

Java-Hamster – Variablen und Datentypen

Bisher hatte unser Hamster noch kein Gedächtnis, d.h. er konnte sich nichts merken. Dieses kann durch sogenannte Variablen geändert werden. Jede Variable kann nur eine bestimmte Art von Information (=“Datentyp“) speichern, die in einer sogenannten „Deklaration“ (=Bekanntmachung) einmal festgelegt wird und danach auch nicht mehr geändert werden kann. Anschließend kann man der Variable einen Wert zuweisen. Wenn dies das erste Mal erfolgt, spricht man von einer „Initialisierung“.



Es stehen folgende Grunddatentypen¹ zur Verfügung:

Java-Datentyp	Beschreibung	Werte	Deklaration	Initialisierung / Zuweisung	Kurzschreibweise Deklaration + Initialisierung
int	ganze Zahl (engl. „integer“)	2, -6	int x;	x = 0;	int x=0;
double	Fließkommazahl (engl. von „double precision floating point number“)	3.5, -1.0	double d;	d = 0.4;	double d=0.4;
char	einzelnes Zeichen (engl. character)	'A', '\$', '2'	char c;	c = '%';	char c='%';
String	Zeichenkette („String of chars“)	"Hi Susi"	String name;	name = "Li";	String name = "Li";
boolean	Wahrheitswert (nach Herrn Boole)	true, false	boolean b;	b = false;	boolean b=false;

Beispiel: Der Hamster möchte herausfinden, wie viele Körner auf dem Feld liegen, auf dem er gerade steht.

```
void main() {
    int anzahlKoerner;    // Deklaration: wir brauchen ein „Gedächtnis“ zum Zählen.

    anzahlKoerner = 0;    // Initialisierung: wir starten mit Null beim Zählen

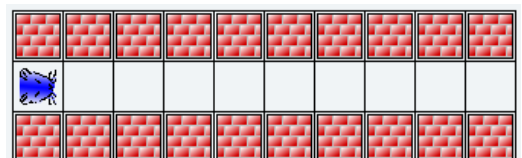
    while (kornDa()) {    // wir machen das Folgende solange wie Körner da liegen ...
        nimm();           // ... und zwar nehmen wir bei jedem Durchlauf ein Korn weg.

        anzahlKoerner = anzahlKoerner + 1;
                        // nach jedem Wegnehmen addieren (+) wir eins (1) hinzu
                        // und merken uns das neue Ergebnis (Zuweisung mit = )
    }

    schreib("Ich habe "+anzahlKoerner+" Körner gefunden! ");
                        // Ausgabe des Ergebnisses auf dem Bildschirm
}
```

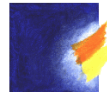
Aufgabe 1: Der vergessliche Hamster

- Der Hamster soll im nebenstehenden Territorium bis zum Ende laufen. Dort merkt er, dass er etwas vergessen hat und muss deshalb umkehren und wieder zurück zum Anfang laufen.



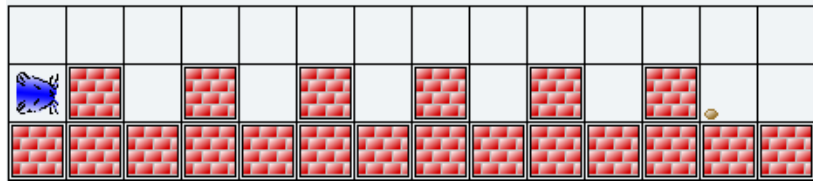
- Jetzt soll der Hamster nicht am Anfang des Tunnels starten, sondern an einem beliebigen Punkt. Zu diesem Punkt muss er dann auch wieder zurücklaufen, nachdem er am Ende des Tunnels umkehren musste.

¹ prinzipiell kann man auch eigene Datentypen deklarieren, aber das kommt später.

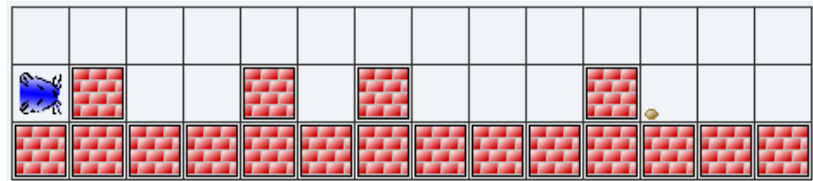


Aufgabe 2: Hürdenlauf

2.1. Der Hamster trainiert für einen Hürdenlauf. Lassen Sie den Hamster über die Hürden springen und das Korn fressen.



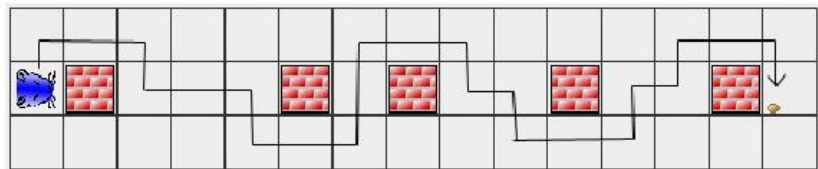
2.2. Wenn noch nicht in 2.1 programmiert, sollen die Abstände zwischen den Hürden nicht festgelegt sein. Testen Sie ihr Programm mit verschiedenen Territorien.



2.3. Lassen Sie den Hamster die Anzahl der Hürden zählen und lassen Sie ihn am Ende das Ergebnis mit `schreib("Ich habe "+anzahlHuerden+" Hürden übersprungen!");` // "anzahlHuerden" ist ein Variable! ausgeben.

Aufgabe 3: Slalom

3.1. Der Hamster ist ein begeisterter Skifahrer. Vor ihm befindet sich ein Slalomparcours mit einem Korn am Ende, das das Ziel markiert. Der Hamster soll den Slalomparcours bewältigen und das Korn fressen.



Hinweis: Slalomstangen bestehen immer nur aus einer Mauer.

3.2. Wenn noch nicht in 3.1 programmiert, verwenden Sie eine *boolean*-Variable, um sich zu merken, in welche Richtung der Hamster letztes Mal um die Stange gefahren ist. Verwenden Sie diese bei der Entscheidung vor der nächsten „Stange“.

Aufgabe 4: Sammeln

4.1. Der Hamster möchte alle Körner aufsammeln und kein Feld dabei auslassen. Verwenden Sie dabei eine *boolean*-Variable.

4.2. Der Hamster möchte wissen, wie viele Körner er insgesamt gesammelt hat und diese Anzahl am Ende des Programms ausgeben.

