WEB DEVELOPMENT

Mentor:- Bhavesh Bansal

Course Structure:-

(i) HTML
(ii) CSS
(iii) Javascript
→ Boaring part

JAVASCRIPT → Intro of javascript code:

(iv) PROJECT
→ Automation code
→ Cricbuzz
→ github
→ Wcat command
→ Hackerrank

for tester profile

HTML, CSS → Browser Activities

(1) Web Scrapping & Automation:
(4-5 projects)
(2.) DOM
(3-4 Projects)
(3.) React js

→ Javascript almost all.
→ JQuery Selector
5* callbacks, Promise, Async wait

→ WEBSITES → Static Website
→ Dynamic Website → Javascript
Quiz
jira
camera

Backend Driven → Server side
Client Side → Browser

React: (Most important):-

→ Components
→ Containers
→ Redux
→ React hooks

→ Instagram Reels
→ Instagram clone
→ Resume builder
→ Google drive

> git → How to use git

Backend Development (Aws, cloud) >
- → Hosting
- → Authorization
- → MongoDB
- → SQL
- → Node.js (API's)

WEB SCRAPPING → data chori

MERN
- M → MySQL / MongoDB
- E → Express.js
- R → React.js
- N → Node.js

→ MERN STACK DEVELOPER

MEAN
- M → MySQL / MongoDB
- E → Express.js
- A → Angular
- N → Node.js

→ MEAN STACK DEVELOPER

Introduction to JAVASCRIPT

Node.js install

Chrome/Edge → Onclient

JAVASCRIPT (Single-threaded language)

→ Programming language

→ javascript can control HTML elements or it is used for making dynamic website.

PPq folder

↓

.js file

① VARIABLES REPRESENTATION

(i) let (ii) var (iii) const

Difference

const can't be assigned new value if it has been already assigned.

```
const a = 10
a = 20
```
error

Run javascript program:

→ node filename.js

Vs engine → Chrome

Node.js → Visual studio code for javascript code.

Online compiler for javascript

→ Coding blocks compiler

javascript Run on

→ Browser or client side.

→ Creating Variable:-

→ let a=10;
console.log(a); o/p → 10

→ let a=10.5;
console.log(a); o/p → 10.5

→ let a="Rupesh";
console.log(a); o/p → Rupesh

, → Refers to space.
\n → New line.
* Array index start with 0 (zero).
* Size of Array start with 1.

→ Loops:-

```
for (let i=1; i<=10; i++) {
    console.log(i);
}
```

o/p →
1
2
3
.
.
10

```
let i=0;
while (i<10) {
    console.log(i);
    i++;
}
```

o/p →
0
1
2
3
.
.
9

Odd/Even program:-

```
for (let i=1; i<10; i++) {
    if (i%2==0) {
        console.log ("even\n", +i);
    }
    else {
        console.log ("odd\n", +i); }
}
```

o/p →
odd
1
even
2
odd
3
.
.
even
10

ARRAYS:-

Can be inserted any data types.

```
let arr = [1, 2, 3, 4];
console.log(arr);
```
o/p → [1, 2, 3, 4]

→ for inserting different value inside the array.

```
let arr = [1, 2, 3, 'Rupesh', true, 10.5];
```
(0, 1, 2, 3, 4, 5 → Index no. of Array)

```
console.log(arr);
```
o/p → [1, 2, 3, 'Rupesh', true, 10.5]

```
console.log(arr[2]);
```
o/p → 3

printing the element inside the array.

Length of array = index+1

```
console.log(arr.length)
```
o/p → 6

Printing the length of array.

```
arr[6] = "Rajat"
console.log(arr);
```
o/p → [1, 2, 3, 'Rupesh', true, 10.5, 'Rajat']

New element added at index no. 6.

(To insert a element at particular index in Array):-

```
arr[2] = 89;
console.log(arr);
```
o/p → [1, 2, 89, 'Rupesh', true, 10.5, Rajat].

previous value at index no. 2 has been overwrite.

→

```
const a = [];
a[1] = 'Rupesh';
console.log(a);
```
o/p → [<1 empty item>, 'Rupesh']

push → Element will be added at the end.
Pop → Last element will be deleted.

:) let arr2 = [];

→ Inserting new element at the end.

```
arr2.push("Rupesh");
console.log(arr2);
```
o/p → ['Rupesh']

→ Removing last element from the array.

```
arr2.pop();
console.log(arr2)
```
o/p → []

→ Pushing the element at particular index.

```
arr2[4] = "hello";
console.log(arr2);
```
o/p → [<4 empty items>, 'hello']

→ Printing the length of array.

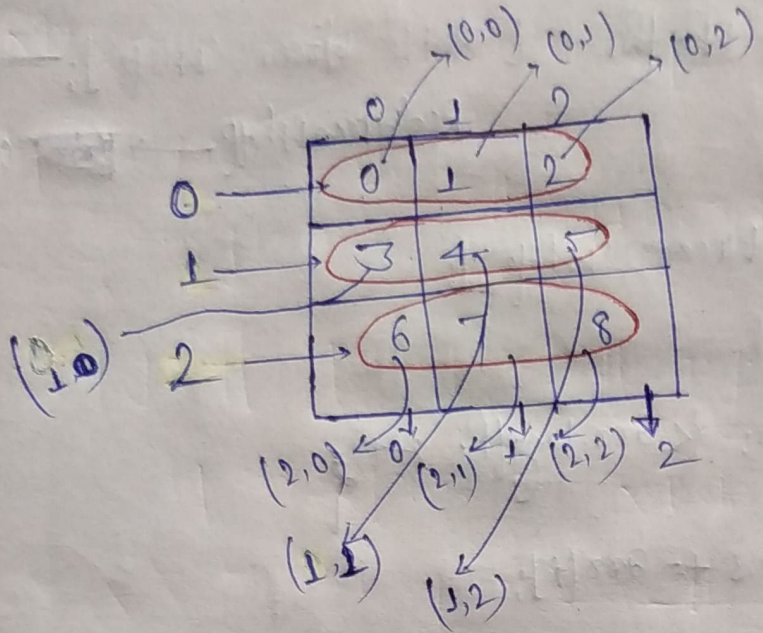
```
console.log(arr2);
console.log(arr2.length);
```
o/p → [<4 empty items>, 'hello']
o/p → 5
↳ length of Array2.

→ To Represent last index of ARRay.

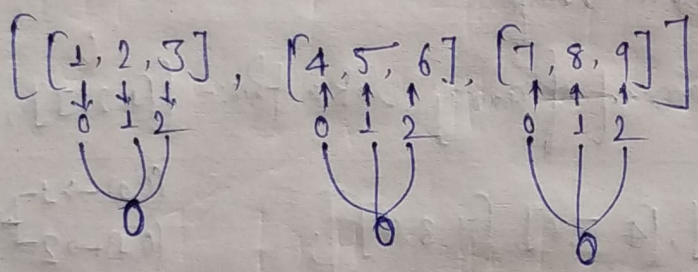
```
console.log(arr2[arr2.length-1]);
```
o/p → hello

→ CREATING EMPTY ARRAY WITH DEFINED SIZE.

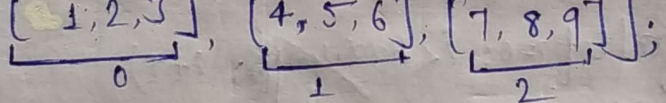
```
let arr1 = new Array(10);
console.log(arr1)
console.log(arr1.length)
```
o/p → [<10 empty item>]
o/p → 10



Real 2-> ARRAY:—



TRAVERSE




```
let arr = [[1, 2, 3], [4, 5, 6], [7, 8, 9]];
    for(let i=0; i<arr.length; i++){
        for(let j=0; j<arr.length.j++){
            console.log(arr[i][j]);
    }
}
```

o/p → 1
2
3
4
.
.
.
10

Ques.(1) Print the element of array like this:- ["Red", "Green", "White"]; ⟶ given array.

RedGreenWhite ⟶ o/p

```
let arr = ["Red", "Green", "White"];
  let ans = "";
    for(let i=0; i<arr.length; i++)
      {
        ans += arr[i];
      }
    console.log(ans);
```

o/p ⟶ RedGreenWhite

Ques.(2) Find sum of element of array like this:- [[1,2,3], [4,5,6], [7,8,9]]

(0) (1) (2)

```
let arr = [[1,2,3], [4,5,6], [7,8,9]];
  let sum = 0;
    for(let i=0; i<arr.length; i++)
      {
        for(let j=0; j<arr.length; j++)
          {
            sum += arr[i][j];
          }
      }
    console.log(sum);
```

(0,0) → 1
(0,1) → 2 → 3
(0,2) → 3 → 6
(1,0) → 4 → 10
(1,1) → 5 → 15
(1,2) → 6 → 21
(2,0) → 7 → 28
(2,1) → 8 → 36
(2,2) → 9 → 45

o/p ⟶ 45

Ques. Print the element of 2D Array

```
let arr = [[1,2,3], [4,5,6], [7,8,9]];
for (let i=0; i<arr.length; i++){
    for (let j=0; j<arr[i].length; j++){
        console.log(arr[j]);
    }
}
```

O/P →
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]
[1, 2, 3]
[4, 5, 6]
[7, 8, 9]

Ques. Print 2D ARRAY in 1D ARRAY form:—

```
let arr = [[1,2,3], [4,5,6], [7,8,9]];
let ans = [];
for (let i=0; i<arr.length; i++){
    for (j=0; j<arr.length; j++){
        ans.push(arr[i][j]);
    }
}
console.log(ans);
```

O/P → [1, 2, 3, 4, 5, 6, 7, 8, 9]

Ques. Increase the element of ARRAY by 2.

```
let arr = [1, 2, 3, 4];
for (let i=0; i<arr.length; i++){
    arr[i] += 2
}
console.log(arr);
```

O/P → [3, 4, 5, 6]

FUNCTION IN JAVASCRIPT

Representation:-

```
function abc(a,b){
    ...
}
abc();
```

→ function define

→ function call.

Ex:

```
function abc (a,b) {
    console.log ("Hello World");
}
abc();
```

O/P → Hello World

map() Method on ARRAY:-

arr.map( ) → function (data, index)

→ Map Method

Ex:

```
let arr = [1, 2, 3, 4];
function print (data, index){
    console.log (index, data);
}
arr.map (print)
```

O/P →

index	data
0	1
1	2
2	3
3	4

Ex:

```
let arr = [1, 2, 3, 4];
function addTwo (d,i){
    return d+2;
}
arr = arr.map (addTwo);
console.log (arr);
```

Overwrite the original value

O/P → [3, 4, 5, 6]

Ques. Print only odd number without filter method:-

```
let arr = [1, 2, 3, 4, 5, 6, 7, 8];
let ans = [];
for (let i=0; i<arr.length; i++){
    if (arr[i] % 2 != 0){
        ans.push(arr[i]);
    }
}
console.log(ans);
```

O/P → [1, 3, 5, 7]

→ filter() Method

Ques. print only odd number using filter() method.

```
let arr = [1, 2, 3, 4, 5, 6, 7, 8];
function removeEven(data, index){
    if (data % 2 == 0){
        return false;
    }else{
        return true;
    }
}
let ans = arr.filter(removeEven);
console.log(ans);
```

O/P → [1, 3, 5, 7]

Map ⟶ Used to change whole Array.

filter ⟶ Used to remove some element / filter the array.

↳ Returns only boolean value.

```
Let arr = [1, 2, 3, 4, 5, 6, 7, 8];

function removeEven (data, index) {
        return (data % 2 != 0)
}

Let ans = arr.filter (removeEven);
Console.log (ans);
```

o/p ⟶ [1, 3, 5, 7]

25th Mar. 2021

String:-

Representation: " " → double quote

' ' → Single quote

\ \ → Back tick.

Advantage

```
let abc = 123;
let str = `Bhavesh ${abc}`;
console.log(str);
```

o/p → Bhavesh 123

Add variable using $ (dollar)

Eg: let abc = 123;

```
let str1 = "bhavesh";
let str2 = 'bansal';
let str3 = 'pepcoding';
console.log (str1, str2, str3);
```

o/p → bhavesh bansal pepcoding

Alternate $ use

```
let str3 = `pepcoding ${abc}`;
console.log (str1, str2, str3);
```

o/p → pepcoding 123

Normal string without $ sign.

```
let str1 = "bhavesh" + abc;
```

o/p → bhavesh 123 bansal pepcoding 123

→ Using function:-

```
function abc() {
    return 123;
}
let str1 = "bhavesh";
let str2 = 'bansal';
let str3 = `pepcoding ${abc()}`;
console.log (str1, str2, str3);
```

o/p → bhavesh bansal pepcoding 123.

→ String Access

↳ By passing index in square bracket.

Ex: let abc = 123;

let str1 = "bhavesh" + abc;

let str2 = 'bansal';

let str3 = `pepcoding ${abc}`;

console.log(str1, str2, str3); o/p → bhavesh123 bansal pepcoding 123

console.log(str1[2]); o/p → a

console.log(str1[7]); o/p → 1

console.log(str3[7]); o/p → n

Ex:

let a = 1;

let b = 2;

let c = 3;

let str1 = "bhavesh";

let str2 = 'bansal';

let str3 = `pepcoding ${a+b+c}`;

console.log(str1, str2, str3); o/p → bhavesh bansal pepcoding6

console.log(str1[6]) o/p → h

→ String TRAVERSE

```
for (let i=0; i<str1.length; i++){
    console.log(str1[i]);
}
```

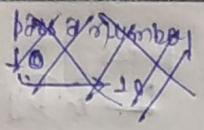
o/p →
b
h
a
v
e
s
h

```
let str = "";
for (let i=0; i<str1.length; i++){
    str += str1[i] + " ";
}
console.log(str);
```

o/p → b h a v e s h

for space

→ parseInt() → it convert the value to Integer.



console.log (parseInt (10)); o/p → 10

Ques:1 Decimal to Binary Conversion:—

Approach 1:

```
let n=10;
let str= " ";
while (n>0) {
    let rem = n%2;
    str = rem + str;
    n = parseInt(n/2);
}
Console.log (str);
```

o/p → 1010

Approach 2:

```
function decimalToBinary (number) {
    let ans= " ";
    while (number >0) {
        let rem = number % 2;
        ans = rem + ans;
        number = Math.floor (number/2);
    }
    console.log (ans);
}
decimalToBinary (parseInt ("10"));
```

o/p → 1010

Ques-2. Binary to decimal Conversion:-

```
function BinaryToDecimal(binary){
    let ans = 0;
    let power = 0;
    for(let i = binary.length-1 ; i>=0; i--){
        let binaryDigit = parseInt(binary[i]);
        ans += (Math.pow(2, power) * binaryDigit);
        power += 1;
    }
    console.log(ans);
}
BinaryToDecimal("1010");
```

O/p → 10

SUB-STRING (Use to get some part of string)

- substring(Startingindex, (Lastindex+1)

"Bhavesh Bansal"
0 1 2 3 4 5 6 7 8 9 10

- Substring(3,7)

Ex:
```
let string = "Bhavesh Bansal";
console.log(string.substring(3,7));
```
O/p → vesh

→ .toUpperCase() Method → To print the character of string in capital letter.

let String = "Bhavesh Bansal";
console.log (string.substring (3,7)); → o/p → vesh
console.log (string.toUpperCase()); → o/p → BHAVESH BANSAL

Ques(1) Print the first character of word in capital letter.

Ex: bhavesh bansal
→ Bhavesh Bansal
o/p

(2) Reverse every word.

Ex: My Name is Bhavesh
o/p
→ yM emaN si hsevahB

(3) String is Palindrome or not

(4) count of every character.

Ex: "adfgashsf-----"

a: 2
d: 1
f: 2
s: 2
|
|

(5) Find the first character of a string which is non-repeating.

: OBJECTS IN JAVASCRIPT :-

```
let obj = {
    key : value
}
```

Example:-

```
let obj = {
    "hello1" : 1
    "hello2" : 2
    "hello3" : 3
}
```

(name of object; key; value (it may be anything like no any discussion of data type))

Program:

```
let obj = {
    "hello1" : 1,
    2: "hello2",
    "hello3": "three"
}
```

```
console.log (obj[2]);          o/p → hello2
console.log (obj["hello1"]);   o/p → 1
console.log (obj["hello3"]);   o/p → three
```

⟹ To change value of object

```
obj[2] = "two";
console.log (obj[2]);
```

o/p → two

⟹ To get arrays of key.

```
console.log (Object.keys (obj));
```

(capital O.)

o/p → ['2', 'hello1', 'hello3']

```
console.log (Object.values (obj));   o/p → ['two', 1, 'three']
```

→ To get key ~~value~~ of object

```
Object.keys (obj)
```

o/p → ["hello1", "hello2", "hello3"]

→ To get only value of object

```
Object.values (obj)
```

o/p → [1, 2, 3]

```
obj["five"] = 5;
console.log (obj);
```

o/p → {'2': 'two', hello1: 1, hello3: 'three', five: 5}

Note: If there is no any key present then it gives undefined output.

```
console.log (obj);
```

o/p → {'2': 'two', hello1: 1, hello3: 'three'}

```
console.log (obj["six"];
```

o/p → undefined

9 Conversion → String ⟶ Array
Array ⟶ String

↳ String.split()
↳ जिसपर split() लगाएँगे वह array हो जाएगा।

String to ARRAY Conversion: —

Ex:

```
let string = "i love pepcoding";
console.log(string.split(" "));
```

o/p → ['i', 'love', 'pepcoding'] ⟵ split on space

```
console.log(string.split(""));
```

⟵ split on empty space/string.

o/p → [
'i', ' ', 'l', 'o',
'v', 'e', ' ', 'p',
'e', 'p', 'c', 'o',
'd', 'i', 'n', 'g'
]

```
Console.log(string.split("i"));
```

o/p → ['', ' love pepcod', 'ng']

```
Console.log(string.split("I"));
```

o/p → ['i love pepcoding']

```
let string = "ii";
console.log(string.split("i"));
```

o/p → ['', '', ''] (Space Space Space)

⇒ ARRAY TO STRING CONVERSION :-

join() → Use

```
let string = "I love peocoding";
console.log (string.split (" ").join (" "));
```

O/P → I love peocoding.

```
console.log (string.split (" ").join ("$"));
```

O/P → I $love $peocoding

```
console.log (string.split (" ").join (" $ "));
```

O/P → I $ love $ peocoding

```
let arr = ["hello", "how", "are", "you"];
console.log (arr.join (" "));
```

O/P → hello how are you

⇒

```
let str = "bhavesh";
console.log (str.charCodeAt (1));
```

// will print ASCII value

O/P → 104

27/03/21

folder → wcat ← file → wcat → left click → open integrated terminal → command

index
0 → Node path
1 → file path (script.js)

argv → pre-define Array.

command → npm i fs → Node Modules

Node package Manager → install

fs → Module → we can do anything related to file system in this module.

WCAT (Mini project)

→ To play with file system.

Saved:

folder → wcat

↓

Open integrated terminal.

↓

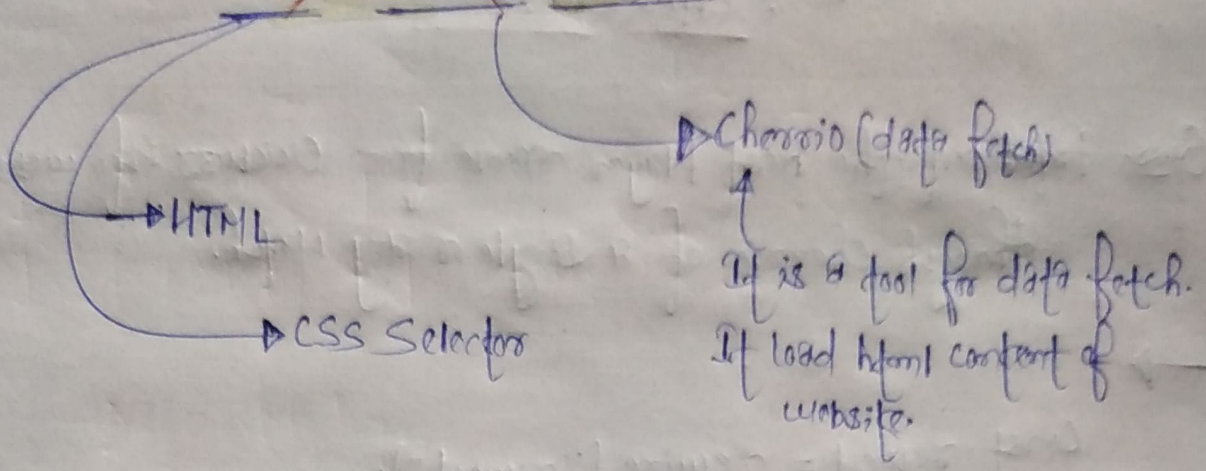
npm i fs

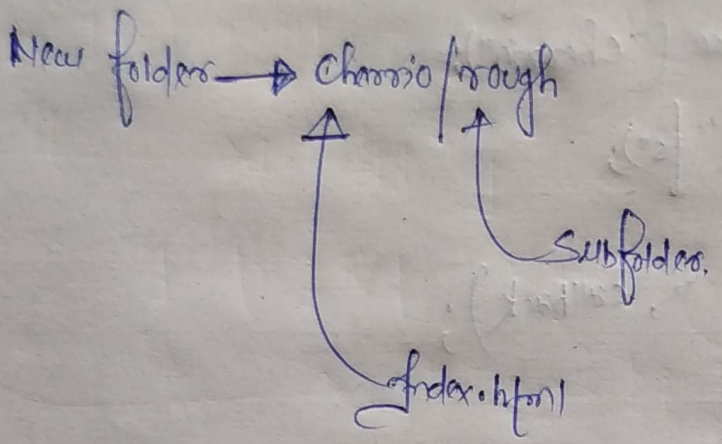
↓

Script.js → code written here.

03/04/21

—: WEB - SCRAPPING :—





→ index.html

Basic of html & color tag.

CSS (cascading style sheet.)

→ Basic of CSS Selector.

Clear CSS selector through question (q5.html to q10.html).

— : CHERRIO : —

Ques. fetch out the best player name from Cricbuzz info website through cherrio of a single or any Match.

→

```
const request = require ("request");
const cherrio = require ("cherrio");
const fs = require ("fs");

request ("                ", callback);
```

Url link of match from Cricbuzz website.

```
function callback (err, res, html) {
    if (!err) {
        fs.writeFileSync ("player.html", html);
        let $ = cherrio.load (html);
        let bestplayer = $(".best-player-name a");
        console.log (bestplayer.attr ("data-hover"));
    }
}
```

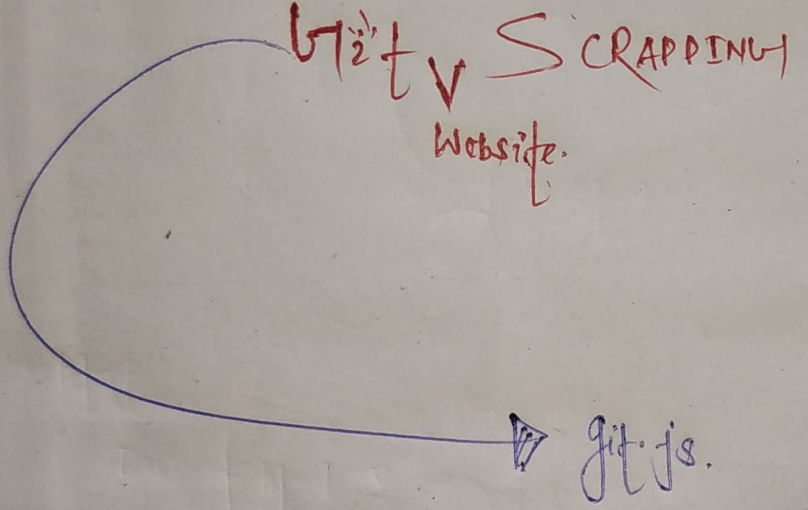
Here: load the html content of website in player.html file.

putting wrapper through $ (symbol) on html file.

Ques → fetch out the Birthday of players from Cricbuzz website.

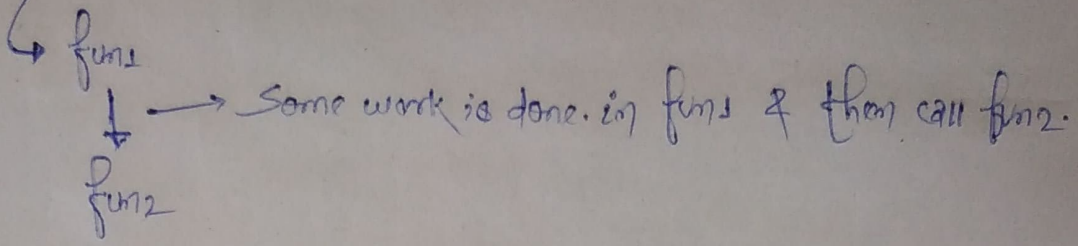
↳ birthday.js

Ques. → ipl.js



4th April 2021

Topic: Callbacks & Cricinfo:-

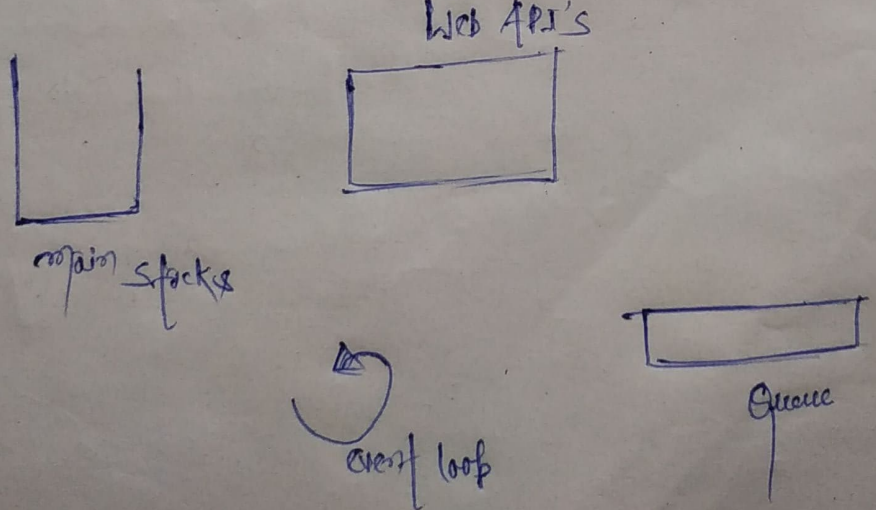


Ex:

```
function print (val) {          ⟶ fun 2

    console.log (val);

}

function add (n1, n2, print) {

    let sum = n1 + n2; }  ⟶ Some work is done
    print (sum);          ⟶ fun 1

}
```

add (2, 3, print)

जब कोई fun1 कुछ काम करके दूसरे fun2 को call कर दे तो fun2 को callback function कहते हैं।



11th Apr 2023

:- PUPPETEER :-

Java dependency → comb-mail.gg.

folder → HackerRank → script.js → open integrated terminal → npm install puppeteer

Cheerio
request ( ) Asynchronous
↓
Web API

puppeteer:-
↳ methods
Asynchronous
await.

Serial → Sync. Method
Parallel → Asyn. Method

async function abc ( ) {
await (p.method1())
p.method2()
}

→ headless: false
↳ Chrome browser dikhega.
→ headless: True
↳ Chrome browser nhi dikhega.

Question. covered in this Puppeteer:
HackerRank (Folder)
(1.) Script.js

Code Submission:- (Hackerrank)

(2.) challenges.js

Challenges creation (HackerRank)

Ques. Create youtube video downloader using puppeteer. I will pass the url of video and you have to download that.

→

```
const puppy = require("puppeteer");
const url = " url ";
```

↳ put youtube video link i.e url here.

```
async function main() {
    let browser = await puppy.launch({
        headless: false,
        defaultViewport: false
    });

    let tabs = await browser.pages();
    let tab = tabs[0];
    let newUrl = url.replace(".youtube", ".ssyoutube");
    await tab.goto(newUrl);
    await tab.waitSelector(".link.link-download.subname.ga-track_
        events.download-icon", {visible: true});
    await tab.click(".link.link-download.subname.ga-track_events.down
        icon");
}

main();
```

22/04/21

:- Callback (Revision) :-

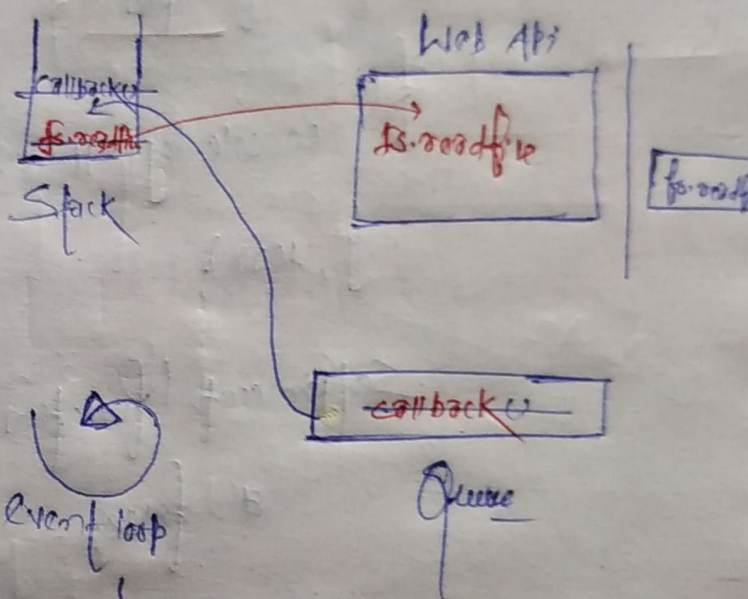
→ $f_1()$ → w_1
$f_2()$ → w_2
callback

} Here: firstly w_1 work done then $f_{n.2}$ callbacks for w_2 work.

fs.readfile(filename, "utf-8", callback);

→ fs.readfile(abc.txt, "utf-8", callback);
function callback (err, data){
console.log(data);
}

fs.readfile(1)
fs.readfile(2)

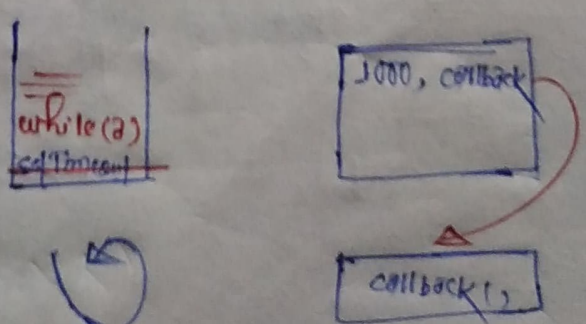


→ check whether stack is empty or not.
→ if stack will be empty then it will put callback() to stack.

→ Let a = true;
setTimeout(function(){
a = false;
}, 1000)
while(a){
console.log(a);
}

o/p → true infinite times

a = true



→ • setTimeout (function, milliseconds)

Executes a function, after waiting a specified number of milliseconds.

→ • setInterval (function, milliseconds)

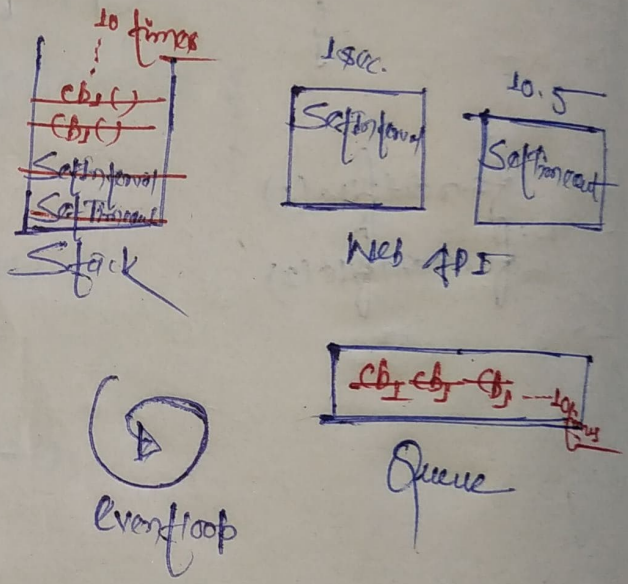
Same as setTimeout(), but repeats the execution of the function continuously

→

```
let a = true;
setInterval (function () {
    CB1
    if(a) console.log("Hello");
}, 1000);

setTimeout (function () {
    a = false;
}, 10500);
```

o/p → Hello → 10 times

Hello



:PROMISES & Async Wait:-

class start: 18:10

Globally install → npm install fs
fs. Module.

fs.readfile → Asynchronous.
fs.readfile Sync → Synchronous
fs.writefile → Asynchronous

(filename, "utf-8", callback)
↓
callback (err, data)

(filename, data, callback)
↓
callback (err)

↳ Math.random()
↓
0 to 0.9999...
0 to < 1

Math.floor (Math.random() * 101)

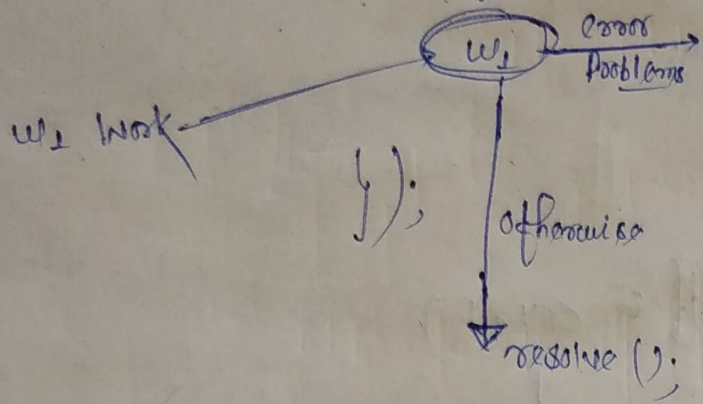
↳ It will give random no. betn 0 to 100

→ Callbacks hell →
↳ difficult to understand Multiple callbacks.

24/04/21

:- PROMISES :-

→ new Promise (function (reject, resolve) {



→ let a = false;

executor function.

new Promise (function (rej, res) {

Constructor.

if (a) { res(); }

else { rej(); }

});

→ let a = 2;

new Promise (function (rej, res) {

just like return abc;

if (a % 2 == 0) { res(); }

else { rej(); }

});

just like throw abc;

→ let a=2;

let aPromise = new Promise(function (req, res) {
if (a%2==0) { res ("yes! a is even"); }
else { rej ("oh! this number is odd"); }
});

aPromise .then (function (data) {
console.log (data);
}) .catch (function (err) {
console.log (err);
});

callback (cb2)

Here:
.then is like a callbacks.

cb1

Note:
.then (on fulfillment)
Mean on resolve
.catch (on Rejection)

o/p → yes! number is even

Note: Here → We can call callback fn. Multiple time.

4. At any point of time.

→ We can modify callbacks at any point of time.

→ readfilepromise.js

```
const fs = require("fs");
let readfilepromise = fs.promises.readfile("1.txt","utf-8");
    console.log(readfilepromise);
```

O/P → Promise { <pending> }

```
readfilepromise.then(function(data){
        console.log(readfilepromise);
        console.log("I ran second");
        return hello;
    })
    .console.log("I ran first");
readfilepromise.then(function(data){
        console.log("I ran last");
    })
```

O/P →
Promise { <pending> }
I ran first
Promise { '12345' }
I ran second
I ran last.

04/05/21

-: Async-await :-

→ Practice on promise topic. (website: codingame.com)
→ Learn Chaining through Quiz Question of codingame website.

→

-: Topic Start :-

async function abc() {

Inplace of return
throw "abc";

o/p → Promise {<rejected> 'abc'}

console.log("hello");
return ("abc");
}

let temp = abc();
console.log(temp);

o/p → hello
Promise {'abc'}

function abc() {
return new Promise(function(res, rej)
{
console.log("hello");
res("abc");
});
}

let temp = abc();
console.log(temp);

o/p → hello
Promise {'abc'}

Conversion of → Asyn fn. to promise fn.

→ We use await only in async.

→ Saare code ko promise me daal data hai.

await always use before promise.

→ Agle ki saari line ko .then me daal data hai.

⇒ File

abc.txt

1 2 3 4 5

```
const fs = require("fs");
async function abc(){
    let data = await fs.promises.readFile("abc.txt", "utf-8");
    console.log(data);
    return undefined;
}
abc().then(function(data){
    console.log(data);
}).catch(function(err){
    console.log(err);
})
```

o/p → 1 2 3 4 5
undefined

o/p
→ 1 2 3 4 5
undefined

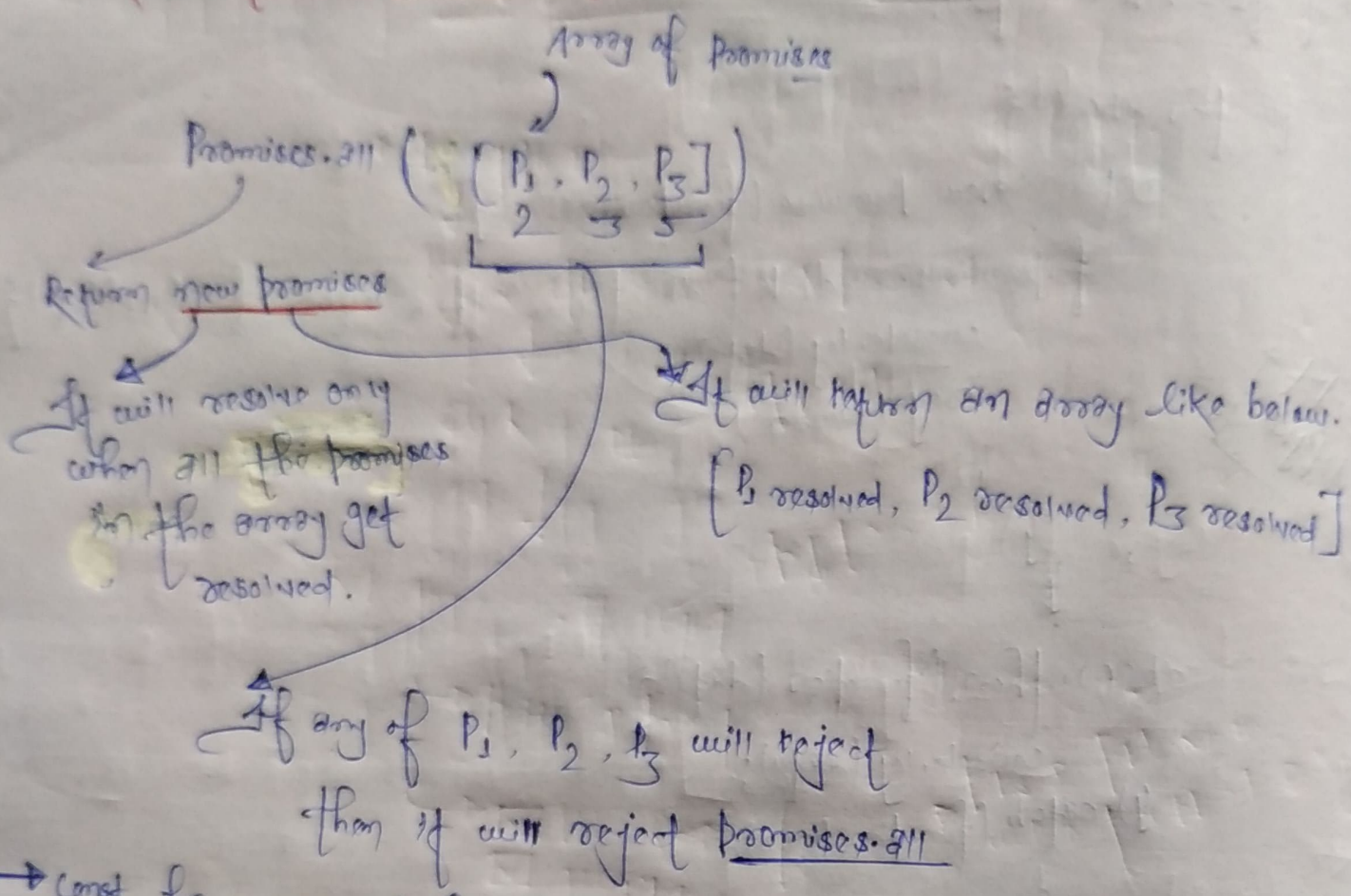
```
const fs = require("fs");

function abc(){
    return new Promise(function(res, rej){
        fs.promises.readFile("abc.txt", "utf-8").then(function(data){
            console.log(data);
            rej("error");
        })
    })
}

abc().then(function(data){
    console.log(data);
}).catch(function(err){
    console.log(err);
})
```

abc.txt
abcdefghi

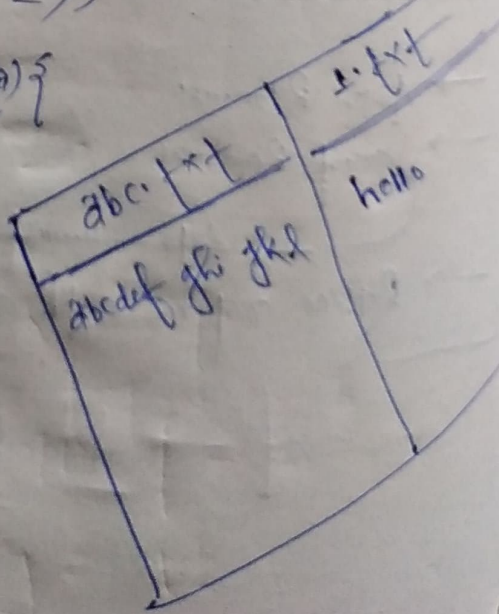
o/p → abcdefghi
error

→ Async Removal
→ await Removal
~~let data = await~~
→ Replace return with res();

→ Run Multiple Web APIs through Promises:-



→ const fs = require("fs");
let P1 = fs.promises.readFile("abc.txt", "utf-8");
let P2 = fs.promises.readFile("1.txt", "utf-8");
let combinedPromise = Promise.all([P1, P2]);
combinedPromise.then(function(data){
console.log(data);
})



O/P → ['abcdef ghi jkl', hello]

02/05/21

—: Experimental Study :—

folder

↓

Selenium

↓ file

script.js

→ npm install selenium-webdriver

→ npm install chromedriver

—: SELENIUM :—

Promise.all([P1, P2, P3, P4]).then(function(data)

↓ promise

↳ will wait until all the promises get resolve.

[[Rows], [Rows], [Rows]]

06/05/21:)

—: Hackathon Details:—

Test on Sundays: (6-10)

↳ callbacks, Promises, async-await

(30 Marks)

hackathon (2 days)

Automation / Webscrapping

↳ html, css, javascript; Selenium, cheerio, puppeteer

any node library.

→ Submission: 09/05/21 (evening 6 PM)

Top-5 Woodies

Certificate to all

Top 3: → 3k (Rank 1)

2k (Rank 2)

1k (Rank 3)

→ Upload the code on Git

Screen recorder

→ Post record video on Linkedin

→ Google form: name, email,

batch, idea, git url,

Linkedin post url.

→ Judgement:

Idea: 4 Marks

Working: 4 Marks

Neat & clean code: 2 Marks

UI: 0 Marks

06/05/21

:- Git :-

2:20:54

Open github.com

↓

login with username & password

↓

Click on New → left side

↓

Repository Name: ~~tempGitbit~~ → project reside here

→ ~~Public~~ & Private.

→ ☑ Add a ~~README~~ file

☑ Add .gitignore → files & foldername will not upload on git

→ Select template: Node

↓

Click on: Creating repository.

↓

Repository Ready

↓

Click on branch & Rename it:

→ master inplace of main

Bcz it is universally accepted.

→ Creat New folder; → temp

↓

Open integrated terminal

→ Download git & install it.

↓ Then

RUN COMMAND>

→ git init → for git initialising. (Normal folder ko git Repository banate hai)

→ To check git version;

git --version

→ To get Repository link

→ click on code (in repository)

↓

copy HTTPS Link

→ git remote add origin Repository Link

This command is used for link local git Repo & Cloud git Repo

→ git pull origin master

This command fetch the file gitignore, README.md file in local git Repo.

Only changes commit in git

↓

→ git add . → To check chage status

→ git status

→ git commit -m first commit → commit Name

→ git push origin master

Push the change to git

IF YOU GET AN ERROR LIKE THIS

* Please tell me who you are.

Run

git config --global user.email "you@example.com"

git config --global user.name "your Name"

Then RUN COMMAND IN TERMINAL

→ git config --global user.email "your email id" → xyz@gmail.com

→ git config --global user.name "your Name" → Rupesh

TO WORK ON A PROJECT WITH SOME DEVELOPERS

→ We fetch 3-4 (According to need) branch from master branch & work individually on each branch.

→ finally, Merge all branch to Master Branch.

→ git branch features → branch name

→ git branch → To check present branch

→ git checkout features → Switched to features branch

→ git push origin features → To push the added branch on git

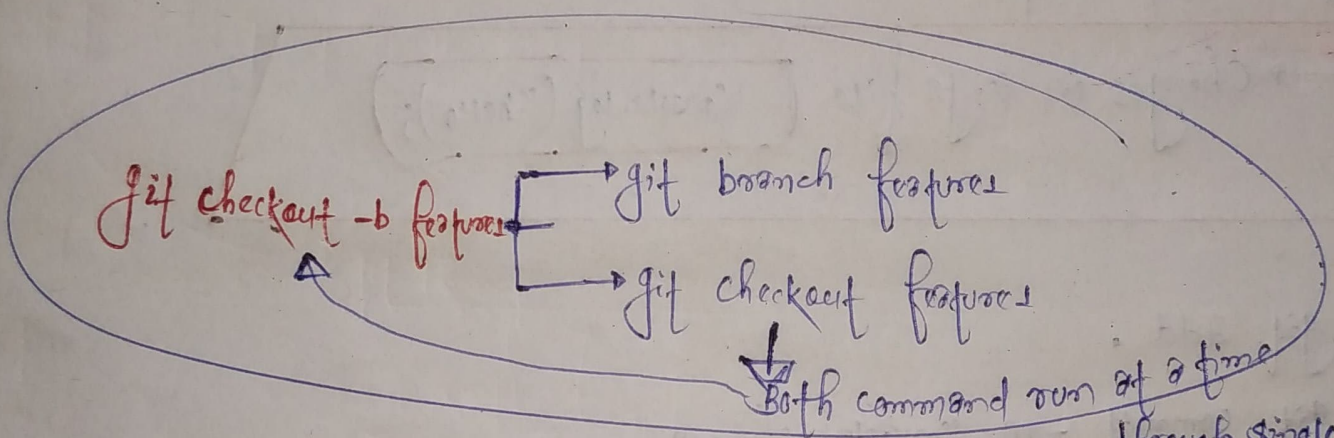
for changes → console.log("hello"); in a.js file.

↳ git add .

↳ git status

↳ git commit -m "add hello" → Message

↳ git push origin features



→ git checkout features

→ git checkout master

→ git merge features ← features Merge to master branch locally.

→ git push origin master ← changes shown on cloud.

↳ git branch -d features → Locally features branch deleted.

↳ git push origin --delete features → features branch deleted from cloud.

01:22:16

Merge conflict → Changes of same position

Senior Developer

↳ Master → Changes → hello1

↳ features → Changes → hello2

→ on a line of code

At Merge time.

Conflict ??

→ git push origin features

→ git checkout master

→ Changes in a.js file `Console.log("hello");`

→ git add .

→ git commit -m "master change"

→ git push origin master

→ git checkout features

Changes in a.js file → `Console.log("hello2");`

→ git add .

→ git commit -m "add hello2"

→ git push origin features

→ git checkout master

01:28:03

Merge features in Master.

→ git merge features

CONFLICT SHOWS
→ click on both

→ git add .

→ git commit -m "merge features"

→ git push origin master