Felder (Arrays) in C#

Wenn eine feste Anzahl von Variablen mit dem gleichen Datentyp notwendig ist, kann man **Felder** bzw. **Arrays** verwenden.

Beispiele:

a) Wenn die durchschnittliche Höchsttemperatur in einem Monat bestimmt werden soll, ist es ungeschickt 31 double-Variablen `temp1, temp2, ... , temp31` zu verwenden. Geschickter ist ein Feld, da man dessen Elemente sehr geschickt in Schleifen bearbeiten kann.
b) Ein typisches Beispiel für Felder in der Mathematik sind Vektoren.

Eigenschaften eines Feldes (Arrays)

n Elemente e_0, e_1, … , e_{n-1} sollen in einem Feld gespeichert werden:

e_0	e_1	...	e_{n-1}

- Alle Daten sind vom gleichen Datentyp
- Die einzelnen Elemente werden mit Hilfe eines ganzzahligen Index („Platznummer“) angesprochen
- Der Index in C# läuft von 0 bis Feldlänge-1

Definition von Feldern

(Eindimensionale) Felder werden in C# folgendermaßen deklariert:
`Datentyp[] Variablenname;`
Damit hat man eine Variable, die ein Feld vom angegebenen Datentyp enthalten kann. Das Feld erzeugt man durch `Variablenname=new Datentyp[AnzahlElemente];`
Dabei kann `Datentyp` jeder beliebige Datentyp sein, also auch Double, String usw.
Nach der Definition eines Feldes ist seine Größe festgelegt, d.h. sie kann nachträglich nicht mehr geändert werden.

Beispiel:

Definition eines Feld aus 31 Double-Elementen mit dem Variablennamen Temperatur:

```
double[] Temperatur;
Temperatur=new double[31];
```

oder kurz: `double[] Temperatur=new double[31];`

Wenn man das Feld gleich mit von 0 verschiedenen Werten initialisieren möchte, geschieht dies durch Angabe der Anfangswert in geschweiften Klammern, wie z. B.:

```
int[] gerade = {2,4,6,8,10,12};
String[] namen = {"Michael","Maike","Jörg","Benjamin","Sebastian");
```

Zugriff auf Elemente eines Feldes

Um auf die einzelnen Eintragungen - also die Daten bzw. *Komponenten* - eines Feldes zugreifen zu können, werden die Plätze, an denen die Daten stehen, nummeriert. Der erste Platz im Feld hat, wie oben angedeutet, die Nummer „0“, der zweite die Platznummer „1“ und der letzte Platz in einem Feld aus n Elementen hat die Platznummer „(n-1)“. Statt Platznummer verwendet man in der Informatik in diesem Fall den Begriff „*Index*“.

Auf die Elemente eines Feldes kann man mit `Variablenname[Index]` zugreifen. Wenn Index außerhalb von Feld-Anfang und Feld-Ende liegt, erhält man in der Regel eine Fehlermeldung.

Der Zugriff auf die Elemente eines Feldes entspricht dem Zugriff auf Elemente eines Strings, aber ein String ist wesentlich mehr als nur ein Feld aus einzelnen Zeichen.

Beispiele:

Mit Hilfe von `gerade[3]` liest man die Zahl 8 aus dem obigen Feld. namen[0] liefert „Michael“ zurück.
Nach der Anweisung `zahl = gerade[4];` hat die Variable zahl vom Typ int den Wert 10.
Einer Komponente eines Feldes weist man den Wert durch

```
Temperatur[3]=25.4;
```

zu. Dadurch wird der vorherige Wert an der Position mit dem Index 3 überschrieben.

Beispiel: Berechnung der Summe von 5 Zahlen

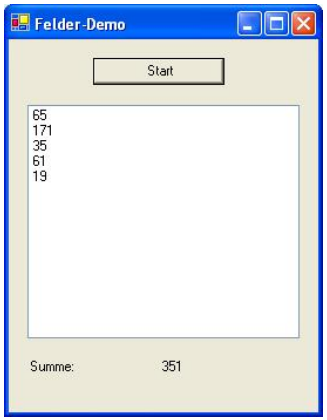
Projekttyp „InfoKursAnwendung“, Formular wie rechts abgebildet, zusätzlich noch die RandomNumbers Komponente aus den InfoKursTools. Eigenschaft Max von RandomNumbers auf 200 setzen.

```
void Button1Click(object sender, System.EventArgs e)
{
      // Feld erzeugen
      int[] zahlen=new int[5];

      // Zufallszahlen in Feld speichern
      // und in Listbox ausgeben
      listBox1.Items.Clear();
      for(int i=0;i<zahlen.Length;i++){
            zahlen[i]=randomNumbers1.Next();
            listBox1.Items.Add(zahlen[i]);
      }

      // Zahlen im Feld summieren
      int summe=0;  // Variable initialisieren
      for(int i=0;i<zahlen.Length;i++){
            summe=summe+zahlen[i];
      }

      // Summe ausgeben
      label2.Text=summe.ToString();
}
```



Felder sind in C# Objekte. Deshalb kann man ihre Länge über die Eigenschaft `Length` ermitteln.

Aufgaben:

1. Ändere das Beispielprogramm so ab, dass man beliebig viele Zahlen addieren kann. Dazu muss der Benutzer gefragt werden, wie viele zufällige Werte erzeugt werden sollen.
2. Ändere das Programm aus Aufgabe 1 so ab, dass der Benutzer jetzt Werte eingeben kann. Nutze dazu die InputBox-Komponente aus den InfoKursTools.
3. Schreibe ein neues Programm, basierend auf dem aus Aufgabe 1, dass jetzt das Produkt der eingegebenen Zahlen bestimmt.
4. Schreibe ein neues Programm, basierend auf dem aus Aufgabe 1, das den arithmetischen und geometrischen Mittelwert der eingegebenen Zahlen bestimmt.
5. Schreibe ein C#-Programm, das ein Feld mit Zufallszahlen füllt und anschließend das Maximum und das Minimum der Zahlen in diesem Feld bestimmt.
6. Schreibe ein Programm, das ein Viereck zeichnet. Zuerst müssen die Koordinaten eingelesen werden, danach kann gezeichnet werden. Verwende zum Zeichnen die Turtle Komponente aus den InfoKursTools.
7. Schreibe ein C#-Programm, das ein n-Eck zeichnet. Zunächst soll die Eckenzahl eingelesen werden, danach alle Eckpunkte zufällig erzeugt werden und danach gezeichnet werden.