Klausur Grundlagen der Webentwicklung

PI/KI Bachelor

Prof. Dr. Kretschmer, WS 2021/2022

Rahmenbedingungen: 120 Minuten Bearbeitungszeit, keine Hilfsmittel. Schreiben Sie Ihre Lösungen in den dafür vorgesehenen Platz. Sollte dieser nicht ausreichen, können Sie auch die Rückseite verwenden. Vermerken Sie das dann bitte auf der Vorderseite. Insgesamt sind 81 Punkte erreichbar. Dies entspricht 100%. Mit 32 Punkten haben Sie bestanden.

Name	Vorname	Matrikelnr.	Studiengang

1	2	3	4	5	6	Summe	Note

Aufgabe 1

(32 Punkte) Beantworten Sie **kurz** folgende Fragen. Das Themengebiet steht jeweils in Klammern.

1.	Was ist ein Codeelement? (Unicode)
2.	Welche möglichen Längen (in Bytes) kann die Codierung eines Zeichens in UTF-8 haben? (Unicode)
3.	Was versteht man unter einem globalen Attribut? (HTML)
4.	Welche zwei Elemente werden bei ungeordneten Listen verwendet? (HTML)
5.	Welche drei Begriffe kürzt RGB ab? (CSS)
6.	Was besagt die Längenangabe `50vmin` ? (CSS)
7.	Nennen Sie zwei strukturelle Pseudoklassen. (CSS)
8.	Aus welchen vier ineinander geschachtelten Boxen besteht eine Box? (CSS)
9.	Wofür steht die Abkürzung DOM und was ist das? (DOM)
10.	Nennen Sie drei primitive Datentypen. (JavaScript)
11.	Was ist ein Versprechen (promise)? (JavaScript)

12.	Wozu wird eine viewBox verwendet? (SVG)
13.	Nennen sie zwei gebräuchliche MIME-Typen. (HTTP)
14.	Was ist ein *shadow host*? (Webkomponenten)
15.	Nennen Sie zwei der sechs Forderungen an eine REST-API, aber nicht „einheitliche Schnittstelle“ mit seinen Unterpunkten. (REST)
16.	Warum entspricht ein Endpunkt wie `/delete/article` nicht den REST-Anforderungen? (REST)

Aufgabe 2

(8 Punkte) Ein XML-Dokument enthält Daten zu den Teilnehmern eines Seminars. Bei der Anmeldung zum Seminar kann jeder Teilnehmer aus einer Liste von Themen bis zu drei Wunschthemen auswählen und diese mit einer Priorität versehen.
Das XML-Element `teilnehmer` hat beliebig viele Kindelemente `stud`.
`stud` hat als Kindelemente erst `vorname`, dann `nachname`, gefolgt von ein, zwei oder drei `wunsch`-Elementen. Das Attribut `matrikel` von `stud` enthält die Matrikelnummer der/des Studierenden, die diese/n identifiziert. Sie besteht aus dem Zeichen `m`, gefolgt von 7 Ziffern.
`wunsch` ist ein leeres Element. Das Attribut `prio` enthält die Zahl 1,2 oder 3. Diese gibt die Priorität des Wunsches an. Das Attribut `thema` enthält das Thema.
Geben Sie eine möglichst gut zutreffende (Teil-)DTD für die XML-Elemente `teilnehmer`, `stud`, `wunsch` und deren Attribute an. Die anderen Kindelemente müssen **nicht** definiert werden. Wenn nichts anderes gesagt ist, sind die Attribute erforderlich.

Aufgabe 3

(8 Punkte) Gegeben sei das linksstehende body-Element einer HTML-Datei.

a) Geben Sie unter den zwei Selektoren auf der rechten Seite die `id`'s der durch die Selektoren selektierten Elemente an.

b) Geben Sie je einen Selektor an, der nur das Element mit der `id` 7 bzw. 20 selektiert und nichts anderes. Dabei dürfen Sie nicht auf das Attribut `id` zugreifen.

```
<body id="1">
  <p id="2">Brettspiele</p>
  <ul id="3">
    <li id="4">
      <p id="5">Schach</p>
      <ul id="6">
        <li id="7">Schnellschach</li>
        <li id="8">
          <p id="9">Blitzschach</p>
          <ul id="10">
            <li id="11">3 Minuten</li>
            <li id="12">5 Minuten</li>
          </ul>
        </li>
        <li id="13">Blindschach</li>
      </ul>
    </li>
    <li id="14">Go</li>
    <li id="15">Gobang</li>
    <li id="16">
      <p id="17">Dame</p>
      <ol id="18">
        <li id="19">normale Dame</li>
        <li id="20">Laskerdame</li>
      </ol>
    </li>
  </ul>
</body>
```

a)

`body>*>li>p`

`ul>li:nth-child(4n-1)`

b)

Selektor für 7:

Selektor für 20:

Aufgabe 4

(8 Punkte)

Geben Sie den Wert der vier Konstanten `r1,r2,r3,r4` an.

```
const f = (g, x) => g(g(g(x)))
const h = x => x + 5
const r1 = f(h, 1)

const cities = ['Berlin', 'Hamburg', 'Trier', 'Saarlouis', 'Bamberg']
const r2 = cities.slice(1, 3).map(v => v.length)
const r3 = cities.filter(v => /a[b-n]/.test(v)).reduce((r, v) => r + v.length, 0)
cities.pop()
cities.shift()
const r4 = cities.map(v => v[0]).join('=')
```

r1=	r2=	r3=	r4=

Aufgabe 5

(15 Punkte)

In dieser Aufgabe geht es um vereinfachte Emmetstrings. Diese bestehen aus Namen von HTML-Elementen, die durch > voneinander getrennt sind. Das Zeichen > bedeutet: *„hat als*

Kindelement“. Hinter dem Namen eines HTML-Elementes kann in geschweiften Klammern Textinhalt für dieses Element stehen.

Beispiel: *Der Emmetstring `div{hallo}>ul>li{welt}` steht für ein `div`-Element mit dem Textinhalt `hallo`, das als Kind ein `ul`-Element hat. Dieses wiederum hat als Kind ein `li`-Element mit dem Textinhalt `welt`.*

Schreiben Sie eine Funktion `createEmmet`. Die Funktion erhält als erstes Argument ein Objekt `appendTo` mit der Schnittstelle `HTMLElement` und als zweites Argument einen Emmetstring. Die Funktion legt den durch den Emmetstring spezifizierten Unterbaum an und hängt ihn hinter das letzte Kindelement des Elementes `appendTo`.

Ergänzen Sie den JavaScript-Code, um die geforderte Funktionalität zu erreichen. Verwenden Sie weder `innerHTML` noch `insertAdjacentHTML`.

```
function createEmmet(appendTo, emmet) {









}
```

```
// zur Erinnerung, wenn Sie match verwenden wollen:
const paragraph = 'The quick brown fox jumps over the lazy dog.'
const capturingRegex = /(?<animal>fox|cat) jumps over/
const found = paragraph.match(capturingRegex)
console.log(found.groups.animal) // 'fox'
```

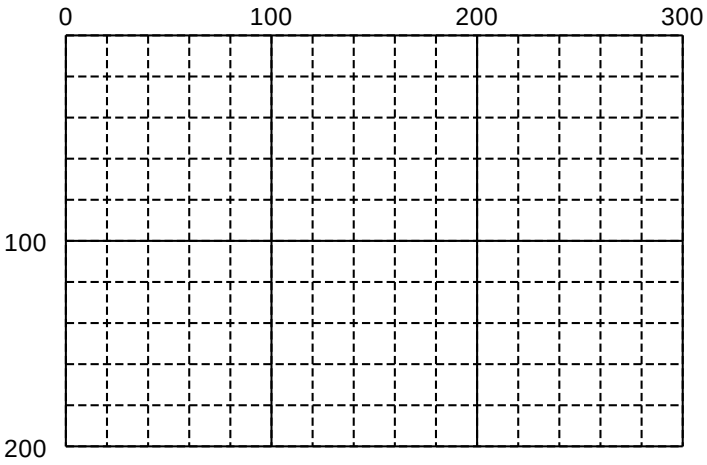
Aufgabe 6

(10 Punkte) Skizzieren Sie die folgenden zwei SVG-Graphiken auf den dafür vorgesehenen Gittern. Folgende Eigenschaften sind per CSS festgelegt:

```
circle,line,rect,path, polygon {
  stroke: black;stroke-width: 2;fill: transparent;
}
```

a)

```
<svg width="300" height="200" version="1.1">
  <circle cx="100" cy="100" r="50" />
  <circle cx="200" cy="100" r="50" />
  <polygon points="200,100 300,0 200,20" />
</svg>
```



b)

```
<svg width="300" height="200" version="1.1">
  <path d="M 300,200 L 220,180" />
  <g transform="translate(-20,60)">
    <rect width="200" height="100" x="40" y="20" />
    <g transform="scale(0.5)">
      <rect width="200" height="80" x="160" y="120" />
    </g>
  </g>
</svg>
```