

Web Engineering Multiple Choice Questions Midterm

Test 1 – SS 2023

What are promises in JavaScript?

- All objects have a promise, from which they inherit properties and methods. Existing promises can be extended at any time ("monkey patching").
- Promises are anonymous functions which allow the special "Fat Arrow" syntax (=>).
- Similar to cookies, promises allow storage of data in the user's browser. However, promises do not have an expiration date.
- **Promises are proxy objects for values that are not yet known. They are used as wrappers around asynchronous computations.**

In the following code, how would you declare the variables message, start, and millis? For each of the three variables, choose the most restrictive scope possible.

```
function time(f) {  
    let message = ``;  
    const start = Date.now();  
    f();  
    const millis = Date.now() - start;  
    if(millis > 6000) {  
        message = `Wow, this function took ${millis/1000} seconds to run!`;  
    } else {  
        message = `The function ran for ${millis/1000} seconds.`;  
    }  
    return message;  
}
```

You are given the vague description to design a URL for a secure http resource available on port 6000: it should serve a path to users, query a username, and target the profile fragment within the resource.

Which of the following URLs is the most compatible with this description?

- <http://example.com:6000/users/username/jc#profile>
- <https://example.com/users?username=jc&port=6000#profile>
- **<https://example.com:6000/users?username=jc#profile>**
- <https://example.com/users?username=jc&port=6000&fragment=profile>

What is wrong with the following HTML structure?

```
<section class="important label">  
    <strong style="border-bottom: 1px solid black;"><h2>Important  
    Announcement</h2></strong>  
    <section>  
        <em>Beware: </em> of all the wrong things in this HTML.  
    </section>  
</section>
```

- An HTML element cannot have multiple classes
- ** must not include <h2> because inline elements are not allowed to contain block elements**
- cannot have a border because inline elements are not allowed to define borders
- <section> cannot be included within another <section>

Match up the ways to include CSS styling in an HTML document

<style> p { color: red; } </style> **embed CSS into HTML document**
<p style="color: red;">stylish</p> **apply inline styling**
<link href="style.css" rel="stylesheet"> **include external stylesheet**

You are given the following HTML and JavaScript code:

```

<label for="grade">Points</label>
<input type="text" id="points">
<div>
    Grade <span id="grade"></span>
</div>

var el = document.getElementById('points')

// Beware that event.toElement.value returns a string
el.addEventListener('change', event =>
grade_dom_insert(points_to_grade(document.getElementById('points').value)));

function points_to_grade(raw_points) {
    points = parseInt(raw_points);
    grades = {
        'Sehr gut' : 90, 'Gut' : 80, 'Befriedigend' : 70,
        'Genügend' : 51, 'Nicht genügend' : 50
    };
    for (let grade in grades) {
        if (grades[grade] < points) {
            return grade;
        }
    }
    return "Not in range";
}
function grade_dom_insert(grade) {
    const gradeEl = document.getElementById('grade')
    gradeEl.innerText = grade;
    grade2color = {
        'Sehr gut' : 'blue', 'Gut' : 'green',
        'Befriedigend' : 'brown', 'Genügend' : 'black'
        'Nicht genügend' : 'red'
    };
    gradeEl.style.color = grade2color[grade];
}

```

What is the browser output if you enter 87 into the input field?

- „Sehr gut“ with a blue font color
- „Not in range“
- Error in JavaScript (no output)
- „**Gut**“ with a green font color

Given this markup:

```

<div style="border: 4px solid black;">
    <div id="outer">
        <span id="inner" style="font-size: 2em;">Some text</span>
    </div>
</div>

```

Which CSS declaration is most likely to result in the following rendering in the browser?

Some text

- `#outer { padding: 5em; margin: 2em; border: 4px solid red; }`
- `#outer { margin: 2em; } #inner { border: 4px solid red; padding: 5em; }`
- **`#outer { padding: 2em; margin: 5em; border: 4px solid red; }`**
- `#outer { padding: 2em; } #inner { margin: 5em; border: 4px solid red; }`

You are reviewing code for an HTML form. While the markup looks the way it should in the browser, you can see how the following code could create accessibility issues:

```
<form action="/login">
    
    <b>Login</b><br>
    <table>
        <tr>
            <td>
                Username<br>
                Password
            </td>
            <td>
                <input name="field_1"><br>
                <input name="field_2"><br>
                <input type="submit" value="Login">
            </td>
        </tr>
    </table>
</form>
```

Which of the following actions would improve the accessibility of this HTML form and which would result in no change to accessibility?

- **Improved Accessibility** Replacing `Login` with `<h1>Login</h1>`
- **No change** Submitting the form with JavaScript, instead of using the `action` attribute
- **Improved Accessibility** Adding labels for inputs
- **Improved Accessibility** Adding alternative text to the image

You want to send an idempotent HTTP request with a message body. Which method can you use?

- GET
- PATCH
- PUT
- POST

Which of the following CSS selectors would you use to target the HTML element (and only that element) highlighted below?

```
<article id="first" class="content">
    <h1>Style</h1>
    <section class="wrapper" id="wrapper">
        <ul class="itemize">
            <li>First</li>
            <li>Second</li>
        </ul>
    <div>
        <ul class="itemize">
            <li>First</li>
            <li>Second</li>
        </ul>
    </div>
    </section>
    <section>
        <p class="content">Another Style</p>
        <ul class="itemize">
            <li>First</li>
            <li>Second</li>
        </ul>
        <div>
            <ul class="itemize">
                <li>First</li>
                <li>Second</li>
            </ul>
        </div>
    </section>
</article>
```

- article#first.content > section#wrapper ul.itemize
- **article#first.content > h1 + section.wrapper > ul.itemize**
- article#first.content > h1 > section#wrapper + ul#itemize

Given this CSS code:

```
h1 {
    --alarm-color: darkred;
}

h1.alarm {
    font-size: 4em;
    color: var(--alarm-color, red);
}

@media only screen and (max-width: 500px) {
    h1.alarm {
        font-size: 1em;
    }
}
```

The element `<h1 class="alarm">...</h1>` has the following styling on a device with 300px width:

`font-size: 1em; color: darkred;`

You send an HTTP POST request with a JSON payload to a web server and receive the following response:

```
HTTP/1.1 200 OK
Date: Mon, 6 Jul 2020 13:00:23 GMT
Server: nginx
Content-Type: application/json;
Content-Length: 33

{"sessionId": "km85jhhgS7jf223y"}
```

What can you tell from the headers of the response?

- False** The server is running nginx with an application that only returns JSON for all endpoints.
- True** The response was sent at 13:00:23 GMT.
- False** The server wants me to store a cookie with a session identifier.
- True** My request was successful and the response contains a body with 33 bytes of JSON.