TU Berlin, Institut für Mathematik
Thomas Slawig

Felder (Arrays)

Felder bestehen aus mehreren Komponenten *desselben Datentyps*. Sie können eindimensional (math. wie ein Vektor) oder mehrdimensional (wie eine Matrix oder ein höherdimensionales Objekt) sein. Es gibt

* statische Felder: Länge zur Compilezeit bekannt, also konstant
* dynamische Felder: Länge erst zur Laufzeit bekannt.

Statische Felder

- Deklaration:

  ```
  double x[10];
  ```

  oder

  ```
  const int n=10;
  double x[n];
  ```

  Zweidimensionales Feld mit $n \cdot m$ `double`-Werten:

  ```
  double x[n][m];
  ```

  Achtung: Indizes beginnen bei 0.

- Direkte Initialisierung:

  `double x[4] = {0.0, 1.0, 2.0, 3.0};` (liefert $x = (0.0, 1.0, 2.0, 3.0)$)

Allgemeines

- Zugriff (auf die i-te Komponente für $i = 0, \ldots, n-1$):

  ```
  y=x[i];
  ```

  Achtung (häufiger Fehler): auf die Komponente `x[n]` zugreifen (existiert nicht).

- Array-Zuweisungen erfolgen komponentenweise:

  ```
  for (i=0;i<n;i++)  x[i]=y[i];
  ```

- Ein Feld kann *nicht Rückgabewert einer Funktion* sein.
- Bei der Übergabe eines Feldes als Parameter an eine Funktion wird immer die Adresse des Feldes übergeben (kein `&`-Operator).

Dynamische Felder

- Deklaration in zwei Schritten:
  1. **Zeiger (Pointer)** deklarieren: `double *x;`
  2. Speicherbedarf anfordern (`n` muss nicht konstant sein): `x = new double[n];`

  oder zusammen: `double *x = new double[n];`

- Mehrdimensionale Felder: Feld von Pointern

  ```
  double **x = new double*[n];
  for(i=0;i<n;i++) x[i]=new double[m];
  ```

- Speicherplatz muss später auch wieder freigegeben werden:

  ```
  delete [] x;
  ```