A1: #

A2: 6

A3: static

A4: in

A5: GET allows accepting one element at a time

A6: The <TR> is used to create a data cell

A7: a {text-decoration:none;}

A8: al

A9: [111, 44, 1, 22]

A10: padding

B1: Explain the functionality of TCP/IP layers and give examples for protocols used in each layer.

The TCP/IP model is a four-layer model that divides network communications into four distinct categories or layers. The model is often referred to as the TCP/IP stack. The four important layers are the application layer, the transport layer, the network layer, and the link layer.

- The Application Layer: The application layer is closest to the end user. And this is the layer that users interact with directly, including protocols such as HTTP, FTP, and SSH. This layer is responsible for providing applications with access to the network.
- The Transport Layer: The transport layer ensures that data is delivered reliably and efficiently from one point to another. This layer handles data transmission between hosts, including protocols like TCP and UDP.
- The Internet Layer: The network layer is responsible for routing data through the web. This layer delivers data packets from one host to another, including the IP protocol.
- The Link Layer: The link layer provides reliable data links between the two nodes — for example, protocols like ethernet and Wi-Fi.

B2: Explain CSS specificity with examples.

If there are two or more CSS rules that point to the same element, the selector with the highest specificity value will "win", and its style declaration will be applied to that HTML element.

Think of specificity as a score/rank that determines which style declaration is ultimately applied to an element. Every CSS selector has its place in the specificity hierarchy.

There are four categories which define the specificity level of a selector:

1. Inline styles - Example: <h1 style="color: pink;">
2. IDs - Example: #navbar
3. Classes, pseudo-classes, attribute selectors - Example: .test, :hover, [href]
4. Elements and pseudo-elements - Example: h1, ::before

B3: Explain the difference between the following,

display: none;

display: inline;

display: block

none: The element is completely removed

inline: Displays an element as an inline element (like). Any height and width properties will have no effect

block: Displays an element as a block element (like <p>). It starts on a new line, and takes up the whole width

B4: Explain event propagation and event bubbling using examples.

Event propagation

Propagation refers to how events travel through the Document Object Model (DOM) tree. The DOM tree is the structure which contains parent/child/sibling elements in relation to each other. You can think of propagation as electricity running through a wire, until it reaches its destination. The event needs to pass through every node on the DOM until it reaches the end, or if it is forcibly stopped.

Event bubbling

Bubbling and Capturing are the two phases of propagation. In their simplest definitions, bubbling travels from the target to the root, and capturing travels from the root to the target. However, that doesn't make much sense without first defining what a target and a root is.

The target is the DOM node on which you click, or trigger with any other event.

For example, a button with a click event would be the event target.

The root is the highest-level parent of the target. This is usually the document, which is a parent of the , which is a (possibly distant) parent of your target element.

For example → see solutionB4_ex_02_event_continuation.html and solutionB4_ex_02_event_propagation.html

C1 → see solutionC1_ex_02.html

C2 → see solutionC2_ex_02.html

C3 → see solutionC3_ex_02.html

C4 → canvas is no longer part of the course.