7. Übung

1. Finden von Fehlern (Wiederholung)

Gegeben ist eine Funktion, die Minimum, Maximum und den arithmetischen Mittelwert aus n Werten berechnet, die in einem Feld w übergeben werden.

```
void auswertung(double *w, int n, double *minimum, double *maximum,
                double *arithmmittel)
{
    int i;
    double s;

   *minimum = w[0];
   *maximum = w[0];
   s=w[0];
   for (i=0; i<n; i++)
   {
      if (w[i]> maximum)  *maximum=w[i];
      if (w[i]<*minimum)   *minimum=w[i];
       s+w[i]=s;
   }
   *arithmmittel = s / n;
   return n;
}
```

Aufgaben:

a) Die Funktion enthält noch 3 Fehler, die eine erfolgreiche Kompilierung verhindern. Markieren Sie die Fehlerstellen und geben Sie für jeden Fehler eine Korrektur an!

b) Wenn die Funktion mit dem Argument 0 für den Parameter n aufgerufen wird, wird das Programm abstürzen. Erweitern Sie die Funktion, damit dieser Fall geeignet abgefangen wird!

2. Zeigerarithmetik

Das unten stehende Programm gibt als erste Adresse den Wert 14FDF4 aus. Wie lauten die drei folgenden ausgegebenen Adressen?

```
struct artikel  {
   int anr;
   float gewicht; };
int main()
{  struct artikel a[20];
   struct artikel *a_ptr = a;

   printf("%X\n",a_ptr);
   a_ptr+=5;
   printf("%X\n",a_ptr);
   a_ptr--;
   printf("%X\n",a_ptr);
   a_ptr = &a_ptr[10];
   printf("%X\n",a_ptr);

   return 0;
}
```

3. Dynamische zweidimensionale Felder

Erzeugen Sie eine n x n double-Matrix dynamisch. Der Wert n wird erst zur Laufzeit des Programms bekannt.

Initialisieren Sie die durch Zeilen- und Spaltenindex angesprochenen Elemente mit (double) (Zeilenindex+Spaltenindex)!

Hinweis:

Sie können von
double **m;
ausgehen und m als dynamisch allokiertes Feld für double-Zeiger auffassen. Die enthaltenen double-Zeiger nehmen die Anfangsadressen von Zeilen-Feldern auf, die wiederum dynamisch allokiert werden können. In C kann m, mit zwei Indexklammern versehen, wie eine Matrix gehandhabt werden.

4. Binärbäume

Gegeben ist eine Strukturdefinition für Elemente eines Binärbaums. Die Elemente sind anhand ihrer Artikelnummer platziert. Der linke Kindknoten verweist immer auf eine kleinere Artikelnummer, der rechte Kindknoten immer auf eine größere Artikelnummer im Vergleich zum Ausgangsknoten.

```
struct artikel{
        int anr;  // Artikelnummer
        char bez[80]; // Bezeichnung
        double preis;
        int bestand;  // Anzahl im Lager
};
struct baum_element {
  struct artikel a;
  struct baum_element *li, *re;
};
```

Aufgaben:

a) Geben Sie eine Funktion an, die einen Artikel anhand seiner Artikelnummer im Binärbaum sucht! Der Binärbaum wird mir seinem Wurzelknoten-Zeiger (Typ: struct baum_element*) übergeben.

b) Geben Sie eine weitere Funktion an, die den Gesamtwert aller Artikel ermittelt!