QUESTIONS FOR JS [WEEKEND]:

TOTAL - 40

🔹 20 basic (Objects + Arrays)

1. **Sum values in object arrays**
   - Input:

```
{ food: [10, 20, 30], travel: [5, 15], bills: [40, 60] }
```

   - Output:

```
{ food: 60, travel: 20, bills: 100 }
```

2. **Count word occurrences in array**
   - Input:

```
["apple", "banana", "apple", "orange", "banana", "apple"]
```

   - Output:

```
{ apple: 3, banana: 2, orange: 1 }
```

3. **Swap keys and values of object**
   - Input:

```
{ a: "x", b: "y", c: "z" }
```

   - Output:

```
{ x: "a", y: "b", z: "c" }
```

4. **Find the largest value key**
   - Input:

```
{ a: 10, b: 50, c: 20 }
```

   - Output:

```
b
```

5. **Flatten object of arrays into one array**
   - Input:

```
{ fruits: ["apple", "banana"], veggies: ["carrot", "pea"] }
```

   - Output:

```
["apple", "banana", "carrot", "pea"]
```

6. **Group people by city**
   - Input:

```
[ { name: "A", city: "Delhi" }, { name: "B", city: "Mumbai" }, { na
me: "C", city: "Delhi" } ]
```

   - Output:

```
{ Delhi: ["A", "C"], Mumbai: ["B"] }
```

7. **Filter object by values > 50**
   - Input:

```
{ a: 20, b: 60, c: 40, d: 90 }
```

   - Output:

```
{ b: 60, d: 90 }
```

8. Find student with highest average mark
   - Input:

```
{ A: [80, 90], B: [70, 75, 85] }
```

   - Output:

```
A
```

9. **Unique values across all object arrays**
   - Input:

```
{ x: [1,2,3], y: [2,3,4], z: [4,5] }
```

   - Output:

```
[1,2,3,4,5]
```

10. **Pick only given keys from object**
    - Input:

```
{ name: "Rahul", age: 23, city: "Noida" }, ["name","city"]
```

    - Output:

```
{ name: "Rahul", city: "Noida" }
```

11. **Find student with highest average marks**
    - Input:

```
{ A: [80, 90], B: [70, 75, 85] }
```

    - Output:

```
A
```

12. **Sort object entries by values (ascending)**
    - Input:

```
{ a: 3, b: 1, c: 2 }
```

    - Output:

```
[["b",1], ["c",2], ["a",3]]
```

13. **Count number of keys in object**
    - Input:

```
{ a: 1, b: 2, c: 3 }
```

    - Output:

```
3
```

14. **Capitalize string values inside object**
    - Input:

```
{ name: "alice", city: "delhi" }
```

    - Output:

```
{ name: "Alice", city: "Delhi" }
```

15. **Convert object to query string**
    - Input:

```
{ name: "Alice", age: 25 }
```

    - Output:

```
"name=Alice&age=25"
```

16. **Count even and odd numbers in array**
    - Input:

```
[1,2,3,4,5,6]
```

    - Output:

```
{ even: 3, odd: 3 }
```

17. **Find common keys between two objects**
    - Input:

```
{ a: 1, b: 2, c: 3 }, { b: 4, c: 5, d: 6 }
```

    - Output:

```
["b","c"]
```

18. **Convert array of objects to lookup by id**
    - Input:

```
[{ id: 1, name: "A" }, { id: 2, name: "B" }]
```

    - Output:

```
{ 1: { id:1, name:"A" }, 2: { id:2, name:"B" } }
```

19. **Check if all values in object are numbers**
    - Input:

```
{ a: 1, b: "hello", c: 3 }
```

    - Output:

```
false
```

🔷 20 intermediate (Objects + Arrays)

1. **Sum all transactions per user**
   - Input:

```
[ { user: "A", amount: 100 }, { user: "B", amount: 200 }, { user: "A", amount: 50 } ]
```

   - Output:

```
{ A: 150, B: 200 }
```

2. **Transform API response to object (id → name)**
   - Input:

```
[ { id: 1, name: "Alice" }, { id: 2, name: "Bob" } ]
```

   - Output:

```
{ 1: "Alice", 2: "Bob" }
```

3. **Remove falsy values from object**
   - Input:

```
{ a: 0, b: null, c: "hello", d: undefined, e: 5 }
```

   - Output:

```
{ c: "hello", e: 5 }
```

4. Check for permissions from roles
    - Input:

```
roles={ admin:["read","write"], user:["read"], staff: ["write"]} ch
eckRole="user", action="write"
```

    - Output:

```
false
```

5. **Transform array of orders into revenue per category**
    - Input:

```
[ { id: 1, category: "electronics", price: 100 }, { id: 2, categor
y: "clothes", price: 50 }, { id: 3, category: "electronics", price:
200 } ]
```

    - Output:

```
{ electronics: 300, clothes: 50 }
```

6. **Remove duplicate objects by id**
    - Input:

```
[ { id: 1, name: "A" }, { id: 2, name: "B" }, { id: 1, name: "A" }
]
```

    - Output:

```
[ { id: 1, name: "A" }, { id: 2, name: "B" } ]
```

7. **Chunk object entries into groups of size**
    - Input:

```
{ a: 1, b: 2, c: 3, d: 4 }, size=2
```

    - Output:

```
[ [["a",1],["b",2]], [["c",3],["d",4]] ]
```

    1. **Find longest string among object values**
        - Input:

```
{ a: "apple", b: "banana", c: "kiwi" }
```

        - Output:

```
banana
```

8. Convert the object where **languages** are the top-level keys, and inside each are **translation strings by key into** an object where **translation keys** are the top-level keys, and inside each you store values per language [HARD**]
    - Input:

```
{ en: { hello: "Hello", bye: "Goodbye" }, fr: { hello: "Bonjour", b
ye: "Au revoir" }, es: { hello: "Hola" } }
```

    - Output:

```
{ hello: { en: "Hello", fr: "Bonjour", es: "Hola" }, bye: { en: "Go
odbye", fr: "Au revoir" } }
```

9. **Build index of ids grouped by category**
   - Input:

```
[ { id: 1, category: "fruit" }, { id: 2, category: "veggie" }, { id: 3, category: "fruit" } ]
```

   - Output:

```
{ fruit: [1,3], veggie: [2] }
```

10. **Remove deeply nested key from object**
    - Input:

```
{ a: { b: { c: 1, d: 2 } } }, remove "c"
```

    - Output:

```
{ a: { b: { d: 2 } } }
```

11. **Check if two objects are deeply equal**
    - Input:

```
{ a: { x: 1, y: 2 } }, { a: { x: 1, y: 2 } }
```

    - Output:

```
true
```

12. **Deep flatten nested arrays inside object**
    - Input:

```
{ a: [1, [2, [3]]], b: [4, [5]] }
```

    - Output:

```
{ a: [1,2,3], b: [4,5] }
```

13. **Find most repeated word across categories**
    - Input:

```
{ fruits: ["apple","apple","banana"], drinks: ["apple","tea"] }
```

    - Output:

```
apple
```

14. **Find intersection of all arrays in object**
    - Input:

```
{ a: [1,2,3], b: [2,3,4], c: [3,4,5] }
```

    - Output:

```
[3]
```

15. **Deep merge two nested objects**
    - Input:

```
{ a: { x: 1, y: 2 } } { a: { y: 3, z: 4 } }
```

    - Output:

```
{ a: { x: 1, y: 3, z: 4 } }
```

16. **Nested object destructuring**
    - Input:

```
{ user: { profile: { name: "Alice", age: 25 } } }
```

    - Output:

```
Alice 25
```

17. **Find top N keys by value**
    - Input:

```
{ a: 10, b: 50, c: 30, d: 40 }, N=2
```

    - Output: