ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Informatik I - D-ITET

Wintersemester 2003/2004

Prof. Markus Gross, Bruno Heidelberger, Richard Keiser, Nicky Kern, Edouard Lamboray, Christoph Niederberger, Tim Weyrich, Felix Eberhard, Manuel Graber, Nathalie Kellenberger, Marcel Kessler, Lior Wehrli

Uebung 6- Musterlösung

1. Programmanalyse

siehe auch u6a1_analyse.cpp

```
      der Pointer c_ptr zeigt auf die Variable mit einem $
      der Pointer d_ptr zeigt auf die Variable mit einem #

      a     b     e[0]  e[1]  e[2]  e[3]  e[4]

7     $10   0     0     1     2     3     4
8     $10   #0    0     1     2     3     4
9     $4    #0    0     1     2     3     4
10    $4    #0    0     4     2     3     4
11
12    $4    #4    0     4     2     3     4
13    $4    4     0     4     #2    3     4
14    4     4     $0    4     #2    3     4
15    4     4     $7    4     #2    3     4
16
17    4     4     6     $4    #2    3     4
18    4     4     6     4     $#7   3     4
19    4     4     7     4     $#7   3     4
20    4     4     7     4     #3    $3    4
21    4     4     7     4     #3    $4    4
```

2. n-dimensionale Vektoren

siehe u6a2_vektor.cpp

3. Dynamische mehrdimensionale Arrays

siehe auch u6a3_array.cpp

```
      // Es gilt:
      int n, i;                  // n: Seitenlaenge, i: Index
      int zeile, spalte;         // Zeile und Spalte
```

a) Erzeugen der Matrix

```
// M ist ein Pointer auf ein Array[n] von Pointern auf int-Arrays
int **M = new int*[n];
for (i=0; i<n; i++)
{
      // M[i] ist ein Pointer auf ein Array[n] von int
      M[i] = new int[n];
}
```

b) Füllen der Matrix

```
for (zeile=0; zeile < n; zeile++)
{
      for (spalte=0; spalte < n; spalte++)
      {
            // Dem Element wird der Wert zugeordnet
            M[zeile][spalte] = (spalte+1) + zeile*n;
      }
}
```

c) Ausgabe

d) Freigeben der Matrix

Regel 1: Fuer jedes `new` muss es ein `delete` geben.
Regel 2: Fuer jedes `new[]` muss es ein `delete[]` geben.

```
// Loeschen der Zeilenarrays
for (i=0; i<n; i++)
{
      delete [] M[i];
}
// Loeschen des Spaltenarrays
delete [] M;
```