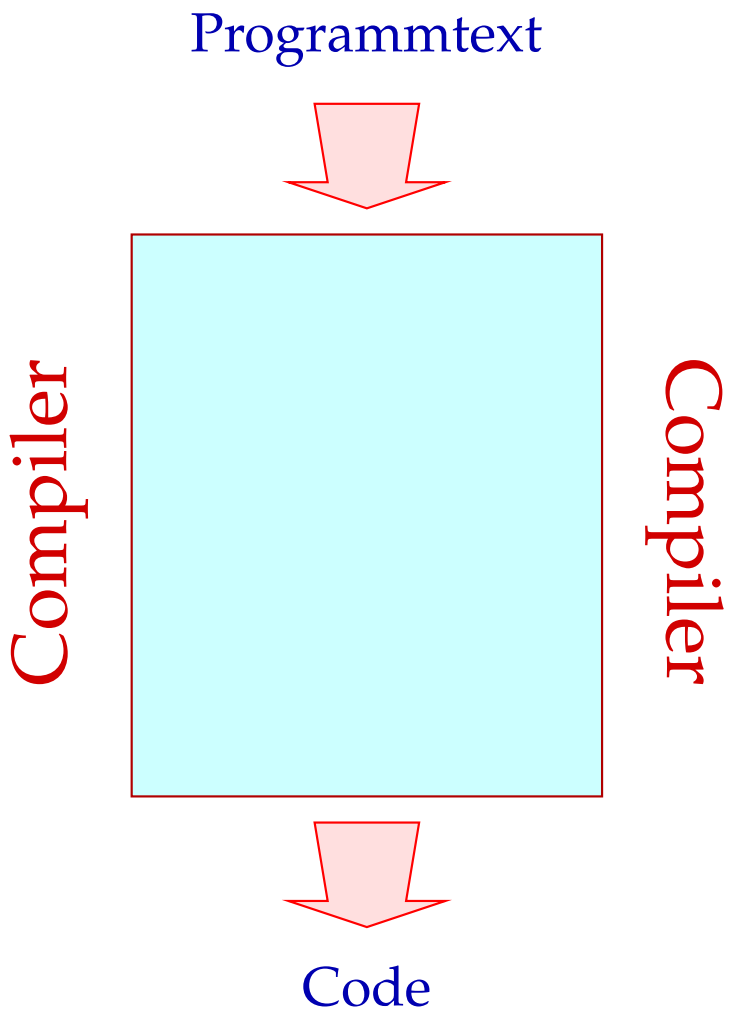
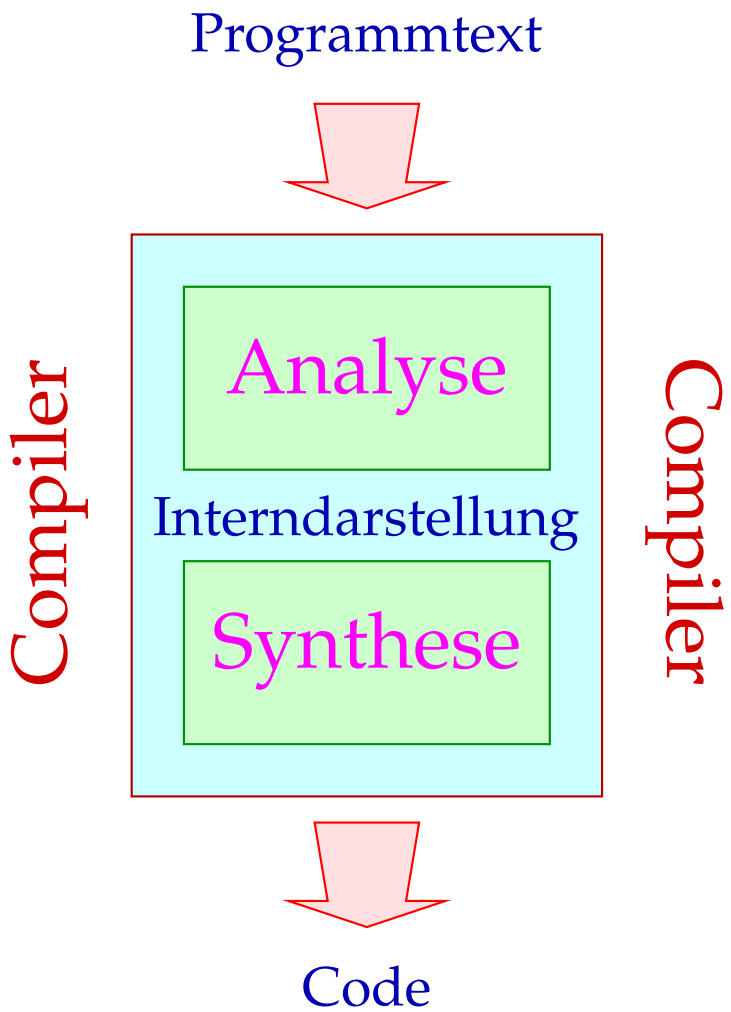
Die Analyse-Phase

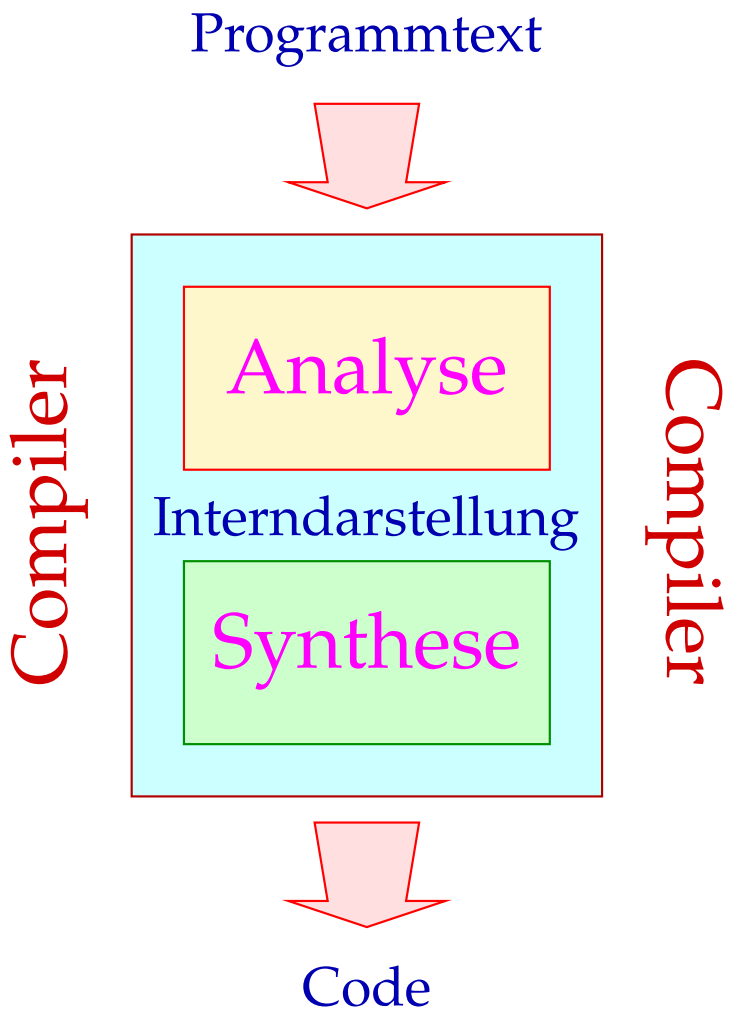
Orientierung:



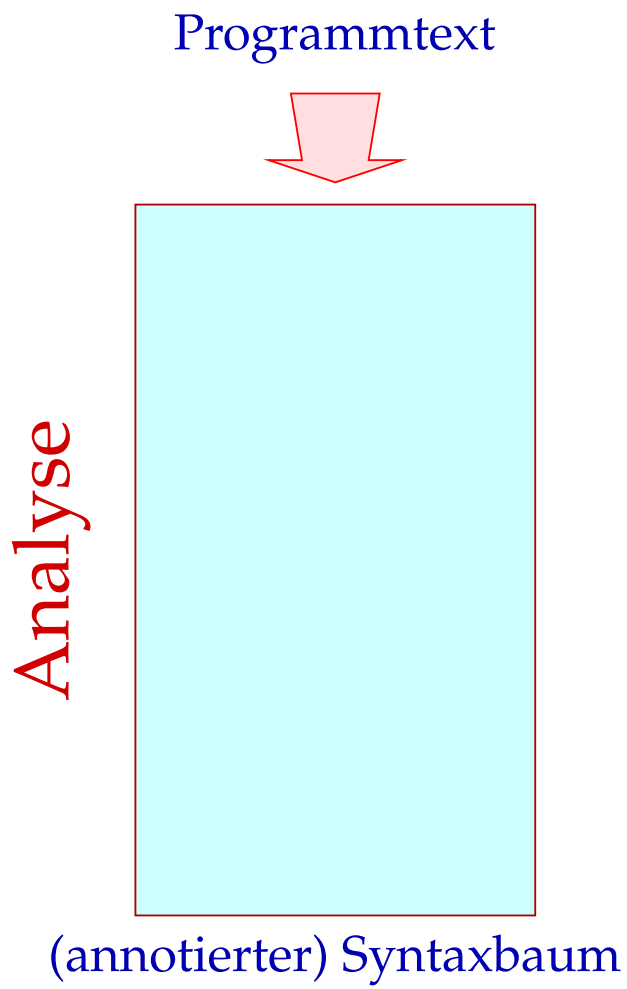
Orientierung:



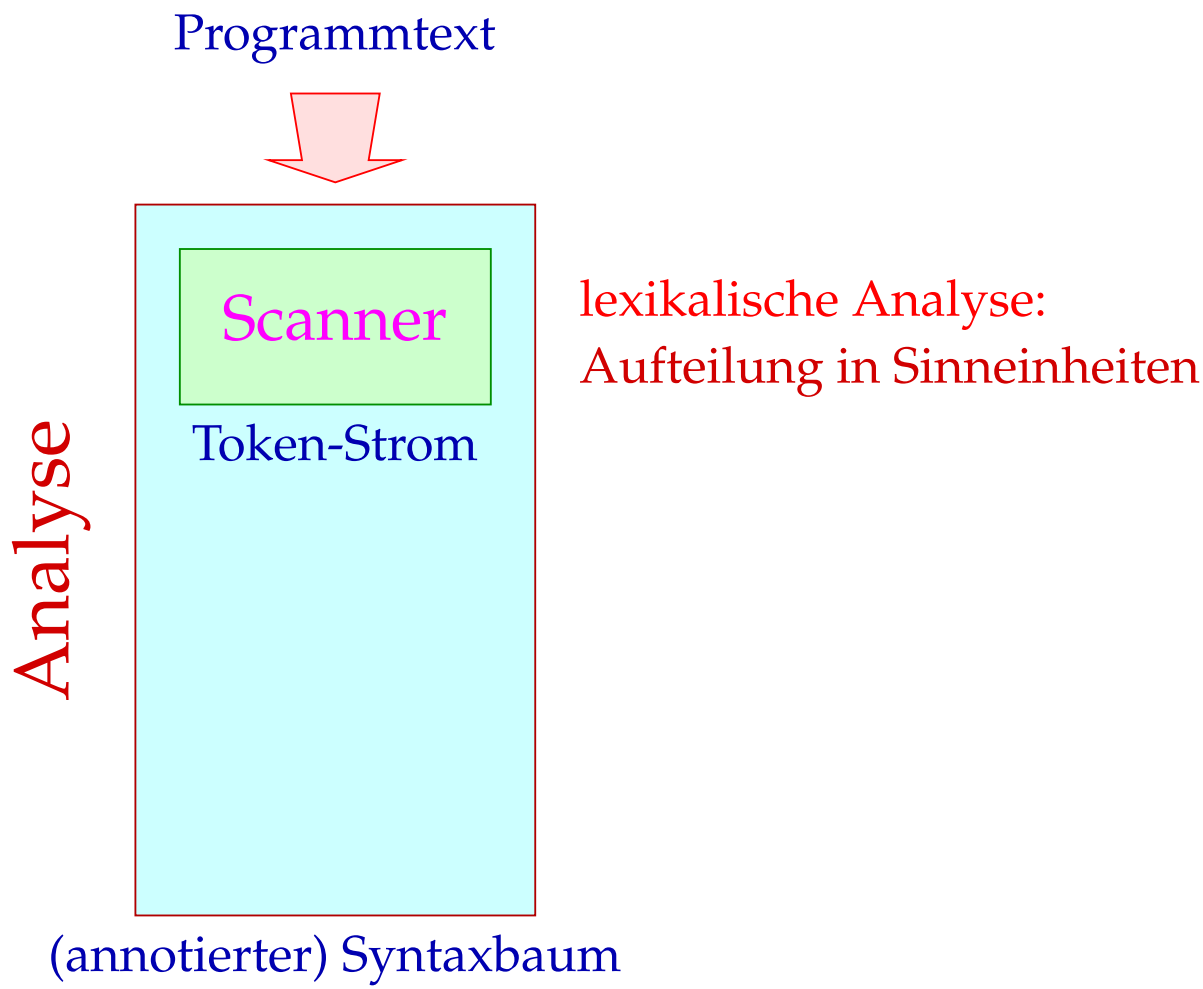
Orientierung:



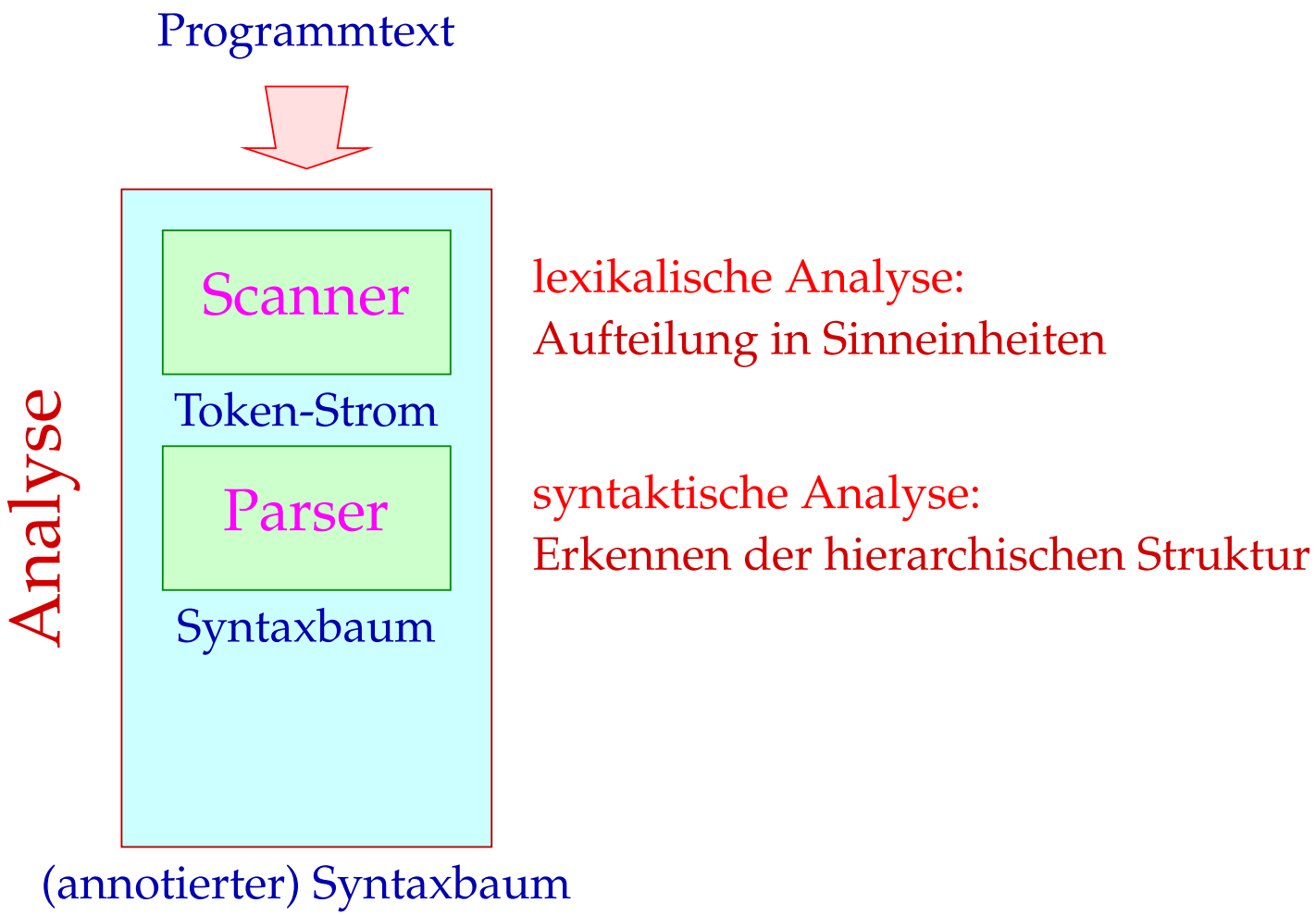
Nachdem wir Prinzipien der Code-Erzeugung kennen gelernt haben, behandeln wir nun die Analyse-Phase :-)



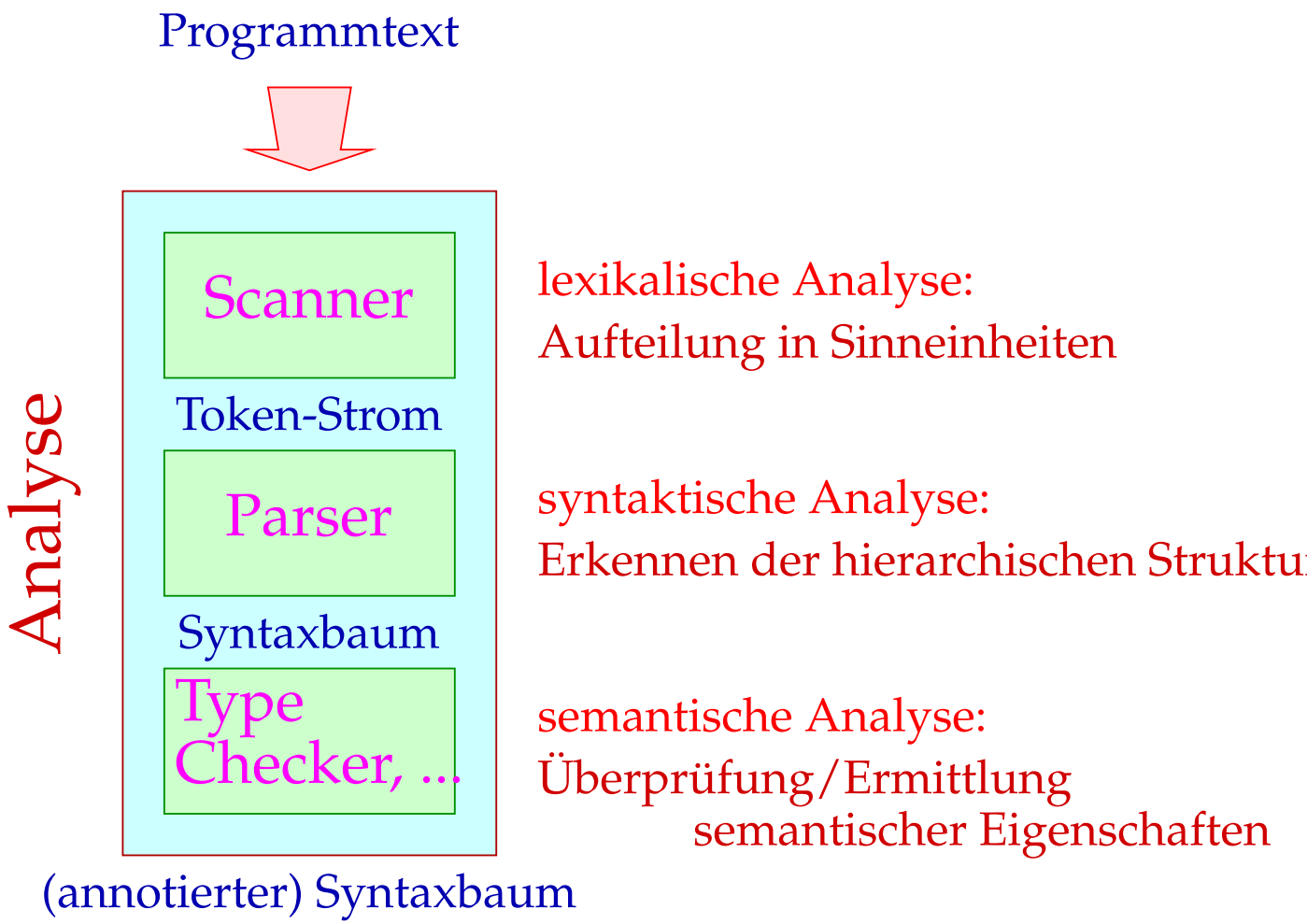
Nachdem wir Prinzipien der Code-Erzeugung kennen gelernt haben, behandeln wir nun die Analyse-Phase :-)



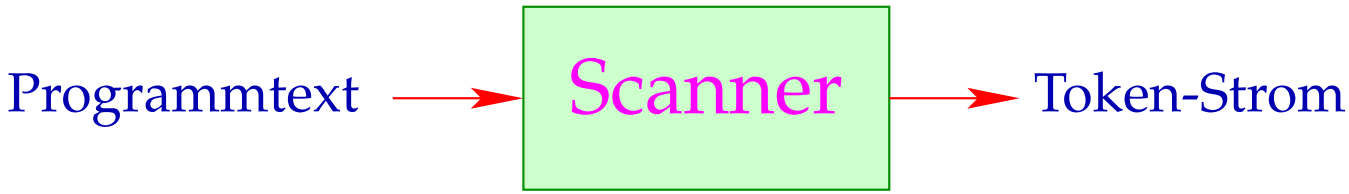
Nachdem wir Prinzipien der Code-Erzeugung kennen gelernt haben, behandeln wir nun die Analyse-Phase :-)



Nachdem wir Prinzipien der Code-Erzeugung kennen gelernt haben, behandeln wir nun die Analyse-Phase :-)

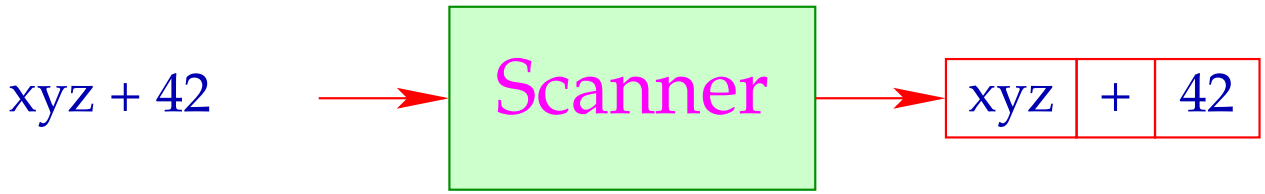


1 Die Lexikalische Analyse



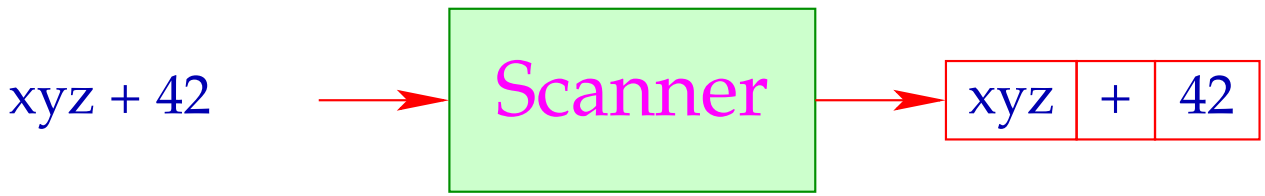
Ein Token ist eine Folge von Zeichen, die zusammen eine Einheit bilden.

1 Die Lexikalische Analyse



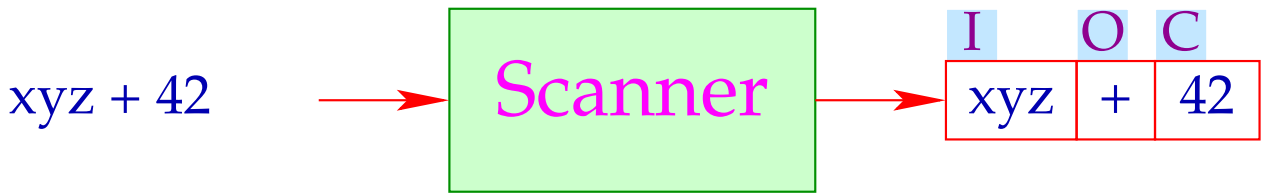
Ein Token ist eine Folge von Zeichen, die zusammen eine Einheit bilden.

1 Die Lexikalische Analyse



- Ein Token ist eine Folge von Zeichen, die zusammen eine Einheit bilden.
- Tokens werden in Klassen zusammen gefasst. Zum Beispiel:
  - → Namen (Identifier) wie `xyz`, `pi`, ...
  - → Konstanten wie `42`, `3.14`, `"abc"`, ...
  - → Operatoren wie `+`, ...
  - → reservierte Worte wie `if`, `int`, ...

1 Die Lexikalische Analyse



- Ein Token ist eine Folge von Zeichen, die zusammen eine Einheit bilden.
- Tokens werden in Klassen zusammen gefasst. Zum Beispiel:
  - → Namen (Identifier) wie `xyz`, `pi`, ...
  - → Konstanten wie `42`, `3.14`, `"abc"`, ...
  - → Operatoren wie `+`, ...
  - → reservierte Worte wie `if`, `int`, ...

Sind Tokens erst einmal klassifiziert, kann man die Teilwörter vorverarbeiten:

- Wegwerfen irrelevanter Teile wie Leerzeichen, Kommentaren,...
- Aussondern von Pragmas, d.h. Direktiven an den Compiler, die nicht Teil des Programms sind, wie include-Anweisungen;
- Ersetzen der Token bestimmter Klassen durch ihre Bedeutung / Interndarstellung, etwa bei:

  $\rightarrow$ Konstanten;

  $\rightarrow$ Namen: die typischerweise zentral in einer Symbol-Tabelle verwaltet, evt. mit reservierten Worten verglichen (soweit nicht vom Scanner bereits vorgenommen :-) und gegebenenfalls durch einen Index ersetzt werden.

$\Longrightarrow$ Sieber