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

|  |  |
| --- | --- |
| **HTTP/1.1** | **HTTP/2** |
| * It Loads resources one after other so if one resource cannot be loaded, it blocks all the other resources behind it. * It relies on the transport layer to avoid buffer overflow, each new TCP connection requires a separate flow control mechanism. * transfer all the requests & responses in the plain text message form. * before sending the request and the response there is a TCP connection established between client & server. again you make a request to the server for image img.jpg & the server gives a response as an image img.jpg. | * It sends multiple streams of data at once so that no one resource blocks any other resource. * multiplexes streams within a single TCP connection, and will have to implement flow control in a different manner. * HTTP/2 works on the binary framing layer instead of textual that converts all the messages in binary format. * HTTP/2 uses HPACK which is used to split data from header. it compresses the header. The server sends all the other files like CSS & JS without the request of the client using the PUSH frame. |

2. **Write a blog about objects and its internal representation in JavaScript**.

An object stores its state in fields (variables in some programming languages) and exposes its behaviour through methods (functions in some programming languages).

Objects are important data types in JavaScript. Objects are different than primitive datatypes (i.e. number, string, Boolean, etc.). Primitive data types contain one value but Objects can hold many values in form of Key: value pair. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.

**Internal Representation of Objects in JavaScripts:**

var <object-name> = {key1: value1, key2: value2,... keyN: valueN};

here value will be either String,integer or etc.

for i.e:

let obj = {

name: “Karthick”

gender: “male”;

Age: 29;

}

Console.log(obj)