1.Difference between Browser JS (console) vs node.js

Browser JS	Node JS
browser.js is mainly used for client-end application like validation on a webpage.	node.js is used for server-side application and it gets executed outside of the browser (Server or back-end process)
browser.js is sandboxed for the safety purpose and have access limited to the browser.	node.js has complete access to the system that is it can read and write directly to the system file thus making changes to the system.
browser.js and not headless	node.js is headless.
Moduling is not mandatory in browser.js	Everything is a module in node.js
Browser.js runs any engine like spider monkey (Firefox), Javascript core (Safari) and V8 in (google chrome)	Node.js runs in V8 engine which is mainly used by google chrome.
"Window" is a predefined global object which has function and attributes; and location is a predefined object.	Node.js doesn't have any predefined object, whereas the "location" object is related to a particular URL, whereas I have "global" which is a predefined object which contains several functions.

2.Watch and Summary 5 points

//https://www.youtube.com/watch?v=SmE4OwHztCc&ab_channel=JSConf

1.HTML parsing

- HTML is forgiving in nature, and parsing isn't straight forward too, some tags can be eliminated e.g.,<head>, <body>,<html>
- **Speculative parsing** – images and script form externals.
- **Reentrant** – Means the parsing process can be interrupted.

2.CSS parsing

- Combining DOM (parse HTML) and CSSOM (parse CSS) to give style resolution to the website.
- **Multiple tress**
    1. Render objects
    2. Render styles
    3. Render layers
    4. Line boxes
- **Not in render tree -** Nonvisual element like head, script and title etc. are not shown on the page. Also will be hidden via display; none.

- **Render Tree -** Render object has reference to DOM node like height and width i.e., visual output, geometric info and layout so on.

  Combining all styles, defaults, complexity around matching rules for each element

  Calculating visual properties.

3. Layout

- **Recursive process -** Transverse renter tree , odes position and size and layout its children
- **Batch layout -** Browser will intelligently batch changes, render tree will flag themselves as dirty, then batch will transverse the tree clearly arranges the items.
- **Immediate layout -** Doing a font size change will relay out the entire document, and it happens with the same browser,
- Certain properties will be accessed via JavaScript e.g., node.offsetheight.

4.**Paint**

- Takes the layed out render tress as base then create layers and is an incremental process and is build up over 12 phases(like background, shadow etc.)
- Create layes from render objects, then position nodes, transparency, overflow etc.
- Many to one relationship a render layer could contain multiple render objects.
- Bitmap from each layer, uploaded to GPU as a texture then it is composites into an image on the screen.
- **Inline critical CSS -** Speed up paint times,

Question No 4

	Question	**Output**
a.	typeof(1)	`Number`
b.	typeof(1.1)	`Number`
c.	typeof('1.1')	`String`
d.	typeof(true)	`Boolean`
e.	typeof(null)	`Object`
f.	typeof(undefined)	`undefined`
g.	typeof([])	`Object`
h.	typeof({})	`Object`
i.	typeof(NaN)	`Number`