1)Write a blog on Difference between HTTP1.1 vs HTTP2

Http1:

- **Head of line blocking:** One of the significant drawbacks of HTTP/1.1 is head-of-line blocking. This occurs when a request for a specific resource is delayed due to the queuing of other requests, even if those requests could be processed more quickly.
- **Multiple Connections:** To overcome the limitations of head-of-line blocking, browsers often open multiple connections to a server. However, this approach can lead to increased latency and resource usage.
- **Textual Representation:** HTTP/1.1 uses textual representation (ASCII) for its messages. While human-readable, it can be less efficient in terms of data transfer compared to binary formats.

Http2:

Multiplexing:

Multiplexing is a crucial improvement in HTTP/2 that allows multiple streams of data to be sent and received in parallel over a single connection. This eliminates the need for multiple connections, reducing latency and optimizing resource usage.

Header compression:

HTTP/2 introduces header compression, which significantly reduces the size of header fields in each request and response. This is a departure from HTTP/1.1, where headers are sent in their entirety with each request.

Binary framing:

HTTP/2 uses binary framing to encode and transmit data. This binary format is more efficient than the textual representation used in HTTP/1.1, reducing overhead and improving data transfer speeds.

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

Objects:

- In JavaScript, objects are composite data types that allow you to store and organize data in key-value pairs. They are versatile and can represent various entities, from simple variables to more complex structures. Objects can be created using the object literal notation or through the `Object` constructor.

```
// Object literal notation
let person = {
  name: 'John',
  age: 30,
  address: {
    city: 'New York',
    country: 'USA'
  }
};

// Object created using the Object constructor
let car = new Object ();
car.brand = 'Toyota';
car.model = 'Camry';
car.year = 2023;
```

Internal representation:

- Under the hood, JavaScript engines use various techniques to represent objects internally. One common approach is using a combination of properties and an internal `[[Prototype]]` property, forming a prototype chain. This allows objects to inherit properties and methods from other objects, creating a hierarchy.

- ***Properties:***
- Properties in JavaScript objects are stored as key-value pairs, where the key is a string or a symbol, and the value can be any valid JavaScript value, including other objects. These properties can be accessed using dot notation (`object. Property`) or bracket notation (`object['property']`).

console.log(person.name); // Output: John
console.log(person['address']['city']); // Output: New York

3)codeketa practice:

1) Write a code to get the input in the given format and print the output in the given format

Input Description:
To take an integer value

Output Description:
Print the integer value

Sample Input :
2
Sample Output :
2

input:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <script src="script.js">
     </script>
  <title>JavaScript Input and Output</title>
</head>
<body>

<script>
  // Input
   input_value = prompt ("Enter an integer: ");
   // Convert the string input to an integer
  input_value = parseInt(input_value);


  // Output
  console.log(input_value);
</script>

</body>
</html>
```

Output:

Ans:4

2) Write a code to get the input in the given format and print the output in the given format

Input Description:
A single line contains integers separated by space

Output Description:
Print the integer list of integers separated by space

```
Sample Input :
2 3 4 5 6 7 8
Sample Output :
2 3 4 5 6 7 8
```

Input:

```
array=['2','3','4','5','6','7','8']
console.log(array.join(' '))
```

output:

```
2 3 4 5 6 7 8
```

3) Write a code to get the input in the given format and print the output in the given format.

Input Description:
First-line indicates two integers which are the size of array and 'K' value. Second-line indicates an integer contains elements of an array.

Output Description:
Print the taken input in the same format.

```
Sample Input :
5 3
1 2 3 4 5
Sample Output :
5 3
1 2 3 4 5
```

Ans:

```
k=['3','222'];
 array2=["1","2","3","4","5"];
console.log(k.join(' '))
console.log(array2.join(' '))
```

output:

```
3 222
1 2 3 4 5
```

4) Write a code to get the input in the given format and print the output in the given format

Input Description:
First-line indicates two integers separated by space. Second-line indicates two integers separated by space. Third-line indicates two integers separated by space.

Output Description:
Print the input in the same format.

```
Sample Input :
2 4
2 4
2 4
Sample Output :
2 4
2 4
2 4
```

Ans:

```
array1=["22","26"]
array2=["11","24"]
array3=["010","20"]
console.log(array1.join(' '))
```

console.log(array2.join(' '))

console.log(array3.join(' '))

output:

```
22 26
11 24
010 20
```

4) Read about IP address, port, HTTP methods, MAC address

- **IP Address:** An IP (Internet Protocol) address is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. It serves two main purposes: identifying the host or network interface and providing the location of the device in the network. IP addresses can be IPv4 (e.g., 192.168.1.1) or IPv6 (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
- **Port:** Ports are used to uniquely identify different applications or services running on a single device within a network. They allow multiple processes to operate on a single device without interfering with each other. Ports are numbered and range from 0 to 65535. Well-known ports (0-1023) are reserved for specific services, while registered and dynamic ports (1024-65535) are used for various applications dynamically.
- **HTTP Methods:** These are actions or operations that indicate the desired action to be performed on a resource identified by a URL. Common HTTP methods include:
    - **GET:** Retrieves data from a specified resource.
    - **POST:** Submits data to be processed to a specified resource.
    - **PUT:** Updates a specified resource.
    - **DELETE:** Deletes the specified resource.
    - **PATCH:** Partially updates a specified resource.
- **MAC Address:** A MAC (Media Access Control) address is a unique identifier assigned to network interfaces for communications on a network segment. It operates at the data link layer of the OSI model and is often physically embedded in network hardware such as network cards. MAC addresses are represented as a series of hexadecimal numbers separated by colons or hyphens (e.g., 00:1A:2B:3C:4D:5E).