(0);
48     }
49     // Jetzt noch Speicher reservieren für die einzelnen Spalten
50     // der i-ten Zeile
51     for(i = 0; i < anz_zeile; i++)
52     {
53        dual[i] = (short int *)malloc(anz_spalte*sizeof(short int));
54        if (NULL == dual[i])
55        {
56           printf("Kein Speicher mehr f%cr Zeile %d\n",char(129),i);
57           getch();
58           exit(0);
59        }
60     }
61     // Aufruf der Belegungsfunktion
62     // Übergabeparamter: Anzahl Eingänge, zweidimensionales Array
63     dualtabelle(anz,dual);
64     // Ausgabe der Tabelle
65     for (i = 1; i <= pow(2.,anz); i++)
66     {
67        printf("\n\t");
68        for (j = 1; j <= anz; j++)
69        {
69           printf(" %1i ",dual[i][j] );
70        }
71     }
72     getch();
73  }
```