JavaScrip 2

1. **Why is it important to write clean code?**
   Writing clean code is important because it allows you to clearly communicate with the next person who works with what you've written. Being able to return to previously written code and understand what it does is key, especially in the software development world.
2. **What is the difference between good comments and bad comments?**
   A good comment tells the reader why this particular code is doing whatever it is doing or explains what a section of code is about to do. A bad comment restates what a particular line of code is doing. Lines of code should be written so that they speak for themselves.
3. **What is an array?**
   Array is a single variable that is used to store different elements. It is often used when we want to store list of elements and access them by a single variable.
4. **What are arrays useful for?**
   An Array is used to store a collection of data, and it is more useful to think of an array as a collection of variables of the same type.
5. **How do you access an array element?**
   document.getElementById("demo").innerHTML = arrayName[0];
6. **How do you change an array element?**
   cars[0] = "Opel";
7. **What are some useful array properties?**
   cars.length // Returns the number of elements
   cars.sort() // Sorts the array
8. **What are some useful array methods?**
   Converting Arrays to Strings - toString()
   Popping - pop()
   Pushing - push()
   Shifting Elements - shift()
   Unshifting Elements - unshift()
   Array length - .length
   Array delete
   Merging Arrays - concat()
   Splicing Arrays - splice()
   Slicing Arrays - slice()
   Automatic toString()
9. **What are loops useful for?**
   Loops are all about doing the same thing over and over again. Often, the code will be slightly different each time round the loop, or the same code will run but with different variables.
10. **What is the break statement?**
    The break statement terminates the current loop, switch , or label statement and transfers program control to the statement following the terminated statement.
11. **What is the continue statement?**
    The continue statement terminates execution of the statements in the current iteration of the current or labeled loop, and continues execution of the loop with the next iteration.

12. **What is the DOM?**
    The DOM (or Document Object Model) is a tree-like representation of the contents of a webpage - a tree of “nodes” with different relationships depending on how they’re arranged in the HTML document.
13. **How do you target the nodes you want to work with?**
    When working with the DOM, you use “selectors” to target the nodes you want to work with. You can use a combination of CSS-style selectors and relationship properties to target the nodes you want. For example, you could use the following selectors to refer to
14. **How do you create an element in the DOM?**
    By using document.createElement(tagName, [options]) creates a new element of tag type tagName.
15. **How do you add an element to the DOM?**
    To add a new element to the HTML DOM, you must create the element (element node) first, and then append it to an existing element.
    For example
    const para = document.createElement("p");//create the element
    const node = document.createTextNode("This is new.");//add an element
    para.appendChild(node);
16. **How do you remove an element from the DOM**?
    div.classList.remove('new');
    // removes "new" class from div
17. **How can you alter an element in the DOM?**
    When you have a reference to an element, you can use that reference to alter the element’s own properties. This allows you to do many useful alterations, like adding/removing and altering attributes, changing classes, adding inline style information and more.
18. **When adding text to a DOM element, should you use textContent or innerHTML?**
    That textContent is preferable for adding text, and innerHTML should be used sparingly as it can create security risks if misused.
19. **Where should you include your JavaScript tag in your HTML file when working with DOM nodes?**
    Inside of the document <head> section in order to keep them contained and out of the main content of your HTML document.
20. **How do “events” and “listeners” work?**
    - Events are actions or occurrences that happen in the system you are programming, which the system tells you about so your code can react to them.
    - To react to an event, you attach an event listener to it. This is a block of code (usually a JavaScript function that you as a programmer create) that runs when the event fires. When such a block of code is defined to run in response to an event, we say we are registering an event handler.
21. **What are three ways to use events in your code?**
    inline, using a property, or using a listener
22. **Why are event listeners the preferred way to handle events?**
    They allow us to add interactive functionality to HTML elements by “listening” to different events that take place on the page, such as when the user clicks a button, presses a key, or when an element loads.
23. **What are the benefits of using named functions in your listeners?**
    So it can be called when a specified event happens, such as when a user clicks a button.

24. **How do you attach listeners to groups of nodes?**
    add a listener to each of them we simply need to iterate through the whole list like so:
    ```
    <div id="container">
      <button id="1">Click Me</button>
      <button id="2">Click Me</button>
      <button id="3">Click Me</button>
    </div>
    ```
25. **What is the difference between the return values of querySelector and querySelectorAll?**
    The difference between querySelector() and querySelectorAll() is that querySelector() returns a single object with the first HTML element that matches the 'selectors', but querySelectorAll() returns an array of objects with all the HTML elements that match the 'selectors'.
26. **What does a "nodelist" contain?**
    A NodeList is a collection of document nodes (element nodes, attribute nodes, and text nodes). HTMLCollection items can be accessed by their name, id, or index number. NodeList items can only be accessed by their index number. An HTMLCollection is always a live collection.
27. **Explain the difference between "capture" and "bubbling".**
    Event capturing means propagation of event is done from ancestor elements to child element in the DOM while event bubbling means propagation is done from child element to ancestor elements in the DOM.
28. **What is the difference between objects and arrays?**
    Objects represent a special data type that is mutable and can be used to store a collection of data (rather than just a single value). Arrays are a special type of variable that is also mutable and can also be used to store a list of values.
29. **How do you access object properties?**
    Three way to access object properties is
    - Dot property accessor: object. property.
    - Square brackets property access: object['property']
    - Object destructuring: const { property } = object.