ChatGBT_Active Recall Questions

What is covered:

HTTP

HTML

CSS

JavaScript

JSON

WGET

Beautiful Soup

HTTP:

1. **What is a simple Protocol?**
    - A simple protocol is the Client-Server Protocol, where the client sends a request and receives a response back. For example, a client could be a computer communicating with an external hard drive and asking for memory space in block 100.
2. **Describe the server code outlined in the conceptual outline.**
    - The server code is structured as a continuous while loop that listens for requests from the client. Upon receiving a request, it serves the request by

processing it (e.g., reading files from the file system) and then sends back a response.

3. **Explain the significance of the HTTP method `GET`.**
    - The `GET` method is used to retrieve a resource (such as a webpage) from the server. It allows the client to request a specific resource identified by a URL path. This method is widely used for fetching web pages and other resources.
4. **What is the purpose of the `Connection: Close` header in an HTTP request?**
    - The `Connection: Close` header in an HTTP request indicates that the client wants the connection to be closed immediately after the request has been completed. This informs the server to close the connection after sending the response back to the client.
5. **Explain the main functionality of TCP in data transmission.**
    - TCP (Transmission Control Protocol) is responsible for ensuring reliable and ordered data transmission between computers. Its main functionalities include breaking down messages into smaller packets, sending each part, ensuring all parts arrive at the destination, arranging the message correctly, and managing the flow of data.
6. **What is the purpose of port forwarding?**
    - Port forwarding is a network configuration method used to allow external devices to access services on a private network from the outside. It facilitates connections to devices on a local network over the internet by directing incoming data to specific applications or services based on port numbers.
7. **Define the term URI and explain its significance.**
    - URI stands for Uniform Resource Identifier, which is a standard format for identifying resources on the web. It can refer to any resource, whether accessible over the internet or not. URIs are significant because they provide a uniform system for naming and locating resources, allowing for consistent access across different systems and platforms.
8. **What are the principles of REST, and why are they important?**

- REST (Representational State Transfer) is an architectural philosophy for designing networked applications. Its principles include using URIs to identify resources, interacting with resources via standard HTTP methods, and maintaining statelessness. These principles are important because they promote scalability, reliability, and interoperability in web applications.

9. **Explain the purpose of URL fragments.**
   - URL fragments are optional parts at the end of a URL that start with `#`. They allow users to refer to specific parts of a document they have accessed. URL fragments are commonly used for linking directly to specific sections of web pages, improving the user experience and facilitating navigation.

HTML

1. **Links**:
   - **Question**: What HTML element is used to create hyperlinks on a web page?
     - **Answer**: The `<a>` (anchor) element is used to create hyperlinks in HTML.
   - **Question**: Explain the difference between absolute and relative paths in the `href` attribute of an anchor tag.
     - **Answer**: Absolute paths specify the full URL, including the protocol (e.g., `http` or `https`) and domain, while relative paths refer to a resource on the same domain and are relative to the current document's location.
   - **Question**: How can you open a link in a new tab using HTML?
     - **Answer**: You can open a link in a new tab by adding the `target="_blank"` attribute to the anchor tag.

2. **Forms**:
    - **Question**: What are the two key attributes of a form element in HTML, and what do they specify?
        - **Answer**: The two key attributes of a form element are `method` and `action`. The `method` attribute specifies the type of HTTP request to be sent (e.g., `GET` or `POST`), and the `action` attribute specifies the URL the form interacts with.
    - **Question**: How do you create a mandatory form field in HTML?
        - **Answer**: You can make a form field mandatory by adding the `required` attribute to the input element.
    - **Question**: What is form validation, and how can it be implemented in HTML?
        - **Answer**: Form validation ensures that user input meets certain criteria before the form is submitted. It can be implemented in HTML using attributes like `required` for mandatory fields or `type` for specific input formats.
3. **Tables**:
    - **Question**: What HTML elements are used to define the structure of a table?
        - **Answer**: The `<table>`, `<thead>`, `<tbody>`, `<tr>`, `<th>`, and `<td>` elements are used to define the structure of a table.
    - **Question**: How can you add borders to a table in HTML?
        - **Answer**: You can add borders to a table by using the `border` attribute in the `<table>` tag or by applying CSS styles to the table elements.
    - **Question**: What is the purpose of the `<th>` element in an HTML table?
        - **Answer**: The `<th>` element is used to define header cells in an HTML table, which typically contain column headings or row headings.

CSS

1. **What is CSS, and what is its primary purpose?**

Answer: CSS, which stands for Cascading Style Sheets, is a styling language used to modify the appearance of HTML elements. Its primary purpose is to alter aspects such as fonts, colors, sizes, and spacing of content within HTML documents. Additionally, CSS allows for the inclusion of animations to enhance the visual experience.

2.

Explain the syntax of a CSS rule.

Answer: A CSS rule consists of a selector and a declaration block. The selector targets the HTML element(s) to be styled, while the declaration block, enclosed within curly braces, contains one or more property-value pairs that specify the styling rules. For example:

```
cssCopy codeselector {
    property1: value1;
    property2: value2;
    /* Additional properties */
}
```

3.

How can you apply multiple CSS styles to different HTML elements simultaneously?

Answer: You can apply multiple CSS styles to different HTML elements simultaneously by using a comma-separated list of selectors. For example:

```
cssCopy codeselector1, selector2 {
    /* CSS properties for both selector1 and selector2 */
}
```

4.
Describe three different methods for specifying the positioning of elements within a CSS grid.

Answer: Elements within a CSS grid can be positioned using individual property declarations, combined row/column start and end values, or a shorthand declaration for specifying both row and column positions. Examples include:

- Individual declarations:

```
cssCopy code.container {
    grid-row-start: 1;
    grid-row-end: 3;
    grid-column-start: 2;
    grid-column-end: 3;
}
```

- Combined start/end values:

```
cssCopy code.container {
    grid-row: 1 / 3;
    grid-column: 2 / 3;
}
```

- Shorthand declaration:

```
cssCopy code.container {
    grid-area: 1 / 2 / 3 / 3;
}
```

5.
What is the purpose of the `fr` unit in CSS Grid Layout?

Answer: The `fr` unit, short for "fraction," is used in CSS Grid Layout to distribute available space dynamically within a grid container. It allows designers to specify the fraction of available space that a grid track (column or row) should occupy. This unit is particularly useful for creating responsive layouts where the size of grid tracks adapts based on available space.

CSS styling

Question 1: What analogy does the note draw between web design and physical craftsmanship?

Answer 1: The note compares web design to crafting a table, emphasizing the importance of balancing aesthetics with functionality. Just like a well-designed table must look appealing while also serving its purpose effectively, webpages must be visually pleasing while providing standard functionality.

Question 2: How does the note describe the relationship between color and design in different cultures?

Answer 2: The note suggests that the use of color in design often relies on cultural associations. However, it emphasizes that these associations can vary across different parts of the world, highlighting the importance of cultural sensitivity in web design.

Question 3: What non-functional changes are illustrated in the note regarding the OK button?

Answer 3: The note showcases various designs of the OK button, indicating that these changes are non-functional, meaning they do not affect the button's functionality. Despite being non-functional, some designs appear more modern than others, suggesting that outdated styles may impact users' perceptions of a webpage's age or aesthetic appeal.

Question 4: According to the note, why is testing with the intended audience important in web design?

Answer 4: Testing with the intended audience serves not only to ensure that users can use the service correctly but also to gather first impressions about design choices. Users may not always interpret design intentions as intended, leading to potential misunderstandings or confusion.

Question 5: What is the significance of responsive design in web development, according to the note?

Answer 5: Responsive design is highlighted as particularly important in web design because it allows webpages to adapt to the capabilities of different devices. With users accessing websites on various devices such as phones and laptops, responsive design ensures optimal viewing experiences by resizing and adjusting layouts accordingly.

Question 6: Describe the concept of whitespace as discussed in the note.

Answer 6: The note explains that whitespace can effectively group items or ideas on a page. It illustrates this with examples of tabular formats, demonstrating how appropriate whitespace, without the need for line separation, enhances readability and perceived quality. Additionally, it suggests that appropriate padding between cell contents and border lines contributes to a higher-quality presentation.

Question 7: What is the 960 grid system, and how does it impact web design?

Answer 7: The 960 grid system is a design framework based on commonly used dimensions in web design. It involves dividing a webpage into 12 columns, each 60 pixels wide, with margins between columns. This system provides a conceptual reference for placing elements, resulting in a layout that optimizes space usage, maintains consistent spacing between elements, and avoids a blocky appearance.

Question 8: How can CSS be linked to HTML, according to the note?

Answer 8: The note explains three ways to link CSS to HTML: inline, internal, and external. It emphasizes the use of an external stylesheet as the most common and efficient method, achieved by using the `<link>` tag within the `<head>` section of the HTML document. This approach keeps the CSS separate from HTML, making code management and maintenance easier for larger projects.

JavaScript Basics:

Question 1: How can you create a JavaScript program that displays "Hello World"?

Answer 1: To create a JavaScript program that displays "Hello World", you can define a function called `doAnything` within a `<script>` tag in an HTML file. Inside this function, use the `alert()` function to display the message "Hello World".

```
htmlCopy code
<script>
    function doAnything() {
        alert('Hello World');
    }
    doAnything();
</script>
```

Question 2: What is the purpose of the execution context in JavaScript?

Answer 2: The execution context in JavaScript serves as a container that holds all the necessary information for running code, including variables and the code itself. There are both global and local execution contexts, with the global execution context being the first context created when a JavaScript program starts.

Question 3: Explain hoisting in JavaScript.

Answer 3: Hoisting in JavaScript refers to the behavior where declarations of variables and functions are moved to the top of their containing scope during compilation. While variable declarations are hoisted, variable initializations remain at their original location. Functions are also hoisted to the top of their containing scope, allowing them to be accessed before they are declared in the code.

Question 4: What is the scope chain in JavaScript?

Answer 4: The scope chain in JavaScript refers to the process by which the JavaScript engine looks up variables in the current scope and then in outer scopes if the variable is not found. This chain continues until the variable is either found or until the global scope is reached. This mechanism allows JavaScript to determine the visibility of variables and functions within different parts of the code.

Question 5: What is the purpose of the call stack in JavaScript?

Answer 5: The call stack in JavaScript is a Last-In-First-Out (LIFO) data structure that keeps track of function calls during code execution. It helps manage the flow of execution by storing information about the active functions and their execution contexts. When a function is called, its execution context is pushed onto the call stack, and when the function returns, its context is popped off the stack.

Question 6: Describe the syntax for creating arrays in JavaScript.

Answer 6: In JavaScript, arrays can be created by enclosing comma-separated values within square brackets. For example:

```
javascriptCopy code
var exampleArray = [1, "tom", 6.666];
```

Arrays can hold mixed data types, and individual elements can be accessed using index notation, starting from 0.

Question 7: What are anonymous functions in JavaScript?

Answer 7: Anonymous functions in JavaScript are functions that do not have a name. They are defined on the fly and can be assigned to a variable or used directly within an expression. Anonymous functions are commonly used as arguments to other functions or for callback functions.

Question 8: Explain the concept of arrow functions in JavaScript.

Answer 8: Arrow functions, introduced in ES6, provide a more concise syntax for writing anonymous functions in JavaScript. They offer a shorter syntax and implicit return for single-line expressions. Arrow functions are defined using the `=>` syntax and can be used to define functions with or without parameters.

Question 9: What role do events play in JavaScript?

Answer 9: In JavaScript, events represent actions or occurrences that happen in the browser, such as clicking a button, hovering over an element, or entering text into a form field. HTML pages can capture these events and pass them to event handlers in JavaScript, allowing developers to execute code in response to user interactions. Event handling is a fundamental aspect of building interactive web applications.

JSON

Question 1: What is JSON, and how does it relate to JavaScript?

Answer 1: JSON, or JavaScript Object Notation, is a text-based data format that follows JavaScript object syntax. It exists as a string and is commonly used to transmit data across a network. JSON files can store various data types, including strings, integers, booleans, arrays, and null values. While JSON resembles JavaScript objects, it is purely a string with a specified data format.

Question 2: How can you convert JSON to native JavaScript objects?

Answer 2: JavaScript provides a global JSON object with methods available to convert between JSON and JavaScript objects. One such method is `JSON.parse()`, which accepts a JSON string as a parameter and returns a corresponding JavaScript object. For example:

```
javascriptCopy code
const jsonString = '{"name": "John", "age": 30}';
const jsObject = JSON.parse(jsonString);
```

Question 3: Explain the significance of the `stringify()` method in JavaScript.

Answer 3: The `stringify()` method in JavaScript is used to convert a JavaScript object into a JSON string. This is particularly useful when you need to send JavaScript objects across a network, as JSON is a text-based format that is easy to transmit. For example:

```
javascriptCopy code
const jsObject = { "name": "John", "age": 30 };
const jsonString = JSON.stringify(jsObject);
```

Question 4: How do you access data within a JSON file using JavaScript?

Answer 4: To access data within a JSON file using JavaScript, you can parse the JSON string into a JavaScript object using the `JSON.parse()` method. Once the JSON string is converted into a JavaScript object, you can use dot notation or bracket notation to access specific properties and values within the object. For example:

```
javascriptCopy code
const jsonString = '{"name": "John", "age": 30}';
const jsObject = JSON.parse(jsonString);

console.log(jsObject.name); // Output: John
console.log(jsObject["age"]); // Output: 30
```

Question 5: How can you access hierarchical data within a JSON file using JavaScript?

Answer 5: To access hierarchical data within a JSON file using JavaScript, you need to chain the required properties together in an array. For example, to access the second superpower of the second member of the superhero squad from the provided JSON:

```
javascriptCopy code
const superHeroes = {
  "squadName": "Justice League",
  "homeTown": "Metro City",
  "formed": 2016,
  "active": true,
  "members": [
    { "name": "Batman", "power": null },
    { "name": "Superman", "power": ["SuperSpeed", "Super Stre
ngth"] }
  ]
};

console.log(superHeroes["members"][1]["power"][1]); // Outpu
t: Super Strength
```

Question 6: What precautions should be taken when working with JSON?

Answer 6: When working with JSON, it's essential to ensure that the JSON string is well-formed, as even a single misplaced comma or colon can cause the JSON file to become invalid. JSON requires double quotes to be used around strings and property names, and single quotes are not valid except when surrounding the entire JSON string. Validating JSON using tools like JSONLint can help identify and fix any syntax errors.

Question 7: How can you obtain JSON data using JavaScript?

Answer 7: To obtain JSON data using JavaScript, you can use an API called Fetch. This API allows you to make network requests to retrieve resources from a server, such as JSON files, via JavaScript. You can then use the `fetch()` function to make the network request and the `json()` method to retrieve the response as JSON. For example:

```
javascriptCopy code
async function fetchData() {
  const response = await fetch('https://example.com/data.jso
n');
  const jsonData = await response.json();
  console.log(jsonData);
}
```

Asynchronous JavaScript:

Question 1: What is asynchronous programming, and when is it typically used?

Answer 1: Asynchronous programming is a technique used when you have a long-running task but still want to handle events while the task is running. It's commonly employed in scenarios such as making HTTP requests using `fetch()`, accessing a user's camera or microphone using `getUserMedia()`, or asking a user to select files using `showOpenFilePicker()`.

Question 2: How does synchronous programming differ from asynchronous programming?

Answer 2: In synchronous programming, code is run and executed line by line on a single thread. This means that the browser waits for each line to finish executing before moving on to the next one. In contrast, asynchronous programming allows the program to continue executing other tasks while waiting for certain operations to complete, thus enabling more responsive user interfaces and better resource utilization.

Question 3: Explain the concept of event handlers in asynchronous programming.

Answer 3: Event handlers are a form of asynchronous programming where you provide a function (the event handler) that will be called not right away, but whenever the specified event happens. In the context of asynchronous operations, event handlers are used to handle the completion of asynchronous tasks, such as HTTP requests or user interactions.

Question 4: What is the XMLHttpRequest API, and how does it relate to asynchronous programming?

Answer 4: The XMLHttpRequest API enables JavaScript to make HTTP requests to a remote server asynchronously. This means that the browser can continue executing other tasks while waiting for the request to complete. XMLHttpRequest objects allow you to attach event listeners that get called when the request finishes, enabling asynchronous handling of network operations.

Question 5: What are callbacks, and how are they used in asynchronous programming?

Answer 5: Callbacks are functions that are passed into another function with the expectation that the callback will be called at the appropriate time. In asynchronous programming, callbacks are commonly used to handle the completion of asynchronous tasks. However, nesting callbacks can lead to complex and hard-to-read code, a situation often referred to as "callback hell" or the "pyramid of doom."

Question 6: What are Promises, and what advantages do they offer over callbacks?

Answer 6: Promises are objects that represent the eventual completion or failure of an asynchronous operation and its resulting value. They provide a more manageable approach to handling asynchronous tasks compared to callbacks. Promises offer advantages such as easier error handling, chaining of asynchronous operations, and improved code readability.

Question 7: How does the `await` keyword simplify asynchronous programming?

Answer 7: The `await` keyword is used inside async functions to pause execution until a Promise is settled (either fulfilled or rejected). This allows you to write asynchronous code that looks and behaves more like synchronous code, making it easier to understand and maintain. The `await` keyword eliminates the need for nested callbacks and simplifies error handling, leading to cleaner and more concise code.

Wget:

Question 1: What is the purpose of web crawlers, and why do robots visit websites?

Answer 1: Web crawlers, also known as robots, spiders, or web crawlers, visit websites for various reasons, including indexing websites for search or archiving purposes, carrying out specific tasks for their owners or programmers, probing sites for security vulnerabilities, pretending to be humans for nefarious purposes, and generating false traffic for scamming advertising networks.

Question 2: How can you use `wget` to download files from the command line?

Answer 2: You can use `wget` to download files from the command line by providing the URL of the file you want to download, like this:

```
bashCopy code
wget <url>
```

Question 3: What happens when you download a webpage using `wget`, and why might it not appear exactly as it does in your browser?

Answer 3: When you download a webpage using `wget`, it won't appear exactly like it does in your browser because the resources referenced on the webpage, such as style sheets or images, may not exist locally on your filesystem. This can lead to differences in appearance.

Question 4: How can you use `wget` to download all requisites to accurately render a page?

Answer 4: You can use `wget` with the `-p` flag to download all the requisites to accurately render a page. For example:

```
bashCopy code
wget -p <url>
```

Question 5: What is the purpose of the `robots.txt` file, and how does `wget` interact with it?

Answer 5: The `robots.txt` file is a standard used by websites to communicate with web crawlers about which parts of the site should be crawled or not. When using `wget` to download more than a single file from a site, it will first check that domain's `robots.txt` file to respect the crawling preferences set by the website owner.

Question 6: How can you download multiple webpages at once from a domain using `wget`?

Answer 6: You can use the `-r` option to perform recursive downloading and the `-l N` option to specify the level of recursion. For example:

```
bashCopy code
wget -r -l N <url>
```

Question 7: What is the purpose of the `-m` flag in `wget`, and how does it work?

Answer 7: The `-m` flag in `wget` is used for mirroring, which means downloading the entire website. It follows links recursively until it runs out of links to follow, effectively creating a local copy of the entire website.

Question 8: What are some ethical concerns related to web crawling, and how can they be addressed?

Answer 8: Ethical concerns related to web crawling include respecting the limits set in the `robots.txt` file, assessing whether web crawling is the best way to obtain website content, considering copyright and republishing issues, avoiding crawling with security risks, and imposing crawl delays to avoid overwhelming servers and blocking human clients. It's important to implement behavior that respects the guidelines and permissions set by website owners.

Beautiful Soup

Question 1: What is BeautifulSoup, and what does it allow you to do?

Answer 1: BeautifulSoup is a Python library used for web scraping. It allows you to load an HTML document as a Python object and interact with it using various methods.

Question 2: How can you install BeautifulSoup on Alpine Linux using pip?

Answer 2: To install BeautifulSoup on Alpine Linux, you need to first install Python and pip using the following commands:

```
bashCopy code
sudo apk add python3 py3-pip
```

Then, use pip to install the BeautifulSoup library:

```
bashCopy code
pip install bs4
```

Question 3: How can you load a page into BeautifulSoup?

Answer 3: To load a page into BeautifulSoup, you can use the `open` function to create a file pointer and then pass it to BeautifulSoup. For example:

```
pythonCopy code
file = "cattax/index.html"
soup = BeautifulSoup(open(file, 'r'))
```

Question 4: What method can be used to print all visible text on a webpage using BeautifulSoup?

Answer 4: You can use the `get_text()` method to print all visible text on a webpage. For example:

```
pythonCopy code
text = soup.get_text()
print(text)
```

Question 5: How can you navigate page elements in BeautifulSoup?

Answer 5: You can navigate page elements in BeautifulSoup by accessing tags directly or by navigating the element hierarchy using methods like `findChildren()`. For example:

```
pythonCopy code
soup.head.findChildren()
```

Question 6: What is the purpose of the `find` method in BeautifulSoup?

Answer 6: The `find` method in BeautifulSoup is used to extract a specific element from a webpage without having to navigate to find the element manually. For example:

```
pythonCopy code
soup.find('strong')
```

Question 7: How can you use BeautifulSoup to find all elements with a specific tag?

Answer 7: You can use the `find_all` method to find all elements with a specific tag. For example:

```
pythonCopy code
soup.find_all('strong')
```

Question 8: How can you access the value of attributes in BeautifulSoup?

Answer 8: You can access the value of attributes in BeautifulSoup by using the tag's attribute as a key. For example:

```
pythonCopy code
soup.head.meta['charset']
```

Question 9: What is the purpose of the `os` library in the provided scraping script?

Answer 9: The `os` library is used in the scraping script to perform certain operating system functions, such as listing the contents of a directory using `os.listdir()`.

Question 10: How can you modify the provided scraping script to print the contents of the 'info' paragraph in each page?

Answer 10: You can modify the script to find the 'info' paragraph using BeautifulSoup's `find` method and then print its text. For example:

```
pythonCopy code
info_paragraph = soup.find('p', class_='info')
print(info_paragraph.text)
```

Question 11: What modification is needed to the scraping script to only print information for leaf nodes?

Answer 11: To only print information for leaf nodes, you can check if the page has a 'container' element. If it does not have a 'container' element, then print the information. Otherwise, skip printing.

Question 12: How can you create and update a Python dictionary to store values scraped from leaf nodes?

Answer 12: You can create and update a Python dictionary by using the page titles as keys and the corresponding content of the 'info' box as values. For example:

```
pythonCopy code
leaf_nodes = {}

# Inside the loop
leaf_nodes[soup.title.text] = info_paragraph.text
```