***JS code cheatsheet***

|  |  |
| --- | --- |
| Square Root of Different Data Types const number1 = 2.25;  const number2 = -4;  const number3 = 'hello';  const result1 = Math.sqrt(number1);  const result2 = Math.sqrt(number2);  const result3 = Math.sqrt(number3);  console.log(`The square root of ${number1} is ${result1}`);  console.log(`The square root of ${number2} is ${result2}`);  console.log(`The square root of ${number3} is ${result3}`); | Swap two variables //create a temporary variable  let temp;  //swap variables  temp = a;  a = b;  b = temp;  console.log(`The value of a after swapping: ${a}`);  console.log(`The value of b after swapping: ${b}`); |
| Get a Random Number between 1 and 10 // generating a random number  const a = Math.random() \* (10-1) + 1  console.log(`Random value between 1 and 10 is ${a}`); | Check if a number is Positive, Negative, or Zero const number = parseInt(prompt("Enter a number: "));  // check if number is greater than 0  if (number > 0) {  console.log("The number is positive");  }  // check if number is 0  else if (number == 0) {  console.log("The number is zero");  }  // if number is less than 0  else {  console.log("The number is negative");  } |
| Odd-even const number = prompt("Enter a number: ");  //check if the number is even  if(number % 2 == 0) {  console.log("The number is even.");  }  // if the number is odd  else {  console.log("The number is odd.");  } | Largest Number among Three Numbers const num1 = parseFloat(prompt("Enter first number: "));  const num2 = parseFloat(prompt("Enter second number: "));  const num3 = parseFloat(prompt("Enter third number: "));  let largest;  // check the condition  if(num1 >= num2 && num1 >= num3) {  largest = num1;  }  else if (num2 >= num1 && num2 >= num3) {  largest = num2;  }  else {  largest = num3;  }  // display the result  console.log("The largest number is " + largest); |
| Check prime number const number = parseInt(prompt("Enter a positive number: "));  let isPrime = true;  // check if number is equal to 1  if (number === 1) {  console.log("1 is neither prime nor composite number.");  }  // check if number is greater than 1  else if (number > 1) {  // looping through 2 to number-1  for (let i = 2; i < number; i++) {  if (number % i == 0) {  isPrime = false;  break;  }  }  if (isPrime) {  console.log(`${number} is a prime number`);  } else {  console.log(`${number} is a not prime number`);  }  } | Factorial const number = parseInt(prompt('Enter a positive integer: '));  // checking if number is negative  if (number < 0) {  console.log('Error! Factorial for negative number does not exist.');  }  // if number is 0  else if (number === 0) {  console.log(`The factorial of ${number} is 1.`);  }  // if number is positive  else {  let fact = 1;  for (i = 1; i <= number; i++) {  fact \*= i;  }  console.log(`The factorial of ${number} is ${fact}.`);  } |
| Fibonacci Series Up to n Terms const number = parseInt(prompt('Enter the number of terms: '));  let n1 = 0, n2 = 1, nextTerm;  console.log('Fibonacci Series:');  for (let i = 1; i <= number; i++) {  console.log(n1);  nextTerm = n1 + n2;  n1 = n2;  n2 = nextTerm;  } | Check armstrong number // program to check an Armstrong number of three digits  let sum = 0;  const number = prompt('Enter a three-digit positive integer: ');  // create a temporary variable  let temp = number;  while (temp > 0) {  // finding the one's digit  let remainder = temp % 10;  sum += remainder \* remainder \* remainder;  // removing last digit from the number  temp = parseInt(temp / 10); // convert float into integer  }  // check the condition  if (sum == number) {  console.log(`${number} is an Armstrong number`);  }  else {  console.log(`${number} is not an Armstrong number.`);  } |
| Sum of Natural Numbers const number = parseInt(prompt('Enter a positive integer: '));  let sum = 0;  // looping from i = 1 to number  // in each iteration, i is increased by 1  for (let i = 1; i <= number; i++) {  sum += i;  }  console.log('The sum of natural numbers:', sum); | HCF/GCD let number1 = prompt('Enter a first positive integer: ');  let number2 = prompt('Enter a second positive integer: ');  // looping until both numbers are equal  while(number1 != number2){  if(number1 > number2) {  number1 -= number2;  }  else {  number2 -= number1;  }  }  // display the hcf  console.log(`HCF is ${number1}`); |
| LCM // take input  const num1 = prompt('Enter a first positive integer: ');  const num2 = prompt('Enter a second positive integer: ');  // higher number among number1 and number2 is stored in min  let min = (num1 > num2) ? num1 : num2;  // while loop  while (true) {  if (min % num1 == 0 && min % num2 == 0) {  console.log(`The LCM of ${num1} and ${num2} is ${min}`);  break;  }  min++;  } | Decimal2binary // take input  const number = parseInt(prompt('Enter a decimal number: '));  // convert to binary  const result = number.toString(2);  console.log('Binary:' + ' ' + result); |
| ASCII Value of Character // take input from the user  const string = prompt('Enter a character: ');  // convert into ASCII value  const result = string.charCodeAt(0);  console.log(`The ASCII value is: ${result}`); | Check string is palindrome function checkPalindrome(string) {  // convert string to an array  const arrayValues = string.split('');  // reverse the array values  const reverseArrayValues = arrayValues.reverse();  // convert array to string  const reverseString = reverseArrayValues.join('');  if(string == reverseString) {  console.log('It is a palindrome');  }  else {  console.log('It is not a palindrome');  }  }  //take input  const string = prompt('Enter a string: ');  checkPalindrome(string); |
| Reverse a string function reverseString(str) {  // return a new array of strings  const arrayStrings = str.split("");    // reverse the new created array elements  const reverseArray = arrayStrings.reverse();    // join all elements of the array into a string  const joinArray = reverseArray.join("");    // return the reversed string  return joinArray;  }    // take input from the user  const string = prompt('Enter a string: ');  const result = reverseString(string);  console.log(result); | Number of vowels in string // defining vowels  const vowels = ["a", "e", "i", "o", "u"]  function countVowel(str) {  // initialize count  let count = 0;  // loop through string to test if each character is a vowel  for (let letter of str.toLowerCase()) {  if (vowels.includes(letter)) {  count++;  }  }  // return number of vowels  return count  }  // take input  const string = prompt('Enter a string: ');  const result = countVowel(string);  console.log(result); |
| Remove property from object // creating an object  const student = {  name: 'John',  age: 20,  hobbies: ['reading', 'games', 'coding'],  greet: function() {  console.log('Hello everyone.');  },  score: {  maths: 90,  science: 80  }  };  // deleting a property from an object  delete student.greet;  delete student['score'];  console.log(student); | Check key exists in object const person = {  id: 1,  name: 'John',  age: 23  }  // check if key exists  const hasKey = 'name' in person;  if(hasKey) {  console.log('The key exists.');  }  else {  console.log('The key does not exist.');  } |
| Clone object using Object.assign() // declaring object  const person = {  name: 'John',  age: 21,  }  // cloning the object  const clonePerson = Object.assign({}, person);  console.log(clonePerson);  // changing the value of clonePerson  clonePerson.name = 'Peter';  console.log(clonePerson.name);  console.log(person.name); | Clone object using spread const person = {  name: 'John',  age: 21,  }  // cloning the object  const clonePerson = { ... person}  console.log(clonePerson);  // changing the value of clonePerson  clonePerson.name = 'Peter';  console.log(clonePerson.name);  console.log(person.name); |
| Clone object using JSON.parse() const person = {  name: 'John',  age: 21,  }  // cloning the object  const clonePerson = JSON.parse(JSON.stringify(person));  console.log(clonePerson);  // changing the value of clonePerson  clonePerson.name = 'Peter';  console.log(clonePerson.name);  console.log(person.name); | Loop in object const student = {  name: 'John',  age: 20,  hobbies: ['reading', 'games', 'coding'],  };  // using for...in  for (let key in student) {  let value;  // get the value  value = student[key];  console.log(key + " - " + value);  } |
| Merge property of two objects // object 1  const person = {  name: 'Jack',  age:26  }  // object 2  const student = {  gender: 'male'  }  // merge two objects  const newObj = {...person, ...student};  console.log(newObj); | Add key-value pair to object  const person = {  name: 'Monica',  age: 22,  gender: 'female'  }  // add a key/value pair  person['height'] = 5.4;  console.log(person); |
| Object to string const person = {  name: 'Jack',  age: 27  }  const result = JSON.stringify(person);  console.log(result);  console.log(typeof result); | String to Base64 const str = "Learning JavaScript";  // encoding the string  const result = window.btoa(str);  console.log(result);  // decoding the string  const result1 = window.atob(result);  console.log(result1); |
| Check leapyear function checkLeapYear(year) {  //three conditions to find out the leap year  if ((0 == year % 4) && (0 != year % 100) || (0 == year % 400)) {  console.log(year + ' is a leap year');  } else {  console.log(year + ' is not a leap year');  }  }  // take input  const year = prompt('Enter a year:');  checkLeapYear(year); | Remove item from array function removeItemFromArray(array, n) {  const newArray = [];  for ( let i = 0; i < array.length; i++) {  if(array[i] !== n) {  newArray.push(array[i]);  }  }  return newArray;  }  const result = removeItemFromArray([1, 2, 3 , 4 , 5], 2);  console.log(result); |
| Array contains a specific value // program to check if an array contains a specified value  const array = ['you', 'will', 'learn', 'javascript'];  const hasValue = array.includes('javascript');  // check the condition  if(hasValue) {  console.log('Array contains a value.');  } else {  console.log('Array does not contain a value.');  } | Append object to array // program to append an object to an array  function insertObject(arr, obj) {  // append object  arr.push(obj);    console.log(arr);  }  // original array  let array = [1, 2, 3];  // object to add  let object = {x: 12, y: 8};  // call the function  insertObject(array, object); |
| Check object is an array function checkObject(arr) {  // check if arr is array  const result = Array.isArray(arr);  if(result) {  console.log(`[${arr}] is an array.`);  }  else {  console.log(`${arr} is not an array.`);  }  }  const array = [1, 2, 3];  // call the function  checkObject(array); | Remove duplicate from array // program to remove duplicate value from an array  function getUnique(arr){  // removing duplicate  let uniqueArr = [...new Set(arr)];  console.log(uniqueArr);  }  const array = [1, 2, 3, 2, 3];  // calling the function  getUnique(array); |
| Create 2D array function twoDimensionArray(a, b) {  let arr = [];  // creating two dimensional array  for (let i = 0; i< a; i++) {  for(let j = 0; j< b; j++) {  arr[i] = [];  }  }  // inserting elements to array  for (let i = 0; i< a; i++) {  for(let j = 0; j< b; j++) {  arr[i][j] = j;  }  }  return arr;  }  const x = 2;  const y = 3;  const result = twoDimensionArray(x, y);  console.log(result); | Extract Given Property Values from Objects as Array function extractValue(arr, prop) {  // extract value from property  let extractedValue = arr.map(item => item[prop]);  return extractedValue;  }  const objArray = [{a: 1, b: 2}, {a: 4, b: 5}, {a: 8, b: 9}];  // passing an array of objects and property 'a' to extract  const result = extractValue(objArray, 'a');  console.log(result); |
| Get file extension function getFileExtension(filename){  // get file extension  const extension = filename.split('.').pop();  return extension;  }  // passing the filename  const result1 = getFileExtension('module.js');  console.log(result1);  const result2 = getFileExtension('module.txt');  console.log(result2); | Set union // perform union operation  // contain elements of both sets  function union(a, b) {  let unionSet = new Set(a);  for (let i of b) {  unionSet.add(i);  }  return unionSet  }  // two sets of fruits  const setA = new Set(['apple', 'mango', 'orange']);  const setB = new Set(['grapes', 'apple', 'banana']);  const result = union(setA, setB);  console.log(result); |
| Set intersection // perform intersection operation  // elements of set a that are also in set b  function intersection(setA, setB) {  let intersectionSet = new Set();  for (let i of setB) {  if (setA.has(i)) {  intersectionSet.add(i);  }  }  return intersectionSet;  }  // two sets of fruits  const setA = new Set(['apple', 'mango', 'orange']);  const setB = new Set(['grapes', 'apple', 'banana']);  const result = intersection(setA, setB);  console.log(result); | Set difference // perform difference operation  // elements of set a that are not in set b  function difference(setA, setB) {  let differenceSet = new Set(setA)  for (let i of setB) {  differenceSet.delete(i)  }  return differenceSet  }  // two sets of fruits  const setA = new Set(['apple', 'mango', 'orange']);  const setB = new Set(['grapes', 'apple', 'banana']);  const result = difference(setA, setB);  console.log(result); |
| Get current url const url1 = window.location.href;  const url2 = document.URL;  console.log(url1);  console.log(url2); | Check If a Variable is of Function Type function testVariable(variable) {    if(variable instanceof Function) {  console.log('The variable is of function type');  }  else {  console.log('The variable is not of function type');  }  }  const count = true;  const x = function() {  console.log('hello')  };  testVariable(count);  testVariable(x); |
| Function overloading // program to perform function overloading  function sum() {    // if no argument  if (arguments.length == 0) {  console.log('You have not passed any argument');  }  // if only one argument  else if (arguments.length == 1) {  console.log('Pass at least two arguments');  }  // multiple arguments  else {  let result = 0;  let length = arguments.length;    for (i = 0; i < length; i++) {  result = result + arguments[i];  }  console.log(result);  }  }  sum();  sum(5);  sum(5, 9);  sum(1, 2, 3, 4, 5, 6, 7, 8, 9); | Check if a Number is Float or Integer // program to check if a number is a float or integer value  function checkNumber(x) {  // check if the passed value is a number  if(typeof x == 'number' && !isNaN(x)){    // check if it is integer  if (Number.isInteger(x)) {  console.log(`${x} is integer.`);  }  else {  console.log(`${x} is a float value.`);  }    } else {  console.log(`${x} is not a number`);  }  }  checkNumber('hello');  checkNumber(44);  checkNumber(3.4);  checkNumber(-3.4);  checkNumber(NaN); |
| Implement stack // program to implement stack data structure  class Stack {  constructor() {  this.items = [];  }    // add element to the stack  add(element) {  return this.items.push(element);  }    // remove element from the stack  remove() {  if(this.items.length > 0) {  return this.items.pop();  }  }    // view the last element  peek() {  return this.items[this.items.length - 1];  }    // check if the stack is empty  isEmpty(){  return this.items.length == 0;  }    // the size of the stack  size(){  return this.items.length;  }    // empty the stack  clear(){  this.items = [];  }  }  let stack = new Stack();  stack.add(1);  stack.add(2);  stack.add(4);  stack.add(8);  console.log(stack.items);  stack.remove();  console.log(stack.items);  console.log(stack.peek());  console.log(stack.isEmpty());  console.log(stack.size());  stack.clear();  console.log(stack.items); | Implement queue // program to implement queue data structure  class Queue {  constructor() {  this.items = {};  this.headIndex = 0;  this.tailIndex = 0;  }  //adds a new element  enqueue(element) {  this.items[this.tailIndex] = element;  this.tailIndex++;  }  //removes an element from head of the queue  dequeue() {  let removedElement = this.items[this.headIndex];  delete this.items[this.headIndex];  this.headIndex++;  return removedElement;  }  //shows the head element of the queue  peek() {  let peekElement = this.items[this.headIndex];  return peekElement;  }  //shows the number of items in queue  size() {  return this.tailIndex - this.headIndex;  }  //checks if queue is empty or not  isEmpty() {  if (this.tailIndex - this.headIndex == 0) {  return true;  }  else {  return false;  }  }  //empty the queue  clear() {  this.items = {};  this.headIndex = 0;  this.tailIndex = 0;  }  }  let queue = new Queue();  // add items to queue  queue.enqueue(8);  queue.enqueue(6);  queue.enqueue(4);  queue.enqueue(2);  console.log("Queue after adding items: ");  console.log(queue.items);  // remove the first item  queue.dequeue();  console.log("Queue after deleting the first item:");  console.log(queue.items);  // show the first item  console.log("First item of the queue = " + queue.peek());  // empty the queue  queue.clear();  console.log("After clearing the queue: ");  console.log(queue.items); |
| Pass a function as parameter function greet() {  return 'Hello';  }  // passing function greet() as a parameter  function name(user, func)  {  // accessing passed function  const message = func();  console.log(`${message} ${user}`);  }  name('John', greet);  name('Jack', greet);  name('Sara', greet); | Dimensions of image const img = new Image();  // get the image  img.src = '//cdn.programiz.com/sites/tutorial2program/files/cover-artwork.png';  // get height and width  img.onload = function() {  console.log('width ' + this.width)  console.log('height '+ this.height);  } |
| Remove all spaces from string // program to trim a string  const string = ' Hello World ';  const result = string.split(' ').join('');  console.log(result); | **Convert Date to Number**  // program to convert date to number  // create date  const d1 = new Date();  console.log(d1);  // converting to number  const result = d1.getTime();  console.log(result); |
| Merge two arrays and remove duplicates // program to merge and remove duplicate value from an array  function getUniqueAfterMerge(arr1, arr2){  // merge two arrays  let arr = [...arr1, ...arr2];  // removing duplicate  let uniqueArr = [...new Set(arr)];  console.log(uniqueArr);  }  const array1 = [1, 2, 3];  const array2 = [2, 3, 5]  // calling the function  getUniqueAfterMerge(array1, array2); | Intersection of two arrays function performIntersection(arr1, arr2) {  const intersectionResult = arr1.filter(x => arr2.indexOf(x) !== -1);    return intersectionResult;  }  const array1 = [1, 2, 3, 5, 9];  const array2 = [1, 3, 5, 8];  const result = performIntersection(array1, array2);  console.log(result); |
| Split array in smaller chunks function splitIntoChunk(arr, chunk) {  while(arr.length > 0) {  let tempArray;  tempArray = arr.splice(0, chunk);  console.log(tempArray);  }  }  const array = [1, 2, 3, 4, 5, 6, 7, 8];  const chunk = 2;  splitIntoChunk(array, chunk); | Sort words in alphabetical order const string = prompt('Enter a sentence: ');  // converting to an array  const words = string.split(' ');  // sort the array elements  words.sort();  // display the sorted words  console.log('The sorted words are:');  for (const element of words) {  console.log(element);  } |
| Digit Frequency function getDigitFrequency(*n*, *d*) {      let cnt = 0;      while (n > 0) {          if (n % 10 === d) {              cnt++;          }          n = Math.floor(n / 10);      }      return cnt;  }  const n = 994543234;  const d = 4;  const f = getDigitFrequency(n, d);  console.log(f); | Is sorted function isSorted(*arr*) {      for (let i = 1; i < arr.length; i++) {          if (arr[i] < arr[i - 1]) {              return false;          }      }      return true;  }  console.log(isSorted([1, 2, 3, 4, 5, 6])); |
| Inverse function inverseAndDisplay(*a*) {      let result = "";      let inverseArr = [];      for (let i = 0; i < a.length; i++) {          for (let j = 0; j < a.length; j++) {              if (a[i] === j) {                  inverseArr[j] = i;              }          }      }      console.log(inverseArr);  }  const n = 5;  const a = [4, 0, 2, 3, 1];  inverseAndDisplay(a); | Leaders function leaders(*arr*) {      const n = arr.length;      let curr = arr[n - 1];      let result = [];      result.push(curr);      for (let i = n - 2; i >= 0; i--) {          if (arr[i] > curr) {              curr = arr[i];              result.push(curr);          }      }      return result.reverse();  }  const arr = [7, 10, 4, 10, 6, 5, 2];  const leadersArr = leaders(arr);  console.log(leadersArr.join(" ")); |
| Left rotate array by one element function leftrotate(*arr*) {      const temp = arr[0];      for (let i = 1; i < arr.length; i++) {          arr[i - 1] = arr[i];      }      arr[arr.length - 1] = temp;      console.log(arr.join(" "));  }    leftrotate([2, 2, 3, 2, 3, 6]); | Maximum consecutive ones (1) function maxConsecutiveOnes(*arr*) {      let res = 0;      let curr = 0;      for (let i = 0; i < arr.length; i++) {          if (arr[i] === 0) {              curr = 0;          } else {              curr++;              res = Math.max(res, curr);          }      }      return res;  }  console.log(maxConsecutiveOnes([1, 1, 0, 1, 1, 1])); // Output: 3 |
| Pair sum function pairSum(*arr*, *num*) {      let cnt = 0;      for (let i = 0; i < arr.length; i++) {          for (let j = 0; j < i; j++) {              if (arr[i] + arr[j] === num) {                  cnt++;              }          }      }      return cnt;  }  const arr = [1, 3, 6, 2, 5, 4, 3, 2, 4];  console.log(pairSum(arr, 9)); | Reverse array: function reverse(*a*) {      let f = 0;      let r = a.length - 1;      while (f < r) {          let temp = a[r];          a[r] = a[f];          a[f] = temp;          f++;          r--;      }  }  const arr = [1, 2, 3, 4, 5];  reverse(arr);  console.log(arr); // Output: [5, 4, 3, 2, 1] |
| Max profit from stock buy and sell function maxProfit(*arr*) {      let profit = 0;      for (let i = 1; i < arr.length; i++) {          if (arr[i] > arr[i - 1]) {              profit += (arr[i] - arr[i - 1]);          }      }      return profit;  }  console.log(maxProfit([1, 5, 3, 8, 12])); // Output: 13 | Subsets function printSubsets(*arr*) {      for (let i = 0; i < arr.length; i++) {          for (let j = i; j < arr.length; j++) {              let subset = "";              for (let k = i; k <= j; k++) {                  subset += arr[k] + "\t";              }              console.log(subset);          }      }  }  const arr = [4, 5, 6, 7];  printSubsets(arr); |
| Container with most water function mostWater(*heights*) {      let i = 0;      let j = heights.length - 1;      let water = 0;      while (i < j) {          const w = j - i;          const h = Math.min(heights[i], heights[j]);          water = Math.max(water, h \* w);          if (heights[i] < heights[j]) {              i++;          } else {              j--;          }      }      return water;  }  const heights = [1, 8, 6, 2, 5, 4, 8, 3, 7];  const res = mostWater(heights);  console.log(res); | Majority element function printMajorityElement(*arr*) {      let val = arr[0];      let count = 1;      for (let i = 1; i < arr.length; i++) {          if (val === arr[i]) {              count++;          } else {              count--;          }          if (count === 0) {              val = arr[i];              count = 1;          }      }      count = 0;      for (let i = 0; i < arr.length; i++) {          if (arr[i] === val) {              count++;          }      }      if (count > arr.length / 2) {          console.log(val);      } else {          console.log("No Majority Element exist");      }  }  const arr = [2,2,3,4,5,4,4,4,4];  printMajorityElement(arr); |