Variablenbelegungstabellen

Jeder von uns kennt Computerprogramme, inzwischen haben wir auch erste eigene Anwendungen geschrieben. Dabei fällt auf, dass – nachdem man ein Programm startet – dieses selten einfach von Anfang bis Ende „durchläuft“ und sich nach dem Start sofort wieder schließt. Meistens ist es so, dass ein Programm zu einem bestimmten Zeitpunkt „wartet“, bevor die nächste Aktion durchgeführt wird. Erst danach rechnet es weiter, bis es irgendwann schließt bzw. beendet wird.

Als Beispiel kann unser allererstes Programm dienen, „Hallo, Welt!“:

Nach dem Start hält das Programm inne und wartet auf eine Eingabe – hier das Klicken auf den „Drück' mich“-Button. Man sagt, das Programm befindet sich in einem bestimmten **Zustand**. Allgemein ist es in der Informatik häufig **wichtig, den aktuellen Zustand eines Programms zu beschreiben**.

Ein Beispiel ist eine beliebige Situation in einem Computerspiel, wenn man sich bspw. mit seiner Figur an einer Stelle in der Spielwelt befindet: ein bestimmtes Level, mit einer bestimmten Ausstattung des eigenen Charakters (z. B. Kräfte), einer bestimmten Position aller Freunde und Feinde auf der Map, einer bestimmten Punktzahl auf dem eigenen Konto etc. Oft ist es möglich, einen solchen Zustand als Spielstand abzuspeichern und bei einem späteren Aufruf des Computerspiels zu laden, um exakt an dieser Stelle fortzufahren.

Aber was speichert das Programm eigentlich in solch eine Spielstandsdatei? Oder abstrakter: **Was definiert einen Programmzustand?**

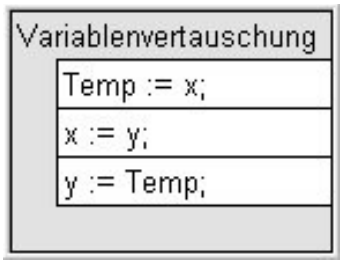
In der Informatik sagt man, dass der momentane Zustand eines Programms durch die
- **Konfiguration der Steuerleitungen** („Programmeinstellungen“)
  und
- die **aktuelle Belegung aller Variablen** („Inhalt“) gekennzeichnet ist.

Wir wollen uns im Folgenden auf die Variablenbelegung konzentrieren. Da diese von essentieller Bedeutung in der Informatik ist, hat man nach effektiven und übersichtlichen Me-

thoden gesucht, um (a) die Belegung von Variablen sowie (b) deren Änderung während eines Programmablaufs zu beschreiben. Die Lösung ist ebenso naheliegend wie einfach: Man benutzt sog. **Variablenbelegungstabellen**. Folgendes Beispiel soll dies verdeutlichen:

Gesucht sei ein Algorithmus zur Vertauschung zweier Variableninhalte. Dieses Problem kann mittels einer Hilfsvariablen gelöst werden:

Algorithmus als Struktogramm:	**Algorithmus in Delphi-Syntax:**

	`Temp := x;  { x-Wert zwischenspeichern }`
	`x := y;  { y-Wert in x uebertragen }`
	`y := Temp;  { y-Wert aus Zwischenspeicher holen }`

Im Grunde ist der Dreizeiler leicht verständlich:

1. Der erste Wert „x“ ist zunächst in der Hilfsvariablen „Temp“ (für „temporär“ ≈ vorübergehend) zwischengespeichert.
2. Nun wird der zweite Wert „y“ in den ersten Wert „x“ gespeichert.
3. Zum Schluss wird der in „Temp“ zwischengespeicherte erste Wert in den zweiten Wert „y“ gespeichert.

Am Ende sind die zwei Variableninhalte vertauscht. **Nach jedem einzelnen Befehl befindet sich der Algorithmus** (bzw. das Programm, so dieser Algorithmus in ein komplettes Programm eingebettet sein sollte) **in einem anderen Zustand**.

Mit einer Variablenbelegungstabelle kann man die aktuelle Belegung aller drei Variablen übersichtlich veranschaulichen. Im folgenden Beispiel gehen wir von der Anfangsbelegung x = 42 und y = 13 aus:

Schritt / **Variable**	**0** (zu Beginn)	**1** (nach 1. Befehl)	**2** (nach 2. Befehl)	**3** (nach 3. Befehl)
x	42	42	13	13
y	13	13	13	42
Temp	–	42	42	42

(Anm.: Natürlich kann man die Variablenbelegungstabelle auch senkrecht anstelle – wie hier – waagerecht aufbauen, indem man Zeilen- und Spaltenbeschriftung vertauscht.)

Der „Variablenzustand“ des Algorithmus' wird hier durch die jeweiligen Spalten beschrieben. Dabei ist es sinnvoll, den ersten Schritt – quasi vor Beginn der Abarbeitung der Befehle – mit „0“ zu nummerieren. Die Markierung „–“ verdeutlicht, dass diese Variable zur

Zeit nicht belegt oder ihr Wert unbekannt ist.

Ihren vollen Nutzen entfaltet die Methode der Variablenbelegungstabelle dann, wenn sich der Algorithmus noch trickreicher (d. h. komplizierter, komplexer) als in unserem obigen Einstiegsbeispiel gestaltet.

Aufgaben

1) Die Variablen a, b und c seien (in dieser Reihenfolge) mit den Werten 1, 2 und 3 belegt. Es werden nacheinander folgende Befehle (Befehlsfolge oder „Sequenz“) ausgeführt. Welche Belegung haben die Variablen nun? Erstellen Sie jeweils eine Variablenbelegungstabelle.

a) Algorithmus:

```
{1}  a := b - c;
{2}  b := -b;
{3}  a := (a + b) - c;
{4}  c := Abs(a);
{5}  b := c + a * b;
```

b) Algorithmus:

```
{1}  a := 49;
{2}  c := Sqrt(a);
{3}  b := a div 5;
{4}  c := b mod 3;
{5}  a := (Sqr(c) - a) * 4
```

c) Algorithmus:

```
{1}  c := c div 3;
{2}  if c = 1 then Inc(b)
       else Dec(a);
{3}  if b < 3 then b := 7 * b + 2;
{4}  a := Sqrt(Sqr(b+1));
{5}  b := Random(11) + 7;
```

Platz für eigene Notizen:

Der Platz reicht nicht? Super, nimm Dir ein weiteres Blatt! ☺