

Lernzettel

Rechnerdarstellung: Zahlen- und
Zeichendarstellung, Binärformat und Codierung

Universität: Technische Universität Berlin
Kurs/Modul: Einführung in die Informatik
Erstellungsdatum: September 19, 2025



Zielorientierte Lerninhalte, kostenlos!
Entdecke zugeschnittene Materialien für deine Kurse:

<https://study.AllWeCanLearn.com>

Einführung in die Informatik

Lernzettel: Rechnerdarstellung: Zahlen- und Zeichendarstellung, Binärformat und Codierung

(1) Grundlagen der Zahlen- und Zeichendarstellung im Rechner.

In einem Computer werden Werte als Bitfolgen gespeichert. Die Art der Darstellung hängt von der Zahlart und dem gewählten Format ab. Wichtige Konzepte:

- Ganzzahlen werden in vorzeichenbehafteten Formaten dargestellt. Die gängigste Darstellung ist das Zweierkomplement.
- Reelle Zahlen verwenden Gleitkommadarstellungen, typischerweise nach dem IEEE-754-Standard (z. B. 32-Bit oder 64-Bit).
- Zeichen werden durch Codierungen wie ASCII, Unicode/UTF-8 gespeichert.

(2) Ganzzahlen: Zweierkomplement und Wertebereich.

Sei eine n-Bit-Ganzzahl mit Bitfolge $b_{n-1}b_{n-2} \dots b_0$. Der Wert ist

$$\text{Wert} = -b_{n-1} 2^{n-1} + \sum_{k=0}^{n-2} b_k 2^k.$$

Beispiel: 8-Bit-Zweierkomplement der Zahl -18 . Zunächst $18 = 00010010_2$. Invertieren $\rightarrow 11101101_2$, $+1$ ergibt 11101110_2 . Also: $-18 = 11101110_2$.

Der Zahlenbereich einer n-Bit-Zweierkomplementdarstellung ist

$$-2^{n-1} \leq x \leq 2^{n-1} - 1.$$

Hinweis: Überläufe treten auf, wenn das Ergebnis außerhalb dieses Bereichs liegt.

(3) Reelle Zahlen: Gleitkomma (IEEE 754).

Gleitkommazahlen bestehen grundsätzlich aus Vorzeichen, Exponent und Mantisse in der Form

$$(-1)^s \cdot (1.f) \cdot 2^{E-\text{bias}},$$

wobei:

- s das Vorzeichenbit ist,
- E der Exponent ist (mit Bias),
- f die Mantisse als Bruchteil bezeichnet wird ($0.f$ bis $0\dots$),
- der Bias dem Exponentenraum entspricht (z. B. 127 bei 8 Exponentbits in single precision).

Für Normalzahlen gilt $E \in [1, 254]$ (Single) bzw. $E \in [1, 2046]$ (Double). Subnormale Zahlen haben $E = 0$ und $\text{Wert} = (-1)^s \cdot 0.f \cdot 2^{1-\text{bias}}$.

Beispiele:

- Single-Precision (32 Bit): 1 Bit Vorzeichen, 8 Bit Exponent, 23 Bit Mantisse.
- Double-Precision (64 Bit): 1 Bit Vorzeichen, 11 Bit Exponent, 52 Bit Mantisse.

Formeln zu konkreten Werten bitte außerhalb des Fließtexts als einfache Gleichungen darstellen, wie oben, und nicht nebeneinander.

(4) Binäres Format, Endianness und Basis-Werkzeuge.

Bits und Bytes bilden die grundlegende Speichereinheit:

- 1 Byte = 8 Bits.
- Hexadezimalnotation erleichtert das Ablesen langer Binärfolgen.

Endianness: Reihenfolge der Bytes in Mehrbyte-Werten.

- Big-endian: das am höchstenwertige Byte wird zuerst gespeichert.
- Little-endian: das niederwertigste Byte wird zuerst gespeichert.

Beispiel für eine 16-Bit-Zahl $N = 0x1234$:

- Big-endian Speicherabfolge: 12 34
- Little-endian Speicherabfolge: 34 12

(5) Zeichencodierungen: ASCII, Unicode, UTF-8.

- ASCII: 7-Bit-Codierung für die ersten 128 Zeichen; flach, einfach, inkompatibel mit vielen Zeichen außer dem Basis-ASCII.
- Unicode: universeller Zeichensatz. UTF-8 ist eine variable Länge Codierung von 1 bis 4 Bytes pro Zeichen.

Beispiele für UTF-8-Codierung:

- U+0041 (A) -> 0x41
- U+00E9 (é) -> 0xC3 0xA9
- U+1F600 (😊) -> 0xF0 0x9F 0x98 0x80

(6) Auswirkungen in der Praxis.

- Speicherform vs. Datei-/Netzwerk-Format: Byte-Order und Codierung müssen konsistent sein.
- Zahlenüberlauf, Rundungsfehler bei Gleitkomma, Loss of Significance.

- Zeichencodierung muss beim Lesen/Schreiben einer Datei übereinstimmen, sonst treten Garbled Text oder Fehler auf.

(7) Checkliste / Merke.

- Zweierkomplement: Vorzeichen durch höchstes Bit, Wertebereich $-2^{n-1} \dots 2^{n-1} - 1$.
- Gleitkomma: Sign, Exponent (Bias), Mantisse; Normalzahlen vs. Subnormale.
- Codierungen: ASCII vs. Unicode/UTF-8; Byte-Order beachten.
- Bei Datei-/Netzwerkkommunikation immer Endianness und Codierung abgleichen.