



Landwirtschaft ist heutzutage zu einer Wissenschaft geworden: Agrarökonomie. Ohne genaue Statistiken über die Erträge wäre die Forschung in diesem Bereich undenkbar. Unser Bauer sollte also für jedes einzelne Huhn zählen können, wie viele Eier es legt. Aber bei einem großen Hühnerhof mit 100 Hühnern bräuchte man ja dann 100 Variablen. Und jetzt kommen noch 10 neue Hühner dazu. Wie soll unser Programm das schaffen? Müssen wir alles umprogrammieren?

Arrays = Liste vieler gleichartiger Variablen

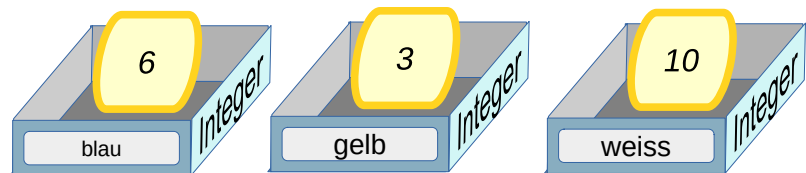
ZIEL: Arrays deklarieren und initialisieren und zum Speichern vieler gleichartiger Werte nutzen können. Algorithmen auf Arrays implementieren können.

Arrays deklarieren, initialisieren und benutzen

Der Bauer links unten verdient sein Geld mit Blumenzucht. Je nach Farbe erhält er einen unterschiedlichen Preis für seine Blumen. Daher möchte er eine Statistik erstellen, wie häufig welche Farbe vorkommt.

Die erste Idee ist drei integer-Variablen mit den Namen `rot`, `blau` und `weiss` zu erstellen:

```
int blau = 6;
int gelb = 3;
int weiss = 10;
```



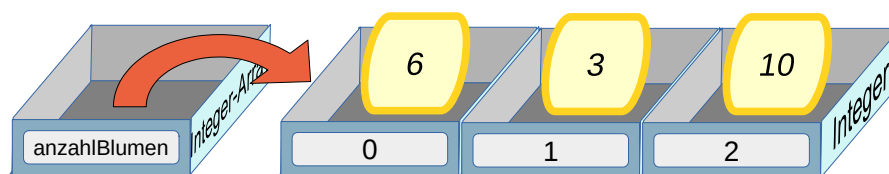
3 Variablen für die Blumenanzahl

Bei drei verschiedenen Farben geht das noch. Aber das Programm soll leicht auf eine große Anzahl verschiedener Blumen erweitert werden können.

Daher erstellt man ein Array:

```
int[] anzahlBlumen;
anzahlBlumen = new int[3];
anzahlBlumen[0] = 6;
anzahlBlumen[1] = 3;
anzahlBlumen[2] = 10;
```

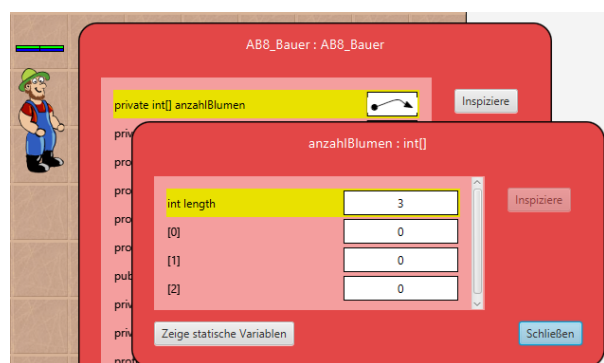
Die einzelnen Speicherplätze spricht man mit ihrer Nummer an. Dabei beginnt die Nummerierung mit 0. Bevor man das Array benutzen kann, muss man mit dem Befehl `new` genügend Speicherplatz reservieren (hier drei `int`-Variablen).



Array mit 3 Speicherplätzen

Aufgaben:

- Viel Speicherplatz:** Beim `AB8_Bauern` gibt es schon ein Attribut `anzahlBlumen`. Initialisiere dieses Attribut wie oben gezeigt im Konstruktor des `AB8_Bauern`. Setze die Startwerte für alle Blumensorten auf 0. Inspiziere den Bauern. Beim Attribut `anzahlBlumen` siehst du nur einen Pfeil (= Verweis). Diesen kannst du doppelklicken und dadurch inspizieren, ggf. musst Du die roten Fenster an einem Rand verbreitern. Dann solltest du sehen, dass das Array die Länge 3 hat und in jedem Eintrag eine 0 steht.





2. **Blumenzähler:** Die Methode `pflueckeBlumen()` lässt den Blumenzüchter auf seinem Feld herumlaufen und alle Blumen pflücken, bis er keine Energie mehr hat. Ergänze diese Methode so, dass er dabei die Anzahl der Blumen zählt. An Position 0 soll die Anzahl der blauen Blumen gespeichert werden, an Position 1 die Anzahl der gelben und an Position 2 die Anzahl der weißen Blumen.

Hinweis 1: `anzahlBlumen[1]++;` // erhöht die Anzahl an der Position 1

Hinweis 2: Mit `pruefe("Blume")` kann man feststellen, welche Farbe die Blume hat (blau = 40, gelb = 30, weiß = 40).

Teste die Methode. Kontrolliere im Inspect-Fenster, ob die Zählung erfolgreich war.

3. **Gesamtertrag:** Implementiere eine Methode `int gibGesamtzahlBlumen()`, die die Gesamtzahl aller gepflückten Blumen zurückgibt. Addiere dazu die Werte der einzelnen Positionen und speichere sie in einer lokalen Variable `ergebnis`. Gib dieses Ergebnis zurück.

4. **Einzelaufstellung:** Untersuche die Methode `gibAnzahlBlumenart(int art)`. Beschreibe, welche Bedeutung das Ergebnis von `gibAnzahlBlumenart(2)` hat. Wie wird hier ausgewählt, welche Position des Arrays ausgelesen wird?

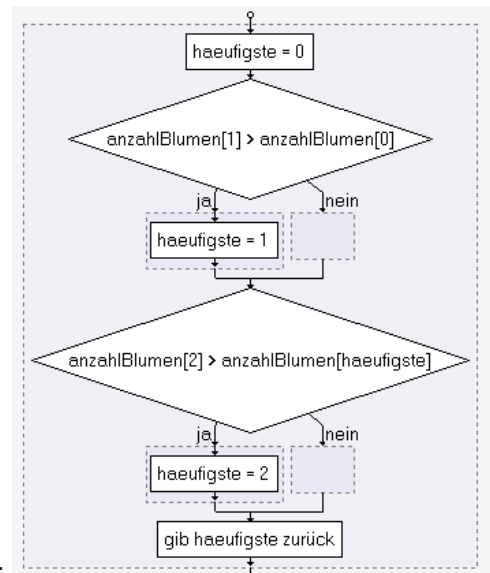
5. **Prozentrechnung:** Implementiere `double gibProzentsatz(int art)`. Diese Methode soll berechnen, wie groß der Prozentsatz der Blumen einer bestimmten Farbe war.

Hinweis 1: $p\% = \frac{W}{G} * 100\%$

Hinweis 2: Wenn man in Java zwei ganze Zahlen teilt, dann bekommt man wieder eine ganze Zahl. Java schneidet die Nachkommastellen einfach ab. Daher muss man die Anzahl der Blumen in eine Dezimalzahl umwandeln:

```
int i = 3;
double e1 = i / 5;           // ergibt 0.0
double e2 = (double) i / 5;  // ergibt 0.6
```

6. **Wachstumswunder:** Implementiere eine Methode `int gibNrHaeufigsteBlume()`, die bestimmt, welche Blumensorte am häufigsten auf dem Feld gewachsen ist. Als Ergebnis soll eine Zahl zwischen 0 und 2 zurückgegeben werden. Setze dazu nebenstehendes Flussdiagramm in Java-Quelltext um.



7. **Wachstumswunder 2:** Implementiere die Methode `int gibAnzahlHaeufigsteBlume()`, die dieses Mal nicht die Nummer, sondern die Anzahl der häufigsten Blume bestimmt.

Hinweis: Nutze dazu die Methode, die du schon bei Aufgabe 6 implementiert hast.



Arrays mit Zählschleifen durchlaufen

Aufgabe 4 hat dir gezeigt, dass man die Position in einem Array auch durch eine Variable angeben kann. Dies ist sehr nützlich, wenn man mit allen Elementen eines Arrays etwas machen möchte, z.B. alle auf 0 setzen:

Variante 1:

```
anzahlBlumen[0] = 0;
anzahlBlumen[1] = 0;
anzahlBlumen[2] = 0;
```

Variante 2:

```
for(int i=0; i<3; i++) {
    anzahlBlumen[i] = 0;
}
```

i nimmt also die Wert 0 bis 2 an. Der Reihe nach werden die einzelnen Array-Elemente auf 0 gesetzt. Auch die Summenbildung kann man mit einer Schleife realisieren, wenn man die Summenbildung zerlegt und immer nur ein weiteres Array-Element zur Summe addiert:

```
summe = ((0 + anzahlBlumen[0]) + anzahlBlumen[1]) + anzahlBlumen[2];
```

Variante 1:

```
summe = 0;
summe = summe + anzahl[0];
summe = summe + anzahl[1];
summe = summe + anzahl[2];
```

Variante 2:

```
summe = 0;
for(int i=0; i<3; i++) {
    summe = summe + anzahlBlumen[i];
}
```

Ersetzt man die Zahl 3 noch durch `anzahlBlumen.length` kann das Array beliebig groß sein und der Algorithmus funktioniert trotzdem noch.

Der Hühnerbauer möchte untersuchen, welche seiner Nester gut platziert und welche nicht. Dafür zählt er die Eier, die er aus den Nestern nimmt.

Aufgaben:

8. **Eierstatistik:** Dekлариere ein weiteres Attribut `anzahlEier`. Initialisiere es im Konstruktor des Bauern, so dass es 26 Plätze für `int`-Zahlen bietet. Setze alle Werte mit Hilfe einer Zählschleife auf 0. Kontrolliere deinen Erfolg im Inspect-Fenster.
9. **Eiersammelei 1:** Implementiere die Methode `laufeSchritte(int anzahl)` so, dass der Bauer die gegebene Anzahl von Schritten nach vorne macht. Er muss nicht kontrollieren, dass dort frei ist.
Teste deine Methode, indem du die Methode `sammleEier()` aufrufst. Wenn du alles richtig programmiert hast, läuft der Bauer zwei Runden außen herum über alle Nester.
10. **Eiersammelei 2:** Ändere die Methode `laufeSchritte` so ab, dass der Bauer die Eier in den Nestern einsammelt. Er soll dabei in dem Attribut `anzahlEier` zählen, wie viele Eier er aus welchem Nest geholt hat. Dazu muss er sich in einem weiteren Attribut `nestNr` merken, beim wievielten Nest er ist. Er testet also nach jedem Schritt, ob er auf einem Nest steht und erhöht die Nummer ggf. um 1. Falls ein Ei im Nest liegt, erhöht er außerdem die entsprechende Speicherstelle des Arrays um 1.
Hinweis: Wenn er eine Runde gelaufen ist, muss `nestNr` wieder zurückgesetzt werden. Teste deine Methode. Erhöhe die Anzahl der Runden, wenn der Test erfolgreich war.
11. **Einzelaufstellung:** Implementiere die Methode `gibAnzahlEierInNest(int nr)`, die zurückgibt, wieviel Eier im angegebenen Nest eingesammelt wurden.
12. **Gesamtertrag:** Implementiere eine Methode `int gibGesamtzahlEier()`, die die Gesamtzahl aller gesammelten Eier zurückgibt. Verwende dazu eine `for`-Schleife.
13. **Durchschnitt:** Implementiere eine Methode `double gibDurchschnitt()`, die die durchschnittliche Anzahl Eier pro Nest zurückgibt. Denke daran, dass du bei der Division eine Umwandlung der Anzahl in Dezimalzahl benötigst. Die Gesamtanzahl der Nester erhältst du mit `anzahlEier.length`.



14. **Nutzlos:** Implementiere eine Methode `int gibAnzahlLeererNester()`, die zurückgibt, in wievielen der Nester kein einziges Ei gefunden wurde.
15. **Lieblingsnest:** Implementiere eine Methode `int gibNrBestesNest()`, die bestimmt, welches Nest den höchsten Ertrag gebracht hat. Als Ergebnis soll die Nummer des Nestes zurückgegeben werden.
Hinweis: Die Methode geht ähnlich wie bei den Blumen. Du musst dir überlegen, was man bei den Blumen ändern müsste, wenn es 4 Sorten gewesen wären. Wie kann man das mit einer for-Schleife vereinfachen? An welcher Stelle muss man den Zähler i einsetzen?
16. **Lieblingsnest 2:** Implementiere die Methode `int gibAnzahlEierBestesNest()`, die dieses Mal nicht die Nummer, sondern die Anzahl der Eier des besten Nestes bestimmt.
Hinweis: Nutze dazu die Methode, die du schon bei Aufgabe 15 implementiert hast.

Textvariablen (Typ String)

Deklaration: `String text1;`

Initialisierung: `text1 = "";`

Veränderung: `text1 = "Hallo";`
`text1 = text1 + "Leute"; // Verlängern des Textes`
`text1 = text1 + i; // hängt die Zahl i an den Text an`

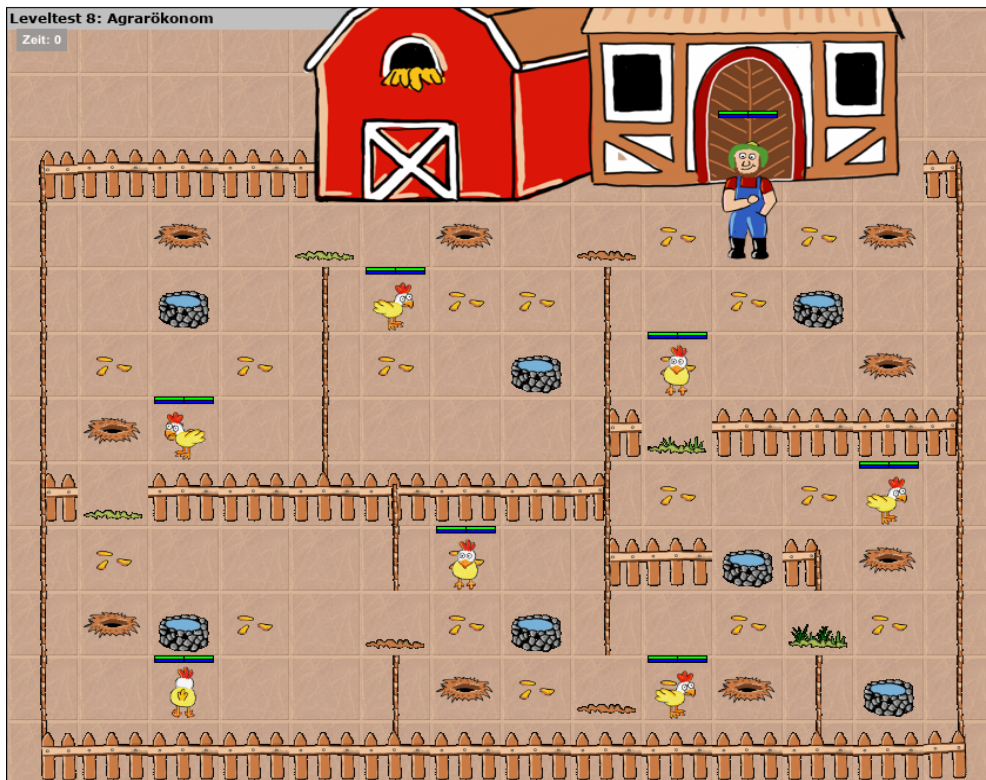
17. **Infotafel:** Implementiere eine Methode die einen Ausgabetext generiert, der für jedes Nest die Anzahl der Eier enthält. Dekлариere dazu eine lokale Variable `text`. Fange mit einem leeren Text an und hänge mit Hilfe einer for-Schleife jede Zahl des Arrays an den Text an. Trenne die Zahlen, indem du dazwischen jeweils ein ", " anhängst.
 Ausgabe ist dann z.B.: 23, 34, 1, 0, usw.



Leveltest 8: Agrarökonom

Du hast festgestellt, dass manche Nester besser von den Hühnern angenommen werden als andere. Diesen Aspekt willst du nun weiter untersuchen, um die optimale Form für das Gehege der Hühner zu finden. Daher bekommt jedes Huhn einen eigenen Bereich, der jeweils unterschiedlich gestaltet ist. Sie variieren in der Größe, Form und Anzahl der Nester.

Dein Bauer soll nun einige Runden durch die Gehege laufen und die Eier einsammeln. Im Anschluss soll er eine nach Anzahl der gesammelten Eier sortierte Liste erstellen.

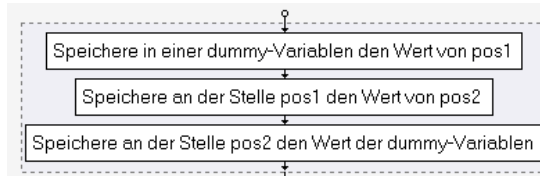


Verändere die Methode `leveltest8()` zunächst so, dass die Anzahl der Eier jedes Huhns gezählt werden. Der Bauer kann an den Grasbüscheln (`istAuf("Gras")`) erkennen, dass er das nächste Gehege betritt.

Deklariere ein Array von Strings und speichere in diesem Array einen Namen für jedes Huhn. Hinweis: Man kann Arrays auch direkt initialisieren:

```
String[] schuelernamen = {"Fred", "Anita", "Martha"};
```

Implementiere eine Methode `vertausche(int pos1, int pos2)`, die die Werte des Anzahl- und des Namensarrays an den beiden übergebenen Positionen vertauscht:



Nutze diese Methode, um das Array der Eieranzahlen und passend dazu das Array der Hühnernamen zu sortieren. Erstelle danach einen aufsteigend sortierten Ausgabetext, der die Hühnernamen und die Anzahl der von diesem Huhn gelegten Eier erstellt (z.B. "Lucmillla: 4, Gacki: 7 usw."). Dieser Text wird vom Leveltest zurückgegeben.