

LU02b - Variablen deklarieren

Vorschrift und Wert

$$x = y + z$$

Aus der Algebra kennen sie bereits solche Rechnungen mit Platzhaltern. Sie wissen, dass jeder Buchstabe stellvertretend für eine bestimmte Zahl steht. Wenn wir diese Gleichung etwas genauer betrachten sehen wir:

- Eine Vorschrift, was zu berechnen ist: $y+z$
- Den Ort, an dem das Ergebnis gespeichert wird: $x=$

Damit sie das Ergebnis der Gleichung berechnen können, müssen die Platzhalter einen bestimmten Wert erhalten. Wenn sie nun wissen, dass $y=4$ und $z=9$ ist, können sie x berechnen.

Variablen im Programm

In der Programmierung nennen wir die Platzhalter **Variablen**. Variablen reservieren Speicherplatz um darin Werte zu speichern. Wenn sie das Ergebnis von $5+7$ speichern wollen, müssen sie eine passende Variable bereitstellen.

In den meisten Programmiersprachen müssen die Variablen **deklariert** werden, bevor sie zum ersten mal verwendet werden.



Die Deklaration legt den Namen (Bezeichner) und Datentyp einer Variable fest. Durch die Deklaration kennt der Compiler die Variable und wir können Sie im Programm verwenden.

Name

Um die Variable zu verwenden, benötigt sie einen Namen (Bezeichner). Der Name einer Variable sollte möglichst sprechend sein. Das bedeutet, dass ihr Name etwas über den Inhalt bzw. Verwendungszweck aussagt. Die Variablennamen sind immer klein geschrieben. Der Name besteht in der Regel aus Buchstaben, Zahlen und dem Underline (_).



Verwenden Sie niemals Leerzeichen, Umlaute oder Sonderzeichen in einem Variablennamen.

Gute Variablennamen

- vorname
- preis
- laenge

Schlechte Variablennamen

- **i** (Nicht sprechend)
- **länge** (Umlaut)
- **grad°** (Sonderzeichen)
- **Anzahl** (gross geschrieben)

Literale (Konstanten mit einem Namen)

In einigen Sprachen hat es sich ebenso zum Standard entwickelt, dass die Namen von Literalen ausschliesslich in Grossbuchstaben geschrieben werden (z.B. MAX_VALUE).

Übung zu Variablennamen

In einem Programm benötigen sie verschiedene Variablen. Legen sie fest wie die Variablen für folgende Aufgaben heissen sollen:

1. Die Variable speichert die Anzahl der Noten in einem Modul
2. Die Variable speichert eine Schulnote
3. Aus der Anzahl und dem Total der Noten wird der Durchschnitt berechnet. Diese Variable soll den Durchschnitt speichern.
4. Sie schreiben ein Programm, dass Lieferscheine erstellt. Dazu benötigen sie Variablen um
 - die Bezeichnung eines Artikels zu speichern
 - den Preis eines Artikels zu speichern
 - die Einheit zu speichern, in denen der Artikel verkauft wird (z.B. Stück, Meter, ...)
 - das Total (berechnet aus Preis * Menge) zu speichern
 - speichern zu können, ob es sich beim Artikel um eine Aktion (Preisreduktion) gehandelt hat oder nicht

Datentyp

Der Datentyp bestimmt die Speicherung und spätere Verwendung der Variablen. Je nach Programmiersprache werden unterschiedliche Datentypen unterstützt und die einzelnen Datentypen heissen unterschiedlich. Grundsätzlich lassen sich die Datentypen wie folgt einordnen:

- Textvariablen
 - Ein Zeichen (Buchstabe, Ziffer, Sonderzeichen)
 - String aus mehreren Zeichen
- Zahlen

- Ganzzahlen
- Fließkommazahlen die auch Dezimalbrüche enthalten können
- Spezielle Datentypen
 - z.B. Ja-Nein (True-False)

Je nach Programmiersprache wird zusätzlich zwischen einer binären und einer dezimalen Speicherform unterschieden.

Übung zu Datentyp

Bestimmen Sie für die Variablen der vorherigen Übung die Datentypen gemäss folgender Liste:

- Zeichen
- Zeichenkette
- Ganzzahl
- Dezimalzahl
- Ja-Nein

Inhalt bzw. Wert

Jede Variable hat zu jedem Zeitpunkt einen bestimmten Wert. Wenn sie eine Variable vereinbaren (Fachbegriff: deklarieren), so wird im Hauptspeicher ein kleines Stück Speicherplatz für diese Variable reserviert. Wird einer Variablen bei der Deklaration kein Wert zugewiesen, so unterscheidet sich das Verhalten resp. der initiale Wert der Variable je nach Programmiersprache.

Initialisierung

Bei der Initialisierung wird einer Variable direkt bei der Deklaration ein Wert zugewiesen. Dadurch hat die Variable schon beim Start des Programmes einen genau definierten Wert. Dies ist aber nicht in allen Programmiersprachen möglich. In Visual Basic zum Beispiel muss die Initialisierung auf einen Startwert in einer separaten Codezeile durchgeführt werden.

Deklaration

Je nach Programmiersprache ist die Syntax für die Deklaration etwas anders. Nachstehend ein paar Beispiele aus den Sprachen Pascal, Java und Visual Basic:

Pascal

```
var anzahl : integer;    // Anzahl ist eine Variable um eine Ganzzahl zu
                           speichern
var total  : real;       // Total ist eine Variable die eine reelle Zahl
                           (Dezimalbrüche) speichert
```

```
var vorname: string; // Vorname speichert einen Text bzw. Zeichenkette
```

1. Das Schlüsselwort var zeigt an, dass eine **Variable** deklariert wird.
2. Der Name oder auch Bezeichner der Variable
3. Nach dem Doppelpunkt folgt der Datentyp

C, Java

```
int total = 5;
```

1. Datentyp
2. Name der Variable
3. (optional) = Wert
4. ;

Visual Basic

```
Dim total As Integer
```

1. Dim
2. Name der Variable
3. As
4. Datentyp

Beachte: Je nach Programmiersprache und Ort der Deklaration (Gültigkeitsbereich) **muss** eine Variable zuerst initialisiert werden, bevor sie im weiteren Programmcode z.B. für eine Ausgabe oder eine Bedingung verwendet werden kann.

Übung

Es werden Ihnen einige Variablen mit Bezeichnung, Datentyp und Initialwert vorgegeben. Schreiben Sie für diese Variablen den korrekten Code inklusive Initialisierung in den drei Programmiersprachen **Java**, **Pascal** und **Visual Basic**. Verwenden Sie dazu das Internet, um die korrekten Notationen zu finden. Achten Sie auf korrekte Gross-/Kleinschreibung.

1. Vorname (Zeichenkette, „Hans“)
2. Körpergrösse (Ganzzahl, 179)
3. Gewicht (Dezimalzahl, 81.5)
4. Verheiratet (Ja-Nein, Ja)
5. Geschlecht als 1-Buchstabe-Kürzel (Zeichen, „m“)



© Marcel Suter

From:

<https://wiki.bzz.ch/> - **BZZ - Modulwiki**

Permanent link:

<https://wiki.bzz.ch/modul/archiv/m319/learningunits/lu02/lu02b-deklaration>

Last update: **2024/03/28 14:07**

