Christian-Albrechts-Universität zu Kiel

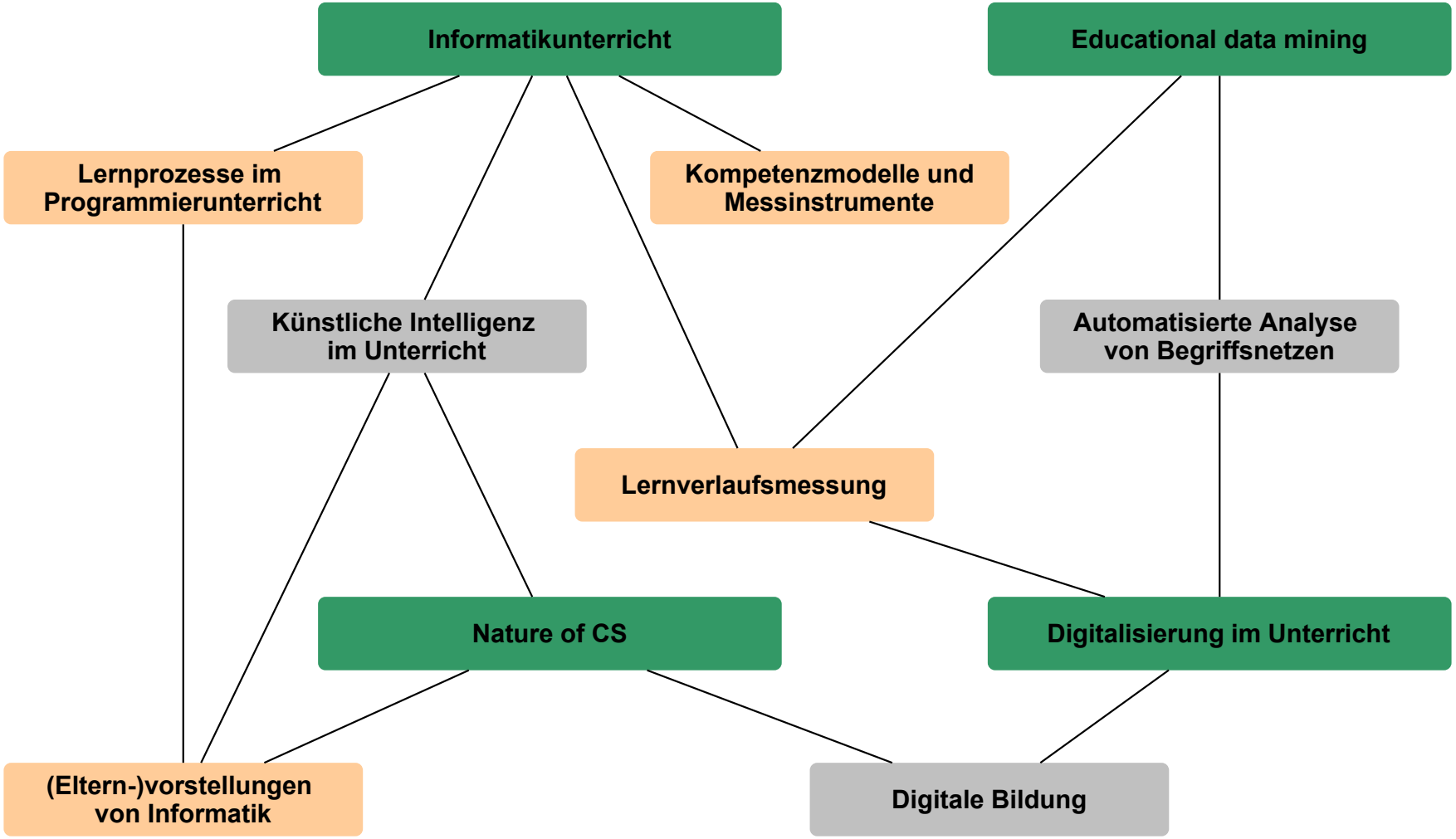
Technische Fakultät

Programmieren lernen und lehren

Prof. Dr. Andreas Mühling

13.01.2020

Überblick


Informatikunterricht
Educational data mining
Lernprozesse im Programmierunterricht
Kompetenzmodelle und Messinstrumente
Künstliche Intelligenz im Unterricht
Automatisierte Analyse von Begriffsnetzen
Lernverlaufsmessung
Nature of CS
Digitalisierung im Unterricht
(Eltern-)vorstellungen von Informatik
Digitale Bildung

Programmieren

Programmieren ist eine „Grundfertigkeit“ (der Informatik!?).

Programmieren lernen ist schwer (Watson & Li, 2014)!

Anfangsunterricht erfolgt zwischen Primarstufe und Universität

Wie lässt sich Kompetenzerwerb „am Anfang“ modellieren, messen und fördern?

Lernverlaufsmessung

Wie entwickelt sich Programmierkompetenz?

Was ist der Wert von z nach Ausführung des Codes?

```
int x = 1; int y = 2; int z = 3;
if (x <= y) {
    if (y >= 4) {
        z = 5;
    } else {
        z = 6;
    }
}
```

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR '17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Was ist der Wert von z nach Ausführung des Codes?

```
int x = 1; int y = 2; int z = 3;
if (x <= y) {
    if (y >= 4) {
        z = 5;
    } else {
        z = 6;
    }
}
```

- Datentyp,
- Variable (Deklaration und Initialisierung),
- Zuweisung,
- Bedingte Anweisung,
- (Bool'scher) Ausdruck,
- Vergleich,
- Verbundanweisung,
- Sequenz (mit/ohne neue Zeile),
- Bezeichner.

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR '17: (S. 47–69). New York: ACM.

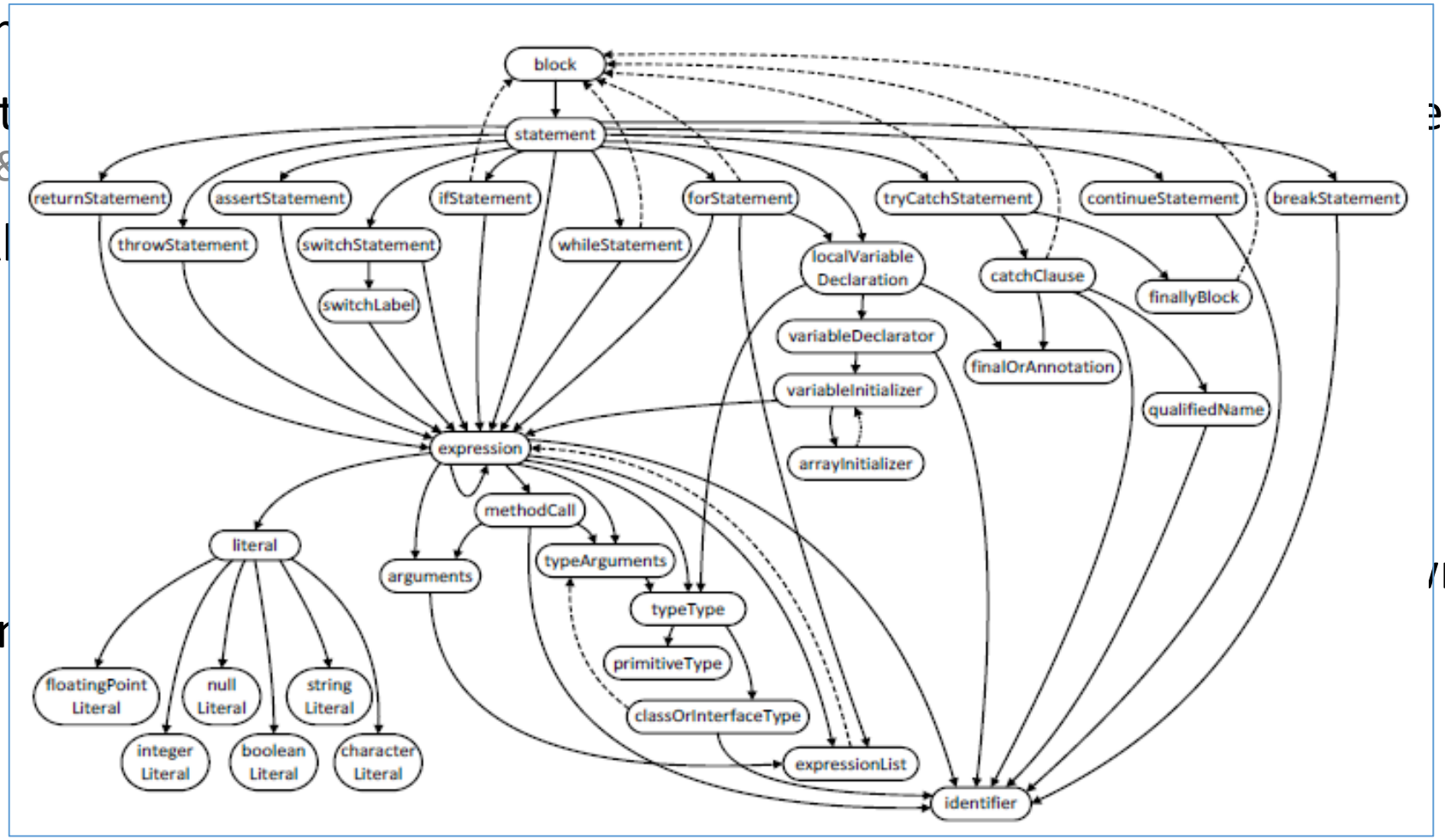
Wie entwickelt sich Programmierkompetenz?

Spezifikur

- Eindeut
  (Berges &
- Klare Al

Idee:

Konzepte
concept ir



Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR '17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Spezifikum:

- Eindeutig definierte Konzepte basierend auf der Grammatik der Sprache (Berges & Mühling, 2012)
- Klare Abhängigkeiten zwischen den Konzepten

Idee:

Konzepte inkrementell überprüfen („bottom-up“), anstelle eines „top-down“ concept inventories (z.B. Caceffo, Wolfman, Booth & Azevedo, 2016)

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR ’17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Eines der Items für *Bezeichner*:

Welches der folgenden Code-Fragemente sind gültige Bezeichner in Java?

(Welches der folgenden sind erlaubte Namen für Variablen in Java?)

3.0, hoehe, “giraffe“, 5, x, 2te_seite

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR ’17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Eines der Items für *Zuweisung:*

Welchen Wert hat b nach der Ausführung des folgenden Java Codefragments?

```
int a; int b;
a = 3;
b = a;
a = 7;
```

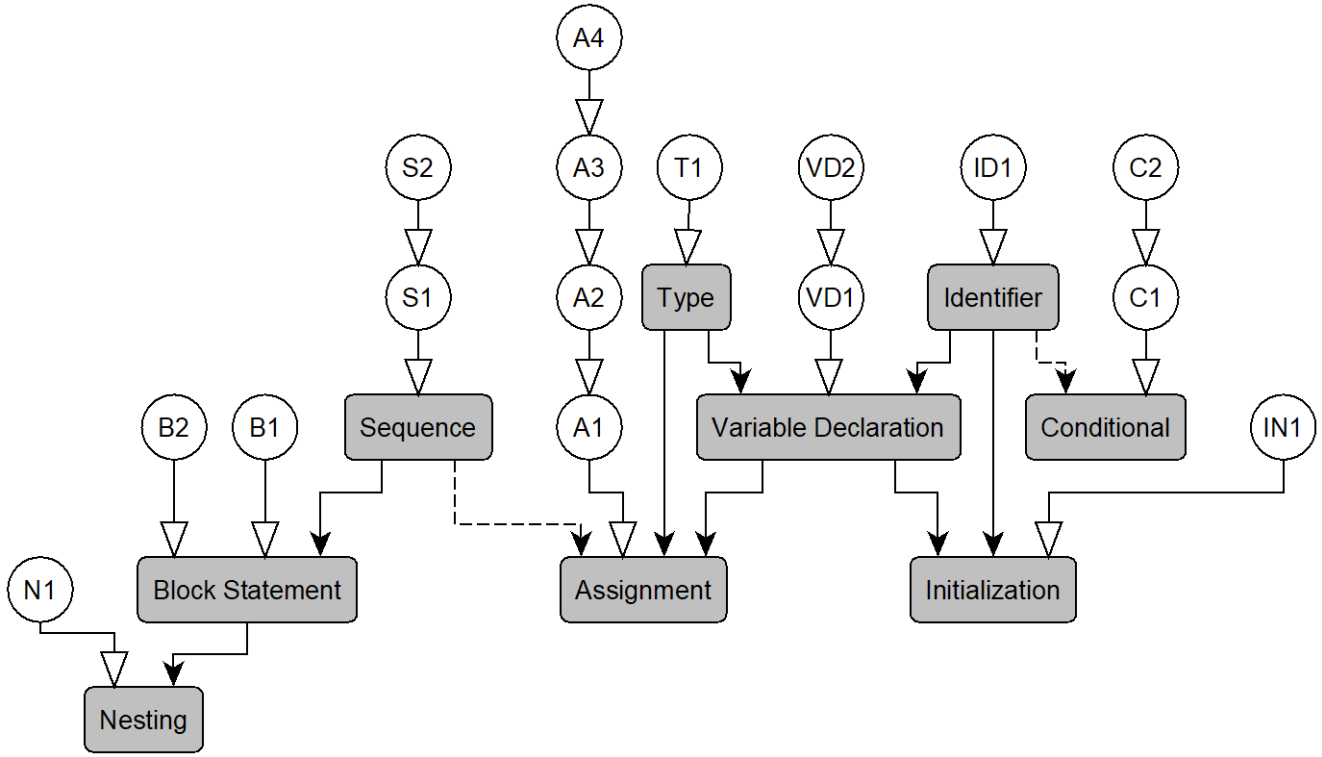
(hängt bereits ab von: *Bezeichner*, *Deklaration, Datentyp, Sequenz*):

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR '17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Was ist der Wert von z nach Ausführung des Codes?

```
int x = 1; int y = 2; int z = 3;
if (x <= y) {
    if (y >= 4) {
        z = 5;
    } else {
        z = 6;
    }
}
```



Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports*. ITiCSE-WGR '17: (S. 47–69). New York: ACM.

Wie entwickelt sich Programmierkompetenz?

Ist so ein Vorgehen machbar?

- Items entwickelt und verfeinert mit Expert:innen bis Einigkeit erzielt wurde
- 46 Items (plus Variationen) reichen um die Java Grammatik „fast“ abzudecken!

Ist es nützlich?

- Mastery-Learning und Lernverläufe benötigten solche Items
- Möglichkeit zum Selbsttest für Lernende
- „Aha-Moment“ für Lehrende

Luxton-Reilly, A., Becker, B. A., Cao, Y., McDermott, R., Mirolo, C., Mühling, A. et al. (2017). Developing Assessments to Determine Mastery of Programming Fundamentals, *Proceedings of the 2017 ITiCSE Conference on Working Group Reports.* ITiCSE-WGR ’17: (S. 47–69). New York: ACM.

Program-Tracing

Aus Begriffsnetzen ermittelte Facetten:

- Datenstrukturen
- Maschine
- Objekt-Strukturen
- Algorithmische Strukturen
- Repräsentation
- Ausführung

Mühling, A., Hubwieser, P. & Berges, M. (2015). Dimensions of Programming Knowledge. In A. Brodnik & J. Vahrenhold (Hrsg.), *Informatics in Schools. Curricula, Competences, and Competitions* (Lecture notes in computer science, Bd. 9378, S. 32–44). Springer International Publishing.

Program-Tracing

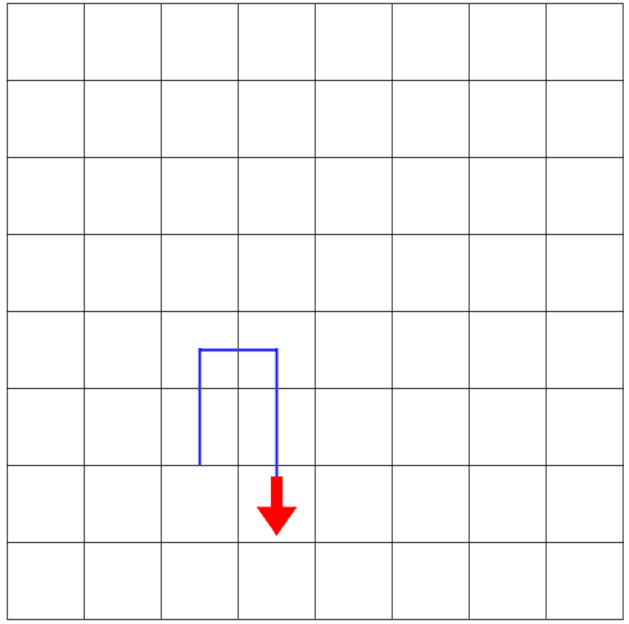
Tracing ist Vorläuferfertigkeit des Codeverstehens und des Programmierens
(Lister et al. 2004; Venables, Tan & Lister, 2009; Kumar, 2013; Nelson, Xi & Ko, 2017)

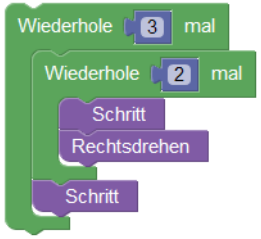
Items zu Kontrollstrukturen lassen sich Rasch-skalieren
(Mühling, Ruf & Hubwieser, 2015)

Tracing von einfachen Programmfragmenten als robuster Indikator?

Bastian, M. Schneier, Y. & Mühling, A. Diagnose von Fehlvorstellungen bei der Ablaufverfolgung von Programmen in einem webbasierten Testsystem, Empirische Pädagogik (im Druck)

Program-Tracing

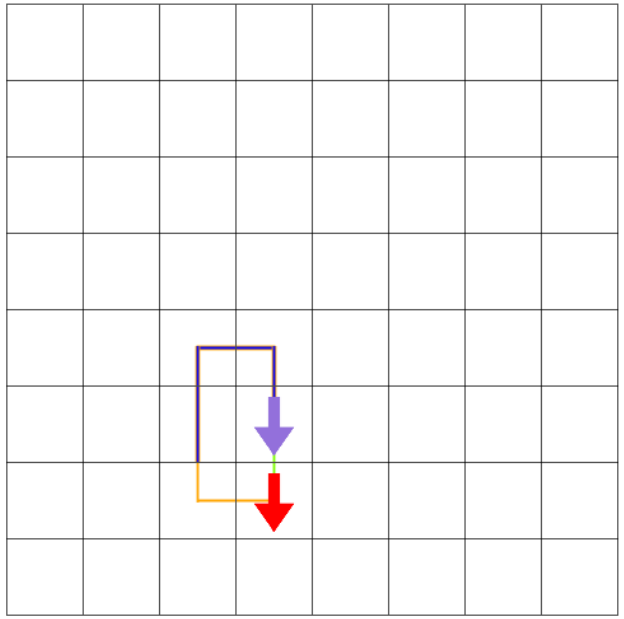


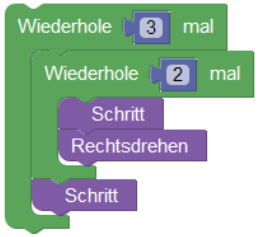


⇄ Anzeige ändern!

Eingabe:
M, R, M, R, M, M

Schritt!
Linksdrehen!
Rechtsdrehen!
Lösung überprüfen!
Zurücksetzen!





⇄ Anzeige ändern!

Eingabe:
M, R, M, R, M, M
Lösung:
M, R, M, R, M, M, R, M, R, M, M, R, M, R, M

Schritt!
Linksdrehen!
Rechtsdrehen!
Nächste Aufgabe
Lösung nochmal anzeigen!

Bastian, M. Schneier, Y. & Mühling, A. Diagnose von Fehlvorstellungen bei der Ablaufverfolgung von Programmen in einem webbasierten Testsystem, Empirische Pädagogik (im Druck)

Program-Tracing

Experimentelle Mixed-Methods-Studie (N = 215) in der Sekundarstufe:

- Items weiterhin Rasch-skalierbar
- Alter (10 – 19, Median: 16) hat keinen signifikanten Einfluss auf Ergebnis.
- Bekannte und neue Fehlvorstellungen erkennbar.

Bastian, M. Schneier, Y. & Mühling, A. Diagnose von Fehlvorstellungen bei der Ablaufverfolgung von Programmen in einem webbasierten Testsystem, Empirische Pädagogik (im Druck)

Program-Tracing

	Dffclt	Dscrmn	P(x=1|x=0)	Beschriebung
I1	-1.642	1.621	0.935	einfache Sequenz
I2	-0.959	1.621	0.826	loop fixed
I6	-0.397	1.621	0.655	if with false
I3	0.096	1.621	0.461	nested two fixed loops
I7	0.319	1.621	0.374	nested if in loop with fixed
I8	0.552	1.621	0.290	nested loop fixed in loop cond
I9	0.742	1.621	0.231	nested loop cond in loop fix
I4	0.782	1.621	0.220	loop with condition
I5	1.659	1.621	0.064	if with true

Bastian, M. Schneier, Y. & Mühling, A. Diagnose von Fehlvorstellungen bei der Ablaufverfolgung von Programmen in einem webbasierten Testsystem, Empirische Pädagogik (im Druck)

Kompetenzmodellierung

Ein Kompetenzmodell für OOP

Aus Begriffsnetzen ermittelte Facetten:

- Datenstrukturen
- Maschine
- Objekt-Strukturen
- Algorithmische Strukturen
- Repräsentation
- Ausführung

Mühling, A., Hubwieser, P. & Berges, M. (2015). Dimensions of Programming Knowledge. In A. Brodnik & J. Vahrenhold (Hrsg.), *Informatics in Schools. Curricula, Competences, and Competitions* (Lecture notes in computer science, Bd. 9378, S. 32–44). Springer International Publishing.

Ein Kompetenzmodell für OOP

Ziel: Messinstrument für Objekt-Strukturen.

- Ausdifferenzierung nach OIH-Hierarchie (Bennedsen & Schulte, 2013)
- Gemeinsame Entwicklung von Items
- Datensammlung und Auswertung zur Validierung

Dreiteilung entlang zweier Ebenen:

1. Interaktion mit einzelnen Objekten
2. Interaktion mit statisch aufgebauten Objektstrukturen
3. Interaktion mit sich verändernden Objektstrukturen

und:

1. Keine Berücksichtigung der Interaktionshistorie
2. Interaktionshistorie, aber keine versteckten Änderungen (Wertsemantik)
3. Interaktionshistorie mit versteckten Änderungen (Referenzsemantik)

Mühling, A., Schulte, C., Bennedsen, J., Budde, L. & Große-Bölting, G. (2019). Assessing students' understanding of object structures. In P. Ihantola & N. Falkner (Hrsg.), *Proceedings of the 19th Koli Calling International Conference on Computing Education Research - Koli Calling '19* (S. 1–10). New York, New York, USA: ACM Press.

Ein Kompetenzmodell für OOP

```
class Item {
	int value;

	public Item(int v) {
		setValue(v);
	}

	public void setValue(int v) {
		value = v;
	}

	public int getValue() {
		return value;
	}
}
```

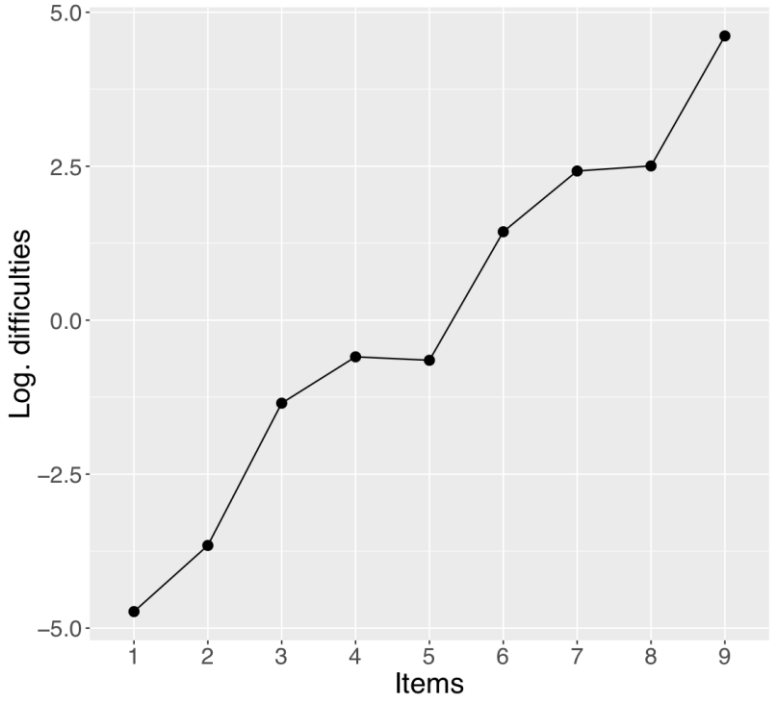
```
Item i1 = new Item(5);
Item i2 = new Item(2);
Item[] a = new Item[2];
a[0] = i1;
a[1] = i2;
System.out.println(a[0].getValue());
```

Statisch aufgbaute Struktur, keine Historie

Mühling, A., Schulte, C., Bennedsen, J., Budde, L. & Große-Bölting, G. (2019). Assessing students' understanding of object structures. In P. Ihantola & N. Falkner (Hrsg.), *Proceedings of the 19th Koli Calling International Conference on Computing Education Research - Koli Calling '19* (S. 1–10). New York, New York, USA: ACM Press.

Ein Kompetenzmodell für OOP

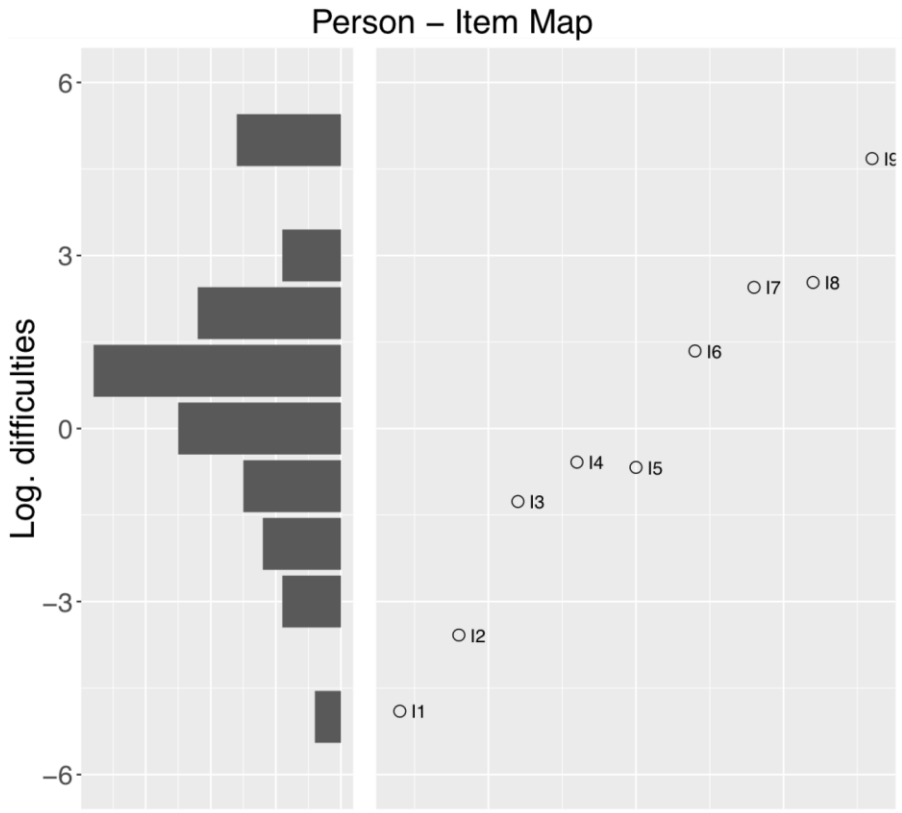
- 9 Items als Ausgangsbasis
- Datensammlung (N=195), erstes Semester Informatik



- Ansteigende Schwierigkeit entlang beider Dimensionen
- Deutlicher Sprung vor „dynamischen Objektstrukturen“ (=> Referenzen)
- Rasch-Modell besser als Birnbaum-Modell

Mühling, A., Schulte, C., Bennedsen, J., Budde, L. & Große-Bölting, G. (2019). Assessing students' understanding of object structures. In P. Ihantola & N. Falkner (Hrsg.), *Proceedings of the 19th Koli Calling International Conference on Computing Education Research - Koli Calling '19* (S. 1–10). New York, New York, USA: ACM Press.

Ein Kompetenzmodell für OOP



Validierung anhand von Selbsteinschätzungsfragen

Korrelation mit:	
Selbsteinschätzung Programmierfertigkeit	0.53***
Jahren Programmiererfahrung	0.42***
Längstes Programm	0.39**
Note in Abschlussklausur	0.37*
Selbsteinschätzung Mathematik	0.21*

Punkte ~ Selbsteinschätzung: $R^2 = 0.21$ **

Punkte ~ Längstes Programm + Jahre Erfahrung: $R^2 = 0.51$ ***

Mühling, A., Schulte, C., Bennedsen, J., Budde, L. & Große-Bölting, G. (2019). Assessing students' understanding of object structures. In P. Ihantola & N. Falkner (Hrsg.), *Proceedings of the 19th Koli Calling International Conference on Computing Education Research - Koli Calling '19* (S. 1–10). New York, New York, USA: ACM Press.

Schüler:innenvorstellungen

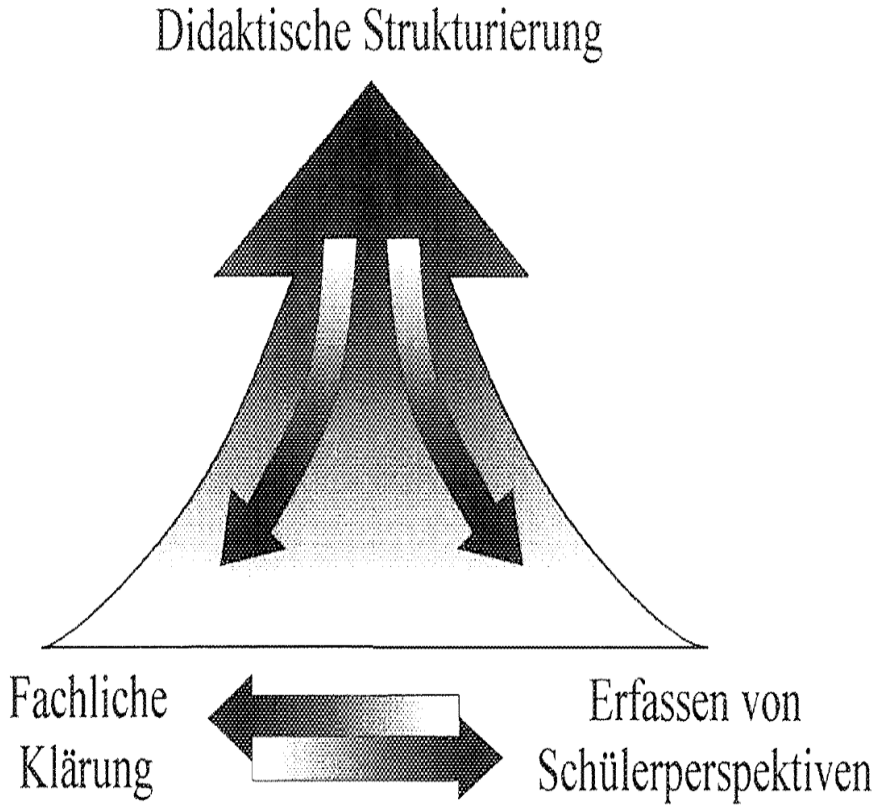
Schüler:innenvorstellungen

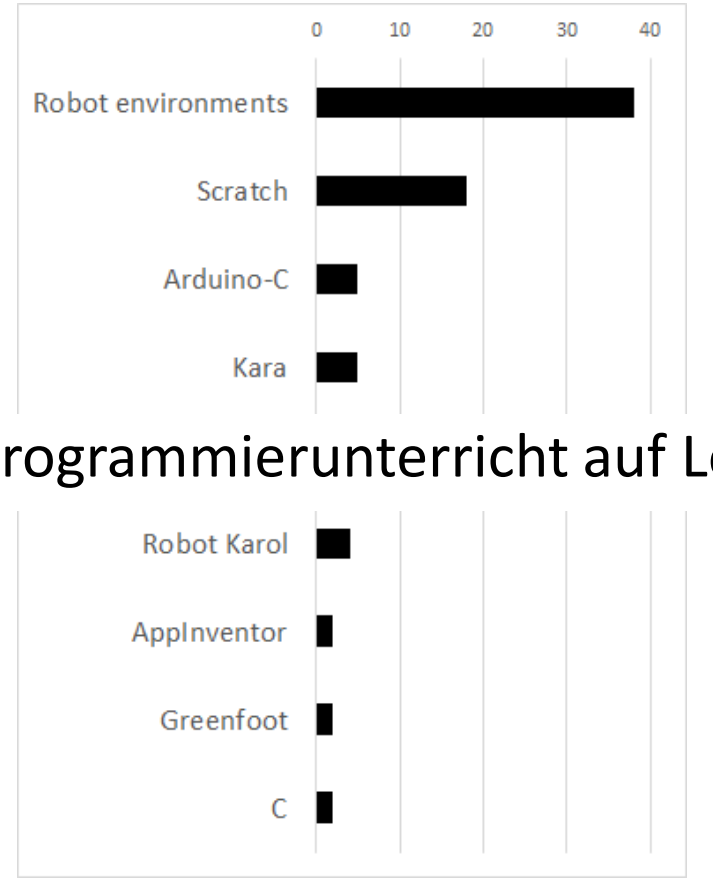
- Grafische Programmiersprachen führen zu besseren Lernerfolgen (Xu, Ritzhaupt et al. 2019)
  - Aber: Auch Umgebungen wie Scratch erzeugen Materialbezogene Last (Çakiroğlu et al. 2018)
  - Transfereffekte von Block- zu Textbasiert entstehen (wenn überhaupt) nicht automatisch (Weintrop & Wilensky 2019)

- Werkzeuge können die Motivation befördern oder nicht (Ruf, Mühling & Hubwieser 2014)
  - Auch auf verschiedene Art!

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen





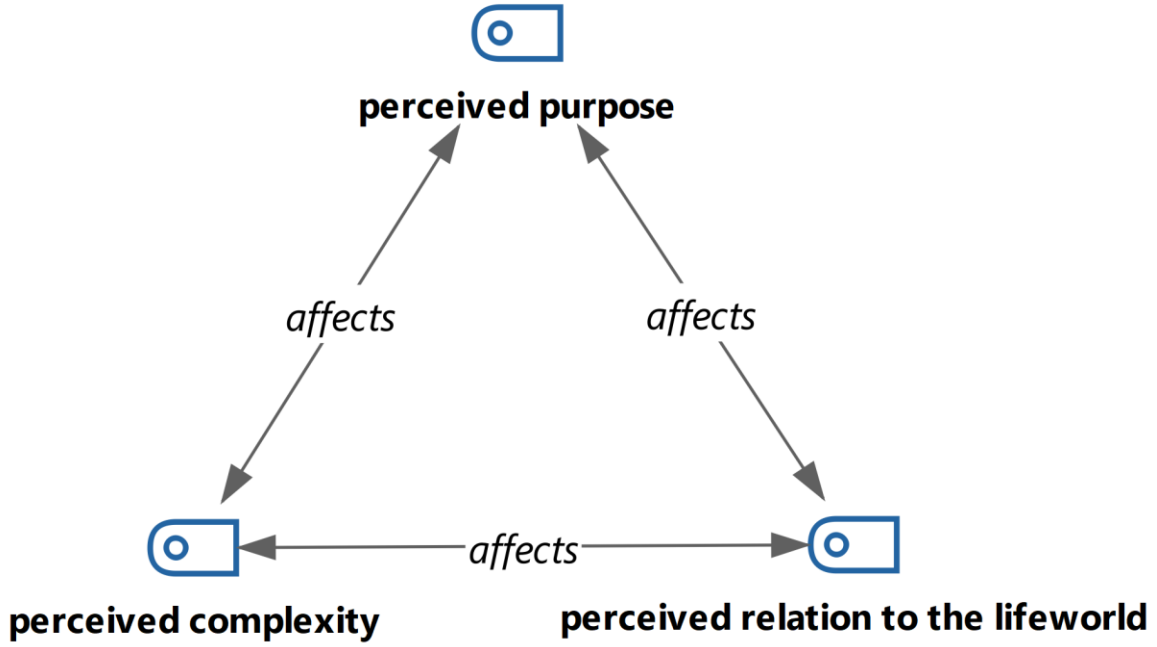
Wie wirkt Programmierunterricht auf Lernende?

(Kattmann, U., Duit, R., Gropengießer, H., & Komorek, M., 1997)

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen

- Interviews mit 7 Schülerinnen und 12 Schülern aus 4 Klassen.
- 2 bzw. 3 Interviewtermine
- Auswertung nach Grounded Theory



Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen

Da gibt es ein paar Elemente, nicht sehr viele, und dann ein paar Codes die man verwenden kann um die Welt zu definieren. Aber an einem Punkt, relativ schnell, war es das dann.

Das Problem mit Scratch ist, dass man nur bestimmte Dinge machen kann.

Es ist begrenzt auf ein Feld in dem man RubyKara mit Kommandos hin- und herschicken kann und das ist eigentlich gar nicht so schwer.

Wir haben zum Beispiel zu Hause – mein Vater hat etwas programmiert so dass die Heizung angeht sobald eine bestimmte Temperatur erreicht wurde. Sowas ist ziemlich nützlich aber soetwas kann man nicht mit, zum Beispiel, Scratch programmieren.

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen

Ich denke, es ist eine sehr einfache Programmiersprache […]. Ich denke, es ist der Anfang, dass es dafür da ist, die Grundlagen von Allem zu verstehen.

Es sind nur diese Blöcke und das ist kein echtes Progarmmieren. Es ist nur etwas zusammenfügen und dann, naja, läuft es. Echtes Programmieren funktioniert komplett anders, mit Befehlen und so weiter. Ja, das hatte ich gedacht, dass wir lernen würden, aber das war überhaupt nicht, was wir gelernt haben.

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen

Wünsche der Schüler:innen vor dem Unterricht:

- Programmiersprache lernen
  - Ich hoffe wir lernen soetwas wie Python
- Verstehen, wie ein Computer funktioniert
  - Ich freue mich drauf zu verstehen, was da wirklich im Hintergrund abläuft
- Etwas mit Alltagsbezug machen
  - Eine App mit der man Musik offline hören kann oder Videos runterladen oder so

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Schüler:innenvorstellungen

Empfundene Lernziele der Schüler:innen nach dem Unterricht:

- Wir haben alles wichtige über das Programm Scratch gelernt, das ganze Programm.
- Vorher kannte ich RobotKarol und Scratch nicht, jetzt weiß ich, wie man die benutzt.

Fazit der Schüler:innen:

- Ich wäre glücklicher, wenn wir etwas gelernt hätten, was mir im Alltag etwas nützt.
- Was das allgemeine Progammieren angeht habe ich gar keinen Fortschritt gemacht.

Braune, G. & Mühling, A. (2020). Learning to program. In T. Brinda (Hrsg.), *Proceedings of the 15th Workshop on Primary and Secondary Computing Education* (ACM Digital Library). New York:ACM, https://doi.org/10.1145/3421590.3421597

Ausblick

- Adaptives Testen und automatische Itemgenerierung
- Umfassendes Kompetenzentwicklungsmodell
- Sicht der Lehrkräfte auf Ihren Unterricht
- …

Vielen Dank!