Klausur Web-Anwendungen, WS17

B_CGT 14.0, B_ECom I14.0, B_Inf 14.0, B_MInf 14.0, B_WInf 14.0,
ITAM 2.0, ITAS 2.0, ITAW 2.0, KAI 2.0

- Gestellt von: Marcus Riemer (**mri**)
- Erlaubte Hilfsmittel: Zeichengeräte, Taschenrechner.
- Dauer: 60 Minuten.
- Diese Klausur besteht inklusive dieses Deckblattes und den Anhängen aus **7 Fragen** auf **14 Seiten**. Es können maximal **43** **Punkte** erreicht werden. Punktzahlen sind in Kästen notiert und einmal als Summe (Σ) je Aufgabe und dann einzeln für jede Teilaufgabe angegeben.
- Sie sollten zu keinem Zeitpunkt zwischen mehreren Seiten blättern müssen, um sich vorgegebene Quelltexte anzusehen. Daher werden alle **Anhänge mit den Quelltexten** zur Klausur **separat ausgeteilt**. Die Heftung der Anhänge sowie der Klausur dürfen **nicht** gelöst werden, Notizen auf dem separat ausgeteilten Anhang werden keinesfalls bewertet. Sie müssen die Anhänge daher auch nicht wieder abgeben.
- Verwenden Sie zur **Lösung** die **freigelassenen Räume auf den Aufgabenblättern** (diese können, müssen aber nicht komplett gefüllt werden). Sollte der Platz nicht ausreichen, verwenden Sie die Rückseite der Aufgabenblätter.
- Der Inhalt der **Rückseiten** wird nur **gewertet**, wenn er eindeutig als Lösung gekennzeichnet ist (Stichwort "Lösung" mit Angabe der Aufgabennummer) und vom vorgesehenen Lösungsfeld auf die zu wertende Rückseite verwiesen wird.

1) Semantik und Syntax von HTML

$\sum 6$

Das in Anhang 1a zu sehende HTML-Fragment ist Kind des <body>-Elements. Es enthält einige Fehler und ist teilweise semantisch unsinnig. Geben sie zwei strukturelle oder syntaktische Fehler. Benennen Sie außerdem einen semantisch unsinnigen Sachverhalt. Beschreiben Sie für jeden gefundenen Fehler knapp die Fehlerursache unter Angabe einer Zeilennummer.

Hinweis: Das Dokument enthält mehr Fehler, als Sie im Rahmen dieser Aufgabe finden müssen. Benennen Sie keinesfalls mehr Fehler als gefordert!

(a) **Syntaktisch** fehlerhaftes HTML in Zeile _____ 2

(b) **Syntaktisch** fehlerhaftes HTML in Zeile _____ 2

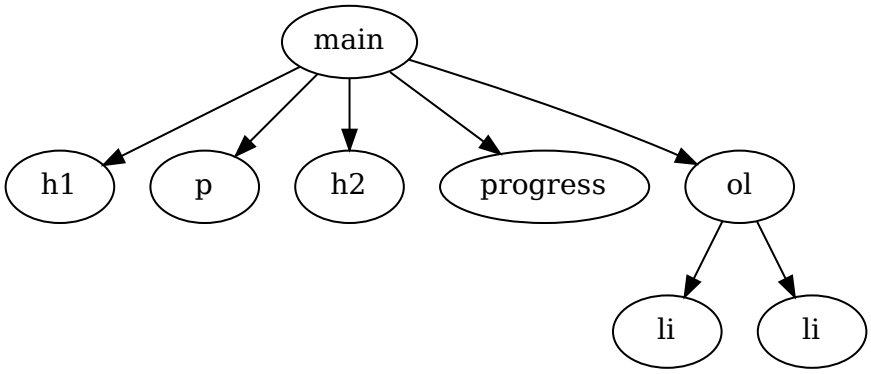
(c) **Semantisch** unsinniges HTML in Zeile _____ 2

2) `Templating` $\boxed{\sum 5}$

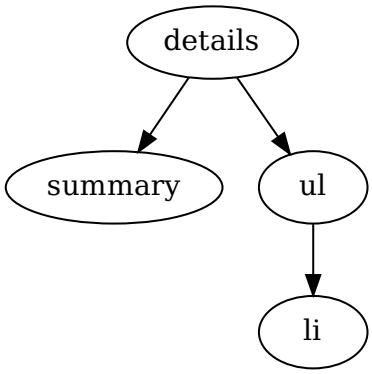
Geben Sie zu dem in Anhang 2a gegebenem Liquid-Template einen passenden Datensatz in `JSON`-Notation an, durch welchen sich die folgenden `HTML`- bzw. `DOM`-Fragmente ergeben.

(a) `<main><h1>readmelater</h1><div>Data Error</div></main>` [1]

(b) [2]



(c) 2



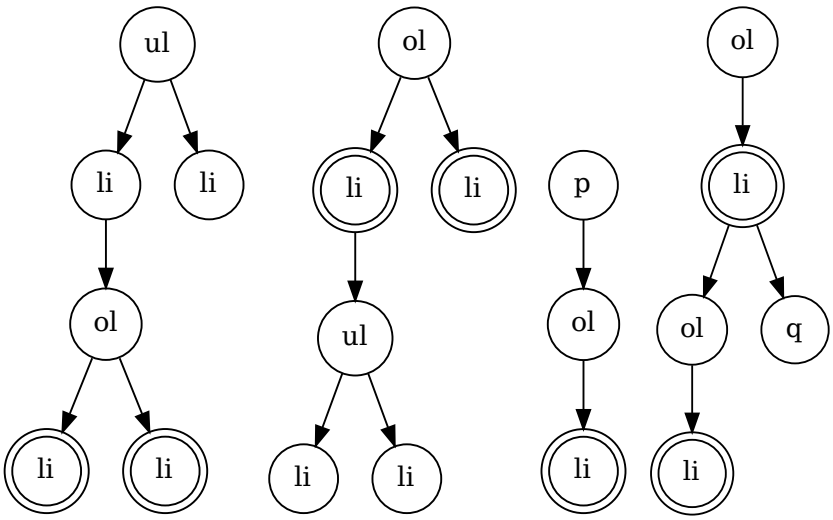
3) CSS-Selektoren

$\sum 4$

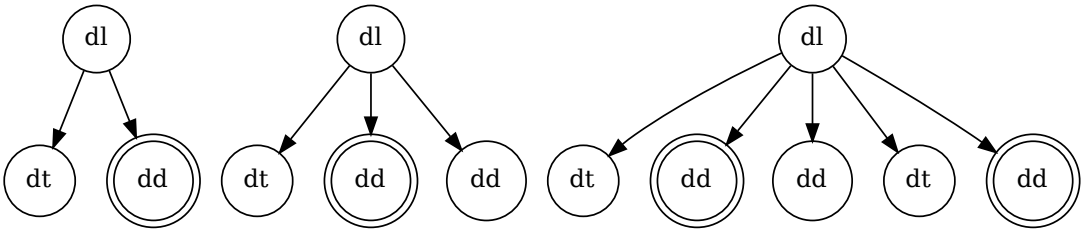
Schreiben Sie jeweils einen CSS-Selektor, der exakt die doppelt eingekreisten Knoten der dargestellten DOM-Bäume selektiert.

Hinweis: Es ist nach genau **einem** Selektor je Aufgabenstellung gefragt, auch wenn mehrere DOM-Bäume referenziert werden. Diese Anforderung schließt die Verwendung des Komma-Operators explizit aus.

(a) 2



(b) 2

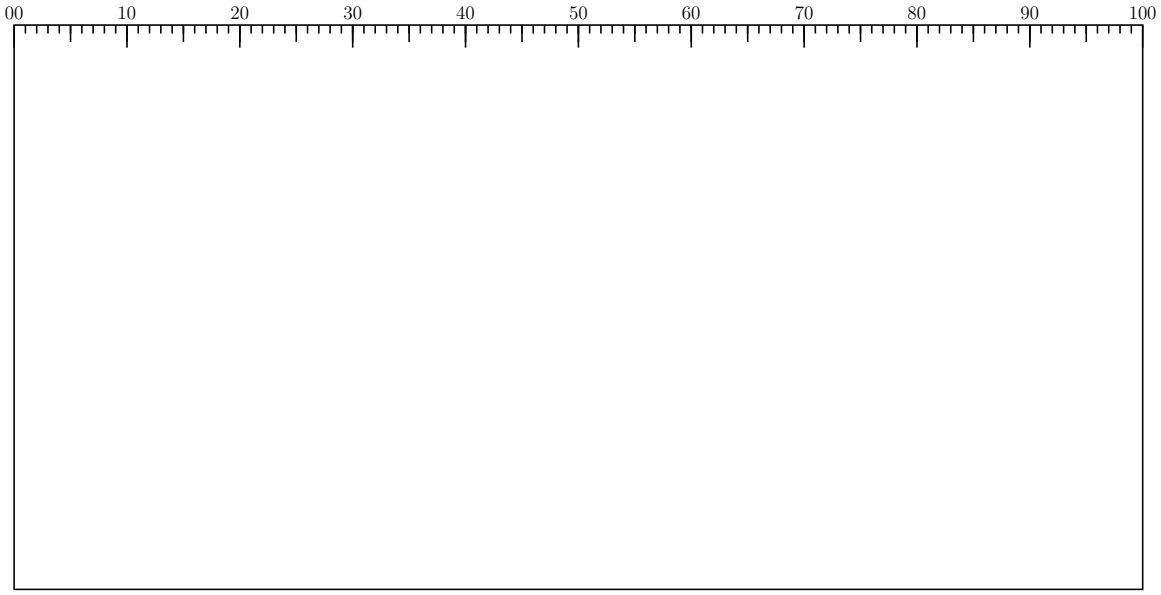


4) Box-Modell und `display: float` ∑7

In Anhang 4 - `HTML` sehen Sie ein `HTML`-Fragment welches mit dem `CSS`-Dokument aus Anhang 4 - `CSS` in einem 100 Pixel breiten Viewport dargestellt wird.

(a) Skizzieren Sie die Darstellung in diesem Viewport. 5

- Gehen Sie davon aus, dass für keinen der Texte ein Zeilenumbruch nötig ist und zeichnen Sie für Bilder den Wert des `src`-Attributes ein.
- Zeichnen Sie Linien für die definierten `border`-Eigenschaften.
- Geben Sie außerdem in jedem Element in Klammern die im Layout benötigte Breite in Pixeln an. Die Höhe aller Elemente können sie frei wählen.
- Zur groben Orientierung sehen Sie unter dem entsprechenden Raum für die Lösung ein Lineal mit Pixelmaß.



(b) Mit welchem `CSS`-Ausdruck ließe sich die `width`-Eigenschaft der Bilder so festlegen, dass jedes Bild in der Darstellung ein Drittel der Breite des Elternelements einnimmt? 2

5) JavaScript ∑ 8

Betrachten Sie die JavaScript-Programme in Anhang: Aufgabe 5. Das dort gezeigte Programm ist syntaktisch einwandfrei und kann problemlos ausgeführt werden.

(a) Was ist das Ergebnis des folgenden Aufrufs? 2

```
1 [justus, peter, bob]
2   .filter(p => p.job === justus.job)
3   .map(p => p.nameLength())
```

(b) Was ist die Ausgabe von `justus.tellJob()`? 1

(c) Erstellen Sie mit dem `new`-Operator eine Instanz `p` von `Person` bei welcher der Aufruf von `p.nameLength()` zu einem Laufzeitfehler führt. 1

(d) Zählen Sie die Namen **aller** Eigenschaften des Objekts `justus` auf. 1

(e) Zählen Sie die Namen **aller** Eigenschaften des Objekts `MyClass.prototype` auf. 1

(f) Ist die Zuweisung `justus.name = "Peter"` möglich, obwohl `justus` mit `const` eingeführt wurde? 2

☐ Ja ☐ Nein

Wenn ja: Geben sie ein Beispiel für eine unzulässige Zuweisung.

Wenn nein: Wie wäre eine Änderung des Namens im Nachhinein dennoch möglich?

6) XML

$\sum 7$

Die XML-Dokumente für diese Aufgabe (zu finden in den Listings 5b, 5c, 5d und 5e) sind syntaktisch einwandfrei.

(a) Welches dieser Dokumente ist im Sinne der Schema-Definition in Anhang 5a valide? [1]

☐ Dokument 1 ☐ Dokument 2 ☐ Dokument 3 ☐ Dokument 4

(b) Benennen Sie je einen Fehler in den übrigen Dokumenten. Nehmen Sie dabei Bezug auf die Zeile, in welcher der Fehler auftritt, die Zeile(n) im XML, in denen die verletzte Restriktion genannt wird und begründen Sie stichwortartig die Art der Verletzung.

i. Dokument _____ in XML-Zeile _____, Schema-Zeile: _____ [2]

ii. Dokument _____ in XML-Zeile _____, Schema-Zeile: _____ [2]

iii. Dokument _____ in XML-Zeile _____, Schema-Zeile: _____ [2]

7) Gemischte Themengebiete $\boxed{\sum 6}$

Setzen Sie exakt die Anzahl der geforderten Kreuze! Sofern Sie mehr Kästchen ankreuzen als die Aufgabenstellung vorsieht, wird die entsprechende Teilaufgabe mit 0 Punkten bewertet.

(a) Welches der folgenden `HTTP`-Verben kann in `HTML`-Formularen eingesetzt werden um große Datenmengen zu übertragen (1 Kreuz)? [1]

☐ `GET` ☐ `HEAD` ☐ `POST` ☐ `PUT` ☐ `PATCH` ☐ `DELETE`

(b) Welche der folgenden Aussagen über Formulare in `HTML` sind wahr (3 Kreuze)? [3]

☐ `<input type='date'>` ermöglicht Datums- und Uhrzeitangaben.

☐ `<input type='submit'>` wird als Knopf dargestellt.

☐ `<input type='radio'>` ermöglicht eine "m aus n"-Auswahl.

☐ `<input type='checkbox'>` wird mittels eines Dropdown-Menüs dargestellt.

☐ `<span>` ermöglicht die Angabe einer Zeitspanne.

☐ `<div>` ermöglicht die Angabe diverser Daten.

☐ `<canvas>` ermöglicht die Eingabe von Ölgemälden.

☐ Das `<datalist>`-Element kann genutzt werden um mittels eines `<input>`-Element vom Typ `text` eine "1 aus n"-Auswahl um beliebige Eingaben zu erweitern.

☐ Das Attribut `id` bestimmt mit welchem Namen der vom Benutzer eingegebene Wert an den Server übertragen wird. Wenn dieses Attribut fehlt berechnet der Browser anhand der Position im `DOM`-Baum eine künstliche `id`.

☐ Ungültige Daten in einem Formular werden mit der `CSS`-Pseudoklasse `:invalid` versehen. Dadurch wird es unmöglich die entsprechenden Daten zu übertragen.

☐ Es ist Aufgabe des Webservers übertragene Formularwerte gegen die `HTML`-Validierungsregeln zu prüfen.

☐ Das Attribut `mandatory` kann genutzt werden um Benutzer auf verpflichtende Eingaben hinzuweisen.

☐ Dateien sollten in einem Formular mit `method='GET'` übertragen werden.

☐ Mit dem `<label>`-Element lassen sich Erläuterungen und Beschriftungen für Eingabeelemente definieren.

(c) Welche Aussage über Parameterübergabe in JavaScript ist wahr (1 Kreuz)? [1]

☐ Der Aufruf einer Funktion mit mehr aktuellen Parametern als formalen Parametern ist ein Syntaxfehler.

☐ Der Aufruf einer Funktion mit weniger aktuellen Parametern als formalen Parametern ist ein Syntaxfehler.

☐ Der Aufruf einer Funktion ist unabhängig von der Anzahl der aktuellen und formalen Parameter möglich.

(d) "Generatoren" sind eine der Möglichkeiten in JavaScript bequem Iterator-Objekte zu erzeugen. Welche der folgenden Aussagen ist wahr (1 Kreuz)? [1]

☐ Generatoren sind Bestandteil von `ECMA`-Script 6 und steht deswegen in jeder konformen Laufzeitumgebung zur Verfügung.

☐ Generatoren sind Bestandteil von node.js und ist deswegen nicht in Browsern verfügbar.

☐ Generatoren sind Bestandteil der `DOM`-Schnittstelle und ist deswegen ausschließlich in Browsern verfügbar.

Anhang: Aufgabe 1

```
1  <header class="welcome">
2    <h1>
3      <li>The Bullet-Blog</ul>
4    </h1>
5    <img id="header-image"
6         src="/logo.svg"
7         alt="Logo">
8  </header>
9  <aside>
10   Besuchen Sie auch unsere anderen Themenseiten zu so wichtigen Themen
11   wie:
12   <ul>
13     <li>Kunst & Kultur</li>
14     <ol>
15       <li><a href="/tannenzapfen">Tannenzapfen</a></li>
16       <li><a href="/bleistifte">Bleistifte</a></li>
17       <li><a href="/kursiv"><q>Die kursive Seite</q></a></li>
18     </ul>
19   </ul>
20 </aside>
21 <nav>
22   <p>
23     Dieses Blog berichtet über Neuigkeiten aus der Welt der unsortierten
24     Aufz&auml;hlungen. Ob tief verschachtelt oder nur auf einer Ebene,
25     hier werden Sie geholfen!
26   </p>
27   <p>
28     Oder um es mit den weisen Worten von Konfuzius zu sagen:
29     <q>Zweimal messen, einmal s&auml;gen!</q>
30   </p>
31 </nav>
```

(a) Fragwürdiges HTML

Anhang: Aufgabe 2

```
{% if page.titles %}
  <main>
    <h1>readmelater</h1>
    {% if page.titles.size > 0 %}
      <p>There are {{ page.titles.size }} titles available</p>
      <h2>Your Progress</h2>
      <progress value="{{ page.read_titles }}"
                max="{{ page.titles.size }}"></progress>
      <ol>
        {% for page in page.titles %}
          <li>{{ page }}</li>
        {% endfor %}
      </ol>
    {% else %}
      <div>Data Error</div>
    {% endif %}
  </main>
{% else %}
  <details>
    <summary>There are no titles</summary>
    <ul>
      {% for error in page.errors %}
        <li>{{ error }}</li>
      {% endfor %}
    </ul>
  </details>
{% endif  %}
```

(a) Zu benutzender Template-Code

Anhang: Aufgabe 4

```html
<h1>Hallo</h1>
<div>
  <img src="1">
  <img src="2">
  <img src="3">
</div>
```

(a) Anhang 4 - HTML

```css
h1 {
    margin-left: 10px;
    width: 20px;
    border-left: 1px solid pen; /* Farbe des Zeichengeräts */
}

div img {
    float: right;
    width: 33%;
    border: 1px solid pen;      /* Farbe des Zeichengeräts */
}
```

(b) Anhang 4 - CSS

Anhang: Aufgabe 5

```
1  class Person {
2    constructor() {
3      this.name = arguments[0];
4      this.job = arguments[1];
5    }
6
7    nameLength() {
8      if (this.name.length > 5) {
9        return ("long");
10     } else {
11       return ("short");
12     }
13   }
14 }
15
16 Person.prototype.greet = function() {
17   console.log(`I am ${this.name}. My name is ${self.nameLength()}`);
18 }
19
20 Person.prototype.tellJob = () => {
21   console.log(`My Job: ${this.job}`);
22 }
23
24 const justus = new Person("Justus", "Detektiv");
25 const peter = new Person("Peter", "Detektiv");
26 const bob = new Person("Bob", "Archiv");
```

(a) Klasse `Person`

Anhang: Aufgabe 6

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="project">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="author" type="xs:string" />
        <xs:element name="source" type="xs:string"
                    minOccurs="1" maxOccurs="unbounded"/>
        <xs:element name="option" type="xs:string"
                    minOccurs="0" maxOccurs="unbounded" />
        <xs:element name="security" minOccurs="0" type="xs:integer" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

(a) XML-Schema

```xml
<project>
  <author>Bill G.</author>
  <security>-12</security>
  <source>defects.c</source>
  <source>drm.c</source>
  <option>crash</option>
</project>
```

(b) Dokument 1

```xml
<project>
  <author>Bill G.</author>
  <source>defects.c</source>
  <source>drm.c</source>
  <option>crash</option>
  <security>-12</security>
</project>
```

(c) Dokument 2

```xml
<project>
  <author>Bill G.</author>
  <security>-12</security>
</project>
```

(d) Dokument 3

```xml
<project>
  <author>Bill G.</author>
  <source>defects.c</source>
  <option>crash</option>
  <source>drm.c</source>
</project>
```

(e) Dokument 4