Notizen AIK ProgTech 3: Function Pointer

Klaus Kusche

Was ist ein Function Pointer?

Ein Function Pointer ist ein Pointer, der nicht auf Daten,
sondern auf *Code* zeigt (nämlich auf den Code einer Funktion).

Achtung:

Pointer auf Funktionen und *Pointer auf Methoden* (die gibt's auch!) sind
zwei komplett *verschiedene* Dinge (Aufruf und interne Darstellung sind unterschiedlich),
man kann sie *nicht mischen*!

Wir behandeln hier nur Function Pointer, d.h. Pointer, die auf globale Funktionen
(oder statische Methoden einer Klasse) zeigen. Pointer auf Methoden lernen wir nicht.

Wozu verwendet man sie?

Function Pointer treten meist als *Parameter* einer Funktion bzw. Methode auf:
Man teilt der Funktion bzw. Methode mit, welche andere Funktion sie intern aufrufen
bzw. verwenden soll (vor allem dann, wenn derselbe Code für viele verschiedene
Funktionen verwendet werden soll).

Beispiele:

- Eine allgemeine Sortierfunktion, die Arrays mit beliebigen Elementen
  in beliebiger Reihenfolge (aufsteigend / absteigend) sortieren können soll,
  bekommt einen Pointer auf die Funktion, die zwei Elemente des Arrays vergleicht.
- Eine Methode, die eine beliebige Funktion auf alle Elemente einer Datenstruktur
  anwendet, bekommt einen Pointer auf die anzuwendende Funktion.

Wie sieht so ein Function-Pointer-Parameter aus?

```
void qsort(void *array, int anzahl, int bytes,
           int (*compFunc)(const void *, const void *));
```

Lies: Der vierte Parameter **compFunc** ist ein Pointer auf eine Funktion,
die zwei **const**-**void**-Pointer als Parameter hat und einen **int** als Returnwert liefert.

Wie übergibt man eine Funktion als Parameter?

Man gibt einfach den Namen der Funktion als Argument an: **`qsort(..., myCmp);`**

Für Funktionsnamen gilt dasselbe wie für Arrays:
Ein Arrayname ohne nachfolgende **[ ]** liefert einen Pointer auf den Anfang eines Arrays,
und ein Funktionsname ohne nachfolgende **( )** liefert einen Pointer auf die Funktion.

Wie ruft man einen Funktion Pointer auf?

Z.B. so: **`result = (*compFunc)(ptr1, ptr2);`**