3. Konversion

3.1 implizit (automatisch) (b) Zwischen Integral- und Gleitkomma-Typen: Vom „schwächeren“ zum „stärkeren“ Ausdruck ohne Compiler-Warnung

bool → char → short → int → long int → float → double → long double

3.2 explizit (cast): Erzwingen der Konversion: ⟨`Typ`⟩ `(...)`

`dat_cast.cc`

Einschub A: Lineare Gleichungssysteme

gegeben: $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$; gesucht: $x \in \mathbb{R}^n$ mit $A\,x = b$

Gauß-Algorithmus (kurz):

(a) Dreiecksform.

(b) Rückwärtseinsetzen.

Verfahren von Gauß–Jordan.

Einschub B: Interpolation

- experimentelle Daten: Approximation an (unbekannte) Funktion
- gesucht: Integral über komplizierte Funktion
  Idee: 1. Approximiere Funktion; 2. Integriere
- Laplace: Konstruktion des eindeutigen Polynoms
- Neville: Rekursion zur Berechnung des Interpolations-Polynoms an einem vorgegebenen Punkt x
  $P_{n,n}$: Approximation an $f(x)$, die die Information von $x_0, \ldots, x_n$ enthält.

(a) $P_{j,0} = f_j$

(b) $P_{j,k} = P_{j,k-1} + \frac{P_{j,k-1} - P_{j-1,k-1}}{\left(\frac{x - x_{j-k}}{x - x_j} - 1\right)}$

$k = 0, 1, \ldots, n; \qquad j = k, \ldots, n$

- Kubische Splines $S(x)$:
  - Motivation: möglichst glatte Kurve durch Punkte + bessere Konvergenz