Variable (von lateinisch varius „verschieden“)
In der Informatik bezeichnet man als Variable meist einen logischen Speicherplatz mit dessen Wert. „Logischer Speicherplatz“ ist hier im Gegensatz zu „physischer Speicherplatz“ zu verstehen: Bei einer Variablen will man sich nicht darum kümmern müssen, wie viele Speicherzellen sie in einem Speicher belegt und wo sie im Speicher steht.

Jede Variable besitzt einen Namen, unter dem man sie ansprechen kann und ihren Wert verändern kann.

In den typisierten Programmiersprachen, und solche betrachten wir in der Regel, muss einer Variablen bei der Deklaration ein Datentyp zu geordnet werden. Der Datentyp legt fest, welche Elemente als Werte der Variablen auftreten und welche Operationen auf die Variable angewendet werden dürfen.

Datentypen bei VIPS:
Wahrheitswert (true oder false), Zeichen, Zeichenkette (bestehend aus mehreren Zeichen), Ganzzahl, Kommazahl

Bsp.: Die Inhalte der Variablen vom Typ Ganzzahl bzw. Kommazahl können addiert werden, für Variablen vom Typ Zeichen macht eine Addition keinen Sinn.

Deklaration von Variablen:
Zu Beginn eines jeden (Unter)-Programms müssen die verwendeten Variablen deklariert werden, d.h. dem Computer muss mitgeteilt werden, welche Variablen angelegt werden sollen, wie die Variablen heißen und welcher Speicherplatz für die Variable reserviert werden soll.

Der rechte Becher soll eine Variablendeklaration darstellen. Es wird eine Variable mit Namen Zähler vom Datentyp Ganzzahl angelegt. Dazu nehmen wir einen Becher der „Größe“ Ganzzahl und beschriften den Becher mit dem Variablennamen „Zähler“. Der Computer reserviert also einen Speicherplatz der Größe Ganzzahl und kann über den Namen Zähler auf den Inhalt des Speicherplatzes zu greifen.

Initialisierung von Variablen:
Wir haben die Variable mit Namen Zähler vom Typ Ganzzahl angelegt. Wir wissen allerdings noch nichts über den Inhalt der Variablen, dieser ist nicht zwangsläufig Null oder leer. Die erste Wertzuweisung einer Variablen nennt man Initialisierung, erst danach wissen wir, welchen Inhalt die Variable hat.

Beispiel: Zähler ← 0; *(An dieser Stelle bekommt die Variable den Wert 0 zugewiesen).*

Die Wertzuweisung einer Variablen geschieht immer von rechts nach links, d.h. links steht der Variablenname und rechts der zugewiesene neue Inhalt. Der ehemalige Inhalt ist nach der Wertzuweisung nicht mehr vorhanden. Der Inhalt kann auch in Form einer Variablen stehen.

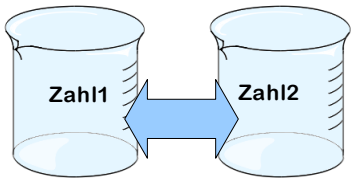
Beispiel: Zähler ← alterZähler;
(An dieser Stelle bekommt die Variable Zähler den Wert von alterZähler zugewiesen, alterZähler behält natürlich seinen Inhalt)

Achtung: Bei der Wertzuweisung mit Variablen müssen alle Variablen vom gleichen Datentyp sein. Eine Zuweisung unterschiedlicher Datentypen ist in der Regeln nicht möglich. Um aus einer Zeichenkette eine Ganzzahl zu machen, kann man bei VIPS den Befehl StrToInt nutzen. Dies ist eine Typumwandlung (cast).
Erhöhung des Zählers

Beispiel: Zähler ← Zähler + 1;
Diese Zuweisung erhöht den Zähler um 1. Wie funktioniert das?
Der Computer beginnt nach dem Pfeil und besorgt sich den Inhalt von Zähler, addiert eine Eins dazu und speichert das Ergebnis als Inhalt in Zähler.

Aufgabe: Der Inhalt zweier Variablen soll vertauscht werden.

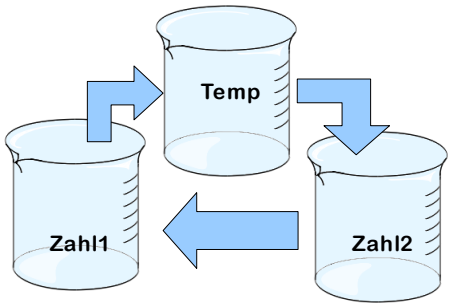
1. Versuch:
Zahl1 ← Zahl2;
Zahl2 ← Zahl1;



Der 1. Versuch scheitert allerdings, da nach der Anweisung Zahl1 ← Zahl2 in beiden Variablen der Inhalt von Zahl2 steht. Der Inhalt von Zahl1 ist verloren gegangen.

2. Versuch:
Neue Variable Temp vom Datentyp Ganzzahl deklarieren;
Temp ← Zahl1;
Zahl1 ← Zahl2;
Zahl2 ← Temp;

Dieser Umweg über die „Hilfs-"Variable löst die Aufgabe.



Globale und Lokale Variablen:

Grundsätzlich unterscheiden wir zwei verschiedene Arten von Variablen, globale und lokale Variablen.

- globale Variablen:
  - werden zu Beginn des Programms deklariert
  - existieren den gesamten Programmverlauf
  - können auch in Unterprogrammen ohne weiteres verwendet werden
- lokale Variablen:
  - werden zu Beginn eines Unterprogramms deklariert und initialisiert
  - existieren nur während das Unterprogramm ausgeführt wird
  - können nicht in weiteren Unterprogrammen verwendet werden
- Parameter:
  - verhalten sich wie lokale Variablen, sind aber keine lokale Variablen, da sie „von außen angesprochen werden können".
  - werden bei Unterprogrammen verwendet, die bestimmte Daten aus den aufrufenden Programmen verarbeiten sollen.
  - Die Deklarierung und Initialisierung erfolgt beim Aufruf des Unterprogramms.
- Konstanten:
  - werden zu Beginn des (Unter-)Programms deklariert und initialisiert.
  - Nach der Initialisierung kann der Konstanten kein (neuer) Wert zugewiesen werden, daher sind Konstanten keine Variablen im engeren Sinne. Der Inhalt ist nicht variable.

Existieren zu einem Zeitpunkt des Programmablaufs eine lokale und eine globale Variable mit gleichem Namen, so kann in dem entsprechenden Unterprogramm nur der Wert der lokalen Variable verändert werden. Die globale Variable kann daher nicht in diesem Unterprogramm angesprochen werden. Bei der Programmierung empfiehlt es sich insgesamt möglichst ohne globale Variablen auszukommen.

Vorteile von lokalen Variablen:

In der modernen Programmierung benutzen viele Programme vorgefertigte (Unter-)Programme. Da wir nicht wissen, welche Variablen diese Unterprogramme benutzen, wäre eine ausschließliche Programmierung mit globalen Variablen unmöglich. Wir wissen ja nicht, welche von unseren globalen Variablen das (Unter-)Programm auch verwendet. Wie der Zufall es so will, verändert das (Unter-)Programm unsere Variablen. Diese Veränderung kann später nicht rückgängig gemacht werden, der Inhalt unserer Variablen ist verloren. Über diese unerwünschte Veränderung werden wir nicht informiert, so dass formal richtig umgesetzte Algorithmen nicht die gewünschten Ergebnisse erzeugen. Ein Fehlersuche ist dabei sehr mühsam und hilft insgesamt nicht bei der Problembeseitigung. Wir wissen ja nicht, welche unserer Variablen noch zufällig den richtigen Wert besitzen und eventuell erst beim nächsten Aufruf des Unterprogramms verändert werden.

Mit lokalen Variablen kann das nicht passieren, da unsere Variablen nicht in (Unter-)Programmen benutzt werden können. Namenskonflikte sind daher ausgeschlossen.

Aber wir können doch globale Variablen benutzen? Jein, natürlich können wir das machen. Aber eventuell wollen wir zu einem späteren Zeitpunkt unsere Programme als Unterprogramme in anderen Programmen benutzen. Dann müssten wir die alten Programme zunächst umschreiben. Dieses geschieht häufiger als gedacht und eine Umschreibung ist bei komplexeren Programmen nicht unbedingt einfach. Die „Philosophie“ der Programmierung verlangt aus diesen Gründen die ausschließliche Verwendung lokaler Variablen. Programme sollen nicht ständig neu geschrieben werden, sondern sie sollen einmal korrekt implementiert werden und danach von anderen Programmen genutzt werden.