Hausaufgabe 2: JavaScript und CSS-Selektoren

Laden Sie ihre Lösung für die Hausaufgabe in einer zip-Datei auf ISIS bis zum 5. Dezember um 23:59 Uhr hoch. Die zip-Datei soll dabei ausschließlich den Ordner `ha2` mit dessen Inhalt wie in der Vorgabe gegeben enthalten (Der Inhalt der Dateien muss natürlich nach Aufgabenstellung bearbeitet werden). Keine weiteren Dateien sollen hinzugefügt oder entfernt werden. Auch soll keine Datei oder Ordner umbenannt werden.

2.1 JavaScript (12 Punkte)

In der Vorlesung haben Sie schon die Grundlagen der Programmierung in JavaScript kennengelernt. Darauf wollen wir nun aufbauen und beschäftigen uns nun mit Funktionen. Für die nachfolgenden Aufgaben bearbeiten Sie lediglich die in der Aufgabe erwähnten Dateien. Die Dateien `index.html` und `data.js` dürfen nicht verändert werden.

Hinweis: Wenn sie die beigefügte index.html in ihrem Browser öffnen, so wird diese Webseite ihre Lösungen für Sie kontrollieren. Sie werden auf dieser Seite einen erwarteten Output sehen welcher bei richtiger Implementierung erwartet wird, und den Output, den Ihre Lösung produziert hat.

2.1.1 Multiplikation

Betrachten Sie die Datei `a01.js`

- Implementieren Sie die Funktion `multiply()`, die zwei Zahlen als Parameter (`number1` und `number2`) übergeben bekommt und diese miteinander multipliziert. Dabei ist der zweite Parameter optional. Wird die Funktion nur mit einem Parameter aufgerufen wird `number1` einfach mit 1 multipliziert. (1 Punkt)
- Implementieren Sie die Funktion `multiplyAll()`, die eine beliebige Anzahl an Parametern übergeben bekommt und diese miteinander multipliziert. (1 Punkt)

  Hinweis: Machen Sie sich zum Lösen dieser Aufgabe mit dem arguments Objekt vertraut. Des Weiteren ist die statische Funktion `Object.values()` hilfreich für die Lösung dieser Aufgabe.

2.1.2 Array

Betrachten Sie die Datei `a02.js`

- Implementieren Sie die Funktion `addArrayElement()`, die ein Element als Parameter übergeben bekommt und dieses zum Array hinzufügt, sofern dieses nicht schon im Array vorhanden ist. (1 Punkt)

  Hinweis: Arbeiten Sie nur auf der Kopie des Arrays.

- Implementieren Sie die Funktion `getArrayElements()`, die N Elemente des Arrays beginnend vom Startindex i ausliest. Dazu bekommt diese zwei Parameter (`number` und `startIndex`) übergeben. Der Parameter `startIndex` kann größer als die Array-Länge sein. Implementieren Sie eine fortlaufende Adressierung, indem Sie das Array wieder vom Beginn durchlaufen.

  Beispiel: Sei L die Array-Länge. Das Element mit dem Index 0 hat auch den Index L und 2 * L und 3 * L ... Der Parameter `number` gibt an wieviele Elemente ausgelesen werden sollen. Das Auslesen wird am Ende des Arrays beendet auch wenn nicht die gewünschte Anzahl an Elementen ausgelesen wurde. (1 Punkt)

- Implementieren Sie die Funktion `deleteArrayElements()`, die N Elemente des Arrays beginnend vom Startindex i ausliest **und** innerhalb dieses Teil-Arrays jedes x-te Element löscht. Implementieren Sie die Parameter `number` und `startIndex` analog zur Funkton `getArrayElements()`. Der Parameter `everyIth` kann auch größer sein als die Array-Länge. Es wird dennoch immer mindestens ein Element gelöscht, nämlich das 0-te Element des Teil-Arrays. Geben Sie sowohl das Array ohne die gelöschten Elemente, als auch die gelöschten Elemente zurück (siehe Vorlage Rückgabewert `newResult` und `removedItems`) (3 Punkte)

2.1.3 Objekte

Betrachten Sie die Datei `a03.js`

- Implementieren Sie die Funktion `addObjectElement()`, die zwei Parameter (`key` und `value`) übergeben bekommt und dieses Wertepaar dem Objekt hinzufügt. Sollte bereits ein Wertepaar mit dem selben Schlüssel (`key`) existieren, wird der Schlüssel umbenannt. Zum Beispiel: Das Wertepaar $\{a : 1\}$ soll dem Objekt hinzugefügt werden, jedoch existiert bereits ein Wertepaar $\{a : 0\}$. In diesem Fall wird $\{a\_1 : 1\}$ dem Objekt hinzufügt. Sollte der Schlüssel a_1 ebenfalls bereits existieren, dann wird der Schlüssel in a_2 umbenannt usw. (3 Punkte)

- Implementieren Sie die Funktion `getObjectElements()`, die ein Array von Schlüsseln übergeben bekommt und für jeden Schlüssel den dazugehörigen Wert zurückgibt. Die entsprechenden Werte sollen in einem Array in analoger Reihenfolge zu den Schlüsseln zurückgegeben werden. Sollte ein Schlüssel nicht existieren wird die Zeichenkette “not found” an der Stelle zurückgegeben. (1 Punkt)

- Implementieren Sie analog die Funktion `deleteObjectElements()`, die ein Array von Schlüsseln übergeben bekommt und für jeden Schlüssel den dazugehörigen Wert löscht. Das resultierende Objekt soll als Rückgabewert zurückgegeben werden. Schlüssel, die nicht existieren werden ignoriert. (1 Punkt)

2.2 CSS (8 Punkte)

In den folgenden Aufgaben sollen Sie das Styling der Bundesliga-Tabelle in der Datei `table.html` vornehmen. Betrachten Sie die Datei und überlegen Sie, welche Regeln Sie in der Datei `style.css` erstellen müssen, um die nachfolgenden Teilaufgaben zu lösen.

Wichtig: Sie dürfen lediglich neue CSS-Regeln erstellen, die HTML-Datei `table.html` darf nicht geändert werden. Ändern Sie auch nicht den Ordner `images` und dessen Inhalt.

2.2.1 Ausrichtung

Bearbeiten Sie ausschließlich die Datei `style.css`

Momentan sind Club-Logos und Vereinsname nicht zentriert zueinander Ausgerichtet auf der vertikalen Ebene. Sorgen Sie dafür, dass der Vereinsname zentriert neben dem Club Logo steht. Des Weiteren sollen die Daten des erstplatzierten Clubs alle **fett** dargestellt werden.(1 Punkt)

2.2.2 Selektoren

Bearbeiten Sie ausschließlich die Datei `style.css`

Verwenden Sie geeignete Selektoren um die folgenden Unteraufgaben zu lösen. **Hinweis:** Alle Änderungen sollen immer die gesamte Zeile betreffen und die angegebenen Farben sind CSS Standard-Farben die mit ihrem Namen direkt verwendet werden können.

1. Die ersten vier Plätze in der Tabelle sollen farblich hervorgehoben werden da diese einen Platz in der Champions-League erhalten. Verwenden Sie `mediumblue` als Hintergrundfarbe (1 Punkt)
2. Der Hintergrund des fünften Platzes soll als Relegationsplatz für die Champions-League mit der Farbe `orange` dargestellt werden. (1 Punkt)
3. Als Teilnehmer der Euro-League soll der sechste Platz die Farbe `green` erhalten. (1 Punkt)
4. Die letzten beiden Plätze sollen mit der Farbe `red` hinterlegt werden. Achten Sie darauf, dass die Bundesliga eventuell die Teilnehmerzahl erhöhen wird und mehr als 18 Teams in Zukunft in der Tabelle sein können. Sorgen Sie dafür, dass Ihre Lösung auch dann noch richtig ist, falls die Tabelle eine andere Größe hätte. (1 Punkt)

5. Der drittletzte Platz soll wieder ein Relegationsplatz aber diesmal für den Abstieg sein und soll wieder `orange` hinterlegt werden. Auch hier gilt, dass Ihre Lösung unabhängig von der Größe der Tabelle sein soll. (1 Punkt)

6. Um die Lesbarkeit zu erhöhen, sollen die jeweiligen Zeilen alternierend etwas heller dargestellt werden als jetzt. Hier gilt, dass `mediumblue` dann `lightblue` sein soll, `red` wird zu `lightcoral`, `orange` zu `lightsalmon` und `grey` wird zu `lightgray`. Der erstplatzierte soll heller dargestellt werden, der zweitplatzierte bleibt wie er ist und so weiter. **Wichtig:** Auch hier gilt, dass die Bundesliga eine beliebige Größe in der Zukunft annehmen könnte, jedoch immer eine gerade Anzahl an Teams. Sorgen Sie dafür, dass Ihre Lösung immer gilt. (2 Punkte)