**Module (JAVASCRIPT BASIC & DOM) – 4**

**(Array and object Question)**

* What is the drawback of declaring methods directly in JavaScript objects?
  + In JavaScript, declaring methods directly in objects can have some drawbacks, primarily related to code organization, inheritance, and memory usage. Here are some potential drawbacks:
    - Code Organization: When methods are declared directly in objects, it can lead to less organized and harder-to-maintain code, especially as the codebase grows. This approach might make it difficult to separate concerns and follow best practices for modular and maintainable code.
    - Inability to Inherit: Objects created with methods declared directly are typically standalone instances, and they don't benefit from prototype-based inheritance. This can limit code reuse and make it harder to implement common functionality across multiple objects.
    - Memory Usage: Each object created with methods directly declared will have its own copy of those methods. This can lead to increased memory usage, especially if you have many instances of similar objects. In contrast, using prototype-based inheritance allows multiple objects to share the same set of methods, potentially reducing memory overhead.
    - Immutability Issues: If methods are declared directly in an object, they can be easily overwritten or modified elsewhere in the code, leading to unexpected behaviour. Using a more structured approach, such as using constructor functions and prototypes, can help enforce encapsulation and reduce the risk of unintended modifications.
    - Readability and Maintainability: Declaring methods directly in objects might make the code less readable, as the structure of the object becomes cluttered with both data properties and methods. This can make it harder for other developers (or even yourself) to understand and maintain the code over time.
  + To address these drawbacks, developers often use constructor functions, classes (introduced in ECMAScript 2015), or other design patterns to create more organized, maintainable, and extensible code. These approaches allow for better separation of concerns, improved code reuse, and adherence to principles like encapsulation and inheritance.