Slide Link: <https://cs.slides.com/colt_steele/big-o-notation>

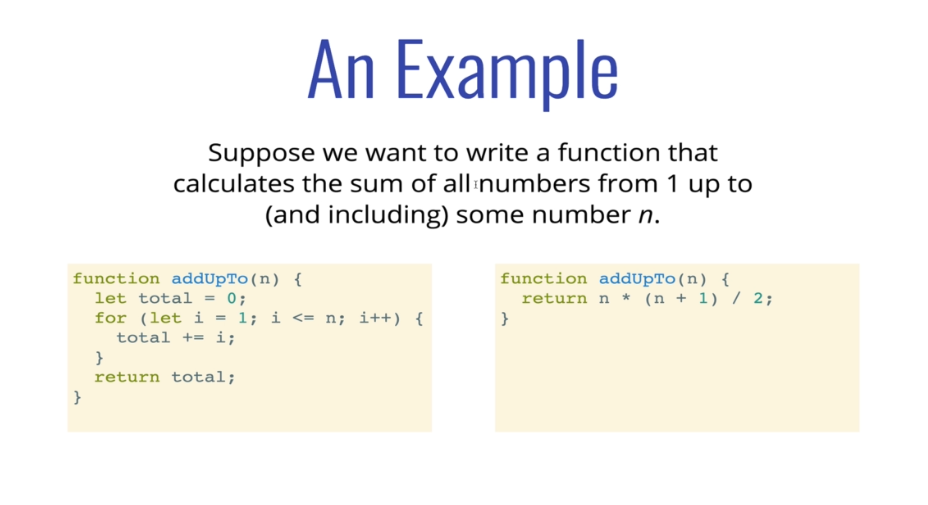
<https://cs.slides.com/colt_steele/built-in-data-structures-25>

<https://cs.slides.com/colt_steele/problem-solving-patterns>

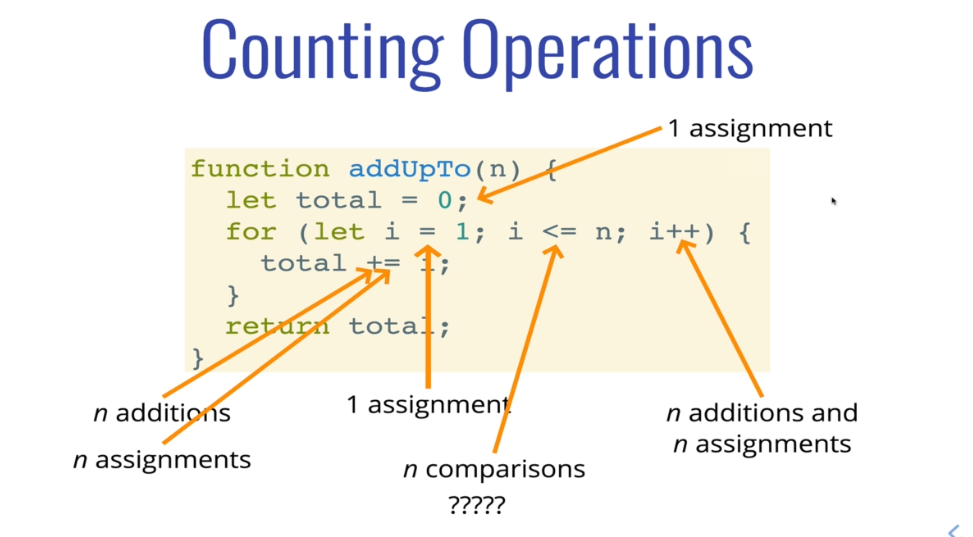
Performance measure korer jonno….Amra number niya care kori na care kori trend niya mane function ta koto tuko time nilo.

Iterate korle loop korle time bashi lage.

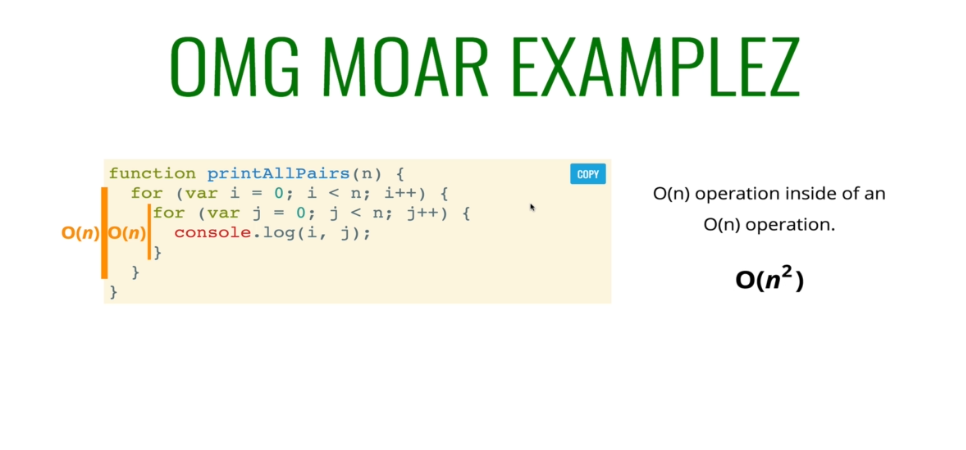
DataStructure and Algorith with Javascript: Big (n)



Counting Operation:

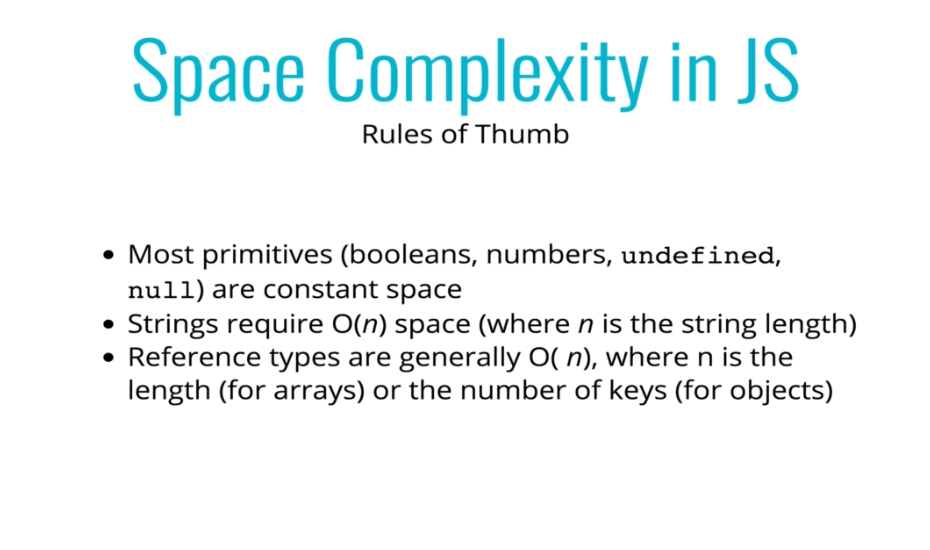


Here 5n+2.

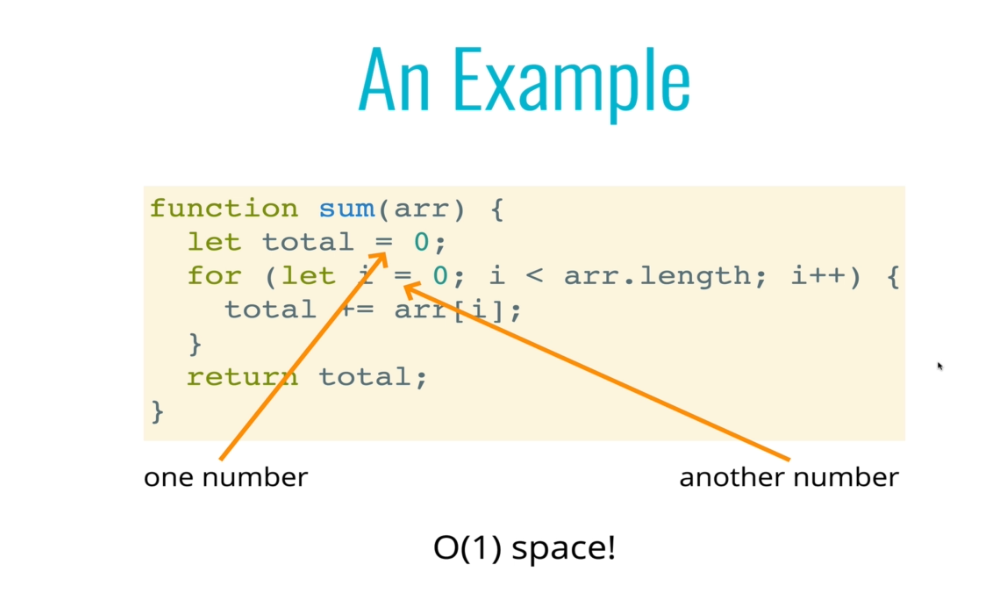


0(n\*n) = 0n2

Space Complexity:

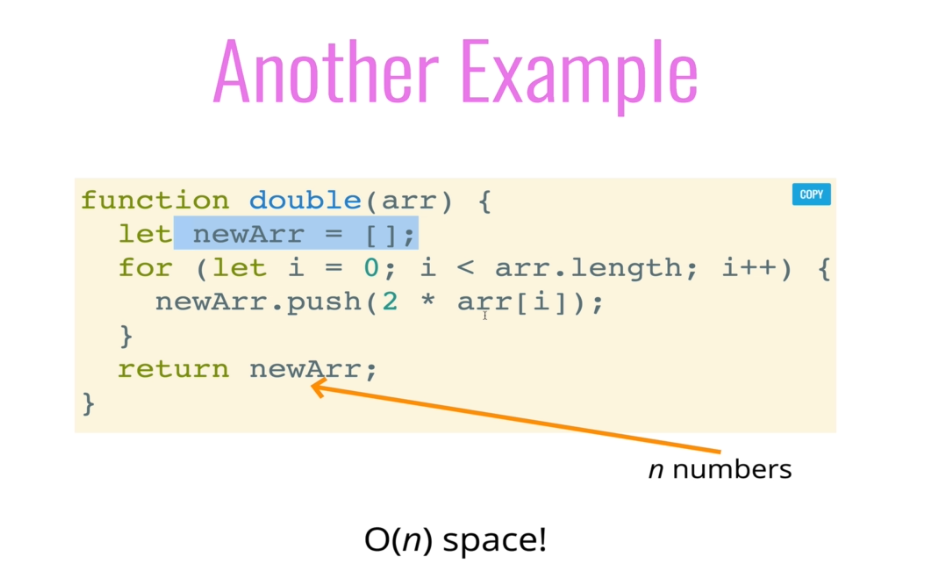


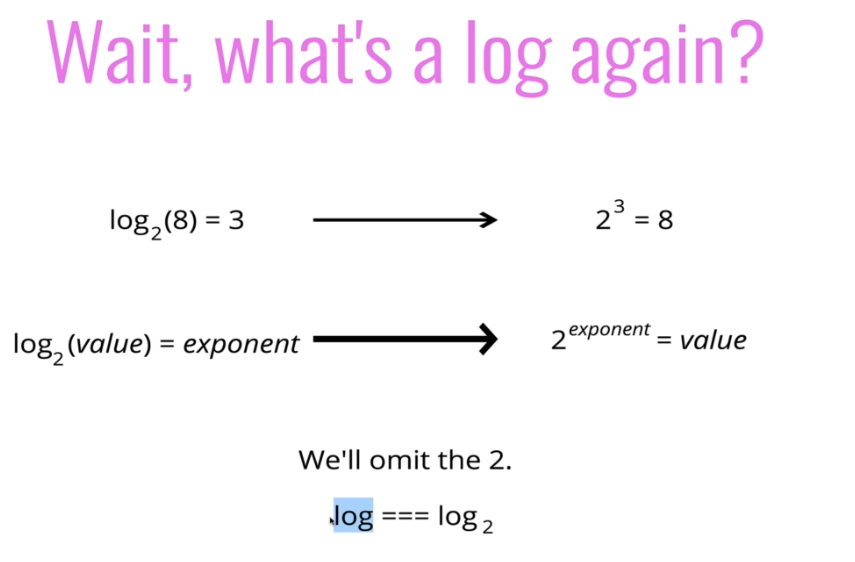
Space count:

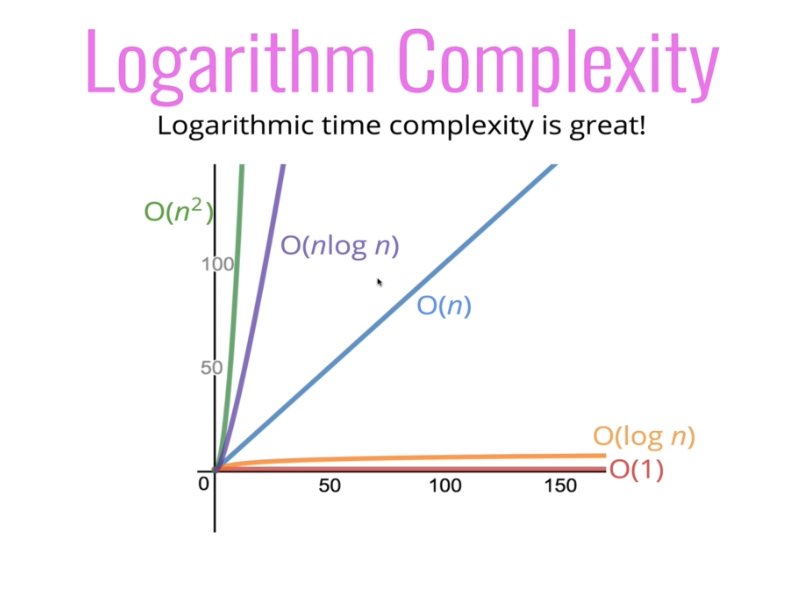


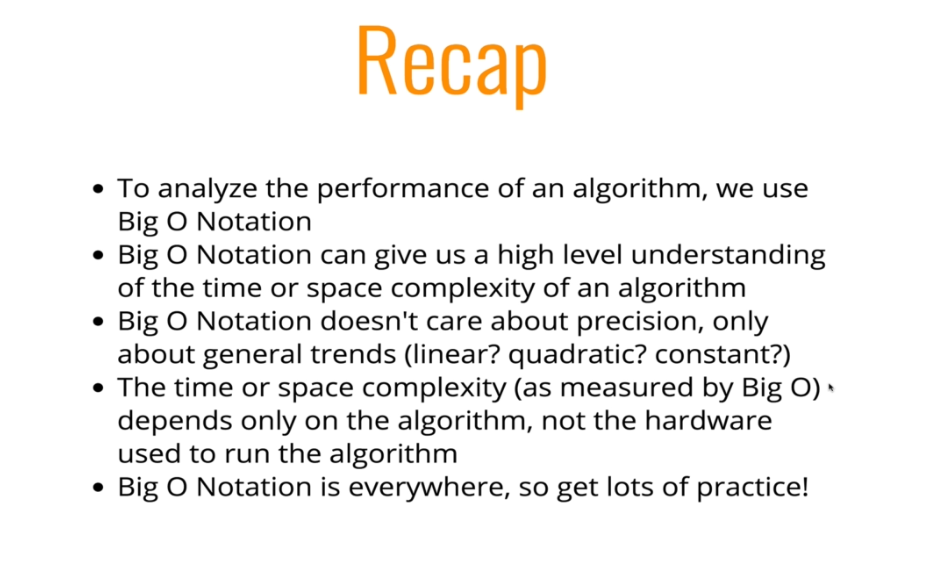
Here 2 constant space. Ai 2 ta e fix r barbe na. memory te jaiga nise.

Or other example:

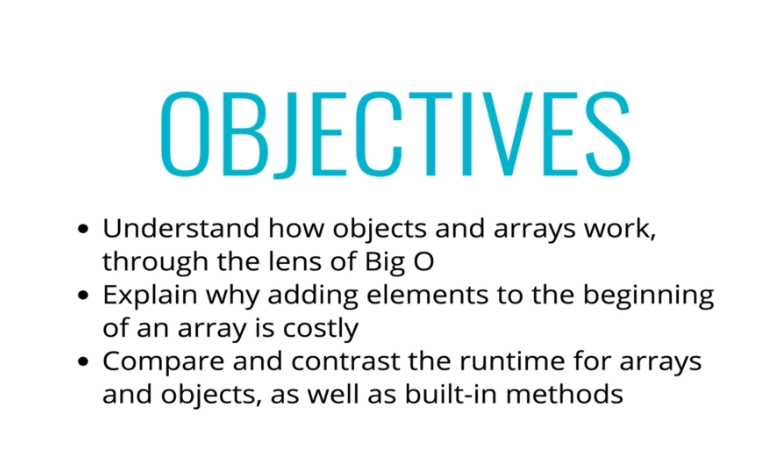


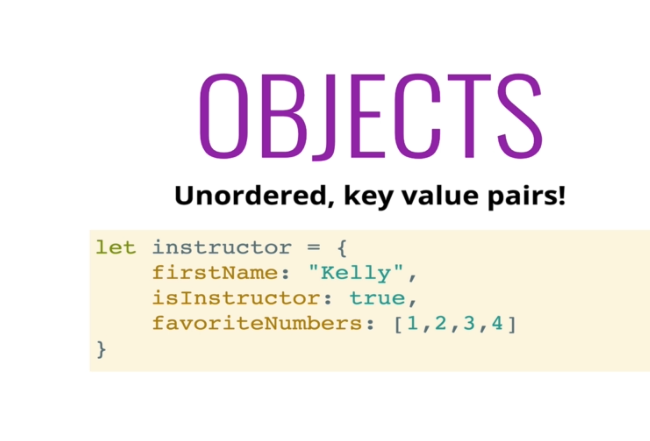






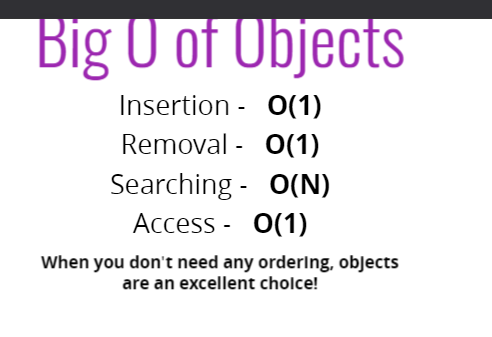
Anayze performance of array & object:

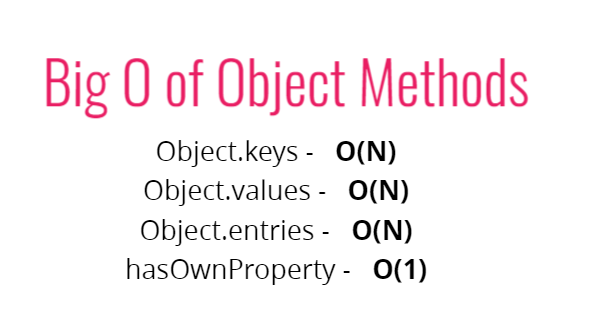




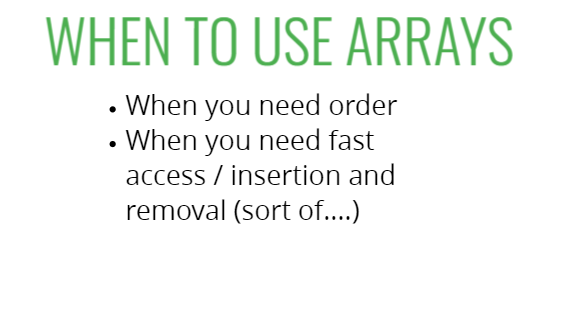


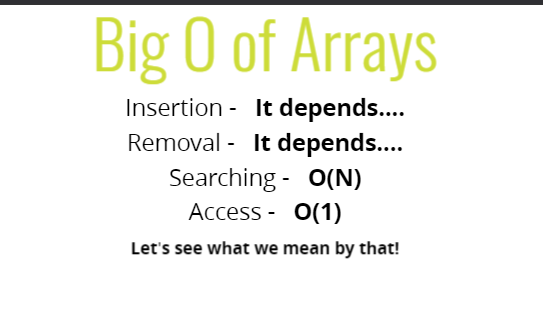
Object better from array.

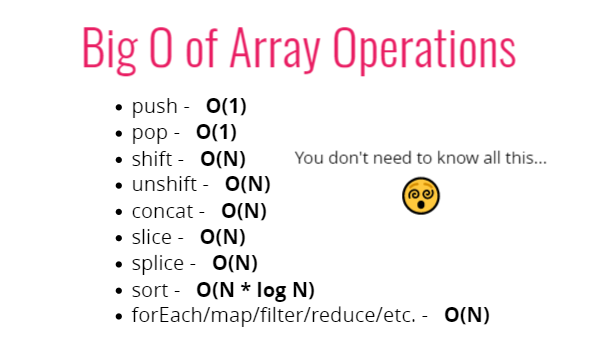


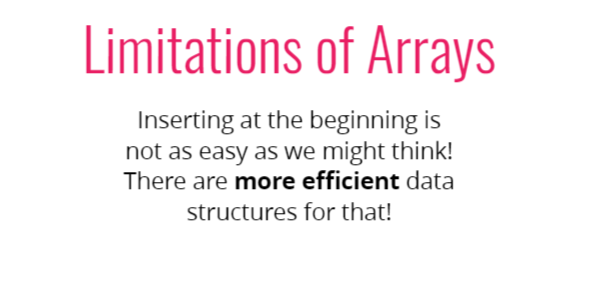












Code:

//1. sum all number

const sum = (val) => {

    let add = 0;

    for (let i = 0; i <= val; i++){

        add += i;

    }

    return add;

}

const total = sum(6);

console.log(total);

//or

const sum = (val) => {

    return val \* (val + 1) / 2;

};

const total = sum(6);

console.log(total);

//down count loop

const sum = (n) => {

   for (let j = n - 1; j >= 0; j--) {

     console.log(j);

   }

};

const total = sum(6);

// 2 loop

const sum = (n) => {

    for (let i = 0; i <= n; i++){

      for (let j = 0; j <=n; j++) {

        console.log(i,j);

      }

    }

};

const total = sum(2);

//Array sum korbo

const sum = (arr) => {

  let total = 0;

  for (let i = 0; i < arr.length; i++) {

    total += arr[i];

  }

  return total;

};

const result = sum([1, 2, 3, 4]);

console.log(result)

//or other example

const sum = (arr) => {

  let newArray = [];

  for (let i = 0; i < arr.length; i++){

    newArray.push(2 \* arr[i])

  }

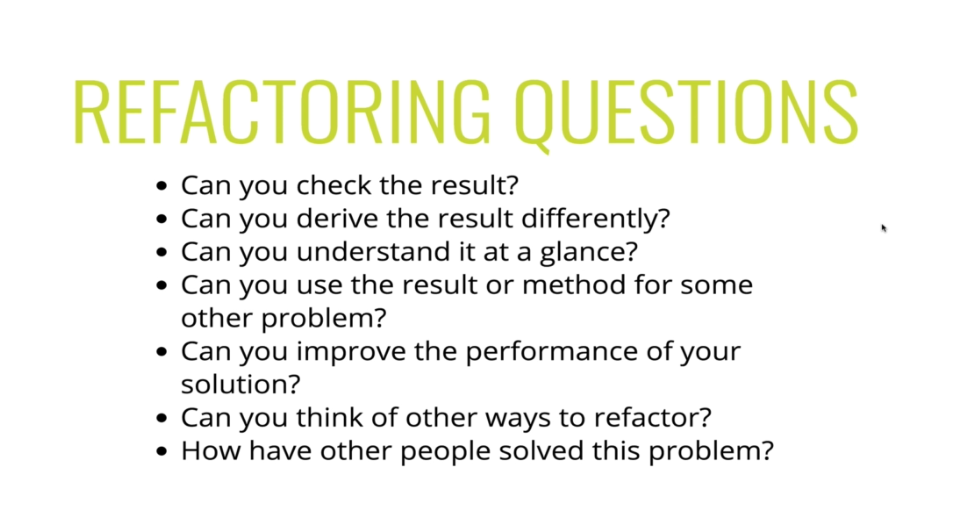
  return newArray;

};

const result = sum([1, 2, 3, 4]);

console.log(result)

Interviewer je question gula amk ask korte pare:



//code refactor

function charCount(str) {

  const obj = {};

  for (let chars of str){

    const char = chars.toLowerCase();

    if (/[a-z0-9]/.test(char)) {

      obj[char] = ++obj[char] || 1;

    }

  }

  return obj;

}

console.log(charCount("Helloo"));

//3. Write a function which takes in a string and returns counts of each character in the string. (return in an object in lowerCase)

//Example:

charCount("Your PIN number is 12334")

/\*{

1 : 1,

2 : 1,

3 : 2,

4 : 1,

y : 1,

....

}\*/

function charCount(str) {

  //make object to return at end

  //loop over string, for each char..

     //if a char is a number/letter and is a key in object, add one to count

     //if a char is a number/letter and is not in object, add it to obj and set value to 1

     //if character is something else(space, period, etc) dont do anything

  //return obj at end

}

Simply Letter Count:

function charCount(str) {

  //make object to return at end

  const result = {};

  //loop over string, for each char..

  for (let i = 0; i < str.length; i++){

    let char = str[i];

    //if a char is a number/letter and is a key in object, add one to count

    if (result[char] > 0) {

      result[char]++;

    }

    //if a char is a number/letter and is not in object, add it to obj and set value to 1

    else {

      result[char] = 1;

    }

  }

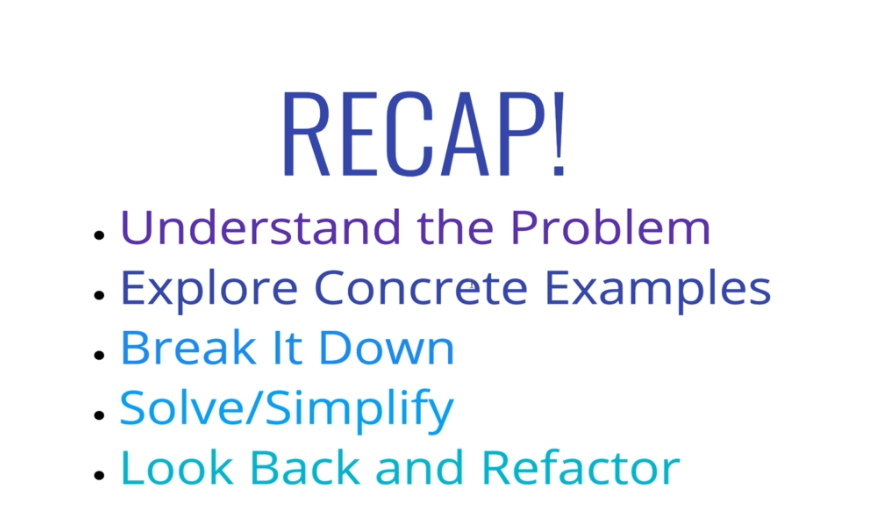
  //if character is something else(space, period, etc) dont do anything

  //return obj at end

  return result;

}

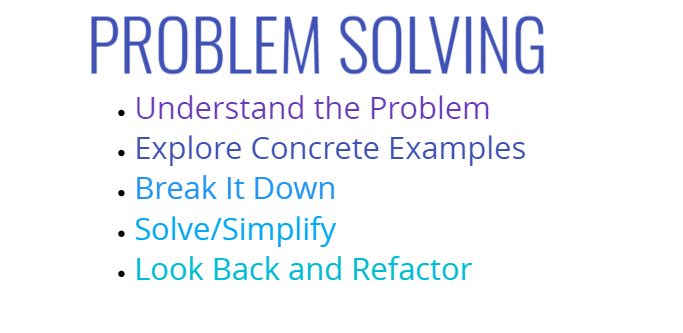
console.log(charCount('Hello'))

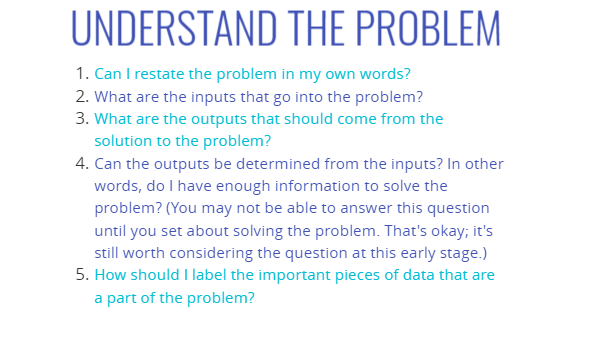


Problem Solving Patterns:

Problem Solving strategies:

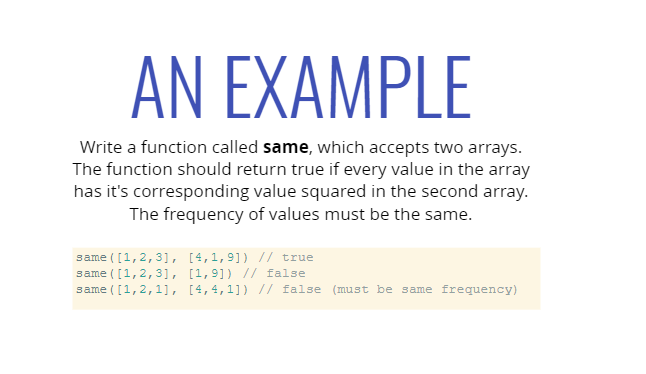












Nornal Solution:

//4. Example

const same = (arr1, arr2) => {

//frequency check

  if (arr1.length !== arr2.length) {

    return false;

  }

  //check all elment

  for (let i = 0; i < arr1.length; i++){

    let correctIndex = arr2.indexOf(arr1[i] \*\* 2)

    if (correctIndex === -1) {

      return false;

    }

    arr2.splice(correctIndex, 1)

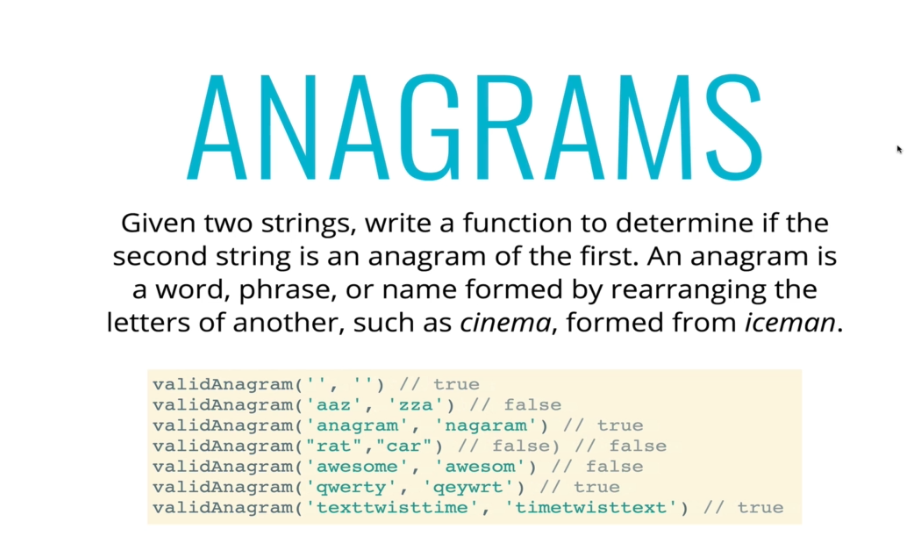
  }

  //return true

  return true;

}

console.log(same([1, 2, 3], [4, 1, 9]));



//5. Example

const validAnagram = (str1, str2) => {

  if (str1.length !== str2.length) {

    return false;

  }

  const lookup = {};

  for (let i = 0; i < str1.length; i++){

    let letter = str1[i];

    lookup[letter] ? lookup[letter] += 1 : lookup[letter] = 1;

  }

  for (let i = 0; i < str2.length; i++){

    let letter = str2[i];

    if (!lookup[letter]) {

      return false;

    } else {

      lookup[letter] -= 1;

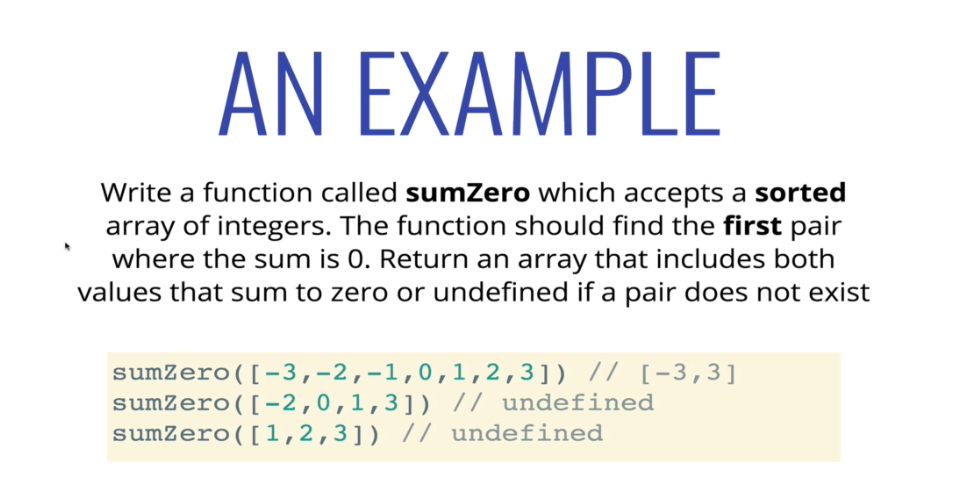
    }

  }

  return true;

}

console.log(validAnagram('anagram', 'nagaram'))



//5. Example

const sumZero = (arr) => {

  let left = 0;

  let right = arr.length - 1;

  while (left < right) {

    let sum = arr[left] + arr[right];

    if (sum === 0) {

      return [arr[left], arr[right]];

    }

    else if (sum > 0) {

      right--;

    } else {

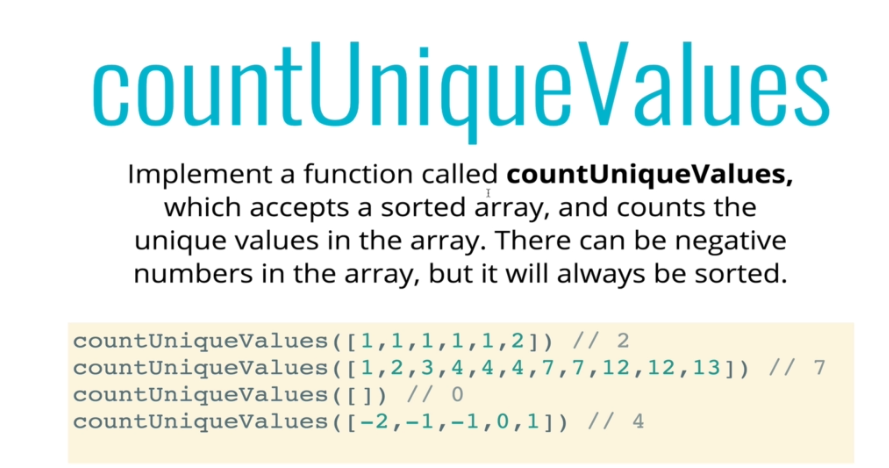
      left++;

    }

  }

}

console.log(sumZero([-3, -2, -1, 0, 1]))



//6. Example

const countUniqueValues = (arr) => {

  if (arr.length === 0) return 0;

  let i = 0;

  for (let j = 1; j < arr.length; j++) {

    if (arr[i] !== arr[j]) {

      i++;

      arr[i] = arr[j];

    }

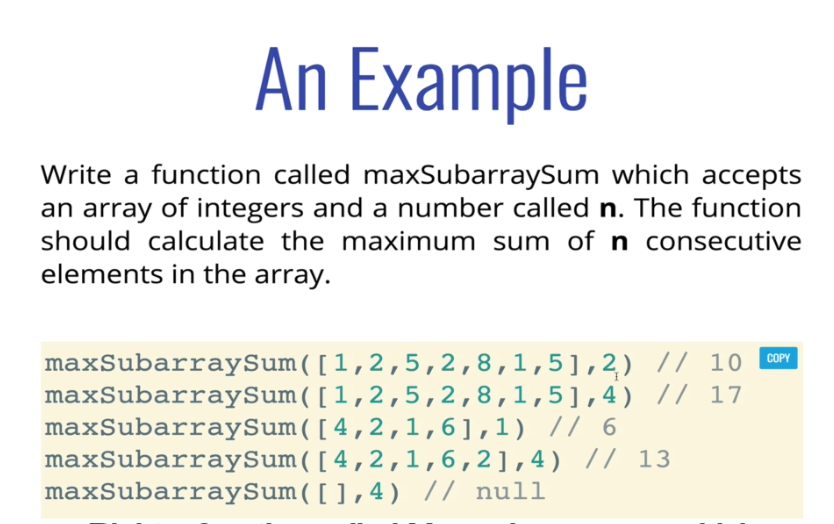
    console.log(i, j)

  }

  return i + 1;

};

console.log(countUniqueValues([1, 1, 2, 3, 3, 3, 4, 5, 5]));



//7. Example

const maxSubrraySum = (arr, num) => {

  if (num > arr.length) {

    return null;

  }

  let max = -Infinity;

  for (let i = 0; i < arr.length - num + 1; i++){

    temp = 0;

    console.log(i)

    for (let j = 0; j < num; j++){

      temp += arr[i + j];

    }

    if (temp > max) {

      max = temp;

    }

    console.log(temp, max);

  }

  return max;

}

console.log(maxSubrraySum([2,6,9,2,1,8,5,6,3], 3))

//or

const maxSubrraySum = (arr, num) => {

  let maxSum = 0;

  let tempSum = 0;

  if (arr.length < num) return null;

  for (let i = 0; i < num; i++){

    maxSum += arr[i];

  }

  tempSum = maxSum;

  for (let i = num; i < arr.length; i++){

    tempSum = tempSum - arr[i - num] + arr[i];

    maxSum = Math.max(maxSum, tempSum);

  }

  return maxSum;

}

console.log(maxSubrraySum([2,6,9,2,1,8,5,6,3], 3))

