- Typsystem $\rightarrow$ definiert die Regeln fuer die konsistente Typisierung von Termen eines Kalkuels | einer konkreten Programmiersprache
- Regeln $\rightarrow$ definieren die algorithmische Realisierung von Typ–Checkern bzw. von Typ–Inferenz Systemen
- Typ–Checker $\rightarrow$ ueberpruefen getypte Programme | Terme auf Typ–Konsistenz
- Typ–Inferenz Systeme leiten aus bekannten Typen von Operanden oder primitiven Operatoren eine (moeglichst) vollstaendige Typisierung von Programme her
- Statisches Typsystem $\rightarrow$ die Typ–Inferenz bzw. das Typ–Checking findet vollstaendig vor Programmausfuehrung statt $\rightarrow$ z.B. im Zuge der Compilation
- Dynamisches Typsystem $\rightarrow$ die Typueberpruefung findet vollstaendig zur Laufzeit statt $\rightarrow$ Bspe sind FL | KiR | LISP | SCHEME etc.
- ein Programm (Term) ist wohl getypt falls zur Laufzeit keine Typinkonsistenzen auftreten
- ein Typsystem ist sicher falls jedes getypte Programm auch wohl getypt ist $\rightarrow$ sichere Typsysteme gestatten vollstaendiges statisches Typ–Checking
- eine Sprache ist streng getypt falls der Typ–Checker nur wohl getypte Programme akzeptiert | zulaesst, anderenfalls ist sie schwach getypt $\rightarrow$ es koennen Typinkonsistenzen auftreten