Round-1

1. Which of the following statements are true?

a)Node js is a server side language

b)Node js is a client side language

c)Node js is a both server and client side language

d)None of the above

Correct Ans: Node js is a both server and client side language

2. Which of the below command is used to install package?

a)npm i <package_name>

b)node i <package_name>

c)npm <package_name> install

d)node <package_name> install

Correct Ans: npm i <package_name>

3. Handlers used to settle the promises in javascript.

a)try/catch/finally

b)then/catch/finally

c)then /finally

d)catch/finally

Correct Ans: then/catch/finally

4. In Javascript which of the below is used to finish the job successfully with the result value.

a)reject(value)

b)resolve(value)

c)catch(value)

d)result(value)

Correct Ans: resolve(value)

5. What is the output of the below code snippet ?

<pre>

function asynchronous () {

return await "done"

}

</pre>

a)error

b)undefined

c)done

d)null

Correct Ans: error

6. In Javascript which of the below is used when any error has occurred.

a)reject(value)

b)resolve(value)

c)catch(value)

d)result(value)

Correct Ans: reject(value)

7. How many states are there in the javascript promise?

a)1

b)2

c)3

d)4

Correct Ans: 3

8. Which of the npm packages is used for API creation?

a)express

b)express-js

c)expression

d)experience

Correct Ans: express

9. Which of the below function in express is used to start the server?

a)start

b)create

c)expression

d)listen

Correct Ans: listen

10. REST stands for?

a)Representational State Transform

b)Representational State Transfer

c)Represent State Transform

d)Represent State Transfer

Correct Ans: Representational State Transfer

11. What does SQL is used to perform operations on?

a)Update Records

b)Insert Records

c)Both A and B

d)None of the above

Correct Ans: Both A and B

12. Which of the following clause cannot be optional in SQL SELECT Statement?

a)WHERE

b)GROUP BY

c)ORDER BY

d)None of the above

Correct Ans: None of the above

13. Which of the below function is used to drop all the tables?

a)sequelize.drop()

b)sequelize.dropAll()

c)sequelize.delete()

d)sequelize.truncate()

Correct Ans: sequelize.drop()

14. Which of the below is equivalent to "Select * from Model" query?

a)Model.findAll()

b)Model.find()

c)Model.selectAll()

d)Model.select()

Correct Ans: Model.findAll()

15. What does the HTTP code 202 represent ?

a)No Content

b)Created

c)Success

d)Accepted

Correct Ans: Accepted

16. Which HTTP method do we use to make an HTTP request to update an existing resource?

a)POST

b)PUT

c)GET

d)CREATE

Correct Ans: PUT

17. Which of the below is not a correct form of accessing a user from a request query string?

a)const { user } = request.query;

b)const user = request.query.user;

c)const user = request.query;

d)None of the above

Correct Ans: const user = request.query;

18. Which HTTP method do we use to make an HTTP request to create a new resource?

a)POST

b)PUT

c)GET

d)CREATE

Correct Ans: POST

19. Which of the content-Type used to send the data in the form of JSON?

a)application/text

b)application/javascript

c)application/json

d)application/xml

Correct Ans: application/json

20. The process of login into the application is called?

a)Authorization

b)Authentication

c)None of the above

d)All the above

Correct Ans: Authentication

21. JWT stand for?

a)Javascript Web Token

b)Json Web Token

c)Java Web Token

d)Json Web Translation

Correct Ans: Json Web Token

22. The process of checking the user access and performing operation is called?

a)Authorization

b)Authentication

c)None of the above

d)All the above

Correct Ans: Authorization

23. CRUD stands for?

a)Create Read Upsert Delete

b)Create Read Update Delete

c)Create Read Upload Delete

d)Create Read Update Decode

Correct Ans: Create Read Update Delete

24. Choose the tool which is used to test the API?

a)Github

b)Heroku

c)Postman

d)Vscode

Correct Ans: Postman

25. Which of the below package is used for testing?

a)test

b)jest

c)nest

d)gest

Correct Ans: jest

26. Which of the below is not a hosting site?

a)AWS

b)Heroku

c)Netlify

d)TCS

Correct Ans: TCS

27. What will be the output of the below code?

<pre>

const arr = [1, 2, 3, 4];

console.log(arr.join(""));

</pre>

a)1,2,3,4

b)[1,2,3,4]

c)1234

d)4321

Correct Ans: 1234

28. What's the output of the following code ?

```
<pre>
doSomething( a, b)
{
if (b==1)
return a;
else
return a + doSomething(a,b-1);
}
console.log(doSomething(2,3));
</pre>
```

a)4

b)2

c)3

d)6

Correct Ans: 6

29. Given an input arr = {2,5,7,99,899}; key = 899;

<pre>

What is the level of recursion for binary search?

</pre>

a)5

b)2

c)3

d)4

Correct Ans: 3

30. What is the worst case complexity of binary search using recursion?

a)O(nlogn)

b)O(logn)

c)O(n)

d)O(n2)

Correct Ans: O(logn)

31. Given an array arr = {5,6,77,88,99} and key = 88;

<pre>

How many iterations are done until the element is found?

</pre>

a)1

b)2

c)3

d)4

Correct Ans: 2

32. Binary Search can be categorized into which of the following?

a)Brute Force technique

b)Divide and conquer

c)Greedy algorithm

d)Dynamic programming

Correct Ans: Divide and conquer

33. How many comparisons will be made to sort the array arr={1,5,3,8,2} using counting sort?

a)0

b)5

c)7

d)6

Correct Ans: 0

34. Which of the following uses the largest amount of auxiliary space for sorting?

a)Bubble

b)Counting

c)Quick

d)Heap

Correct Ans: Counting

35. Which of the following information is stored in a doubly-linked list's nodes?

a)Value of node

b)Address of next node

c)Address of previous node

d)All the above

Correct Ans: All the above

36. What is the optimal time complexity to count the number of nodes in a linked list?

a)O(nlogn)

b)O(logn)

c)O(n)

d)O(n2)

Correct Ans: O(n)

37. The term Push and Pop is related to

a)queue

b)stack

c)linkedlist

d)array

Correct Ans: stack

38. In which data structure, element is inserted at one end called Rear and deleted at other end called Front.

a)queue

b)stack

c)linkedlist

d)array

Correct Ans: queue

39. When the function calls another function then the details of the previous function are stored in Stack?

a)TRUE

b)FALSE

Correct Ans: TRUE

40. The process of inserting an element in the stack is called?

a)Enqueue

b)Insert

c)Push

d)Pop

Correct Ans: Push

Round-2

Question: 1

Question name: Footwear pair formation

Problem Statement

Ranvier is working in a footwear manufacturing company where a pair of footwears are packed but mistakenly some of the packed boxes having extra one footwear.
Given an array of integer as a size of the footwear. Ranvier has to re pair the footwear if any additional footwears are in the packed box.

Input Format

- First line contains a one integer number of boxes N.
- Next line contains Ai space separated integers denoting the pair of footwear.

Output Format

- Among the 2D array Ranvier has to re pair the array

Constraints

- n == mat[i].length
- 1 <= n <= 50
- 1 <= matrix[i][j] <= 105.

Sample Input 1

3
3 3 8
9 8 9
15 15

Sample Output 1

3 3

9 9

15 15

8 8

Explanation of Sample 1

First line represents n represent the number of rows ie. 3
Input: matrix = [[3,3,8],[9,8,9],[15,15]]
Output: [[3,3],[9,9],[15,15], [8, 8]]
Explanation:
1st subarray (3, 3) are pair and 8 is the odd one
2nd subarray (9, 9) are pair and 8 is the odd one
3rd subarray (15, 15) are pair and nothing is the odd here
At last form a pair using the odd one from all the subarray, here (8, 8) are the remaining and we can form a pair using that.

Sample Input 2

2
7 8
8 7

Sample Output 2

7 7

8 8

Explanation of Sample 2

First line represents m x n ie. 2 x 2 matrix (2 row , 2 columns)
Input: matrix = [[7,8],[8,7]]
Output: [[7,7],[8,8]]
Explanation:
1st subarray (7, 8) are not pair and both are odd one
2nd subarray (8, 7) are not pair and both are odd one
At last form a pair from the remaining element so we are forming a pairs as (7, 7) and (8, 8)

Template

```
var pairFormation = function (matrix) {

    let pairedArray = [];
    let unPairedArray = [];

    matrix.forEach(item => {
        let [resultPairs, nonResultPairs] = checkPairs(item);
       if (resultPairs.length)
            pairedArray.push(resultPairs);
        unPairedArray.push(...nonResultPairs);
    })
    let [resultPairs, nonResultPairs] = checkPairs(unPairedArray);
        pairedArray.push(...resultPairs);
    return pairedArray;
};

const checkPairs = (arr) => {
        const pairs = {};
        const resultPairs = [];
        const nonResultPairs = [];
        // Implementation logic here
        return [resultPairs, nonResultPairs];
}
```

```
const n = parseInt(readline());
let inpArr = [];
for (let i = 0; i < n; i++) {
        inpArr.push(readline().split(' ').map(item => parseInt(item)));

}

pairFormation(inpArr).map(item => console.log(item.join(' ')));
```

Solution

```
var pairFormation = function (matrix) {

   let pairedArray = [];
   let unPairedArray = [];

   matrix.forEach(item => {
        let [resultPairs, nonResultPairs] = checkPairs(item);
      if (resultPairs.length)
           pairedArray.push(…resultPairs);
        unPairedArray.push(...nonResultPairs);
   })
   let [resultPairs, nonResultPairs] = checkPairs(unPairedArray);
        pairedArray.push(...resultPairs);
```

```
    return pairedArray;
};

const checkPairs = (arr) => {
        const pairs = {};
        const resultPairs = [];
        const nonResultPairs = [];
        arr.forEach(item => {
                if (pairs[item])
                        pairs[item] += 1;
                else
                        pairs[item] = 1;
        })
        Object.keys(pairs).map(item => {
                if (pairs[item] > 1)
                        resultPairs.push([item, item]);
                else
                        nonResultPairs.push(item);
        })
        return [resultPairs, nonResultPairs];
}

const n = parseInt(readline());
```

```
let inpArr = [];

for (let i = 0; i < n; i++) {

        inpArr.push(readline().split(' ').map(item => parseInt(item)));


}

pairFormation(inpArr).map(item => console.log(item.join(' ')));
```

Solution: https://www.ideone.com/H7NEG7

Question: 2

Question name: Movie Time

Problem Statement

Given a two array A and B of integers. First array A represents the movie start time and the second array B represents the movie end time. Arrange the movie based on the running time.

Explanation

Example :

Input: 220 230 445 600

430 400 600 800

Output: 155 170 200 210

Constraints

$1<=A_i<=10^9$

$1<=B_i<=10^9$

Input Format

Input should be array of integer

Output Format

Running time in Ascending order

Sample Input 1 :

220 230 445 600

430 400 600 800

output :

155 170 200 210

Sample Input 2 :

200 200 345 700

530 400 600 800

output :

100 200 255 330

Template:

```
const movie = (start, end) => {
    let duration = [];
    // Implementation logic here
```

```
        return duration;
}

const quick = (arr, low, high, emp) => {
   // Implementation logic here
}

const start = readline().split(' ').map(item => parseInt(item));
const end = readline().split(' ').map(item => parseInt(item));
let op = movie(start, end);
console.log(op.join(' '));
```

Solution:

```
const movie = (start, end) => {
        let duration = [];
        for (let i = 0; i< start.length; i++) {
                duration.push(end[i] - start[i]);
        }

        quick(duration, 0, duration.length - 1);
        return duration;
}

const quick = (arr, low, high, emp) => {
   if (low >= high)
```

```
    return;
let start = low;
let end = high;
let mid = Math.floor((Math.random() * high) + low);
// let mid = Math.floor((start + end) / 2);
let pivot = arr[mid];

while (start < end) {
    while (arr[start] < pivot){
        start++;
    }

    while (arr[end] > pivot) {
        end--;
    }

    if (start <= end) {
        temp = arr[start];
        arr[start] = arr[end];
        arr[end] = temp;
        start++;
        end--;
```

```
        }
    }
    quick(arr, low, end);
    quick(arr, start, high);
}

const start = readline().split(' ').map(item => parseInt(item));
const end = readline().split(' ').map(item => parseInt(item));
let op = movie(start, end);
console.log(op.join(' '));
```

IDEOne link - https://www.ideone.com/lXi26Q

Round-3

Create a TODO API with the below functionality

Below are the API endpoints for the TODO Application

1) SignIn and SignUp API with below parameters
    a. id
    b. Email
    c. Password
    d. Created At

2) If user signed in successfully then return the Authorization token (JWT Token)

Round-4

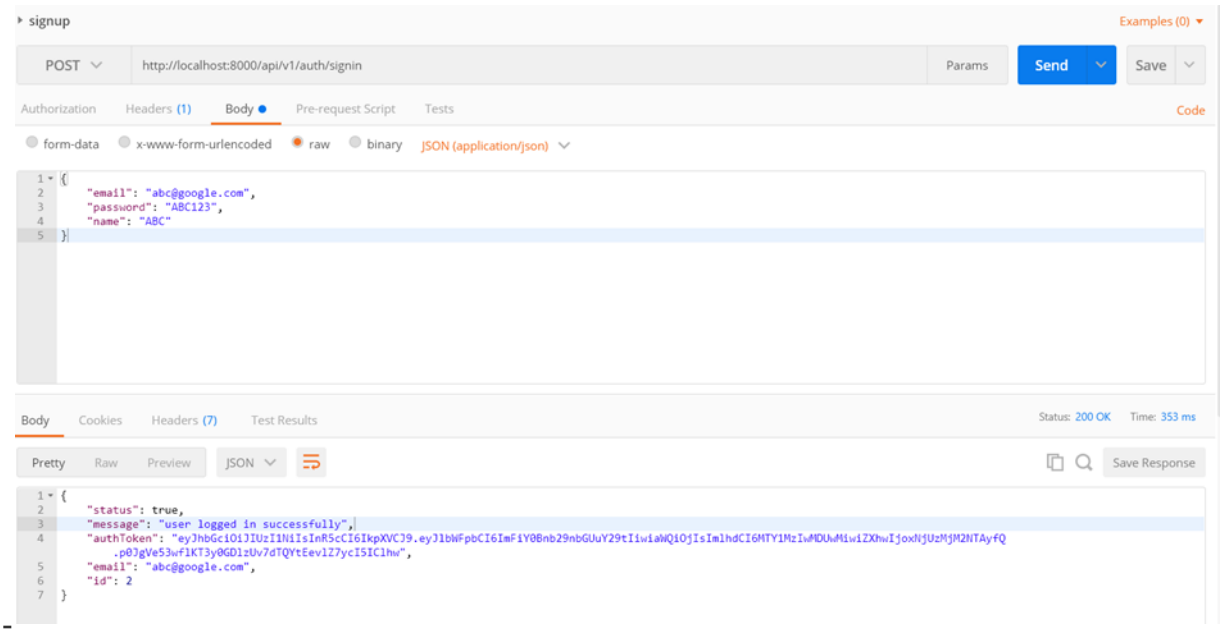
3) Create TODO with below fields
    a. id
    b. Todo title
    c. Is Completed
    d. Created At

4) Every request an Authorization Bearer token has to send in the header for Authorization.

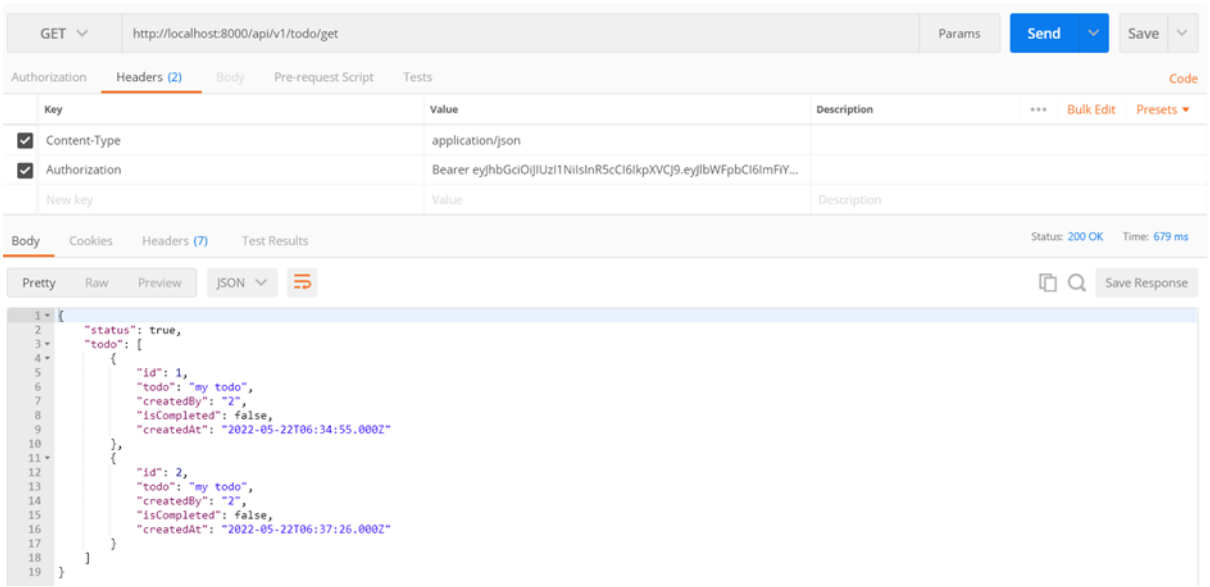
5) Update Todo with todo id and fields which need to update and it will allow only update the fields which the user had created.

6) Get Todo list which the particular user has created and apply filter in the request by using query string

Signin Postman:



Get TODO Postman:



Round 3 & 4 Solution:

https://github.com/Saravananslb/todo-api