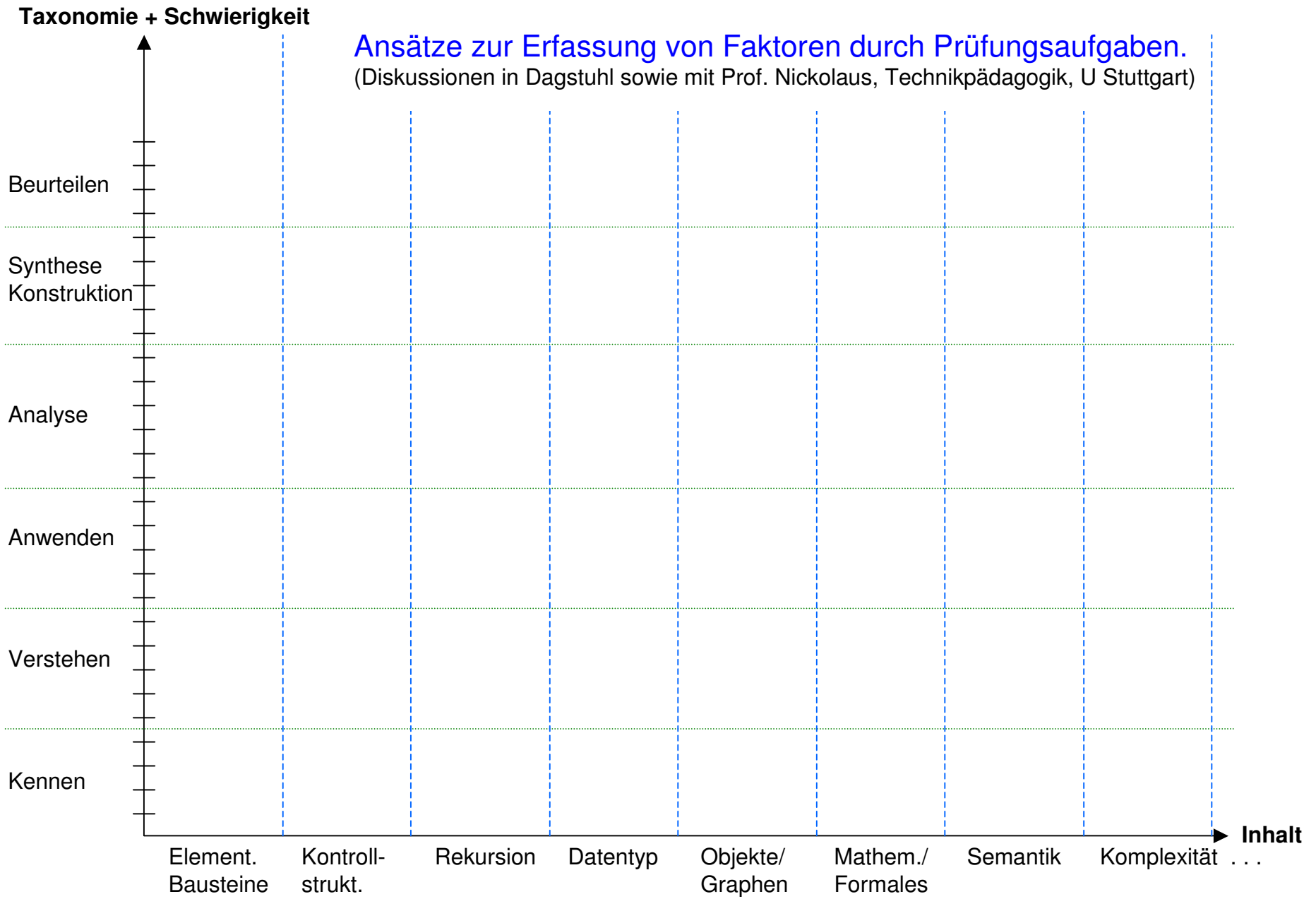
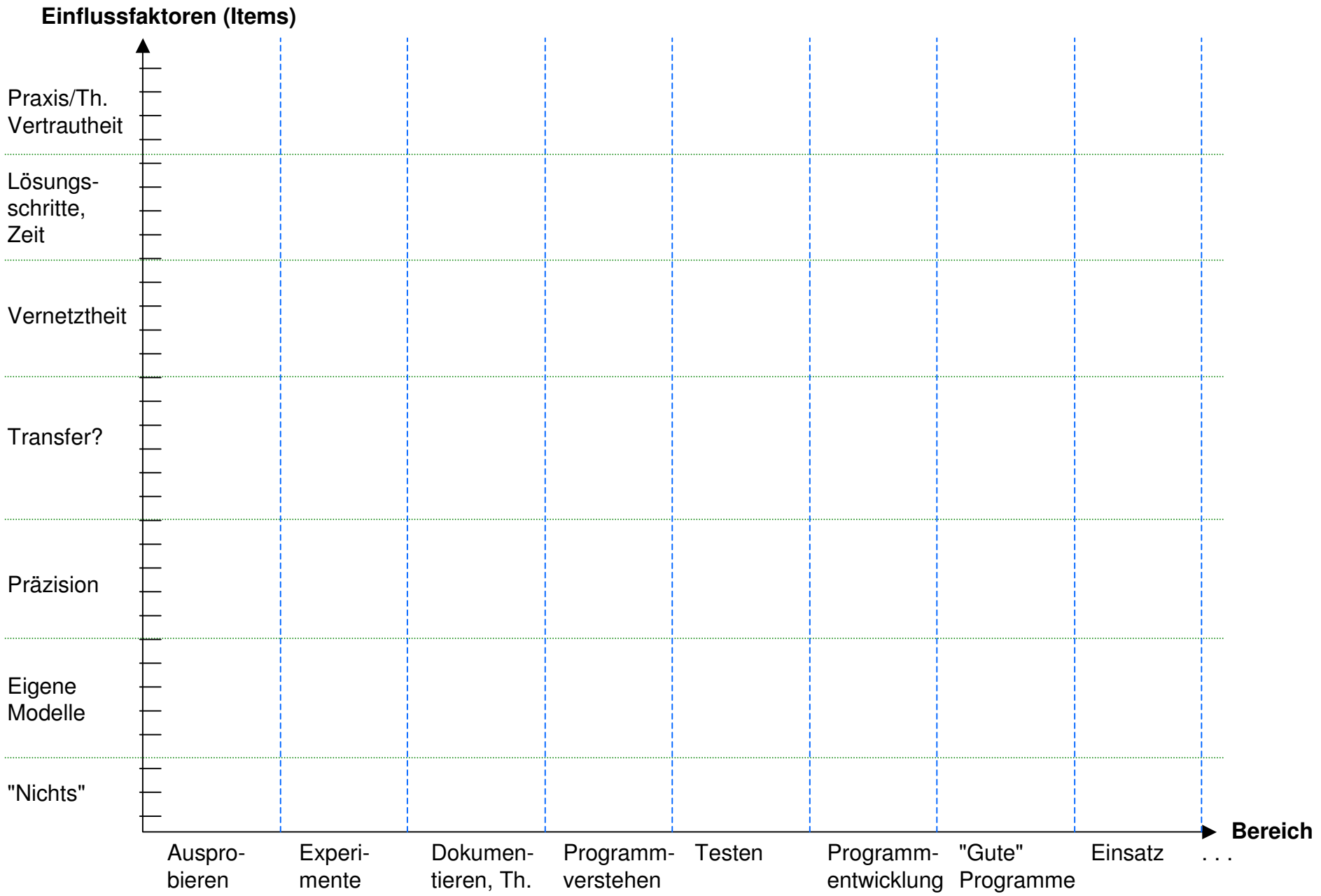

Taxonomie + Schwierigkeit
Ansätze zur Erfassung von Faktoren durch Prüfungsaufgaben.
(Diskussionen in Dagstuhl sowie mit Prof. Nickolaus, Technikpädagogik, U Stuttgart)
Beurteilen
Synthese
Konstruktion
Analyse
Anwenden
Verstehen
Kennen
Inhalt
Element.
Bausteine
Kontroll-
strukt.
Rekursion
Datentyp
Objekte/
Graphen
Mathem./
Formales
Semantik
Komplexität . . .
V. Claus, April 2009



(Th. = "Theoretisches")

Anwendung dieser Schemata (Teil 1).

Ansatz 1: Gegeben ist eine Aufgabe, evtl. mit Teilaufgaben. Füge deren Zuordnung zu Inhalten und Bereichen bzgl. Taxonomie, Schwierigkeit bzw. Einflussfaktoren in die Schemata ein. Beispiel:

Aufgabe 1 Listenbearbeitung

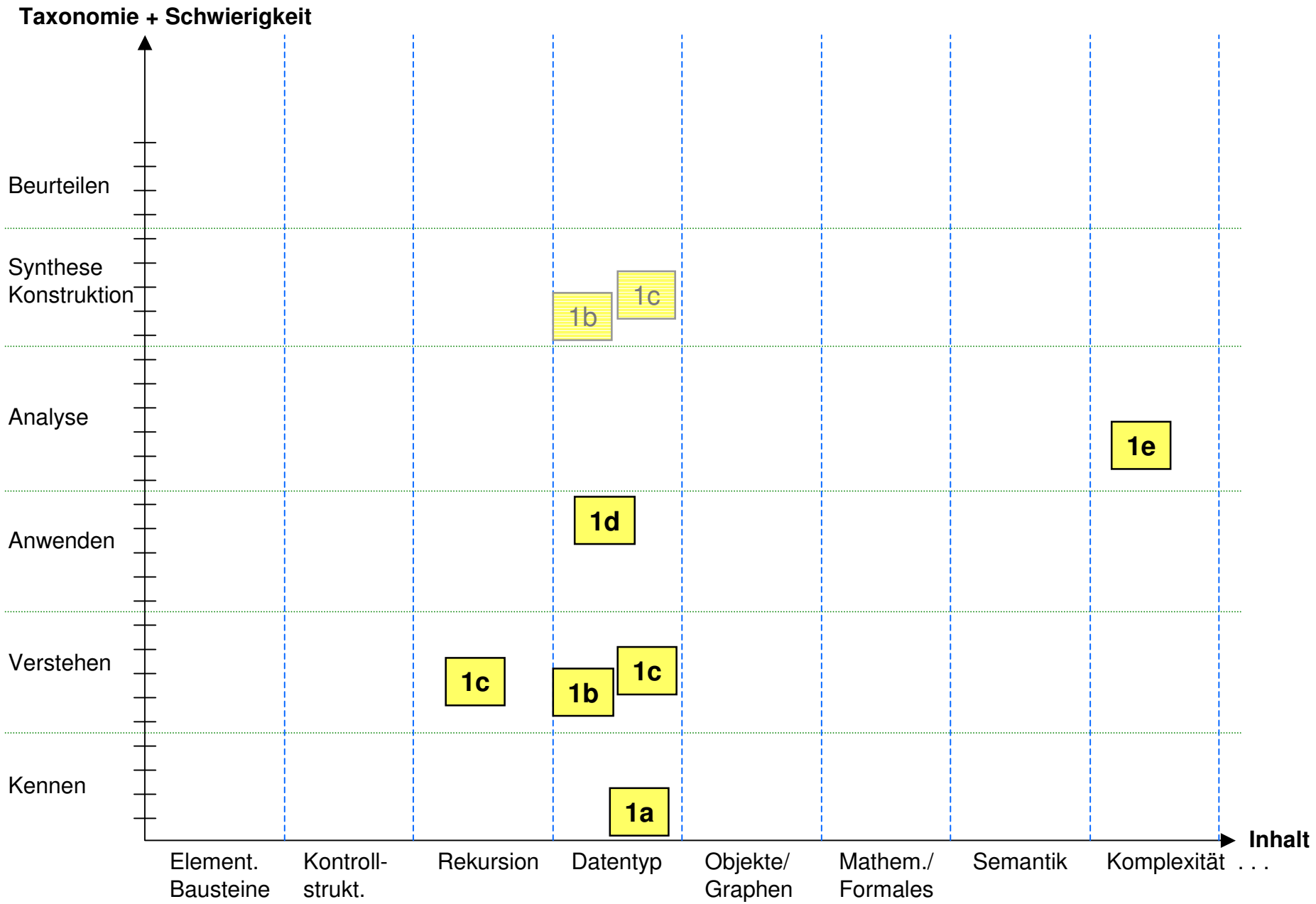
a) Geben Sie den Datentyp Liste an.

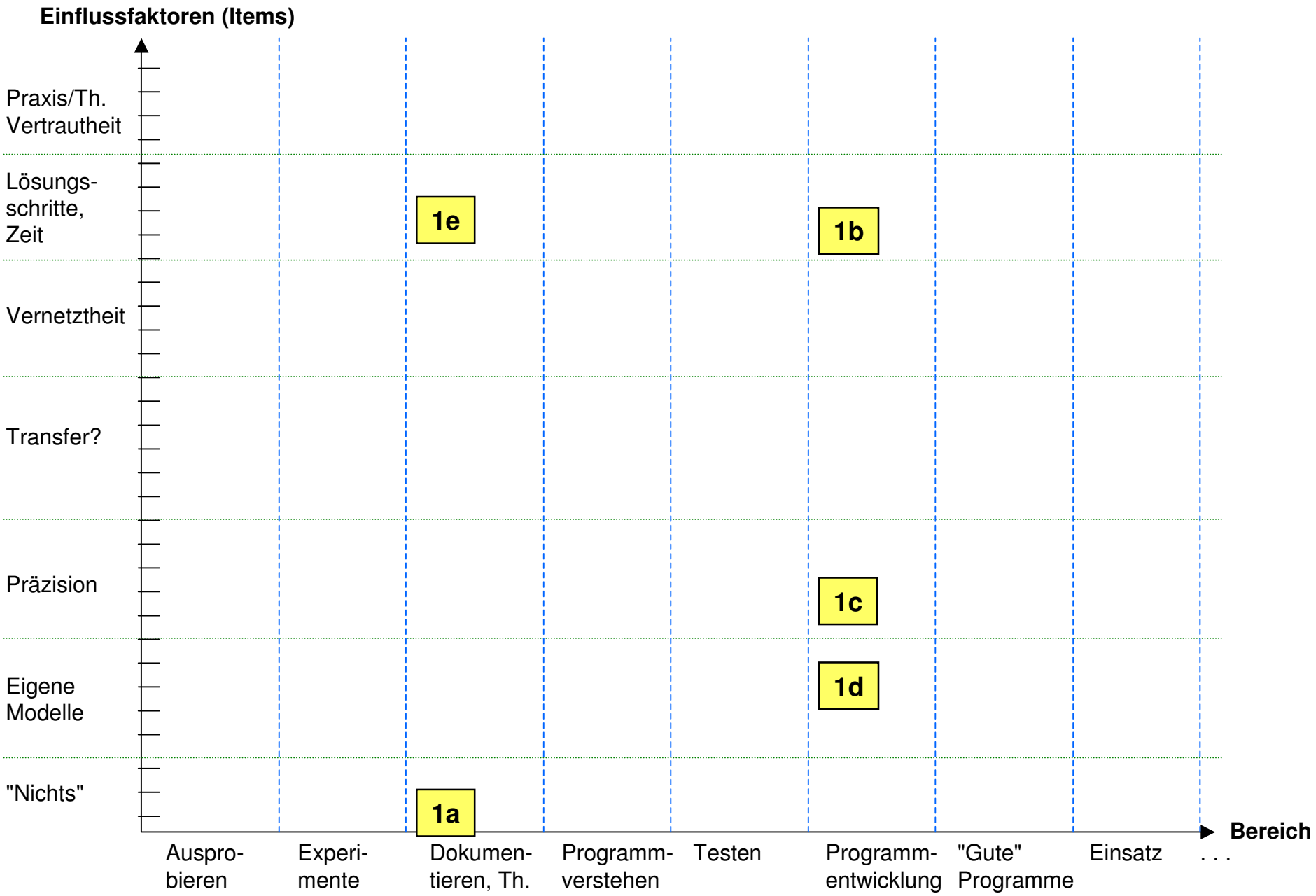
b) Gegeben sind eine Liste L ganzer Zahlen und eine Zahl s. Schreiben Sie ein Programm / eine Prozedur, welches die Zahl s am Ende und am Anfang von L einfügt.

c) Eine Liste L soll gespiegelt werden.

d) Es soll festgestellt werden, ob eine Liste L' in der zyklischen Liste L enthalten ist (d.h., ob L' eine Teilliste von LL und Länge(L') ≤ Länge(L) ist).

e) Berechnen Sie die Komplexität Ihres Programms aus Aufgabe d).


Taxonomie + Schwierigkeit
Beurteilen
Synthese
Konstruktion
Analyse
Anwenden
Verstehen
Kennen
1b
1c
1e
1d
1c
1b
1c
1a
Inhalt
Element.
Bausteine
Kontroll-
strukt.
Rekursion
Datentyp
Objekte/
Graphen
Mathem./
Formales
Semantik
Komplexität . . .


Einflussfaktoren (Items)
Praxis/Th. Vertrautheit
Lösungs-schritte, Zeit
Vernetztheit
Transfer?
Präzision
Eigene Modelle
"Nichts"
1e
1b
1c
1d
1a
Bereich
Auspro-bieren
Experi-mente
Dokumen-tieren, Th.
Programm-verstehen
Testen
Programm-entwicklung
"Gute" Programme
Einsatz
. . .

Anwendung dieser Schemata (Teil 2).

Ansatz 2: Man gebe vor, welche Gebiete in den Schemata abgedeckt werden sollen und konstruiert hiernach Aufgaben. Beispiel: Es soll der Bereich "Testen" in einer Aufgabe abgeprüft werden, wobei möglichst viele Inhalte mitgeprüft werden sollen. Das könnte folgendes Schema ergeben:

Schema 1: Kennen und Rekursion

Verstehen und Datentyp

Anwenden und Komplexität

Analyse und Semantik

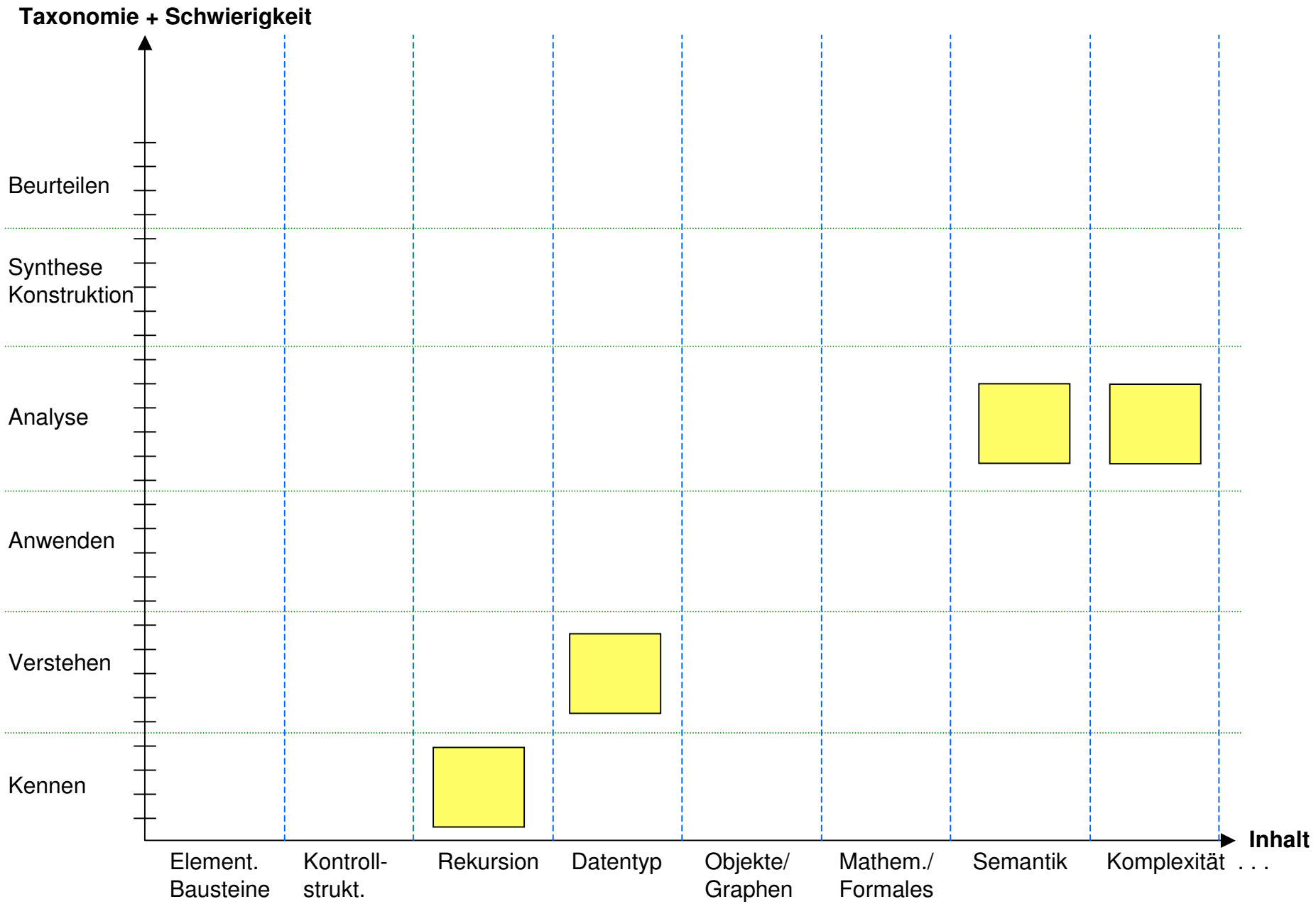
Schema 2: Eigenes Modell und Ausprobieren

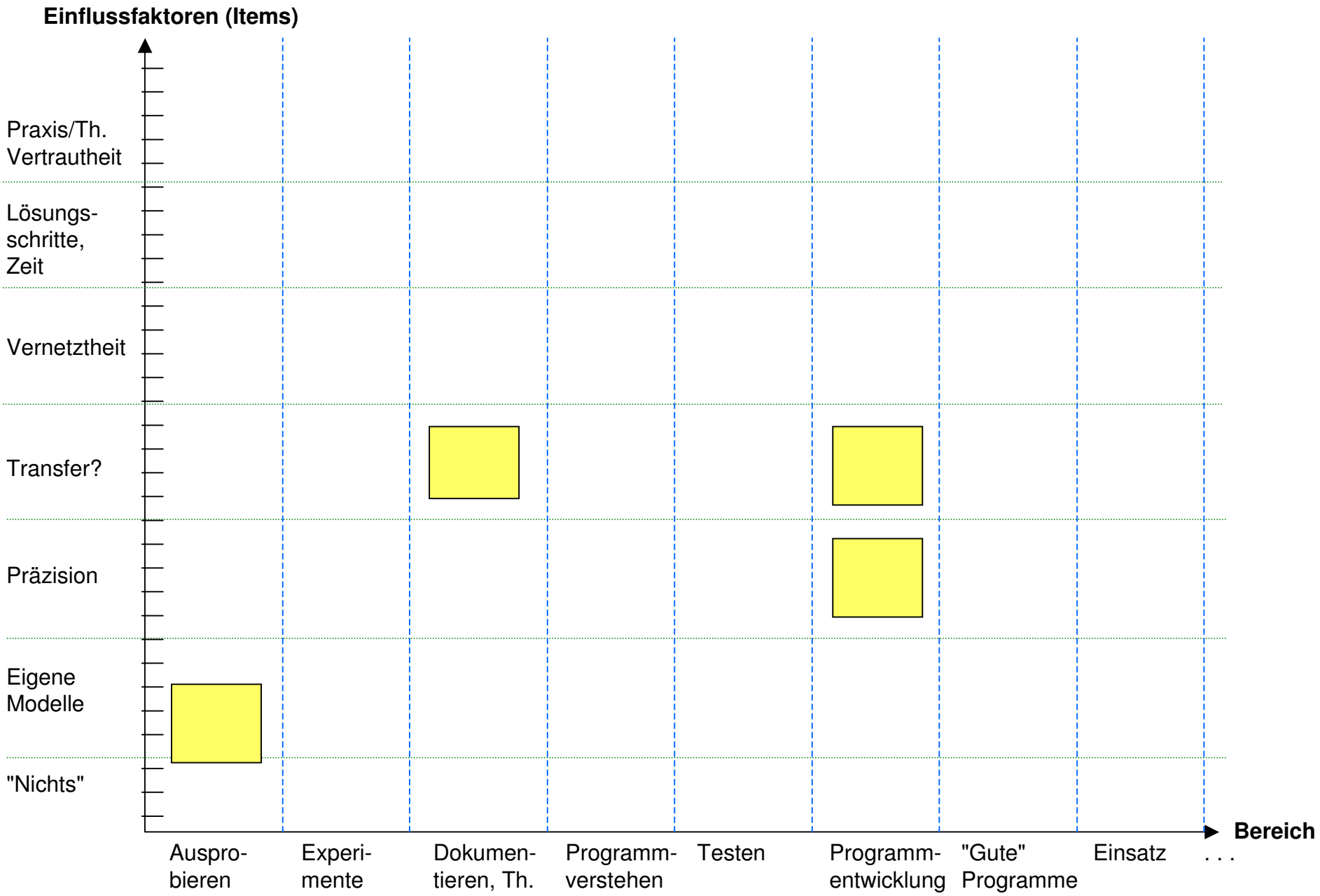
Nichts und Testen

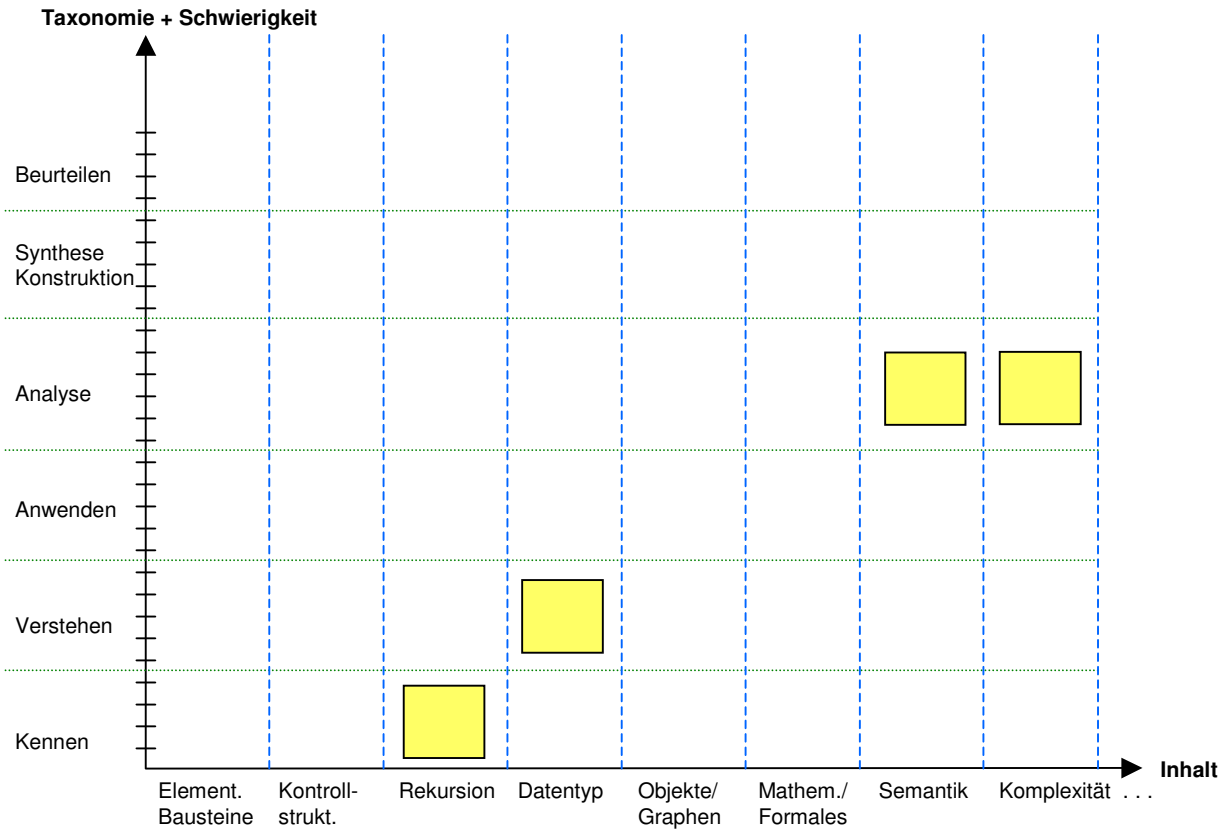
Präzision und Testen

Transfer und Testen

(Man kann nun diese Gebiete in den Schemata als Wunschvorstellung notieren. Anschließend muss man die Aufgabe so konstruieren, dass die Teilaufgaben die oben festgelegten Zuordnungen abdecken.)


Taxonomie + Schwierigkeit
Beurteilen
Synthese
Konstruktion
Analyse
Anwenden
Verstehen
Kennen
Inhalt
Element.
Bausteine
Kontroll-
strukt.
Rekursion
Datentyp
Objekte/
Graphen
Mathem./
Formales
Semantik
Komplexität . . .


Einflussfaktoren (Items)
Praxis/Th. Vertrautheit
Lösungs-schritte, Zeit
Vernetztheit
Transfer?
Präzision
Eigene Modelle
"Nichts"
Bereich
Auspro-bieren
Experi-mente
Dokumen-tieren, Th.
Programm-verstehen
Testen
Programm-entwicklung
"Gute" Programme
Einsatz
. . .



Notieren von Schema 1 in der Taxonomie-Übersicht. Auf gleiche Weise kann man Schema 2 notieren.

Eine Aufgabe mit diesem "Muster" untersucht einen Algorithmus, der relativ wenig Grundverständnisse benötigt und bei dem die Eigenschaften nicht offensichtlich sind. Zusammen mit dem zweiten "Muster" soll zugleich ein Programm entwickelt werden. Denkt man über diese Vorgaben nach, so liegen das binäre Suchen, Varianten der Quersummenbildung, eine Variante von Bubble-Sort, irgendeine rekursive Darstellung (f(x) = if x < 5 then x else f(f(x-5)+1) fi) nahe, wobei man sich zugleich auf die Minimierung der Konstanten beim Aufwand konzentrieren könnte.

Wir entscheiden uns hier für eine einfache Rekursion, die den Fibonaccizahlen nachgebildet ist.

Aufgabe 2 (Undurchsichtige?) Rekursion

Betrachte die Rekursion $f(x) = \begin{cases} x & , \ 0 \le x \le 2 \\ f(x-2) + f(x-3) - f(x-1) & , \ \text{für } x \ge 3. \end{cases}$

Dies definiert eine Funktion f: $\mathbf{N}_0 \rightarrow \mathbf{Z}$

a) Geben Sie die Werte f(x) für x = 3, 4, 5, 6, 7, 8, 9, 10 an.

b) Die Funktion lautet in Ada:

```
function F(X: Integer) return Integer is
begin
  if (X < 3) then return X;
  else return (F(x-2)+F(x-3)-F(x-1));
  end if;
end;
```

Schätzen Sie ab, wie viele Aufrufe von F für die Berechnung von f(x) erforderlich sind ($x \ge 0$).

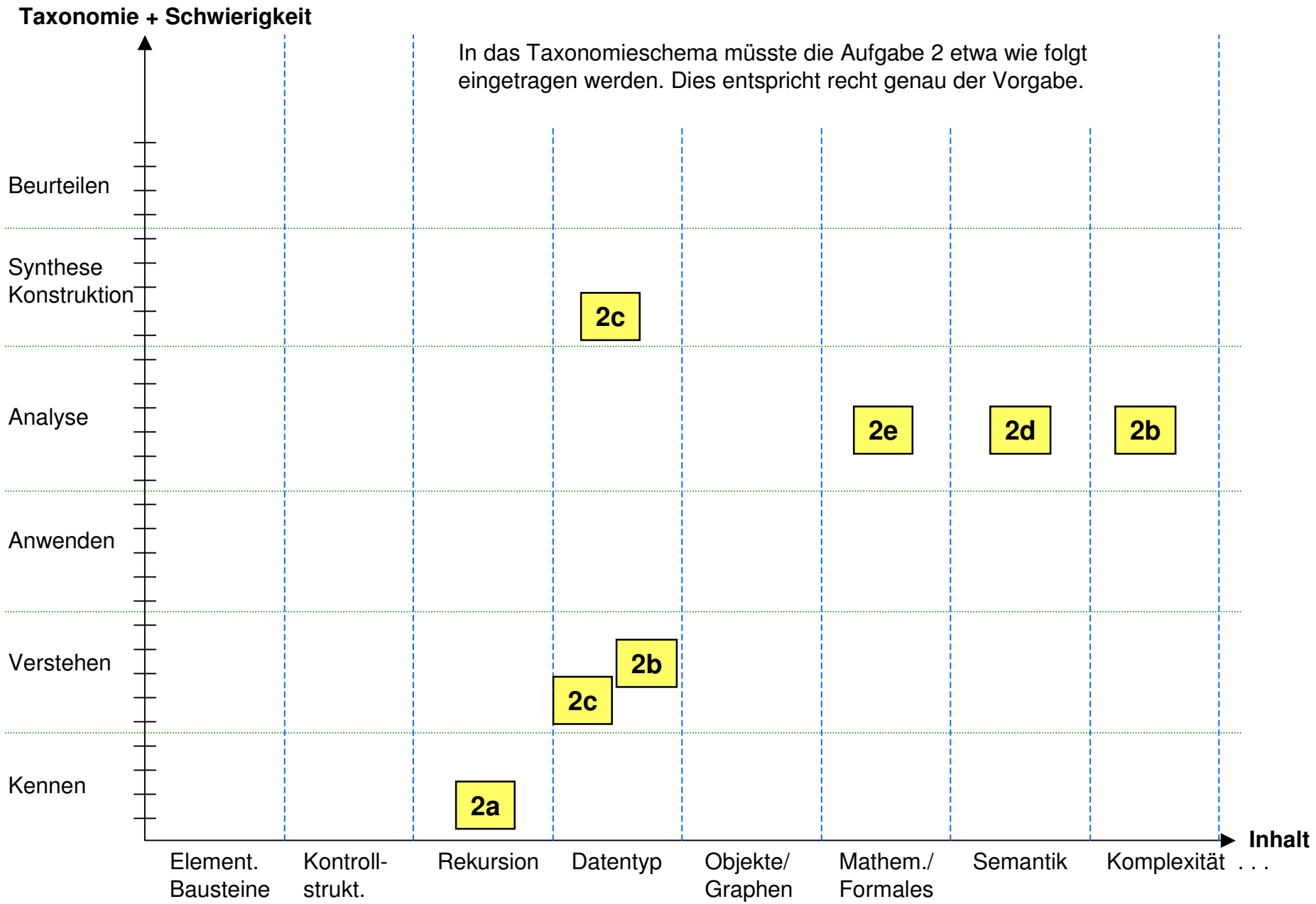
c) Schreiben Sie ein iteratives Programm, welches F(x) in Laufzeit O(x) berechnet.

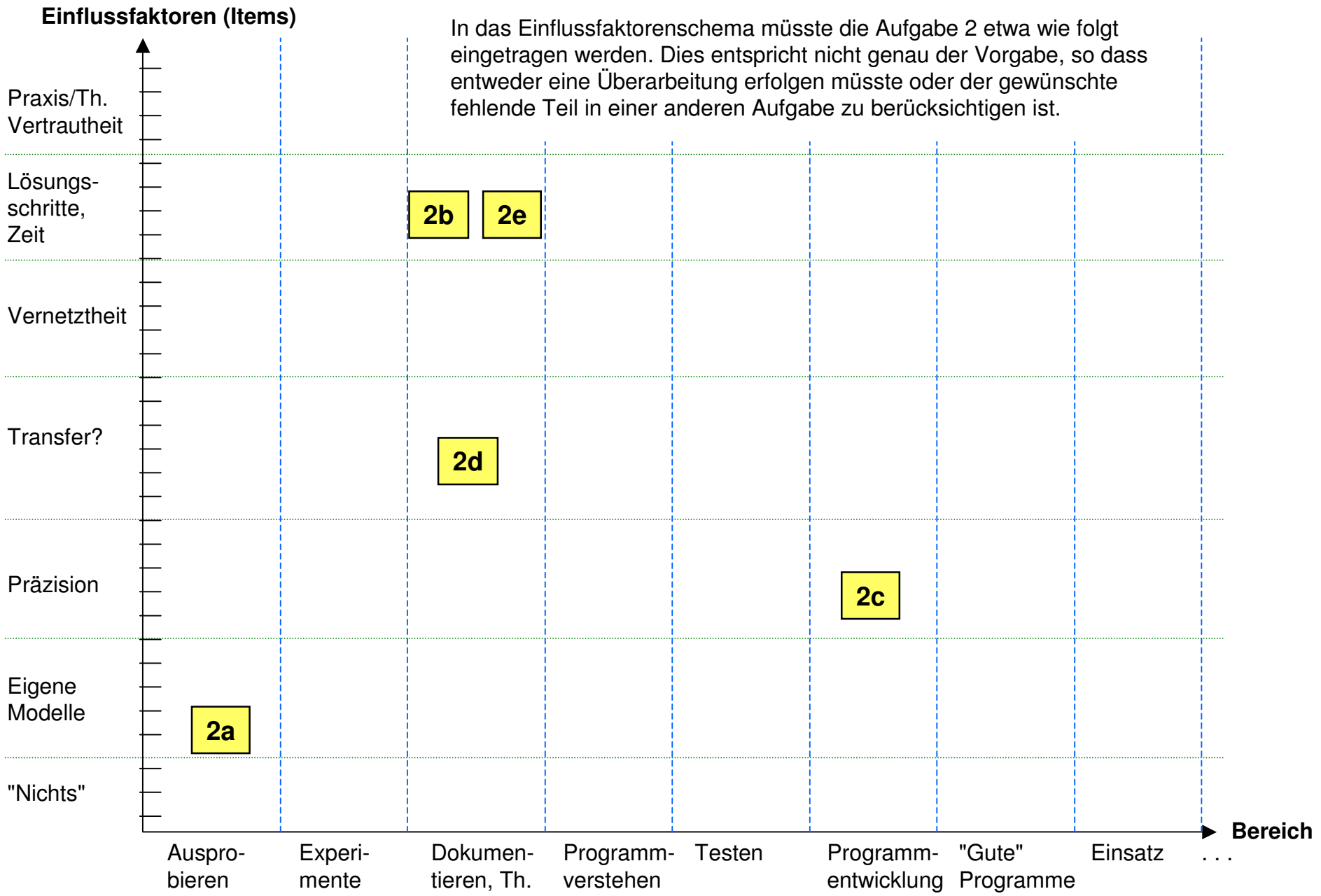
d) Statt f betrachten wir die Funktion $g(x) = \begin{cases} x + 5 & , \ 0 \le x \le 2 \\ g(x-2) + g(x-3) - g(x-1) & , \ \text{für } x \ge 3. \end{cases}$

Beweisen oder widerlegen Sie: Dann gilt $g(x) = f(x) + 5$ für alle x.

e) Wir behaupten: Es gibt Konstanten a und b mit $f(x) = \begin{cases} x & , \ \text{für alle gerade Zahlen x} \\ a - b \cdot x & , \ \text{für alle ungerade Zahlen x.} \end{cases}$

Beweisen Sie diese Aussage und geben Sie a und b an.


Taxonomie + Schwierigkeit
In das Taxonomieschema müsste die Aufgabe 2 etwa wie folgt eingetragen werden. Dies entspricht recht genau der Vorgabe.
Beurteilen
Synthese Konstruktion
Analyse
Anwenden
Verstehen
Kennen
2c
2e
2d
2b
2b
2c
2a
Inhalt
Element. Bausteine
Kontroll-strukt.
Rekursion
Datentyp
Objekte/ Graphen
Mathem./ Formales
Semantik
Komplexität . . .


Einflussfaktoren (Items)
In das Einflussfaktorenschema müsste die Aufgabe 2 etwa wie folgt eingetragen werden. Dies entspricht nicht genau der Vorgabe, so dass entweder eine Überarbeitung erfolgen müsste oder der gewünschte fehlende Teil in einer anderen Aufgabe zu berücksichtigen ist.
Praxis/Th. Vertrautheit
Lösungs-schritte, Zeit
Vernetztheit
Transfer?
Präzision
Eigene Modelle
"Nichts"
2a
2b
2e
2d
2c
Bereich
Auspro-bieren
Experi-mente
Dokumen-tieren, Th.
Programm-verstehen
Testen
Programm-entwicklung
"Gute" Programme
Einsatz
. . .