Zen Task Day 2

1. **List 5 difference between Browser JS(console) v Node JS**.

S.no	Browser JS (console)	Node JS
1.	"window" is a predefined global object which has functions and attributes, that have to deal with window that has been drawn.	Node doesn't have a predefined "window" object because it doesn't have a window to draw anything.
2.	"location" is another predefined object in browsers, that has all the information about the url we have loaded.	"location" object is related to a particular url; that means it is for page specific. So, node doesn't require that.
3.	"document", which is also another predefined global variable in browsers, has the html which is rendered.	Ofcourse Node doesn't have "document" object also, cause it never have to render anything in a page.
4.	Browsers may have an object named "global", but it will be the exact one as "window".	Node has "global", which is a predefined global object. It contains several functions that are not available in browsers, cause they are needed for server side works only.
5.	Browsers don't have "require" predefined. You may include it in your app for asynchronous file loading.	"require" object is predefined in Node which is used to include modules in the app.

2.Watch & summary 5 points

- When you open a website, the HTML parser in the browser will look into the raw HTML data and process it to build the DOM tree.

- After that when the parser come across external resources like CSS or JavaScript file, it goes off to fetch those files. The parser will continue as a CSS file is being loaded, although it will block rendering until it has been loaded and parsed. The JavaScript file is loaded in the background.

- The CSS files parsed to build the CSSOM tree just like the DOM tree to process and execute the CSS file in the proper way.

- How and when the JavaScript files are loaded will determine exactly when this happens, but at some point they will be parsed, compiled and executed.

- Merging the DOM tree and CSSOM tree to build the render tree. Now we have a complete render tree the browser knows what to render but not where to render. Therefore the layout of the page must be calculated. The rendering engine traverses the render tree, starting at the top and working down, calculating the coordinates at which each node should be displayed.Once that is completed, the final step is to take that layout information and paint the pixels to the screen.

Zen Task Day 2

4. Execute the below code and write your description in txt file

console.log(type of **(1)**); - number

console.log(type of **(1.1)**); - number

console.log(type of **('1.1')**); - string

console.log(type of **(true)**); - boolean

console.log(type of **(null)**); - object

console.log(type of **(undefined)**); - undefined

console.log(type of (**[])**); - object

console.log(type of (**{})**); - object

console.log(type of **(NaN**)); - number