Gliederung des Kapitels 1.3

1.3 Daten und ihre Strukturierung

~~1.3.1 Elementare Datentypen~~

~~1.3.2 Konstruktoren (für Datenbereiche)~~

~~1.3.3 Relationen, Graphen, Referenzen~~

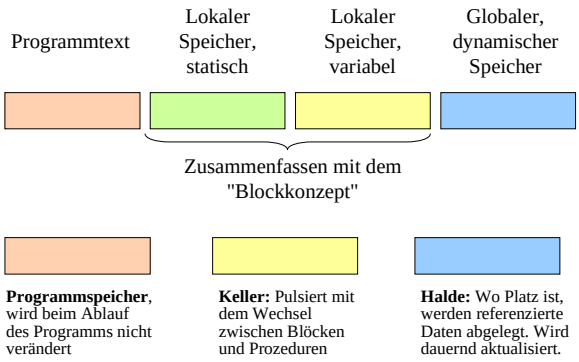
1.3.4 Keller und Halde

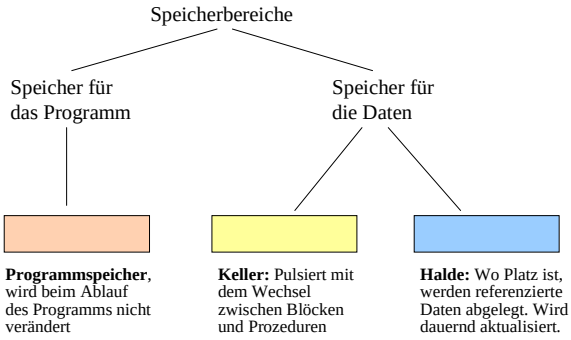
1.3.4 Keller und Halde

Variablen müssen irgendwo abgelegt werden. Wir haben drei Möglichkeiten kennen gelernt:

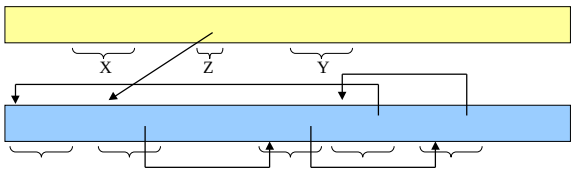
1. Deklaration im Programm mit genauen Angaben über Typ und Größe zur Übersetzungszeit (z.B.: K: integer).
2. Auswertung einer Deklaration mit Festlegung von Größe und Typ zur Laufzeit (z.B.: array [I..J] of ...).
3. Anlegen und Löschen im Anweisungsteil des Programms (mit dem Allocator new).

Da auch das Programm irgendwo abgelegt werden muss, gibt es mehrere Speicherbereiche, die zu einem Programm gehören, wobei durch das sog. Blockkonzept die Bereiche "1." und "2." zusammengefasst werden:






```
program ... X: integer; Z: ...; ...begin declare Y: ...; Z := new TYP; ... end.
```



Wie pulsiert der Keller durch Blöcke und Prozeduren? siehe 1.4.1.
Wie werden die Daten in der Halde verwaltet? siehe Abschnitt 3.1.
Wie werden viele Programme verwaltet? siehe Abschnitt 3.1.

Insbesondere vor Anfängern soll eine Programmiersprache die Verwaltung der Daten und die Details, wie ein Programm vom Rechner ausgeführt wird, verbergen. Wer programmiert, möchte aber hierauf Einfluss nehmen, weil die Effizienz eines Programms davon abhängt.

Ada bietet hierfür Blöcke (und Prozeduren und Pakete), um die zwischenzeitlich benötigten Daten hintereinander im Speicher anzulegen, und die Halde (in Ada "storage pool" genannt) an, um access-Datentypen anzulegen, zu bearbeiten und wieder zu löschen.

Wir wenden uns im folgenden Kapitel 1.4., das mit dem Kapitel 1.5 über die Sprachelemente von Ada verzahnen, den Programmierkonzepten zu.