### **JS EXES - basics**

#### **Section 1: Variable Swap Without Temp**

**Goal:** Swap the values of two variables a and b without using a third variable.  
**Twist:** The variables are strings.

example:

let a = “some string”;

let b = “other string”;

results should be:  
a = “other string”  
b = “some string”  
without using a third variable to achieve this.

#### **Section 2: 2D Grid Count**

**Goal:** Count how many "X" values appear in a 5x5 2D array.  
**Twist:** Ignore "x" (lowercase) and log coordinates of each "X".

#### **Section 3: Nested Object Read**

**Goal:** Access and log the third child’s name from a nested object:

let family = { parents: {}, children: [ {name: "Ali"}, {name: "Lea"}, {name: "Mona"} ] };

**Twist:** Also print all sibling names in one comma-separated string.

#### **Section 4: Manual Reverse**

**Goal:** Reverse an array manually using a loop.  
 **Twist:** Do not use .reverse() or .unshift().

#### **Section 5: Type Collector**

**Goal:** Given a mixed array, count how many of each type: number, string, object, boolean.  
 **Twist:** Also count how many values are null.

#### **Section 6: Filter Without .filter()**

**Goal:** From a number array, manually create a new array of numbers > 100.  
 **Twist:** Track how many comparisons were made.

#### **Section 7: Conditional Object Filler**

**Goal:** Create an object with keys "even" and "odd", and push numbers 1–50 into the correct array.  
 **Twist:** Count how many are divisible by both 2 and 3.

#### **Section 8: Grid Checker**

**Goal:** Create a 3x3 grid using nested arrays and mark all diagonal values with "#".  
 **Twist:** Mark both diagonals (top-left to bottom-right and vice versa).

#### **Section 9: Frequency Counter**

**Goal:** Given a string, count how many times each letter appears using an object.  
 **Twist:** Ignore spaces and punctuation, and make it case-insensitive.

#### **Section 10: Flattening Matrix**

**Goal:** Flatten a 2D number array into one 1D array.  
 **Twist:** Remove all negative numbers during the flattening.

#### **Section 11: Object Key Transformer**

**Goal:** Take an object and produce a new object where all keys are uppercase.  
 **Twist:** Also sort keys alphabetically before printing.

#### **Section 12: Triangular Loop Output**

**Goal:** Use nested loops to print a triangle made of "\*" like this:

\*

\* \*

\* \* \*

**Twist:** Use variables to control size and symbol.

#### **Section 13: Match Two Arrays**

**Goal:** Print how many elements exist in both arrays A and B.  
 **Twist:** Arrays can have repeated values, but count matches once.

#### **Section 14: Key-Value Reverser**

**Goal:** Create a new object where the values become keys and the keys become values.  
 **Twist:** Only reverse if the value is a string or number.

#### **Section 15: Identity Grid**

**Goal:** Create a 5x5 array where cells with row === column contain 1, others contain 0.  
 **Twist:** Mark row 2 fully with 9.

#### **Section 16: Object Conditions**

**Goal:** Given a list of user objects, print all users who are "active" and over 18.  
 **Twist:** Also count how many inactive users are under 18.

#### **Section 17: Manual Min/Max**

**Goal:** Find min and max values in an array using a loop.  
 **Twist:** Do not use Math.min or Math.max.

#### **Section 18: Rotating Array**

**Goal:** Rotate an array left by 1 step:

[1, 2, 3] → [2, 3, 1]

**Twist:** Repeat this 3 times in a loop.

#### **Section 19: Manual JSON Print**

**Goal:** Given an object, manually create a string that looks like JSON:

{ "name": "Ali", "age": 25 }

(add “ “ to each key)

**Twist:** No use of JSON.stringify.

#### **Section 20: Complex Board Validator**

**Goal:** Validate that a 4x4 game board has at most two "O"s in each row and column.  
 **Twist:** Print a message if the board violates the rule.