TW-04 TEAM LEAD VERSION

Meeting Agenda

- Icebreaking
- Workshop Activities - Tuesday
- Teamwork Activities - Friday
  - Questions
  - Interview Questions
- Video of the week
- Retro meeting
- Case study / project

Teamwork Schedule

Ice-breaking 10m

- Personal Questions (Study Environment, Kids etc.)
- Any challenges (Classes, Coding, studying, etc.)
- Ask how they're studying, give personal advice.
- Remind that practice makes perfect.

Workshop Activities (Tuesday) 90m

1.Anagram Check

Anagrams are words or phrases formed by rearranging the letters of another, such as `listen` and `silent` or `triangle` and `integral`.

Write a JavaScript function called areAnagrams that takes two strings as input and returns true if the two strings are anagrams of each other, and false otherwise.

- Your function should be case-insensitive and ignore spaces and punctuation.
- It should return true if the input strings have the same set of characters (ignoring order), and false otherwise.

Some anagram examples:

```
console.log(areAnagrams("listen", "silent")); // Should print true
console.log(areAnagrams("triangle", "integral")); // Should print true
console.log(areAnagrams("debit card", "bad credit")); // Should print true
console.log(areAnagrams("Dormitory", "dirty room")); // Should print true
console.log(areAnagrams("The Morse Code", "Here come dots")); // Should print true
console.log(areAnagrams("Astronomer", "Moon starer")); // Should print true

console.log(areAnagrams("hello", "world")); // Should print false
console.log(areAnagrams("apple", "banana")); // Should print false
console.log(areAnagrams("hello", "holla")); // Should print false
console.log(areAnagrams("race", "racing")); // Should print false
```

Solution :

```
function areAnagrams(str1, str2) {
    // Remove spaces and convert to lowercase
    const cleanedStr1 = cleanAndNormalize(str1);
    const cleanedStr2 = cleanAndNormalize(str2);

    // Check if lengths are different
    if (cleanedStr1.length !== cleanedStr2.length) {
      return false;
    }

    // Sort the characters in the cleaned strings
    const sortedStr1 = sortString(cleanedStr1);
    const sortedStr2 = sortString(cleanedStr2);

    // Compare the sorted strings
    return sortedStr1 === sortedStr2;
  }

  function cleanAndNormalize(str) {
    let cleanedStr = '';
    for (let i = 0; i < str.length; i++) {
      const char = str[i].toLowerCase();
      if (char !== '\t' && char !== '\n' && char !== '\r') {
        cleanedStr += char;
      }
    }
    return cleanedStr;
  }

  function sortString(str) {
    return str.split('').sort().join('');
  }


  // Example usage:
  const input1 = "listen";
  const input2 = "silent";
  console.log(areAnagrams(input1, input2)); // Should print true

  console.log(areAnagrams("debit card", "bad credit")); // Should print true
console.log(areAnagrams("Dormitory", "dirty room")); // Should print true
console.log(areAnagrams("The Morse  Code!", "Here come dots!")); // Should print
true


/*
  Here's how the solution works:

The cleanAndNormalize function converts the input strings to lowercase and removes
spaces.

The sortString function sorts the characters in the cleaned strings.

The areAnagrams function checks if the lengths of the cleaned and sorted strings
are the same and then compares them.
*/
```

2. Write a js code that will uniqe the elements of the array containing repetitive elements.

```javascript
function removeDuplicates(arr) {
  const uniqueArray = [];
  for (const item of arr) {
    if (!uniqueArray.includes(item)) {
      uniqueArray.push(item);
    }
  }
  return uniqueArray;
}

const arrayWithDuplicates = [1, 2, 2, 3, 4, 4, 5];
const uniqueArray = removeDuplicates(arrayWithDuplicates);
// uniqueArray will be [1, 2, 3, 4, 5]
```

3. Write a js code that find the intersection of two JavaScript arrays (common elements in both arrays)

```javascript
function findIntersection(arr1, arr2) {
  const intersection = [];

  for (let i = 0; i < arr1.length; i++) {
    for (let j = 0; j < arr2.length; j++) {
      if (arr1[i] === arr2[j]) {
        intersection.push(arr1[i]);
        break; // Break to avoid duplicate entries
      }
    }
  }

  return intersection;
}

const array1 = [1, 2, 3, 4, 5];
const array2 = [3, 4, 5, 6, 7];

const result = findIntersection(array1, array2);
console.log(result); // [3, 4, 5]
```

Team Work Activities (Friday)

Ask Questions 20m

1. After the following code, what is the value of a.length?

```
var a = ['dog','lion','hen'];
a[100] = 'horse';
```

A. 101
B. 3
C. 4
D. 100

Answer : A

2. What will be the output of this code?

```
let array = [1, 2, 3];
array[6] = 9;
console.log(array[5]);
```

A. 1
B. 2
C. undefined
D. 5

Answer : C

3. Which definition below best describe an array?

A. An array is a function identifier that can hold more than one parameter.
B. An array is a beam of light.
C. An array is a special variable, which can hold more than one value at a time.
D. An array is a special function, which can hold more than one value at a time.

Answer : C

4. Which method is used to add one or more elements to the end of an array?

A. unshift()
B. append()
C. addToEnd()
D. push()

Answer : D

5. What is the purpose of the pop() method in JavaScript?

A. It removes the first element of an array.
B. It removes the last element of an array.br> **C.** It adds an element to the beginning of an array.
D. It reverses the order of elements in an array.

Answer : B

6. Which method is used to remove elements from the beginning of an array?

A. shift()
B. pop()
C. splice()
D. remove()

Answer : A

7. Which method is used to combine two or more arrays in JavaScript?

A. combine()
B. merge()
C. join()
D. concat()

Answer : D

8. Which method is used to remove elements from an array without creating gaps in the array?

A. splice()
B. slice()
C. filter()
D. remove()

Answer : A

9. What does the reverse() method do in JavaScript?

A. Sorts the array in ascending order
B. Sorts the array in descending order
C. Reverses the order of elements in the array
D. Removes all elements from the array

Answer : C

10. Which method is used to create a shallow copy of an array in JavaScript?

A. copy()
B. slice()
C. clone()
D. duplicate()

Answer : B

11. Which method is used to sort the elements of an array in place, without creating a new array?

A. arrange()
B. order()
C. sorted()
D. sort()

Answer : D

12. Write a js code that can reach the result with splice command

```
let myArray = [1, 2, 3, 4, 5];

myArray.splice(2, 1, 6);

console.log(myArray); // [1, 2, 6, 4, 5]
```

13. Which array iteration method is used to execute a provided function for each array element and has no return value?

A. forEach()
B. map()
C. filter()
D. reduce()

Answer : A

14. Write js code that transfers the even numbers in the array to a new array using the filter() method

```
const numbers = [0, 1, 2, 3, 4, 5];
const evenNumbers = numbers.filter((num) => num % 2 === 0);
console.log(evenNumbers) // [0, 2, 4]
```

15.Write js code computes the sum of all numbers in the array using reduce() method.

```
const numbers = [1, 2, 3, 4, 5];
const sum = numbers.reduce((accumulator, currentValue) => accumulator +
currentValue, 0);
console.log(sum) //15
```

16. How can you create a responsive grid layout in CSS Grid?

A. By setting fixed row and column sizes in pixels.
B. By defining a static number of rows and columns.
C. By using media queries to adjust grid properties at different screen sizes.
D. By setting a fixed grid container width.

Answer : C

17. What does CSS Grid Layout provide for web developers?

A. A way to create only one-dimensional layouts
B. A method to style text and fonts
C. An animation framework
D. A powerful two-dimensional layout system

Answer : D

18. What CSS property sets the space between grid rows and columns in CSS Grid?

A. grid-cell-gap
B. grid-spacing
C. grid-rows-columns-gap
D. grid-gap

Answer : D

19. In CSS Grid, which property can be used to control the placement of a grid item within the grid container?

A. grid-item-placement
B. grid-area
C. grid-cell
D. grid-position

Answer : B

20. Which property is used to define the number and size of grid columns in CSS Grid?

A. grid-template-columns
B. grid-gap
C. grid-column
D. grid-row

Answer : A

Interview Questions 20m

1. Explain `reduce()` method in Javascript

Answer : .reduce() runs a callback for every array element just like .map() does. The only difference is that reduce() passes the result of this accumulator from one array element to the other. Some built-in reduce() functions are: Array.prototype.reduce(), and the reduceRight() method which are used to apply functions against accumulators from 1. Left to right and 2. Right to left respectively.

The accumulator either contains the initial value or the return value from the previous call. The accumulator could be any string, integer, object, etc. It is the net result of the function. The present value of the accumulator is simply the element that is being worked against.

Accumulators must be passed when .reduce() is being called.

2. What is the difference between slice and splice?

Answer:

Some of the major difference in a tabular form

Slice	Splice
Doesn't modify the original array(immutable)	Modifies the original array(mutable)
Returns the subset of original array	Returns the deleted elements as array
Used to pick the elements from array	Used to insert or delete elements to/from array

3. What is the difference between .map() and .forEach()?

Answer :

.map() and .forEach() are both array methods that allow you to loop through an array, but they differ in what they return.

```
 .map() returns a new array with the same length as the original array, where each
element is the result of applying a callback function to the original element.
 .forEach() does not return anything, but it simply executes a callback function on
each element of the array.
```

Example:

```
const numbers = [1, 2, 3, 4, 5];

const doubledNumbers = numbers.map(num => num * 2);

console.log(doubledNumbers); // [2, 4, 6, 8, 10]

numbers.forEach(num => console.log(num * 2)); // 2, 4, 6, 8, 10
```

☕

Coffee Break 10m

☕

Video of the Week 15m

- forEach() method

Case study/Project 15m

The case study will be solved by the students during the week and by the team on Friday Team Work.

- CSS Grid

- iOS Calculator

Retro Meeting on a personal and team level 10m

Ask the questions below:

- What went well?
- What could be improved?
- What will we commit to do better in the next week?

Closing 5m

- Next week's plan

- QA Session
