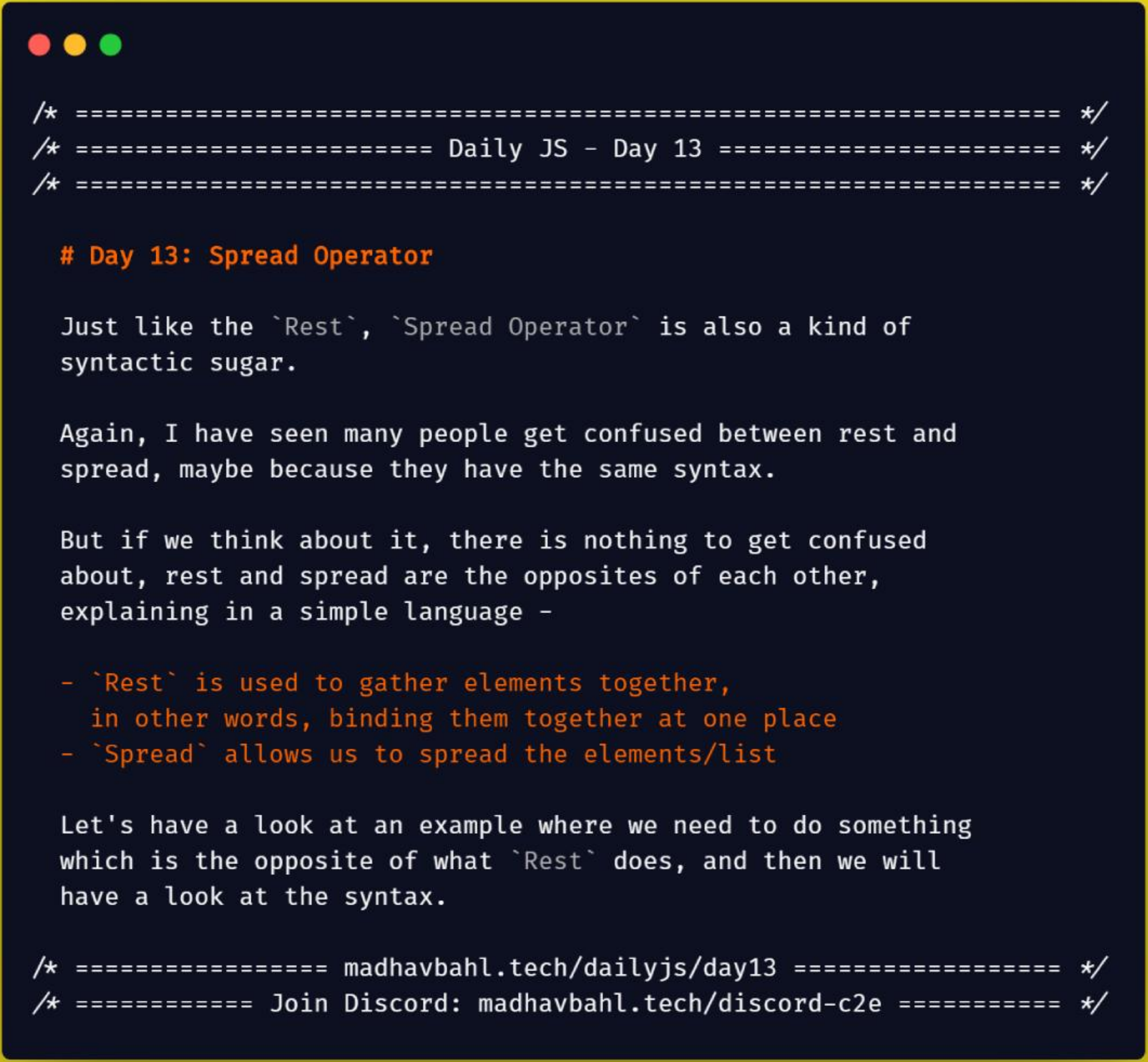
THE LEAN PROGRAMMER | Code To Express

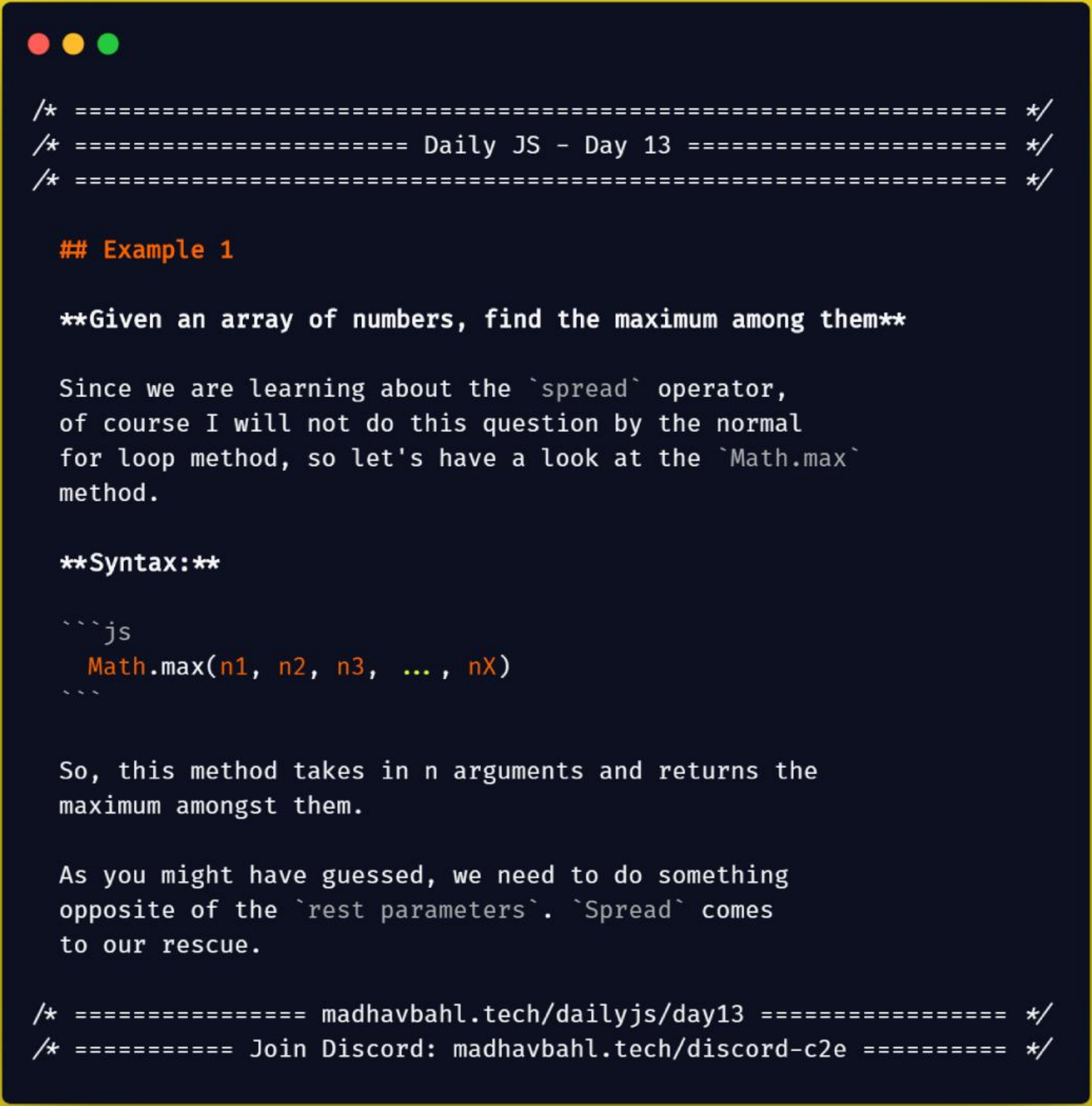
Day 13, Today's Topic

Spread Operators

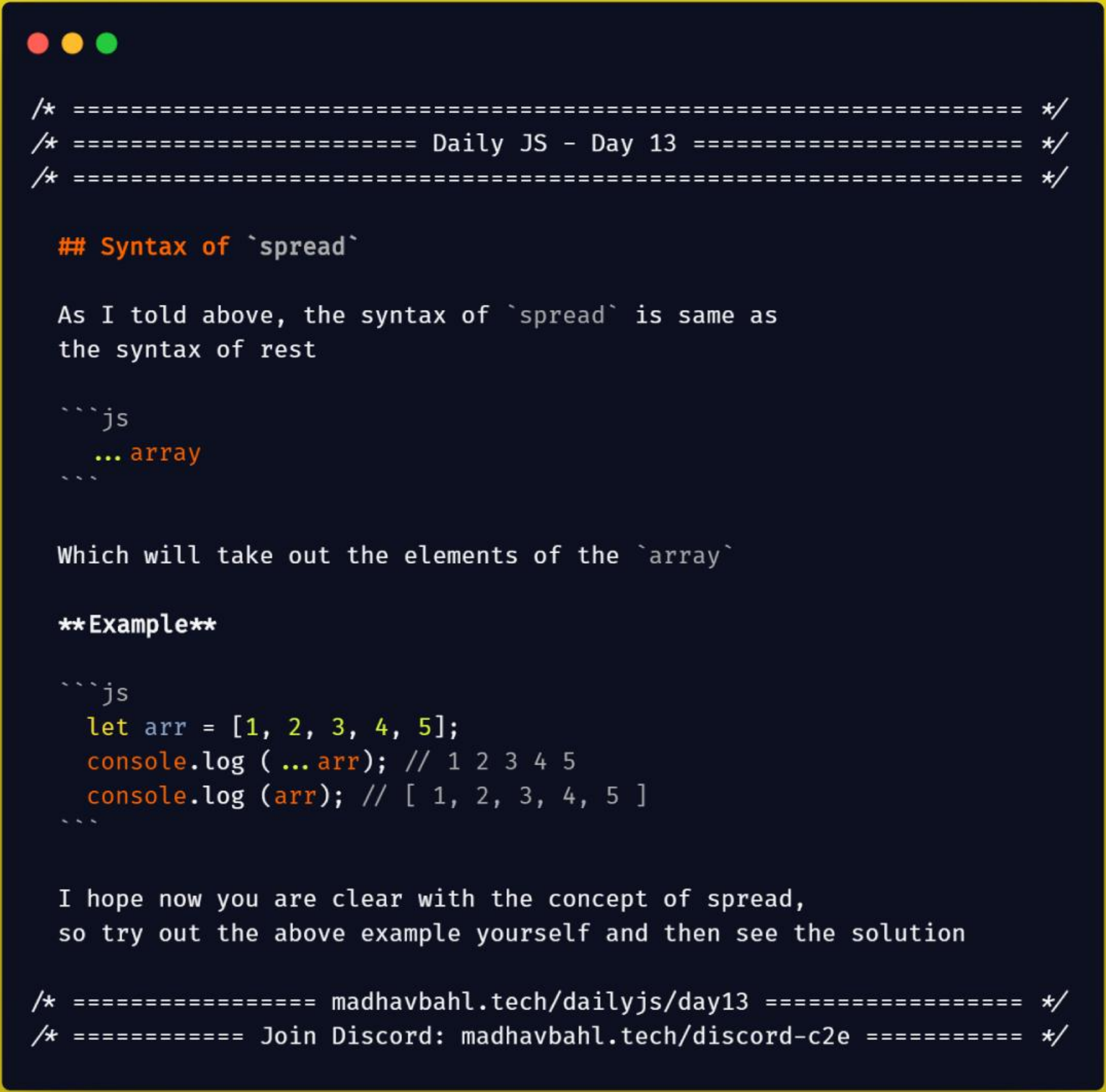
DAILY JS



```
/* ================================================================ */
/* ==================== Daily JS - Day 13 ===================== */
/* ================================================================ */

  # Day 13: Spread Operator

  Just like the `Rest`, `Spread Operator` is also a kind of
  syntactic sugar.

  Again, I have seen many people get confused between rest and
  spread, maybe because they have the same syntax.

  But if we think about it, there is nothing to get confused
  about, rest and spread are the opposites of each other,
  explaining in a simple language -

  - `Rest` is used to gather elements together,
    in other words, binding them together at one place
  - `Spread` allows us to spread the elements/list

  Let's have a look at an example where we need to do something
  which is the opposite of what `Rest` does, and then we will
  have a look at the syntax.

/* ================ madhavbahl.tech/dailyjs/day13 ================ */
/* =========== Join Discord: madhavbahl.tech/discord-c2e =========== */
```



~~~
/* ============================================================= */
/* ===================== Daily JS - Day 13 ===================== */
/* ============================================================= */

  ## Example 1

  **Given an array of numbers, find the maximum among them**

  Since we are learning about the `spread` operator,
  of course I will not do this question by the normal
  for loop method, so let's have a look at the `Math.max`
  method.

  **Syntax:**

  ```js
    Math.max(n1, n2, n3,  ... , nX)
  ```

  So, this method takes in n arguments and returns the
  maximum amongst them.

  As you might have guessed, we need to do something
  opposite of the `rest parameters`. `Spread` comes
  to our rescue.

/* =============== madhavbahl.tech/dailyjs/day13 =============== */
/* ========== Join Discord: madhavbahl.tech/discord-c2e ========== */
~~~



~~~
/* ================================================================== */
/* ======================== Daily JS - Day 13 ======================== */
/* ================================================================== */

  ## Syntax of `spread`

  As I told above, the syntax of `spread` is same as
  the syntax of rest

  ```js
    ...array
  ~~~

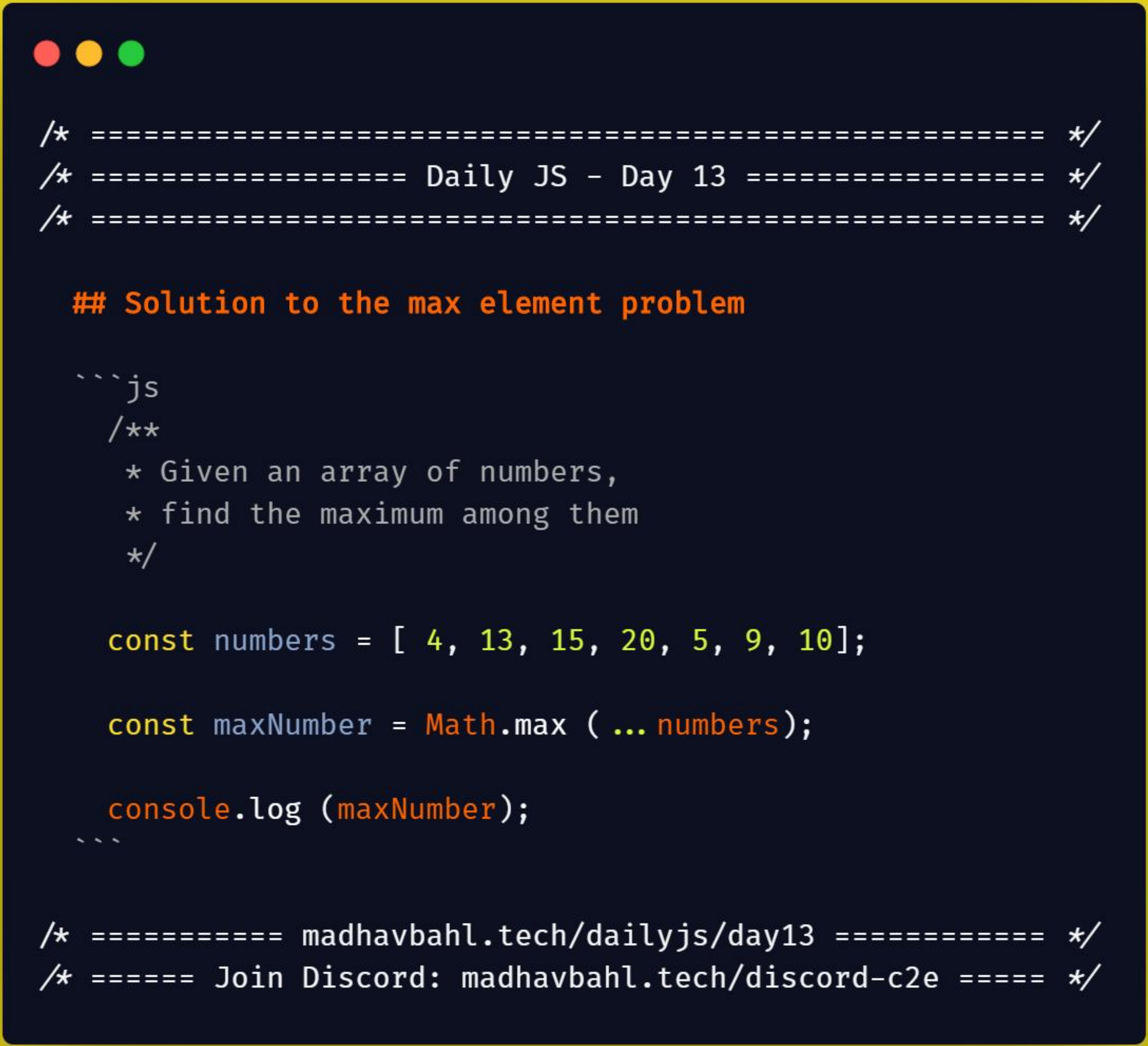
  Which will take out the elements of the `array`

  Example

  ```js
   let arr = [1, 2, 3, 4, 5];
   console.log (...arr); // 1 2 3 4 5
   console.log (arr); // [ 1, 2, 3, 4, 5 ]
  ```

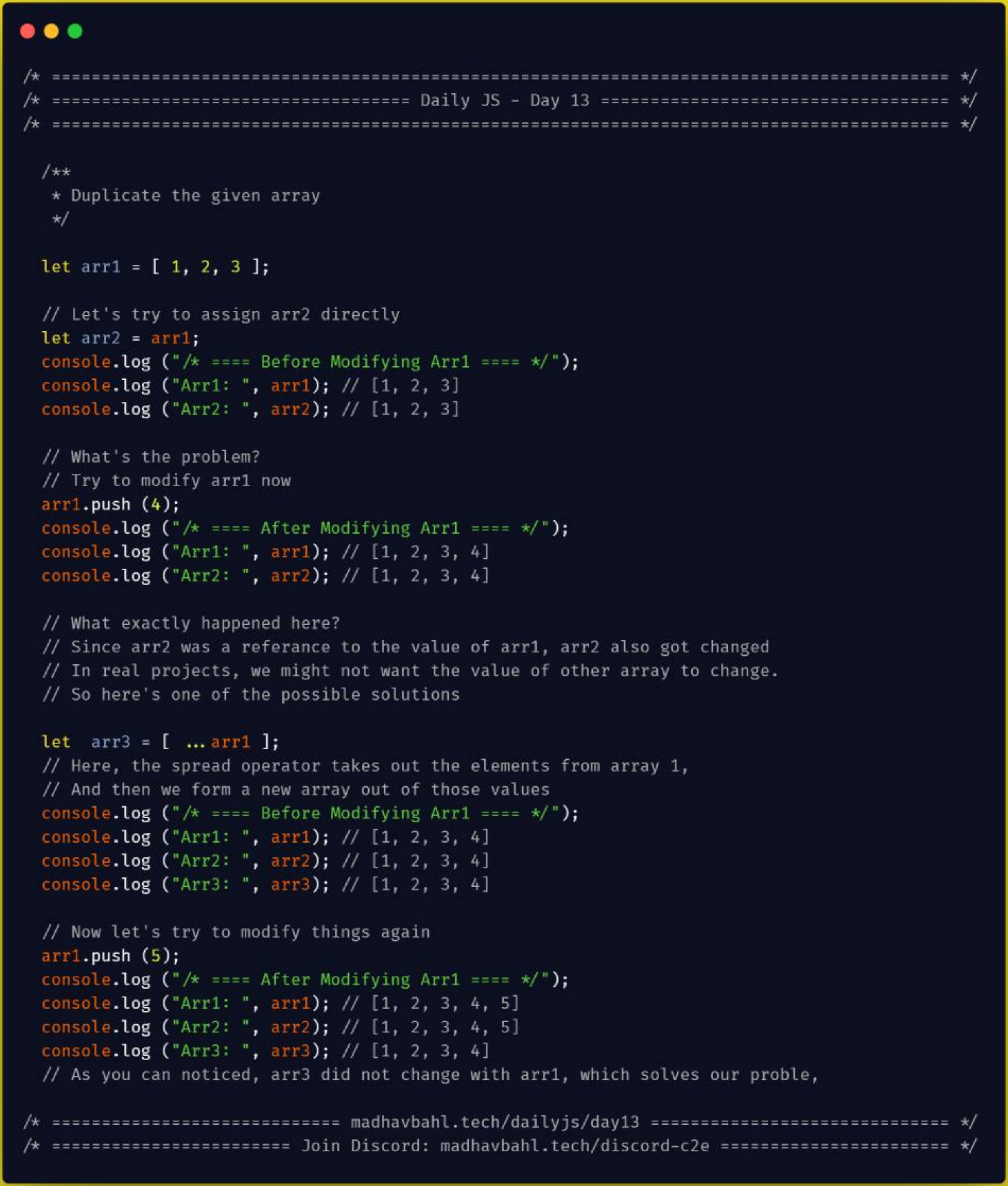
  I hope now you are clear with the concept of spread,
  so try out the above example yourself and then see the solution

/* ================= madhavbahl.tech/dailyjs/day13 ================= */
/* =========== Join Discord: madhavbahl.tech/discord-c2e =========== */
```
```

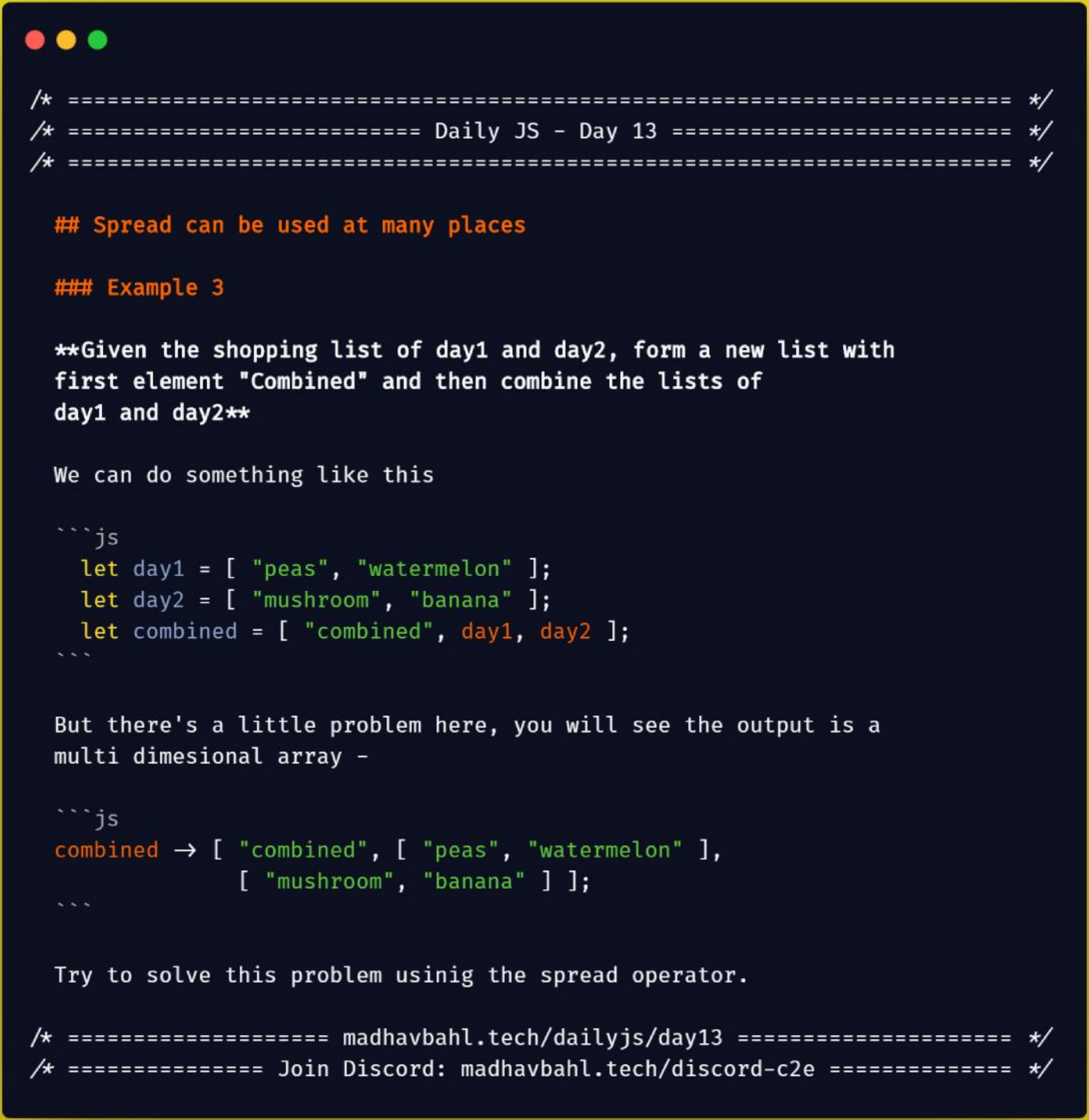


~~~
/* ============================================================ */
/* ================== Daily JS - Day 13 ================== */
/* ============================================================ */

  ## Solution to the max element problem

  ```js
    /**
     * Given an array of numbers,
     * find the maximum among them
     */

    const numbers = [ 4, 13, 15, 20, 5, 9, 10];

    const maxNumber = Math.max ( ... numbers);

    console.log (maxNumber);
  ```

/* =========== madhavbahl.tech/dailyjs/day13 ============ */
/* ====== Join Discord: madhavbahl.tech/discord-c2e ===== */
~~~

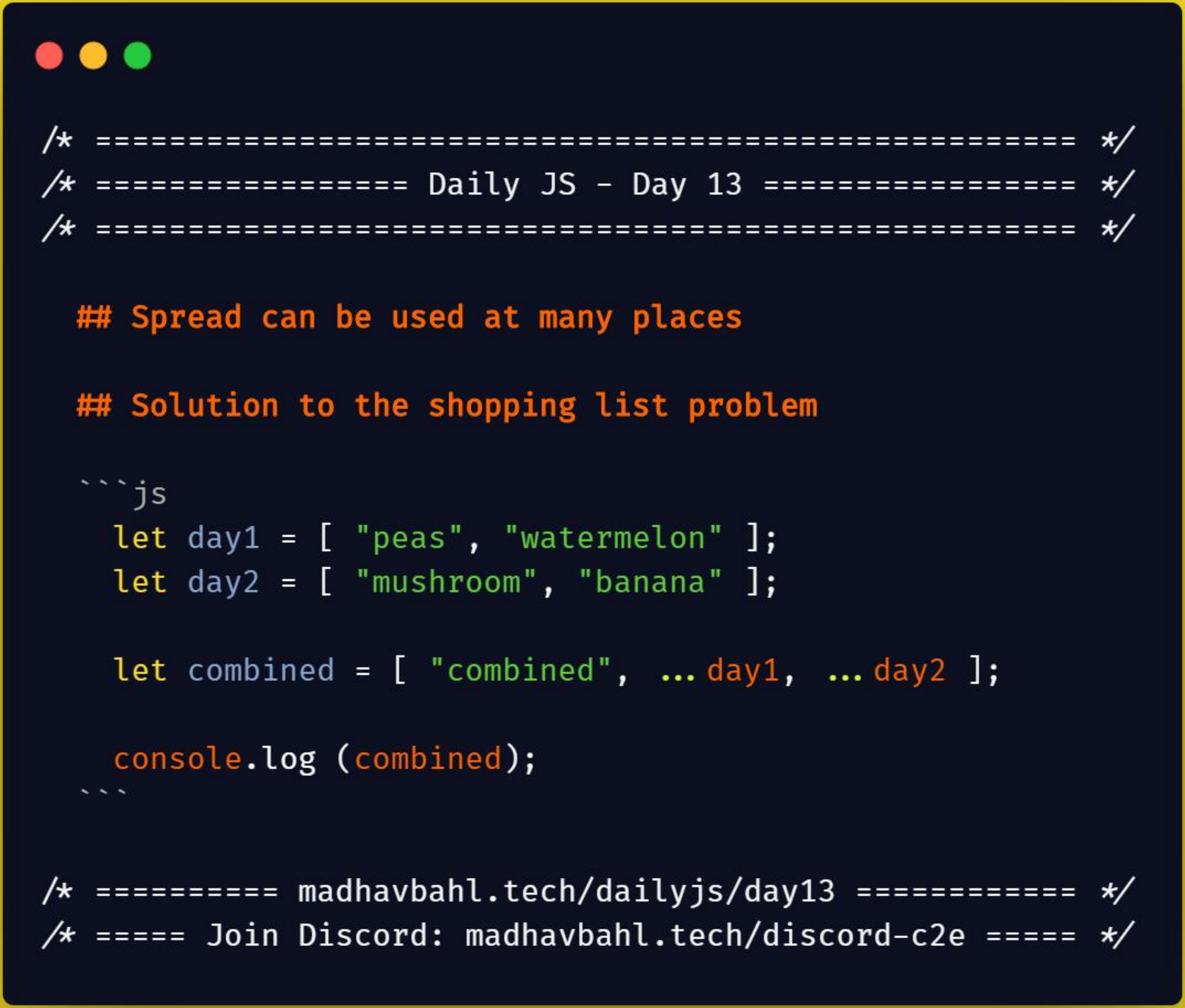
```
/* ============================================================ */
/* =================== Daily JS - Day 13 =================== */
/* ============================================================ */

  ## Spread can be used at many places

  ### Example 2

  **Duplicate the given array**

  To duplicate a given array, many people directly do
  something like this

  ```
    var arr2 = arr1;
  ```

  This can cause some trouble since it isn't actually
  duplicating your array, but rather it creates a
  referance to arr1. Let's see what happens -

/* =========== madhavbahl.tech/dailyjs/day13 ============ */
/* ====== Join Discord: madhavbahl.tech/discord-c2e ===== */
```



```
/* ========================================================================================== */
/* ==================================== Daily JS - Day 13 =================================== */
/* ========================================================================================== */

  /**
   * Duplicate the given array
   */

  let arr1 = [ 1, 2, 3 ];

  // Let's try to assign arr2 directly
  let arr2 = arr1;
  console.log ("/* ==== Before Modifying Arr1 ==== */");
  console.log ("Arr1: ", arr1); // [1, 2, 3]
  console.log ("Arr2: ", arr2); // [1, 2, 3]

  // What's the problem?
  // Try to modify arr1 now
  arr1.push (4);
  console.log ("/* ==== After Modifying Arr1 ==== */");
  console.log ("Arr1: ", arr1); // [1, 2, 3, 4]
  console.log ("Arr2: ", arr2); // [1, 2, 3, 4]

  // What exactly happened here?
  // Since arr2 was a referance to the value of arr1, arr2 also got changed
  // In real projects, we might not want the value of other array to change.
  // So here's one of the possible solutions

  let  arr3 = [ ...arr1 ];
  // Here, the spread operator takes out the elements from array 1,
  // And then we form a new array out of those values
  console.log ("/* ==== Before Modifying Arr1 ==== */");
  console.log ("Arr1: ", arr1); // [1, 2, 3, 4]
  console.log ("Arr2: ", arr2); // [1, 2, 3, 4]
  console.log ("Arr3: ", arr3); // [1, 2, 3, 4]

  // Now let's try to modify things again
  arr1.push (5);
  console.log ("/* ==== After Modifying Arr1 ==== */");
  console.log ("Arr1: ", arr1); // [1, 2, 3, 4, 5]
  console.log ("Arr2: ", arr2); // [1, 2, 3, 4, 5]
  console.log ("Arr3: ", arr3); // [1, 2, 3, 4]
  // As you can noticed, arr3 did not change with arr1, which solves our proble,

/* ============================= madhavbahl.tech/dailyjs/day13 ============================== */
/* ======================== Join Discord: madhavbahl.tech/discord-c2e ======================= */
```



~~~
/* ================================================================= */
/* ======================== Daily JS - Day 13 ======================== */
/* ================================================================= */

  ## Spread can be used at many places

  ### Example 3

  **Given the shopping list of day1 and day2, form a new list with
  first element "Combined" and then combine the lists of
  day1 and day2**

  We can do something like this

  ```js
    let day1 = [ "peas", "watermelon" ];
    let day2 = [ "mushroom", "banana" ];
    let combined = [ "combined", day1, day2 ];
  ```

  But there's a little problem here, you will see the output is a
  multi dimesional array -

  ```js
  combined → [ "combined", [ "peas", "watermelon" ],
               [ "mushroom", "banana" ] ];
  ```

  Try to solve this problem usinig the spread operator.

/* ================== madhavbahl.tech/dailyjs/day13 =================== */
/* ============== Join Discord: madhavbahl.tech/discord-c2e ============= */
~~~



~~~
/* ============================================================ */
/* ================= Daily JS - Day 13 ================= */
/* ============================================================ */

  ## Spread can be used at many places

  ## Solution to the shopping list problem

  ```js
    let day1 = [ "peas", "watermelon" ];
    let day2 = [ "mushroom", "banana" ];

    let combined = [ "combined", ...day1, ...day2 ];

    console.log (combined);
  ```

/* ========== madhavbahl.tech/dailyjs/day13 ============= */
/* ===== Join Discord: madhavbahl.tech/discord-c2e ===== */
~~~

THE LEAN PROGRAMMER | CodeTo Express

Thank you!

Feel free to reach out...

Email: theleanprogrammer@gmail.com
Web: madhavbahl.tech/
Github: github.com/MadhavBahlMD
LinkedIn: linkedin.com/in/madhavbahl/
Insta: instagram.com/theleanprogrammer/

DAILY JS