QUESTION 2 : Write a blog about objects and its internal representation in Javascript

OBJECTS -

In JavaScript, objects are a fundamental data type and are used to represent and store data.

OBJECT'S INTERNAL REPRESENTATION -

1. **Object as a Key-Value Store:**
   - JavaScript objects are essentially collections of key-value pairs, where each key is a string or a symbol, and each value can be of any data type.
2. **Dynamic Properties:**
   - Objects in JavaScript can have properties added or removed dynamically during runtime, making them versatile for representing data structures.
3. **Object Literal Notation:**
   - Objects can be created using literal notation, where properties and values are defined within curly braces. For example:
   - let person = { name : 'john', age : 30 }
4. **Property Access:**
   - Object properties can be accessed using dot notation (object.property) or bracket notation (object['property']), providing flexibility in property naming.
5. **Internal Representation as Hash Table:**
   - Internally, JavaScript engines often use a hash table-like structure to store object properties efficiently, allowing for quick retrieval and modification.
6. **Prototype Chain:**
   - Objects in JavaScript may be linked to a prototype object. If a property is not found in the object itself, the JavaScript engine

looks up the prototype chain to find the property.

7. **Functions as Objects:**
   - In JavaScript, functions are a type of object. They can have properties and methods like any other object, in addition to being callable.
8. **Object Methods:**
   - Objects can have methods, which are functions stored as properties. Methods can operate on the object's data and provide a way to encapsulate functionality.
9. **Object Serialization:**
   - Objects can be serialized into JSON (JavaScript Object Notation) format for data interchange between systems, making it easy to send and receive data.
10. **Object Cloning:**
    - Objects can be cloned using various methods, such as the Object.assign() method or the spread operator (...), allowing the creation of independent copies of objects.