Die C-Stdlib

Matthias Bracht, Steffen Lang, Marcus Echter, 17.05.2005

C ist ursprünglich eine sehr „kleine“ Sprache ohne spezifizierte Ein- und Ausgabe. Es gibt nur einfache Datentypen und wenige Schlüsselwörter. Bis auf funktionsartige Schlüsselwörter (sizeof) und compiler-known functions gibt es auch keine Funktionen. Für die Ein- und Ausgabe verwendet man dann entweder Systemaufrufe des Betriebssystems oder Funktionen der Standardbibliothek, die im Jahr 1989 vom ANSI als Teil des ANSI-C-Standards verabschiedet wurde und die einen einheitlichen Standard für oft benötigte Funktionen, Typen und Makros vorschreibt. Die Standardbibliothek schreibt folgende Headerdateien vor (Einbinden mit #include):

<assert.h>	Fehlersuche mit Testpunkten, wenn NDEBUG nicht gesetzt ist
<complex.h>	Komplexe Zahlen (C99)
<ctype.h>	Tests für Zeichenklassen, z.B. isupper(c), toupper(c)
<errno.h>	Codes von Systemfehlern
<fenv.h>	Kontrolle der Gleitpunktzahlen-Umgebung (C99)
<float.h>	Wertebereiche für Gleitpunktzahlen
<inttypes.h>	Konvertierungs- und Formatierungsfunktionen für Integers (C99)
<iso646.h>	alternative Operatorschreibweisen (z.B. and statt &&, and_eq statt &=) (NA1)
<limits.h>	Beschränkungen für Ganzzahlen
<locale.h>	Lokale Einstellungen (z.B. Sprache, Währung)
<math.h>	Mathematische Funktionen
<setjmp.h>	Globale Sprünge aus Funktionen heraus
<signal.h>	Signalbehandlung z.B. für Interrupts
<stdarg.h>	Bearbeitung von variablen Argumentlisten: va_list ap; va_start(va_list ap, lastarg); type va_arg(va_list ap, type); va_end(va_list ap);
<stdbool.h>	Verwendung von Typ bool und true (1) und false (0) (C99)
<stddef.h>	Typdefinitionen, NULL und errno
<stdint.h>	Integertypen vorgegebener Breite (z.B. intN_t mit N=8, 16, 32, 64) (C99)
<stdio.h>	Ein-/Ausgabe
<stdlib.h>	Stringumwandlung, Zufallszahlen (z.B. rand), Speicherverwaltung (z.B. malloc, free), Environment-Funktionen (z.B. exit, system), Such-/Sortierfunktionen (z.B. qsort), Integer-Arithmetik (z.B. abs)
<string.h>	Zeichenketten-Verarbeitung
<tgmath.h>	Typengenerische Mathematikfunktionen (C99)
<time.h>	Zeit- und Datumsfunktionen
<wchar.h>	stdio.h- und string.h-Funktionen für erweiterten Zeichensatz (z.B. fgetwc())(NA1)
<wctype.h>	ctype.h-Funktionen für erweiterten Zeichensatz (z.B. iswupper(c), towupper(c)) (NA1)

Universale Bibliotheken binden allerdings große Mengen von Code ein, so dass es besonders im Bereich der Embedded Systems sinnvoll sein kann, auf Funktionen aus der Standardbibliothek zu verzichten, um eine geringere Codegröße und Laufzeit zu erzielen.

Ein- / Ausgabe

Dateien sind in C sequentielle Datenströme. Bevor mit einer Datei gearbeitet werden kann muss diese eröffnet werden. Dazu dient die Funktion fopen aus stdio.h. Eine Ausnahme bilden die drei Standardströme stdin, stdout und stderr, welche vom Betriebssystem vordefiniert sind und nicht extra eröffnet werden müssen.
Zur formatierten Ein- bzw. Ausgabe dienen die Funktionen fscanf bzw. fprintf sowie leicht abgewandelte Varianten. Dagegen wird durch fgetc bzw. fputc eine zeichenweise Ein- bzw. Ausgabe realisiert, wobei es auch hier noch weitere ähnliche Funktionen gibt.

Strings in C

Strings besitzen in C keinen eigenen Datentyp, sondern werden in der Standardbibliothek und vielen Funktionen als Arrays vom Typ char dargestellt, wobei als Terminierungszeichen am Ende eine '\0' angefügt wird. Theoretisch ist jedoch auch eine andere Darstellung von Zeichenketten möglich.
In der Standarddefinitionsdatei string.h werden viele Funktionen zum Arbeiten mit Strings deklariert. Darunter sind beispielsweise strcpy zum Kopieren oder strcat zum Verketten von Zeichenketten. Für die meisten dieser Funktionen gibt es auch eine Variante, welche eine zusätzliche Zahl als Argument erwartet, die die maximale Länge der bearbeiteten Strings angibt. Diese sollten verwendet werden um zu verhindern, dass man die Arraygrenzen überschreitet.
Darüber hinaus gibt es in stdlib.h noch Funktionen die der Konvertierung von Strings in Zahlen dienen, wie atoi für int oder atof für double.
Die häufigste Fehlerquelle bei der Arbeit mit Strings ist, dass die Länge der Arrays nicht beachtet wird und es somit zu Speicherfehlern kommt.