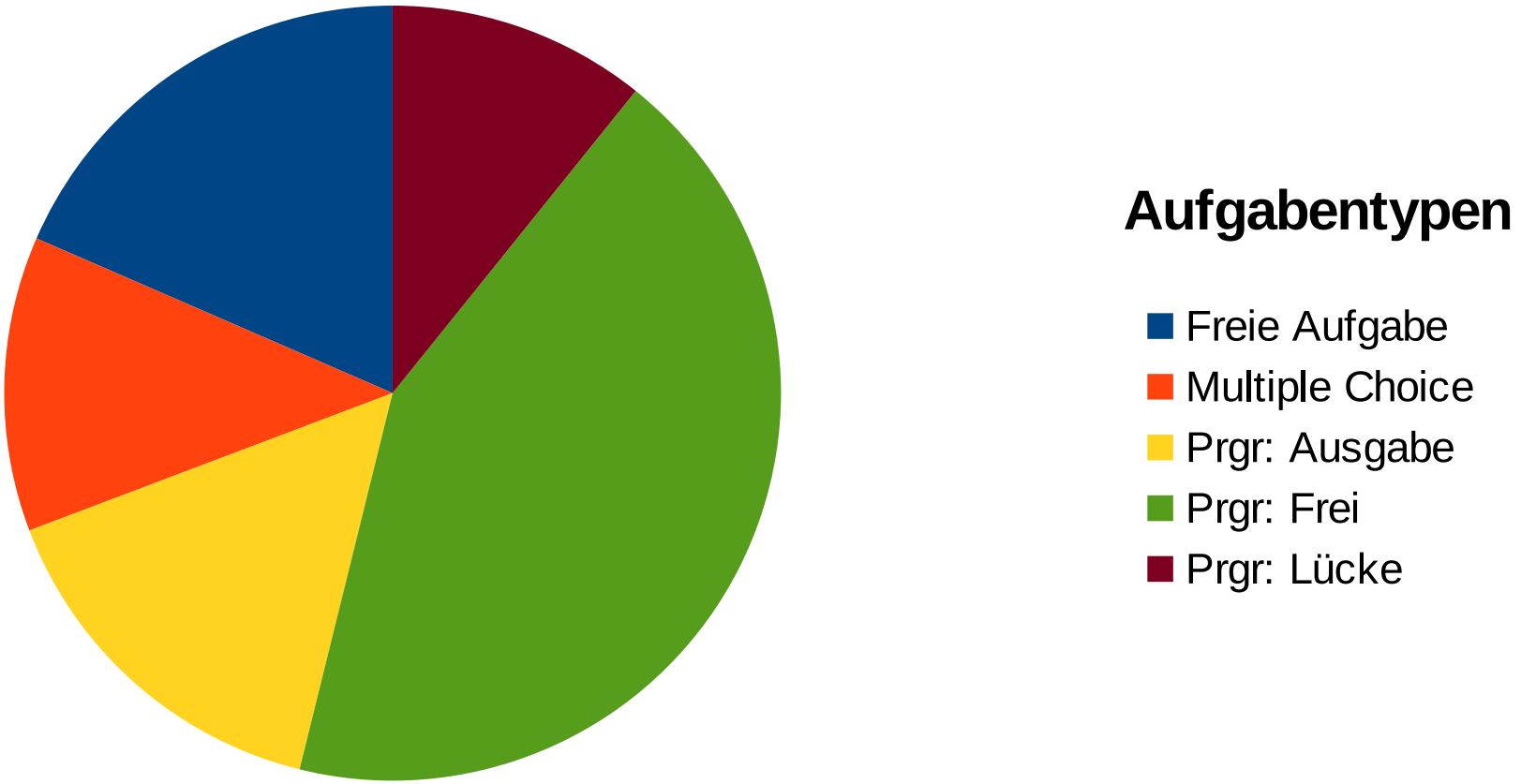
Elektronische Prüfungen in der Informatik

Ein Pilotversuch

Carsten Gips, FH Bielefeld (carsten.gips@fh-bielefeld.de)

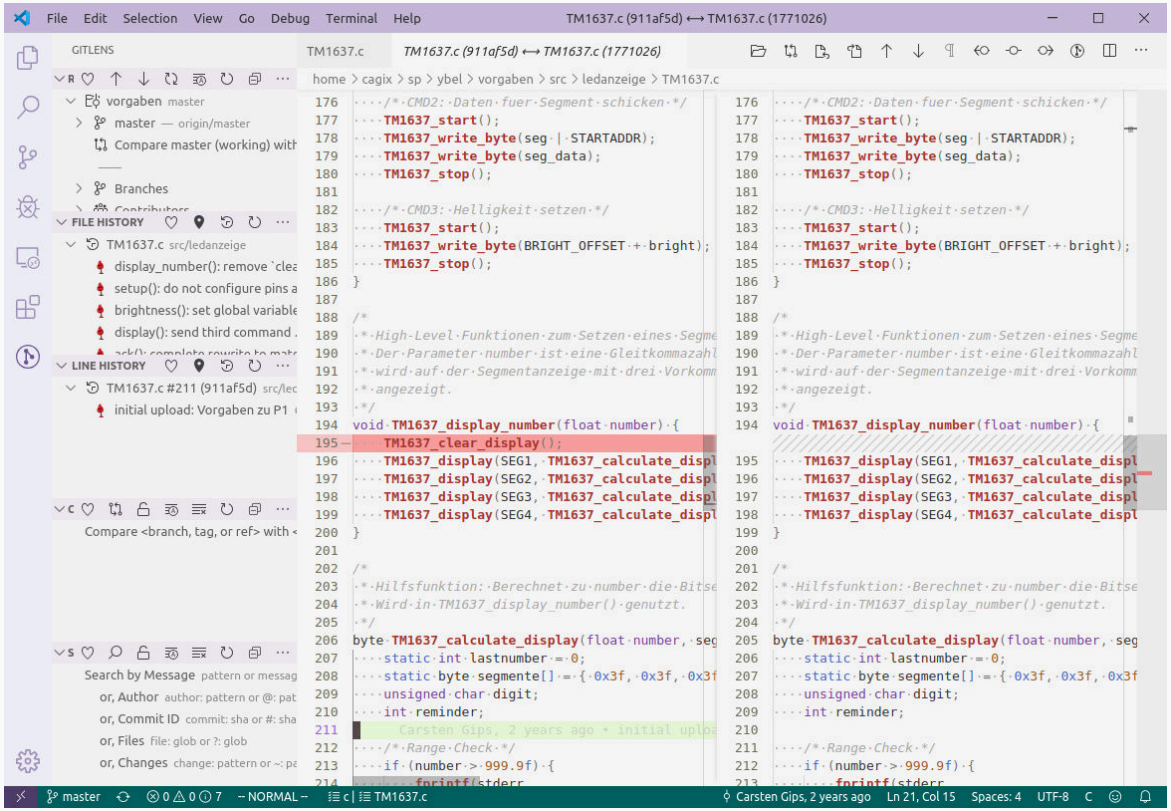
Ausgangssituation: Typisches Prüfungsszenario in der Informatik

Klausuren in Programmierfächern



Programmieren auf Papier ist praxisfremd

- Keine Unterstützung durch IDE, Compiler/Interpreter und andere Tools



- Keine Hilfe aus dem Netz (Dokumentation, API, Stack Overflow, . . .)

Korrektur der Lösungen ist sehr aufwändig

- Handschrift (!)
- Syntaxfehler: Grenze zw. noch akzeptabel und fehlerhaft
- Oft „kreative“ Lösungsansätze: Manuelles Nachprüfen am Rechner ...

$\Rightarrow$ Typisch ca. 50..80 Teilnehmer mit ca. 15 Seiten pro Klausur

Pilotversuch: Elektronische Prüfungen mit ILIAS

Vorbereitung: Fragetypen im ILIAS

Tests erstellen und verwalten

Das Test- und Assessment-System in ILIAS besteht technisch gesehen aus drei Grundkomponenten, die im Folgenden kurz erläutert werden sollen:

- **Fragen**

  Die Fragen bilden die Grundlage für alle Tests, die mit dem Assessment-System vorgenommen werden können. Zu einer Frage können Sie einen Fragentext, Punktezahlen, mögliche Antworten, Bearbeitungszeiten etc. vergeben. Momentan sind im Test- und Assessment-System neun Fragetypen vorgesehen:

Auswahl	Eintragung	An- und Zuordnung	Markieren	Eigene Antwort verfassen	Sonstiges
Single-Choice-Fragen	Lückentexte	Anordnungsfragen (vertikal und horizontal)	Fragentyp „Hotspot/Imagemap“	Freitext-Fragen	Java-Applet-Fragen
Multiple-Choice-Fragen	Fragentyp „Numerische Antwort“	Zuordnungsfragen	Fragentyp „Fehler/Worte markieren“	Datei-Upload-Fragen	Flash-Applet-Fragen
KPrim-Choice-Fragen	Fragentyp „Begriffe benennen“				
	Formelfragen				

- **Fragenpools**

  Fragenpools sind Container, die Sie zum Sammeln von Fragen verwenden **können**. Ob Sie einzelne Fragen nach inhaltlichen, zufälligen oder anderen Kriterien zu Fragenpools zuordnen, bleibt Ihnen vollständig überlassen (s. Kap. Fragenpools erstellen und verwalten).

Vorbereitung: Übertragen der Aufgaben

Codeanalyse Haskell [ID: [redacted]]

Text einfügen | Ok

Betrachten Sie den folgenden Haskell-Code:

```
1  take 4 $ [(x,y) | x <- [1..], y <- [1..]]
2
3  take 4 $ [x+1 | x <- [1..], odd x]
4
5  take 4 $ [(n, even n) | n <- [1..]]
6
7  head (tail [1..4])
8
9  (head . tail) [1..4]
10
11 head . tail $ [1..4]
12
13 head . tail [1..4]
14
15 zip [1 .. 2] ["one", "two", "three"]
16
17 [(x, y) | x <- [1 .. 2], y <- ["one", "two", "three"]]
```

Geben Sie für alle Zeilen die Ergebnisse der Auswertung der jeweiligen Ausdrücke an.

- Zeile 1 (1P):
- Zeile 3 (1P):
- Zeile 5 (1P):
- Zeile 7 (0.5P):
- Zeile 9 (0.5P):
- Zeile 11 (0.5P):
- Zeile 13 (1P):
- Zeile 15 (1P):
- Zeile 17 (1.5P):

Hinweis: Wenn eine Zeile nicht ausgewertet werden kann, etwa durch Syntax- oder Typ-Fehler, geben Sie bitte als Antwort "**Feh**

Frage *

Absatz

Betrachten Sie den folgenden Haskell-Code:

```
1  take 4 $ [(x,y) | x <- [1..], y <- [1..]]
2
3  take 4 $ [x+1 | x <- [1..], odd x]
4
5  take 4 $ [(n, even n) | n <- [1..]]
```

Pfad: p » img

Stunden: 0 Minuten: 8 Sekunden: 0

Absatz

- Zeile 1 (1P): [gap 1][(1\, 1)\, (1\, 2)\, (1\, 3)\, (1\, 4)],[(1\,1)\, (1\,2)\, (1\,3)\, (1\,4)],[(1\,1)\,(1\,2)\,(1\,3)\,(1\,4)][/gap]
- Zeile 3 (1P): [gap 2][2\, 4\, 6\, 8],[2\,4\,6\,8][/gap]
- Zeile 5 (1P): [gap 3][(1\, False)\, (2\, True)\, (3\, False)\, (4\, True)],[(1\,False)\, (2\,True)\, (3\,False)\, (4\,True)],[(1\,False)\,(2\,True)\,(3\,False)\,(4\,True)][/gap]
- Zeile 7 (0.5P): [gap 4]2[/gap]
- Zeile 9 (0.5P): [gap 5]2[/gap]

Pfad: p

Um eine Lücke in den Text einzufügen, setzen Sie den Cursor an die entsprechende Position und nutzen Sie die Schaltfläche "Textlücke". Anschließend stehen weiter unten entsprechende Bearbeitungsbereiche zur Verfügung. Ebenso können Sie die Lücken zur Bearbeitung im Lückentext direkt anklicken.

Textlücke

Zwischen Groß- und Kleinschreibung wird nicht unterschieden

Wenn Sie hier einen Wert eintragen, werden Textlücken, welche keinen eigenen Wert für eine maximale Länge definieren, sowie numerische Lücken mit dieser Länge erzeugt, so dass es nicht möglich ist eine größere Anzahl an Zeichen einzugeben. Für numerische Lücken ist darüber hinaus zu beachten, dass das Dezimaltrennzeichen dabei mit gezählt wird.

☑

Wenn ausgewählt, werden Lücken mit identischen Lösungen entsprechend der Vorgaben bewertet, selbst wenn die gleiche Lösung mehrere Male verwendet wurde. Ist das Kontrollkästchen nicht ausgewählt, wird nur die erste verwendete identische Lösung bewertet.

Textlücke

Länge des Textfeldes

80

Ist ein Wert größer 0 eingetragen, wird diese Lücke mit der hier angegebenen Länge erzeugt. Ist kein Wert angegeben, wird diese Lücke mit der global angegebenen Textfeldlänge erzeugt.

Werte *

Antwort-Text	Punkte
[(1, 1), (1, 2), (1, 3), (1, 4)]	1 + −
[(1,1), (1,2), (1,3), (1,4)]	1 + −
[(1,1),(1,2),(1,3),(1,4)]	1 + −

Lücke entfernen

LÜCKE 2

Durchführung der E-Klausur

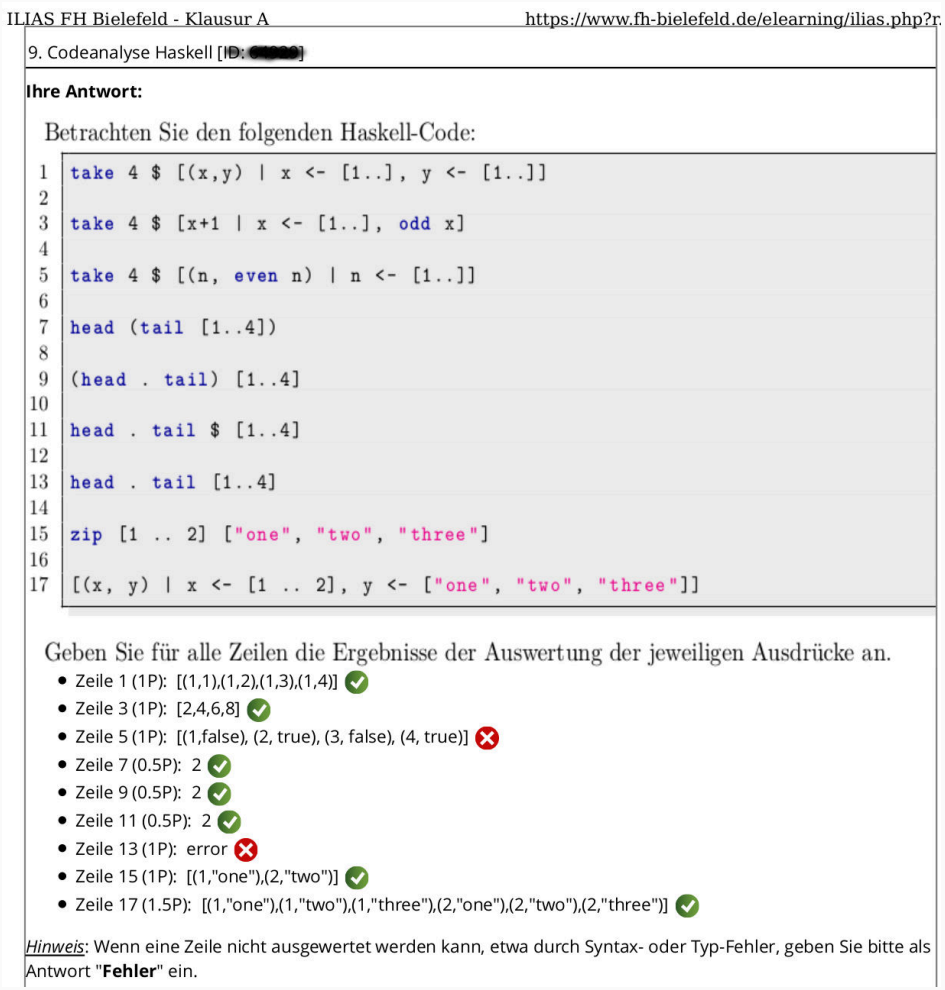
- Fragetypen-Demo für Teilnehmer (im Vorfeld)

- Teilnehmer in speziellen ILIAS-Kurs eintragen
- Test online schalten, wenn alle eingeloggt
- Bearbeitungszeit lässt sich auf Server begrenzen

- Ergebnis:
  - Ausdrucken der (teilautomatisch bewerteten) Klausur (PDF) plus Deckblatt
  - Teilnehmer unterschreiben ihre Klausur

Korrektur und Bewertung

- MC vollautomatisch
- Lückentexte: teilautomatisch
- Freitext: manuelle Korrektur



ILIAS FH Bielefeld - Klausur A https://www.fh-bielefeld.de/elearning/ilias.php?r

9. Codeanalyse Haskell [ID: [redacted]]

Ihre Antwort:

Betrachten Sie den folgenden Haskell-Code:

```
1  take 4 $ [(x,y) | x <- [1..], y <- [1..]]
2
3  take 4 $ [x+1 | x <- [1..], odd x]
4
5  take 4 $ [(n, even n) | n <- [1..]]
6
7  head (tail [1..4])
8
9  (head . tail) [1..4]
10
11 head . tail $ [1..4]
12
13 head . tail [1..4]
14
15 zip [1 .. 2] ["one", "two", "three"]
16
17 [(x, y) | x <- [1 .. 2], y <- ["one", "two", "three"]]
```

Geben Sie für alle Zeilen die Ergebnisse der Auswertung der jeweiligen Ausdrücke an.

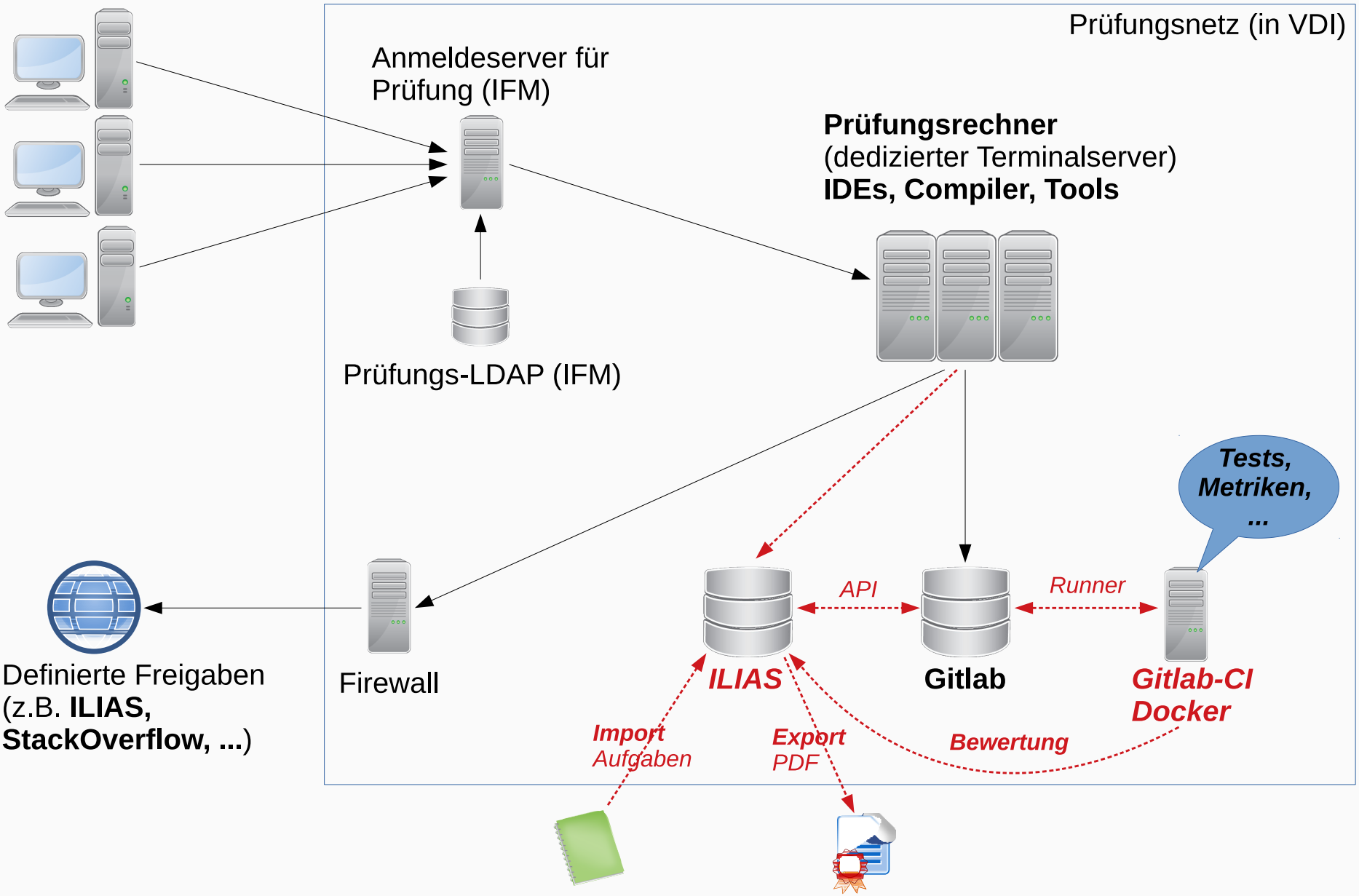
- Zeile 1 (1P): [(1,1),(1,2),(1,3),(1,4)] ✓
- Zeile 3 (1P): [2,4,6,8] ✓
- Zeile 5 (1P): [(1,false), (2, true), (3, false), (4, true)] ✗
- Zeile 7 (0.5P): 2 ✓
- Zeile 9 (0.5P): 2 ✓
- Zeile 11 (0.5P): 2 ✓
- Zeile 13 (1P): error ✗
- Zeile 15 (1P): [(1,"one"),(2,"two")] ✓
- Zeile 17 (1.5P): [(1,"one"),(1,"two"),(1,"three"),(2,"one"),(2,"two"),(2,"three")] ✓

Hinweis: Wenn eine Zeile nicht ausgewertet werden kann, etwa durch Syntax- oder Typ-Fehler, geben Sie bitte als Antwort "**Fehler**" ein.

⇒ Korrektur auf Ausdruck oder im ILIAS

Zukunft: Konzeptskizze

Konzept für Prüfungsumgebung für die Informatik



Vielen Dank! Fragen?