Matthias Heming 25. November 2009

Hinweise zur Phase »Verzweigungen«

1 Kompetenzformulierung

Schülerinnen und Schüler ...

- nutzen Verzweigungen zur Untersuchung von Eingabedaten.
- nutzen Struktogramme zur Visualisierung von Programmabläufen mit Verzweigungen.
- führen Mehrfachverzweigungen auf mehrere aufeinanderfolgende einfache Verzweigungen zurück.
- unterscheiden Abfragen auf der Ebene der Benutzungsschnittstelle und der Programmiersprache.

2 Detaillierte Zielsetzung

In Form des *Tiereraten*-Programms werden Verzweigungen als unterschiedliche Abläufe einer Spielrunde eingeführt. Die zugrunde liegenden Daten des Programms, die durch das Laden entsprechender XML-Dateien variiert werden können, werden dabei als eine Art binärer Suchbaum interpretiert, der nicht unbedingt balanciert sein muss. Intuitiv wird bereits auf eine Unterscheidung zwischen inneren Knoten und Blättern hingearbeitet, da die zu stellenden Fragen immer in inneren Knoten und die resultierenden Tierarten immer als Blätter vorkommen.
Analog zur bedingten Anweisung innerhalb der Programmiersprache, kann das Programm nur erweitert werden, wenn die zu verwendenden Fragen auf Entscheidungsfragen reduziert werden. Schülerinnen und Schüler sollen jedoch erkennen, dass dieses keine tatsächliche Einschränkung darstellt. Fragen mit mehreren Antwortmöglichkeiten können durch geeignete Umformulierungen auf mehrere Entscheidungsfragen zurückgeführt werden..
Selbst ein vorgegebenes Programm nachzuprogrammieren, ist höchst wahrscheinlich nur für wenige Schülerinnen und Schüler motivierend. Dennoch soll dies innerhalb der fünften Aufgabe geschehen, um sich mit der praktischen Umsetzung der Verzweigung auf der Ebene der Programmiersprache auseinanderzusetzen.
In der folgenden Aufgabe werden nun jedoch die Benutzungsschnittstelle von der Programmierebene getrennt. Während die Fragestellungen an der Benutzungsschnittstelle durchaus komplexer werden und nicht nur ja/nein als Antwort erwarten, bleibt die Bearbeitung der Daten auf der Ebene der Programmiersprache dieselbe.
Zur Visualisierung werden Struktogramme Programmablaufplänen vorgezogen. Struktogramme sind näher an der Implementierung, Programmablaufpläne verführen schnell dazu, Sprungbefehle umsetzen zu wollen. Durch die zweite Aufgabe soll jedoch deutlich werden, dass es nicht nur diese eine mögliche Darstellungsform gibt.

3 Mögliche Weiterarbeit

Die Datenspeicherung des *Tiereraten*-Programms geschieht in Form einer XML-Datei. Wenn auch der bisherige Exportmechanismus keine Einrückung berücksichtigt und deswegen nur schwer lesbar ist, so können dennoch eigene XML-Dateien erzeugt werden bzw. welche mit Einrückung zur Modifikation vorgegeben werden.

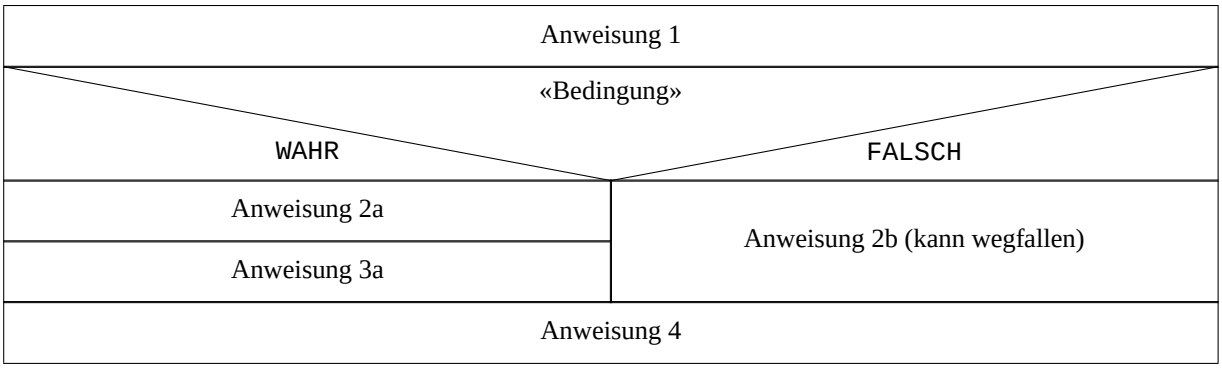
Damit demonstriert dieses Programm, wie XML als Beschreibungssprache nach international festgelegten Normen zur eigenen Datenstrukturierung personalisiert werden kann. Die passende *Documenttype Definition* ist online zugänglich und bewusst sehr einfach gefasst.
Neben dem Thema der Datenstrukturierung stellt sich ebenfalls die Frage, welche Bedingungen automatisiert auf ihren Wahrheitswert überprüft werden können, was überhaupt ein Wahrheitswert ist und warum es nicht direkt regnet, wenn die Sonne mal nicht scheint? Welche Möglichkeiten sich bei dem Thema Aussagenlogik ergeben, soll hier jedoch nicht weiter ausgeführt werden.

4 Genderaspekt

Die Thematik mit Tierarten ist wahrscheinlich recht neutral formuliert. Es stellt sich jedoch die Frage, welchen Einfluss eine anderer thematischer Zusammenhang – z. B. dass Erraten der letzten Freizeitbeschäftigung oder des Mobiltelefonmodells – auf die Motivation hätte.

5 Für das Merkheft

Aufbau eines Struktogramms:



In Python

```
anweisung1()
if <<Bedingung>>:
        anweisung2a()
        anweisung3a()
else:
        anweisung2b()
anweisung4()
```

Vergleiche als Bedingungen:

`==` Wird genutzt, um zu fragen, ob zwei Objekte den gleichen Werte haben.
Beispiel: Ist `a=5` festgelegt worden, so ist die Bedingung `a==3` nicht erfüllt, die Bedingung `a==5` hingegen schon.

`!=` Wird genutzt, um zu fragen, ob zwei Objekte unterschiedliche Werte haben.
Beispiel: Ist `a="Hallo"` festgelegt worden, so ist die Bedingung `a!="Hallo"` nicht erfüllt, `a!="hallo"` jedoch schon (es wird Groß- und Kleinschreibung unterschieden).