FR 6.1 – Mathematik
Institut für Angewandte Mathematik
Prof. Dr. V. John

Saarbrücken, 23.01.2007

Übungsaufgaben zur Vorlesung Modellierung und Programmierung

Serie 09

zu erledigen in den Übungen bis 02.02.2007

Bei den Aufgaben handelt es sich um Programmieraufgaben, die innerhalb der Übungszeiten (**und auch außerhalb dieser Zeiten !**) im Computer–Pool bearbeitet werden können. Die korrekte Bearbeitung wird innerhalb der Übungen vom zuständigen Bremser kontrolliert.

1. Aufgabe :
   Man schreibe eine Funktion `matrix_alloc`, die als Eingabeparameter die Spaltenanzahl `M` und die Zeilenanzahl `N` einer Matrix verlangt. Die Funktion soll gemäß Vorlesung (Beispiel 8.9) dynamisch Speicher fr diese (M x N)–`(double)` Matrix reservieren und als Rückgabewert einen Doppelzeiger von Typ double auf diesen Speichplatz zurückliefern.

   Man schreibe nun ein Hauptprogramm, das eine (N x N)-Matrix A

   $$A = (a_{i,j})_{i=1\ldots N\,,\,j=1\ldots N}$$

   mit den Werten $a_{i,j} = i + j \quad (i = 1 \ldots N\,, j = 1 \ldots N)$ initialisiert. Die Dimension N soll vom Benutzer erfragt werden. Die Speicherreservierung soll mit Hilfe der Funktion `matrix_alloc` erfolgen. Man gebe die Matrix anschließend auf dem Bildschirm aus. Zum Schluss soll der allokierte Speicherplatz wieder freigegeben werden.

2. Aufgabe :
   Man schreibe eine neue Funktion `matrix_alloc`, bei der, im Gegensatz zu Beispiel 8.9, zuerst der gesamte Speicherplatz für die Matrix allokiert wird und danach die Zeiger auf die Zeilen gesetzt werden. Das Funktionieren dieser Funktion überprüfe man, indem man den zweiten Teil der ersten Aufgabe mit der neuen Funktion abarbeitet.

3. Aufgabe :
   Welchen Wert hat die Variable `a` beim Programmdurchlauf in den Zeilen 8, 10, 16, 18, 21, 23, 30?

```
/* 1 */       void funktion_1(int *);
/* 2 */
/* 3 */       int a=1;
/* 4 */
/* 5 */       int main()
/* 6 */       {
/* 7 */         int *p1=&a;p1[0]=p1[0]+1;
/* 8 */
/* 9 */         a=a+1;
/* 10 */
/* 11 */        {
/* 12 */           int a=-7;
/* 13 */           int *p2;
/* 14 */           p2=&a;
/* 15 */           p2[0]=p2[0]-1;
/* 16 */
/* 17 */           funktion_1(p1);
/* 18 */
/* 19 */           p2=p1;
/* 20 */           *p2=*p2-1;
/* 21 */
/* 22 */        }
/* 23 */
/* 24 */        return 0;
/* 25 */      }
/* 26 */
/* 27 */      void funktion_1(int * p1)
/* 28 */      {
/* 29 */             *p1=*p1+2;
/* 30 */
/* 31 */              p1=p1+1;
/* 32 */      }
```