Chapter 2 (Variables for string)  
  
1. Declare any variable in the camelCase format.  
  
Answer: let myVariableName = "exampleString";  
  
  
2. Declare a variable of your choice without defining it and then assign it a string:

Answer: let myVariable;

myVariable = "This is a string assignment";  
  
  
  
3: Declare the variable teamName and Alert your Team name.  
  
Answer: let teamName = "MyTeam";

alert(`Our team name is ${teamName}`);  
  
  
4. This statement has already been coded. var bestMan = "Charlie"; Assign the variable a new string.  
  
Answer: var bestMan = "Charlie";

bestMan = "David"; // Assigning a new string to bestMan  
  
  
  
  
  
Chapter 3 (Variables for numbers)  
  
1. Declare a variable “caseQty”  
  
Answer: let caseQty;  
  
2. Assign to the variable caseQty, which has already been declared, the value 144.  
  
Answer: Assign the value 144 to the variable **caseQty**:  
 caseQty = 144;  
  
3. Rewrite this statement so the variable can be used in a math operation. var num = "9";  
  
Answer: Rewrite the statement so the variable can be used in a math operation. Convert the variable **num** to a number:  
  
var num = "9";

var numAsNumber = Number(num); // Now, numAsNumber can be used in math operations.  
  
4. In one statement declare a variable. In a second statement assign it the sum of 2 numbers.  
  
Answer: Declare a variable and assign it the sum of two numbers in one statement:  
  
let sum = 5 + 7; // Here, 'sum' will hold the value 12 (the sum of 5 and 7).  
  
  
5. What is the value of orderTotal? var merchTotal = 100; var shippingCharge = 10; var orderTotal = merchTotal + shippingCharge; Try it yourself.  
  
Answer: The value of **orderTotal** would be 110. It's the sum of **merchTotal** (100) and **shippingCharge** (10):  
  
var merchTotal = 100;

var shippingCharge = 10;

var orderTotal = merchTotal + shippingCharge;  
  
  
6. In the first statement declare a variable and assign it a number. In the second statement, change the value of the variable by adding it together with a number.  
  
Answer: Declare a variable, assign it a number, and then change its value by adding it to another number in a second statement:  
  
let myNumber = 20; // Declare and assign a number (20)

myNumber = myNumber + 5; // Change the value by adding it to 5, now 'myNumber' is 25.  
  
In the sixth task, we first assign the value 20 to **myNumber**, and then we change its value to 25 by adding 5 to it.  
  
  
  
Chapter 4 (Variable names Legal and Illegal)  
  
1. Correct this statement. var product cost = 3.45;  
  
Answer: var productCost = 3.45;  
  
2. Rewrite this using camelCase. var Nameofband;  
  
Answer: Rewrite using camelCase:  
 var nameOfBand;  
  
3. In a single statement declare a legally-named variable and assign a number to it.  
  
Answer: Declare a legally-named variable and assign a number to it in a single statement:  
let myNumber = 42; // Example of declaring a legally-named variable and assigning a number  
  
4. Declare a variable that is a combination of your first and last names. Use camelCase.  
  
Answer: Declare a variable that is a combination of your first and last names using camelCase:  
let firstName = "John";

let lastName = "Doe";

1. let fullName = firstName + lastName; // Combining first and last names into a camelCase variable  
     
     
   5. List the legal and Illegal Variables.  
     
   Answer: Legal and Illegal Variable Names:

Legal Variable Names:

* **myVariable**
* **\_privateVar**
* **$specialVar**
* **camelCaseVar**
* **number123**

Illegal Variable Names:

* **2ndPlace** (starts with a number)
* **my-variable** (contains a hyphen)
* **my variable** (contains a space)
* **for** (reserved keyword)
* **#hashTag** (contains special characters other than underscores and dollar signs)

Remember that variable names in JavaScript must start with a letter, underscore (\_), or dollar sign ($), followed by letters, numbers, underscores, or dollar signs. They cannot start with a number or contain spaces, hyphens, or other special characters (except underscores and dollar signs). Additionally, it's good practice to use meaningful names that describe the purpose of the variable.

Chapter 5 (Math Expression I)   
  
1. What is the name and symbol of the arithmetic operator that gives you the remainder when one number is divided by another?  
  
Answer: The name of the arithmetic operator that gives you the remainder when one number is divided by another is called the "modulo operator," and its symbol is **%**.  
  
  
2. What is the value of num? var num = 20 % 6;  
  
Answer: The value of **num** is **2**. This is because **20 % 6** calculates the remainder when 20 is divided by 6, which is 2.  
  
3. In a single statement, declare the variable largeNum and assign it the result of 1,000 multiplied by 2,000.  
  
Answer: Declare the variable **largeNum** and assign it the result of 1,000 multiplied by 2,000 in a single statement:  
let largeNum = 1000 \* 2000;  
  
4. Assign to a variable the value represented by one variable subtracted from the value represented by another variable?  
  
Answer: Assign the value represented by one variable subtracted from the value represented by another variable. Let's assume you have two variables **a** and **b**:  
let a = 10;

let b = 5;

let result = a - b; // 'result' now holds the value 5 (10 - 5)  
  
  
5. Assign to a variable the remainder when one number is divided by another. The variable hasn't been declared beforehand. Make up the variable name.  
  
Answer: Assign to a variable the remainder when one number is divided by another. Let's make up a variable name, say **remainderResult**, and assume you have two numbers, **x** and **y**:  
let x = 17;

let y = 4;

let remainderResult = x % y; // 'remainderResult' now holds the value 1 (17 % 4)  
  
  
6. Code an alert that displays the result of a multiplication on 2 numbers.  
  
Answer: Code an alert that displays the result of a multiplication of two numbers:  
  
let number1 = 5;

let number2 = 8;

let result = number1 \* number2;

alert(`The result of multiplying ${number1} and ${number2} is ${result}`);  
  
This code will display an alert with the result of multiplying **number1** and **number2**.  
  
  
  
Chapter 6 (Math Expression II)  
  
1. Code a short form of x = x + 1; Use either of the two legal expressions.  
Answer: hort form of **x = x + 1** using the increment operator:   
x++;

// or

x += 1;  
  
2. If x has a value of 100, what is the fastest way to reduce it to 99 with a math expression?  
  
Answer: The fastest way to reduce **x** from 100 to 99 with a math expression is to use the decrement operator: x--;  
  
3. var x = 50; var y = x++;  
  
Answer: The value of **y** after the code execution will be **50**. The **x++** operation first assigns the current value of **x** to **y** and then increments **x**.  
  
4. var y = 50; var z = --y;  
  
Answer: The value of **z** after the code execution will be **49**. The **--y** operation first decrements **y** and then assigns the new value to **z**.  
  
5. In a single statement, decrement num and assign its original value to newNum.  
  
Answer: In a single statement, decrement **num** and assign its original value to **newNum**:  
let newNum = --num;  
  
6. In a single statement add 1 to a variable and assign its original value to another variable.  
  
Answer: In a single statement, add 1 to a variable and assign its original value to another variable:  
let originalValue = 10;

let newValue = ++originalValue;  
  
  
7. Assign a number value to a variable. Increment the variable. Display the new value in an alert  
  
Answer: Assign a number value to a variable, increment the variable, and display the new value in an alert:  
let myNumber = 5;

myNumber++; // Increment the variable

alert(`The new value is: ${myNumber}`);  
This code will display an alert with the new value of **myNumber** after incrementing it.  
  
Chapter 7 (Math Expression III)  
  
1. var calculatedNum = 2 + (2 \* 6); What is the value of calculatedNum?  
Answer: The value of **calculatedNum** in the first statement is 14. It's calculated as 2 + (2 \* 6), which is 2 + 12.  
  
2. var calculatedNum = (2 + 2) \* 6; What is the value of calculatedNum?  
Answer: The value of **calculatedNum** in the second statement is 24. It's calculated as (2 + 2) \* 6, which is 4 \* 6.  
  
3. var calculatedNum = (2 + 2) \* (4 + 2); What is the value of calculatedNum?  
Answer: The value of **calculatedNum** in the third statement is 24. It's calculated as (2 + 2) \* (4 + 2), which is 4 \* 6.  
  
4. var calculatedNum = ((2 + 2) \* 4) + 2; What is the value of calculatedNum? Note: Try all the above equations yourself.  
Answer: The value of **calculatedNum** in the fourth statement is 18. It's calculated as ((2 + 2) \* 4) + 2, which is (4 \* 4) + 2.  
  
5. Write a statement that assigns to cost the result of 2 + 2 \* 4 + 10, clarified with parentheses, producing 56.  
Answer: To assign the result of 2 + 2 \* 4 + 10, clarified with parentheses, producing 56, you can write: let cost = (2 + 2) \* (4 + 10);  
This will result in **cost** having a value of 56.  
  
6. Write a statement that assigns to units the result of 2 + 2 \* 4 + 10, clarified with parentheses, producing 20.  
Answer: To assign the result of 2 + 2 \* 4 + 10, clarified with parentheses, producing 20, you can write: let units = 2 + (2 \* 4) + 10;  
This will result in **units** having a value of 20.  
  
7. Write a statement that assigns to pressure the result of 4 / 2 \* 4, clarified with parentheses, producing 5.  
Answer: To assign the result of 4 / 2 \* 4, clarified with parentheses, producing 5, you can write:  
let pressure = (4 / 2) \* 4;  
This will result in **pressure** having a value of 5.  
  
  
Chapter 8 (Concatenating Text Strings)  
  
1. var num = "2" + "2"; What is the value of num? Include quotation marks.  
Answer: The value of **num** is "22" (including the quotation marks). This is because the **+** operator concatenates the two strings "2" and "2" to form "22".  
  
2. message = ("Hello," + "Dolly"); What is the value of message? (Include the quotation marks.) Alert the statement  
Answer: The value of **message** is "Hello,Dolly" (including the quotation marks). To alert the statement, you can use the following code: alert(message);  
  
3. alert("33" + 3); What message displays in the alert box?  
Answer: The message that displays in the alert box is "333" (without quotation marks). This is because JavaScript performs concatenation when you use the **+** operator with strings, even if one of the operands is a number.  
  
4. Write an alert that displays the concatenation of the two parts of "Pakistan Zindabad".  
Answer: To display the concatenation of the two parts of "Pakistan Zindabad" in an alert, you can use this code: alert("Pakistan " + "Zindabad");  
  
5. Write a statement that assigns to a variable the concatenation of a string with a number  
Answer: To assign to a variable the concatenation of a string with a number, you can write: let result = "The number is " + 42; // Assuming you want to concatenate with the number 42  
  
6. Assign strings to two variables. Then concatenate them and assign the result to a third variable.  
Answer: Here's how you can assign strings to two variables, concatenate them, and assign the result to a third variable:  
let str1 = "Hello,";

let str2 = " world!";

let combinedStr = str1 + str2; // 'combinedStr' now holds "Hello, world!"  
In this code, **str1** and **str2** are first combined using the **+** operator, and the result is stored in the **combinedStr** variable.  
  
  
Chapter 9 (Prompts)  
  
1. Code a prompt with the message "Enter first name". The user's response is assigned to firstName.  
Answer: Code a prompt with the message "Enter first name". The user's response is assigned to **firstName**: let firstName = prompt("Enter first name");  
  
2. Code a prompt with the message "Country?" and the default answer "China". The user's response is assigned to country.  
Answer: Code a prompt with the message "Country?" and the default answer "China". The user's response is assigned to **country**: let country = prompt("Country?", "China");  
  
3. Correct this statement var yourName = prompt(Enter Your Name");  
Answer: var yourName = prompt("Enter Your Name");  
  
4. Code a prompt that specifies a string as the message Include a default input.  
Answer: To code a prompt that specifies a string as the message and includes a default input, you can do it like this: let userInput = prompt("Please enter something", "Default Text");  
  
5. Assign strings to two variables. Code a prompt specifying the first variable as the message and the second variable as the default response. Assign the user's response to a third variable.  
  
Answer: Assign strings to two variables and then code a prompt specifying the first variable as the message and the second variable as the default response. Assign the user's response to a third variable:   
let message = "Enter your favorite color";

let defaultResponse = "Blue";

let userResponse = prompt(message, defaultResponse);  
  
  
6. Display a prompt, including both a message and a default response. Display the user's response in an alert.  
  
Answer: Display a prompt including both a message and a default response and then display the user's response in an alert: let messageWithDefault = "What's your favorite food?";

let defaultAnswer = "Pizza";

let userResponse = prompt(messageWithDefault, defaultAnswer);

Chapter 10 (if statments)  
  
1. var city = "Karachi" if (city = "Karachi") { console.log("The City OF Lights") Correct the above statement: Also try this statement by yourself  
Answer: var city = "Karachi";

if (city === "Karachi") {

console.log("The City OF Lights");

}  
  
2. This is the first line of an if statement: if (x === y) { Complete the statement. If the condition is true, display a box that asks the user value of z? and assign it to another variable.  
  
Answer: Completing the if statement and adding user input logic:  
if (x === y) {

var z = prompt("Enter the value of z?");

}  
  
This code will display a prompt asking the user for the value of **z** if the condition **x === y** is true.  
  
  
3. Code an if statement that tests if ZipCode is "10010" so, Alert that "Karachi". if not then alert ("Please write correct city")  
Answer: Code an if statement that tests if **ZipCode** is "10010" and alerts "Karachi" if true, otherwise alerts "Please write the correct city":  
var ZipCode = "10010";

if (ZipCode === "10010") {

alert("Karachi");

} else {

alert("Please write the correct city");

}  
  
4. Code an if statement. Test whether a variable has a particular numerical value. If so, assign a new value to that variable, as in x = 1;  
Answer: Code an if statement that tests whether a variable has a particular numerical value and assigns a new value to that variable if the condition is met. For example, if **x** has the value 1:  
var x = 1;

if (x === 1) {

x = 42; // Assign a new value to x if the condition is true

}  
In this code, if **x** is equal to 1, it will be assigned a new value of 42.  
  
  
  
Chapter 11 (Comparison Operators)  
  
1. Code the first line of an if statement that tests whether one variable is unequal to another. (Use !)  
  
Answer: Code the first line of an if statement that tests whether one variable is unequal to another using the **!==** operator:  
if (variable1 !== variable2) {

// Rest of your code here

}  
  
2. Code the first line of an if statement that tests whether the value represented by a variable is greater than or equal to the value represented by another variable.  
  
Answer: Code the first line of an if statement that tests whether the value represented by a variable is greater than or equal to the value represented by another variable using the **>=** operator:   
if (variable1 >= variable2) {

// Rest of your code here

}  
  
3. Code an if statement. Test whether a variable is unequal to a particular number. If so, assign a number to that variable.  
  
Answer: Code an if statement that tests whether a variable is unequal to a particular number and assigns a number to that variable if the condition is true:  
if (variable !== 42) {

variable = 42; // Assign a number to the variable if it's not equal to 42

}  
  
4. Code an if statement that tests whether a number is unequal to a different number. If the condition is true (it will be), display a congratulations alert.  
  
Answer: Code an if statement that tests whether a number is unequal to a different number and displays a congratulations alert if the condition is true:  
if (number1 !== number2) {

alert("Congratulations!");

}  
  
  
5. Code a prompt asking for your first name. Code an if statement that tests whether the name you entered is unequal to another name. If the condition is true (it will be), display an alert that says "No match"  
  
Answer: Code a prompt asking for your first name and an if statement that tests whether the name you entered is unequal to another name and displays an alert "No match" if the condition is true:  
let enteredName = prompt("Enter your first name");

if (enteredName !== "John") {

alert("No match");

}  
In this code, if the name you entered is not "John," it will display the "No match" alert.  
  
Chapter 12 (if…else and else if statements)  
  
1. Code an if statement that tests whether the value represented by a variable is greater than or equal to the value represented by another variable. If so, display an alert. If not, display a different alert.  
  
Answer: Code an if statement that tests whether the value represented by a variable is greater than or equal to the value represented by another variable. If so, display an alert. If not, display a different alert:  
let variable1 = 5;

let variable2 = 10;

if (variable1 >= variable2) {

alert("Variable1 is greater than or equal to Variable2");

} else {

alert("Variable1 is less than Variable2");

}  
  
  
2. Write a program using if else and else if statement which take marks from user. And the program will calculate your percentage and give you grade A/C to Your percentage. (MARKSHEET)  
  
Answer: Here's a program that takes marks from the user and calculates the percentage and grade using if, else if, and else statements:  
  
let marks = parseFloat(prompt("Enter your marks:"));

let percentage = (marks / 100) \* 100;

if (percentage >= 90) {

alert(`Your percentage is ${percentage}%, and your grade is A.`);

} else if (percentage >= 80) {

alert(`Your percentage is ${percentage}%, and your grade is B.`);

} else if (percentage >= 70) {

alert(`Your percentage is ${percentage}%, and your grade is C.`);

} else {

alert(`Your percentage is ${percentage}%, and you need to improve your performance.`);

}  
  
3. This is the if statement that begins the code. if (a === 10) { alert("a is 10"); } If a isn't 10, display an alert that says The correct value of a is \_\_\_\_ Note: Try this by yourself  
  
Answer: This is the if statement that begins the code:   
if (a === 10) {

alert("a is 10");

} else {

alert("The correct value of a is something other than 10");

}  
  
  
4. Prompt the user to enter a city. If the city is Karachi, display an alert acknowledging it is Karachi. If not, check to see if it's Lahore. If it is, display an alert acknowledging it's Lahore. Otherwise, display a different alert.  
  
Answer: Prompt the user to enter a city and display appropriate alerts based on the city entered:  
  
let city = prompt("Enter a city:");

if (city === "Karachi") {

alert("It's Karachi!");

} else if (city === "Lahore") {

alert("It's Lahore!");

} else {

alert("It's neither Karachi nor Lahore.");

}  
  
In this code, it first checks if the city is Karachi, then if it's Lahore, and finally, it handles any other city with the "otherwise" case.  
  
  
Chapter 13 (Testing sets of conditions)  
  
1. Code the first line of an if statement that tests whether both are true: a has the same value as b and c has the same value as d.  
Answer: Code the first line of an if statement that tests whether both are true: **a** has the same value as **b** and **c** has the same value as **d**:   
if (a === b && c === d) {

// Rest of your code here

}  
  
2. Code the first line of an if statement that tests whether either or both are true: a has the same value as b or c doesn't have the same value as d.  
Answer: Code the first line of an if statement that tests whether either or both are true: **a** has the same value as **b** or **c** doesn't have the same value as **d**:  
if (a === b || c !== d) {

// Rest of your code here

}  
  
  
3. Code the first line of an if statement that tests whether I. name is either "Hamza" or "Arsalan". II. age is Less than 60.  
  
Answer: Code the first line of an if statement that tests whether:

* **name** is either "Hamza" or "Arsalan."
* **age** is less than 60.

if ((name === "Hamza" || name === "Arsalan") && age < 60) {

// Rest of your code here

}  
  
4. Declare two variables and assign them number values. If the first variable is less than the second variable or greater than the second variable, display an alert.  
  
Answer: Declare two variables and assign them number values. If the first variable is less than or greater than the second variable, display an alert:  
let num1 = 5;

let num2 = 10;

if (num1 < num2 || num1 > num2) {

alert("The condition is true!");

}  
  
  
5. Declare 2 variables. Assign one of them your first name and the other one your last name. Code 2 prompts, asking for your first and your last name. If your answers match the two variables, display an alert.  
  
Answer: Declare two variables, one for your first name and the other for your last name. Then, code two prompts asking for your first and last name. If your answers match the two variables, display an alert:  
let firstName = "John";

let lastName = "Doe";

let userFirstName = prompt("Enter your first name:");

let userLastName = prompt("Enter your last name:");

if (userFirstName === firstName && userLastName === lastName) {

alert("Your first and last names match!");

In this code, it checks if the user's input for first and last names matches the predefined variables **firstName** and **lastName**, and if they match, it displays an alert.  
  
  
Chapter 14 (If statements nested)  
  
1. Code an if statement enclosing a nested if. If password is not empty, then check if password is not greater than 5 , then display an alert that says "Password must be greater than 5" if greater than 5 then Alert "OK".  
  
Answer: Code an if statement enclosing a nested if to check the password conditions:  
let password = prompt("Enter a password:");

if (password !== "") {

if (password.length <= 5) {

alert("Password must be greater than 5");

} else {

alert("OK");

}

} else {

alert("Password cannot be empty");

}  
  
  
2. Try this statement by yourself if (a === 1) { if (c === "Max") { alert("OK"); } }  
  
Answer: if (a === 1) {

if (c === "Max") {

alert("OK");

}

}  
  
3. Code the first line of an if statement that avoids the nesting above by testing for multiple conditions. if (a === 1) { if (c === "Max") { alert("OK"); } }  
Answer: Code the first line of an if statement that avoids the nesting by testing for multiple conditions:  
if (a === 1 && c === "Max") {

alert("OK");

}  
  
This single if statement checks both conditions without nesting.  
  
4. Declare two variables and assign them the same number value. Test two conditions, using nested if statements. Test whether the first variable equals the second variable and also whether it is less than or equal to the second variable. If the test passes—and it will—display an alert message.  
  
Answer: Declare two variables and assign them the same number value. Test two conditions, using nested if statements. Test whether the first variable equals the second variable and also whether it is less than or equal to the second variable. If the test passes, display an alert message:  
  
let num1 = 5;

let num2 = 5;

if (num1 === num2) {

if (num1 <= num2) {

alert("Both conditions are true.");

}

}  
  
In this code, it first checks if **num1** is equal to **num2**, and then, if that condition is true, it checks if **num1** is less than or equal to **num2**, and if both conditions are true, it displays the alert message.  
  
  
Chapter 15 (Array I)  
1. Declare an empty array.  
Answer: let emptyArray = [];  
  
2. Code an array with 1 string element?  
Answer: let stringArray = ["Hello"];  
  
3. var alphabet = ["h","i","j","k"]. Now print the letter “j” in alert using array index  
Answer: Code to print the letter "j" in an alert using array index:  
var alphabet = ["h", "i", "j", "k"];

alert(alphabet[2]); // This will alert "j" because it's at index 2 in the array  
  
  
4. var alphabet=["h","i","j","k", “l”,”m”, “n”, “o”]. Find the total length of array.  
Answer: Code to find the total length of the array:  
var alphabet = ["h", "i", "j", "k", "l", "m", "n", "o"];

var lengthOfArray = alphabet.length;

alert("The total length of the array is: " + lengthOfArray);  
  
5. Declare an array with one element and Add a second element with index in array. Create an alert whose message is the new element.  
Answer: Declare an array with one element and add a second element with an index in the array. Create an alert whose message is the new element:  
let myArray = ["First Element"];

myArray[1] = "Second Element";

alert(myArray[1]); // This will alert "Second Element"  
  
In this code, we first declare an array with one element, and then we add a second element at index 1. Finally, we display the second element in an alert.  
  
  
Chapter 16 (Array II)  
  
1. Code an array with 1 string element. Add an additional element to the array using push. Create an alert whose message is the last element.  
  
Answer: Code an array with 1 string element, add an additional element to the array using **push**, and create an alert whose message is the last element:  
let myArray = ["First Element"];

myArray.push("Second Element"); // Add a new element to the end of the array

alert(myArray[myArray.length - 1]); // This will alert "Second Element" (the last element)  
  
  
2. var Alphabet=["h","i","j","k"] Remove the last element from the array Alphabet.  
  
Answer: Remove the last element from the array **Alphabet**:  
var Alphabet = ["h", "i", "j", "k"];

Alphabet.pop(); // Remove the last element from the array

console.log(Alphabet); // This will print the updated array without the last element  
  
  
3. var Alphabet=["h","i","j","k"] Add a new element, a number, to the end of an array.  
  
Answer: Add a new element, a number, to the end of an array:  
var Alphabet = ["h", "i", "j", "k"];

Alphabet.push(42); // Add the number 42 to the end of the array

console.log(Alphabet); // This will print the updated array with the new element  
  
In the above code examples, **push** is used to add an element to the end of an array, and **pop** is used to remove the last element from the array.  
  
  
  
  
Chapter 16 (Array III)  
  
1. var sizes = ["S", "M", "XL", "XXL", "XXXL"]. Remove the first element of an array.  
  
Answer: Remove the first element of the array **sizes**:  
var sizes = ["S", "M", "XL", "XXL", "XXXL"];

sizes.shift(); // Remove the first element of the array

console.log(sizes); // This will print the updated array without the first element  
  
2. var sizes = ["S", "M", "XL", "XXL", "XXXL"]. Add three number elements to the beginning of an array.  
  
Answer: Add three number elements to the beginning of the array **sizes**:  
var sizes = ["S", "M", "XL", "XXL", "XXXL"];

sizes.unshift(1, 2, 3); // Add three number elements to the beginning of the array

console.log(sizes); // This will print the updated array with the new elements at the beginning  
  
3. Declare an array with one element. Add a second element to the beginning of the array. Create an alert whose message is the new first element.  
  
Answer: Declare an array with one element, add a second element to the beginning of the array, and create an alert whose message is the new first element:  
let myArray = ["First Element"];

myArray.unshift("Second Element"); // Add a new element to the beginning of the array

alert(myArray[0]); // This will alert "Second Element" (the new first element)  
  
4. var sizes = ["S", "M", "XL", "XXL", "XXXL"]. Insert "L" into the array between "M" and "XL".  
  
Answer: Insert "L" into the array **sizes** between "M" and "XL":  
var sizes = ["S", "M", "XL", "XXL", "XXXL"];

sizes.splice(2, 0, "L"); // Insert "L" at index 2 (between "M" and "XL")

console.log(sizes); // This will print the updated array with "L" inserted  
  
  
5. var sizes = ["S", "M", "XL", "XXL", "XXXL"]. Copy the first 3 sizes of the array and put them into a new array, regSizes.  
  
Answer: Copy the first 3 sizes of the array **sizes** and put them into a new array **regSizes**:  
var sizes = ["S", "M", "XL", "XXL", "XXXL"];

var regSizes = sizes.slice(0, 3); // Copy the first 3 sizes into a new array

console.log(regSizes); // This will print the new array containing the first 3 sizes  
  
  
6. var pets = ["dog", "cat", "ox", "duck", "frog"]. Add 2 elements after "dog" and remove "cat", "ox", and "duck".  
  
Answer: Add 2 elements after "dog" and remove "cat," "ox," and "duck" from the **pets** array:  
var pets = ["dog", "cat", "ox", "duck", "frog"];

pets.splice(1, 3, "elephant", "giraffe"); // Add 2 elements after "dog" and remove "cat," "ox," and "duck"

console.log(pets); // This will print the updated array  
  
7. var pets = ["dog", "cat", "ox", "duck", "frog"]; Remove "cat" and "ox".  
  
Remove "cat" and "ox" from the **pets** array:  
var pets = ["dog", "cat", "ox", "duck", "frog"];

pets.splice(1, 2); // Remove "cat" and "ox"

console.log(pets); // This will print the updated array without "cat" and "ox"  
  
  
8. var pets = ["dog", "cat", "ox", "duck", "frog", "flea"]; Reduce it to "duck" and "frog" using slice.  
  
Answer: Reduce the **pets** array to "duck" and "frog" using **slice**:  
  
var pets = ["dog", "cat", "ox", "duck", "frog", "flea"];

var reducedPets = pets.slice(3, 5); // Reduce the array to "duck" and "frog"

console.log(reducedPets); // This will print the new array containing "duck" and "frog"  
  
  
  
  
Chapter 17 - 20 (for Loops)  
  
1. Write a statement in which loop is to run 10 times.  
  
Answer: for (let i = 0; i < 10; i++) {

// Loop code here

}  
  
2. Code the first line of a for loop with the usual counter name, usual starting value, and usual increment. Run it 12 times using <= to specify how many loops.  
  
Answer: Code the first line of a for loop with the usual counter name, starting value, and increment. Run it 12 times using **<=** to specify how many loops:  
for (let i = 0; i <= 12; i++) {

// Loop code here

}  
  
  
3. What are the 5 characters missing from this code, excluding any spaces that are missing? Type them in order, with no spaces or commas between them. for var i = 0 i <= 4 i++ Note: Complete this statement by yourself  
  
Answer: To complete the code with the missing characters:  
for (var i = 0; i <= 4; i++) {

// Loop code here

}  
  
  
4. Code the first line of a for loop with a counter name that's not i. Code the usual starting value and usual increment. Run it 100 times using < to specify how many loops.  
  
Answer: Code the first line of a for loop with a counter name that's not **i**, the usual starting value, and the usual increment. Run it 100 times using **<** to specify how many loops:  
for (let count = 0; count < 100; count++) {

// Loop code here

}  
  
  
5. Code the first line of a for loop with the usual counter and the usual starting value. Run it 3 times using > to specify how many loops. Decrement it with each iteration.  
  
Answer: Code the first line of a for loop with the usual counter, starting value, and run it 3 times using **>** to specify how many loops. Decrement it with each iteration:  
for (let i = 3; i > 0; i--) {

// Loop code here

}  
  
6. The statement assigns the number of elements in the array to the variable.  
  
Answer: The statement that assigns the number of elements in the array to the variable would typically look like this:  
var arrayLength = yourArray.length;  
  
7. Set a variable named “flag” with an initial Boolean value of your choice.  
  
Answer: Set a variable named "flag" with an initial Boolean value of your choice:  
var flag = true; // or false, depending on your choice  
  
  
8. Code the first line of a for loop with the usual counter, the usual starting value, and the usual incrementing. Limit the number of loops by the number of elements in the array pets.  
  
Answer: Code the first line of a for loop with the usual counter, starting value, and increment. Limit the number of loops by the number of elements in the array **pets**:  
for (let i = 0; i < pets.length; i++) {

// Loop code here

}  
  
9. Coding Exercise: Set a for loop to run 10 iterations. On the second iteration, display the counter in an alert. (It should be 1.) Break out of the loop.  
  
Answer: Coding Exercise:  
for (let i = 0; i < 10; i++) {

if (i === 1) {

alert(i);

break;

}

}  
  
  
  
  
10. Create an array which contains user names Code a prompt with the message "Enter first name". The user's response is assigned to firstName. Code the first line of a for loop with the usual counter, the usual starting value, and the usual incrementing. Limit the number of loops by the number of elements in the array user names. Code an if statement that tests the presense of (user name) in an array. If user name match Alert that "Enter". if not then alert ("Please write correct user name").  
  
Answer: Create an array that contains user names, code a prompt to enter the first name, and then use a for loop to check if the entered name matches any name in the array:  
var userNames = ["Alice", "Bob", "Charlie", "David"];

var firstName = prompt("Enter first name");

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

if (firstName === userNames[i]) {

alert("Enter");

break;

} else {

alert("Please write the correct user name");

break;

}

}  
  
  
  
  
11. Complete this code to display an alert if a match isn't found. var matchFound = false; for (var i = 0; i < list.length; i++) { if (userInput === list[i]) { alert("Match found"); matchFound = true; break; } };  
  
  
Answer: Complete this code to display an alert if a match isn't found:  
var matchFound = false;

for (var i = 0; i < list.length; i++) {

if (userInput === list[i]) {

alert("Match found");

matchFound = true;

break;

}

}

if (!matchFound) {

alert("Match not found");

}  
  
  
  
12. var firstArr = [“a”, “b”, “c”, “d”, “e”, “f”]; var secondArr = [1, 2, 3, 4, 5, 6]; Code the first line of a for loop with the usual counter, the usual starting value, and the usual incrementing. Limit the number of loops by the number of elements in the array firstArr. In the scope of main loop Code the nested loop. Limit the number of nested loops by the number of elements in the array secondArr. After that concatenate the both loops. Expected Output: a1 a2 a3 a4 … f6  
  
Answer: Code to concatenate elements from two arrays as described:  
var firstArr = ["a", "b", "c", "d", "e", "f"];

var secondArr = [1, 2, 3, 4, 5, 6];

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

for (let j = 0; j < secondArr.length; j++) {

console.log(firstArr[i] + secondArr[j]);

}

}  
  
This code will concatenate each element from the first array with each element from the second array, producing the expected output.

Chapter 21 (Changing Case)  
  
1. Type the characters that are missing from this code. var allLower = userInput.toLowerCase; Note: Correct this statement by yourself.  
  
Answer: To correct the statement, you should add parentheses to the **toLowerCase** method to make it a function call, like this:  
var allLower = userInput.toLowerCase();  
  
2. Convert the string represented by x to lower-case and assign the result to the same variable.  
  
Answer: To convert the string represented by variable **x** to lower-case and assign the result to the same variable, you can do this:  
x = x.toLowerCase();  
  
3. Convert the string represented by y to upper-case and assign the result to the same variable.  
  
Answer: To convert the string represented by variable **y** to upper-case and assign the result to the same variable, you can do this:  
y = y.toUpperCase();  
  
4. Convert the string represented by a variable to lower-case and assign the result to a second variable that hasn't been declared beforehand.  
  
Answer: To convert the string represented by a variable to lower-case and assign the result to a second variable that hasn't been declared beforehand, you can do something like this:  
var originalString = "YourOriginalString";

var lowerCaseString = originalString.toLowerCase();  
  
5. Convert the string represented by an array element to lower-case and assign it to a variable that hasn't been declared beforehand.  
  
Answer: To convert the string represented by an array element to lower-case and assign it to a variable that hasn't been declared beforehand, you can do something like this:  
var myArray = ["SomeString"];

var newArrayElement = myArray[0].toLowerCase();  
  
6. Display in an alert the upper-case version of the string represented by a variable.  
  
Answer: To display in an alert the upper-case version of the string represented by a variable, you can do this:  
var myString = "YourString";

alert(myString.toUpperCase());  
  
7. var cityName = “kaRacHi”; Convert the string represented by a cityName in Capitalisation is the writing of a word with its first letter in uppercase and the remaining letters in lowercase.  
  
Answer: To convert the string represented by the **cityName** variable to Capitalisation, you can do this:  
var cityName = "kaRacHi";

cityName = cityName.charAt(0).toUpperCase() + cityName.slice(1).toLowerCase();  
  
This will capitalize the first letter and make the rest of the letters lowercase, resulting in "Karachi."  
  
  
Chapter 22 - 25 (Strings)  
  
1. "captain" has been assigned to variable “sameWords”. You want to slice "ap" out of it.  
  
Answer: To slice "ap" out of the string assigned to the variable **sameWords**, you can do this:  
var sameWords = "captain";

var slicedPart = sameWords.slice(1, 3); // This will slice "ap"  
  
  
2. The number of characters in the string will be assigned to the variable.  
  
Answer: To find the number of characters in a string and assign it to a variable, you can do this:  
  
var myString = "SomeString";

var stringLength = myString.length; // This will store the number of characters in the string  
  
  
3. The string "elephant" has been assigned to the variable animal. Slice the four middle characters out of the string and assign it to the variable seg, which hasn't been declared beforehand.  
  
Answer: To slice the four middle characters out of the string assigned to the variable **animal** and assign it to a new variable **seg**, you can do this:  
var animal = "elephant";

var seg = animal.slice(1, 5); // This will slice "leph"  
  
  
4. Find the number of characters in the string represented by a variable and assign the number to a second variable.  
  
Answer: To find the number of characters in the string represented by a variable and assign the number to a second variable, you can do this:  
var myString = "SomeString";

var stringLength = myString.length; // This will store the number of characters in the string  
  
  
5. In a first statement measure how many characters there are in a string represented by a variable. In a second statement slice all but the first character and last 3 characters of the string and assign it to a second variable that hasn't been declared beforehand.  
  
Answer: To measure how many characters there are in a string represented by a variable and slice all but the first character and last 3 characters of the string and assign it to a second variable, you can do this:  
var myString = "SomeString";

var stringLength = myString.length; // This will store the number of characters in the string

var slicedString = myString.slice(1, stringLength - 3); // This will slice the string as described  
  
  
6. var text = "To be or not to be."; var indx = text.indexOf("be"); What is the value of indx?  
  
Answer: The value of **indx** will be 3 because "be" starts at index 3 in the string.  
  
  
7. var text = "To be or not to be."; var indx = text.lastIndexOf("be"); What is the value of indx? Note: Try the above both examples by yourself.  
  
Answer: The value of **indx** will be 15 because the last occurrence of "be" starts at index 15 in the string.  
  
  
8. Find the index of the first character of the last instance of "go" in the string represented by the variable text and assign the number to the variable indx, which hasn't been declared beforehand.  
  
Answer: To find the index of the first character of the last instance of "go" in the string represented by the variable **text** and assign it to a new variable **indx**, you can do this:  
var indx = text.lastIndexOf("go");  
  
  
9. Code the first line of an if statement that tests whether a segment with an index represented by indexNum exists in a string.  
  
Answer: To code the first line of an if statement that tests whether a segment with an index represented by **indexNum** exists in a string, you can do this:  
if (typeof yourString[indexNum] !== 'undefined') {

// Code to handle the segment's existence

}  
  
  
10. In this string "abcde", what character is at index 2? (Use charAt)  
  
Answer: In the string "abcde," the character at index 2 is "c."  
  
11. Find the 10th character in the string represented by text and assign it to the variable cha, which hasn't been declared beforehand.  
  
Answer: To find the 10th character in the string represented by **text** and assign it to a new variable **cha**, you can do this:  
var cha = text.charAt(9);  
  
  
12. Find the last character in the string represented by str and assign it to x, which hasn't been declared beforehand.  
  
Answer: To find the last character in the string represented by **str** and assign it to a new variable **x**, you can do this:  
var x = str.charAt(str.length - 1);  
  
  
13. Find the the 5th character in a string represented by input and assign it to cha, which hasn't been declared beforehand.  
  
Answer: To find the 5th character in a string represented by **input** and assign it to a new variable **cha**, you can do this:  
var cha = input.charAt(4);  
  
  
14. Code the first line of an if statement that tests whether the 3rd character of a string represented by a variable is a particular character.  
  
Answer: To code the first line of an if statement that tests whether the 3rd character of a string represented by a variable is a particular character, you can do this:  
if (yourString.charAt(2) === 'YourCharacter') {

// Code to handle the condition

}  
  
  
15. Code a for loop that cycles through all the characters of a string represented by a variable and assigns each character to an element of an array that has been declared beforehand. In the string represented by reply replace the first instance of "no" with "yes" and assign the revised string to revisedReply, which hasn't been declared beforehand.  
  
Answer: To code a for loop that cycles through all the characters of a string represented by a variable and assigns each character to an element of an array that has been declared beforehand, you can do this:  
var myString = "YourString";

var charArray = [];

for (var i = 0; i < myString.length; i++) {

charArray.push(myString.charAt(i));

}  
  
  
16. In a string represented by str replace the first instance of "1" with "one" and assign the revised string to newStr, which hasn't been declared beforehand.  
  
Answer: To replace the first instance of "1" with "one" in a string represented by **str** and assign the revised string to a new variable **newStr**, you can do this:  
var newStr = str.replace("1", "one");  
  
  
17. If you want all instances replaced, enter 3 characters that need to appear in this statement. var y = x.replace("a", "z");  
  
Answer: To replace all instances of a particular character in a string, you would need to use a regular expression with the global flag. For example:  
var y = x.replace(/a/g, "z"); // This will replace all instances of "a" with "z"  
  
  
Chapter 26 (Rounding Numbers)  
  
1. Form a statement that rounds a number to the nearest integer.  
  
Answer: To round a number to the nearest integer, you can use the **Math.round()** function:  
var roundedNumber = Math.round(yourNumber);  
  
  
2. Round up a number represented by origNum and assign it to roundNum, which hasn't been declared beforehand.  
  
Answer: To round up a number represented by **origNum** and assign it to a new variable **roundNum**, you can use the **Math.ceil()** function:  
var roundNum = Math.ceil(origNum);  
  
  
3. Round down a number represented by origNum and assign it to roundNum, which hasn't been declared beforehand.  
  
Answer: To round down a number represented by **origNum** and assign it to a new variable **roundNum**, you can use the **Math.floor()** function:  
var roundNum = Math.floor(origNum);  
  
  
4. Round a number represented by a variable and assign the result to a second variable that hasn't been declared beforehand.  
  
Answer: To round a number represented by a variable and assign the result to a second variable, you can use either **Math.round()**, **Math.ceil()**, or **Math.floor()** depending on your rounding preference. For example:  
var originalNumber = 7.75;

var roundedNumber = Math.round(originalNumber); // Rounds to the nearest integer  
  
  
5. Round .5 to 0 and assign it to a variable that hasn't been declared beforehand.  
  
Answer: To round 0.5 to 0 and assign it to a variable that hasn't been declared beforehand, you can do this:  
var roundedValue = Math.floor(0.5);  
  
  
  
Chapter 27 (Random Numbers)  
  
1. Convert a random number generated by JavaScript to a number in the range 1 to 50  
  
Answer: To convert a random number generated by JavaScript to a number in the range 1 to 50, you can use the **Math.random()** function in combination with multiplication and addition like this:  
var randomNum = Math.floor(Math.random() \* 50) + 1; // This generates a random number between 1 and 50 (inclusive).  
  
2. Generate a random number and assign it to a variable that hasn't been declared beforehand.  
  
Answer: To generate a random number and assign it to a variable that hasn't been declared beforehand, you can do this:  
var randomNumber = Math.random(); // Generates a random number between 0 (inclusive) and 1 (exclusive).  
  
3. You have to create a dice in JavaScript with the use of pseudorandom number.  
  
Answer: To create a JavaScript dice roll using a pseudo-random number, you can use the following code to simulate a 6-sided dice:  
var diceRoll = Math.floor(Math.random() \* 6) + 1; // Generates a random number between 1 and 6.  
  
4. You have to create a toss (head/tail) in JavaScript with the use of pseudo-random number.  
  
Answer: To create a JavaScript coin toss (head/tail) using a pseudo-random number, you can use the following code to simulate a fair coin toss:  
var coinToss = Math.random() < 0.5 ? "head" : "tail"; // Generates "head" with a 50% chance and "tail" with a 50% chance.  
  
These code snippets use **Math.random()** to generate pseudo-random numbers between 0 (inclusive) and 1 (exclusive), and then they manipulate these numbers to achieve the desired ranges and outcomes.  
  
  
  
Chapter 28, 29 (Converting Strings)  
  
1. How do you convert a string to an integer in JavaScript?  
  
Answer: To convert a string to an integer in JavaScript, you can use the **parseInt()** function.   
  
2. Write a JavaScript function to convert the string "123" to an integer.  
  
Answer: To convert the string "123" to an integer using JavaScript, you can use the **parseInt()** function like this:  
var str = "123";

3. How can you convert a string containing a decimal number to a floating-point number in JavaScript?  
  
Answer: To convert a string containing a decimal number to a floating-point number in JavaScript, you can use the **parseFloat()** function or simply use the **Number()** constructor.

Using **parseFloat()**:  
  
var str = "3.14";

var num = parseFloat(str);  
  
Using **Number()**:  
  
var str = "3.14";

var num = Number(str);

4. How can you check if a string can be successfully converted to an integer or decimal in JavaScript before performing the conversion?  
  
Answer: To check if a string can be successfully converted to an integer or decimal in JavaScript before performing the conversion, you can use **parseInt()** or **parseFloat()** and check the result for **NaN** (Not-a-Number). For example:

To check if a string can be converted to an integer:  
var str = "123";

var num = parseInt(str);

if (!isNaN(num)) {

// Conversion successful

} else {

// Conversion failed

}  
  
To check if a string can be converted to a floating-point number:  
  
var str = "3.14";

var num = parseFloat(str);

if (!isNaN(num)) {

// Conversion successful

} else {

// Conversion failed

}

5. How can you convert a number to a string in JavaScript?  
  
Answer: To convert a number to a string in JavaScript, you can use the **toString()** method or concatenate it with an empty string (**""**).  
  
6. Write a JavaScript function to convert the number 42 to a string.  
  
Answer: To convert the number 42 to a string using JavaScript, you can use the **toString()** method like this:  
var num = 42;

var str = num.toString();  
  
  
7. Can you convert a string representing a decimal number (e.g., "3.14") to an integer in JavaScript? If so, how?  
  
Answer: Yes, you can convert a string representing a decimal number (e.g., "3.14") to an integer in JavaScript, but it will truncate the decimal part. You can use **parseInt()** for this, but it will only consider the whole part of the number:  
var str = "3.14";

var num = parseInt(str); // num will be 3  
  
If you want to round the decimal number and then convert it to an integer, you can do something like this:  
var str = "3.14";

var num = Math.round(parseFloat(str)); // num will be 3  
  
  
Chapter 30 (Controlling the length of decimals)  
  
1. Code a statement that rounds a number represented by num to 4 places, converts it to a string, and assigns it to newNum, which hasn't been declared beforehand.  
  
Answer: To round a number represented by **num** to 4 decimal places, convert it to a string, and assign it to a new variable **newNum**, you can do this:  
var num = 12.3456789;

var newNum = num.toFixed(4); // This rounds to 4 decimal places and converts to a string  
  
  
2. In a single statement round a number represented by a variable to 2 places, convert it to a string, convert it back to a number, and assign it to the same variable.  
  
Answer: To round a number represented by a variable to 2 decimal places, convert it to a string, then back to a number, and assign it to the same variable, you can do this in a single statement:  
var myNumber = 7.8965;

myNumber = parseFloat(myNumber.toFixed(2));  
  
  
3. Code the first line of an if statement that tests whether the number represented by num, rounded to 2 digits and converted to a string, has more than 4 characters in it.  
  
Answer: To code the first line of an if statement that tests whether the number represented by **num**, rounded to 2 digits and converted to a string, has more than 4 characters in it, you can do this:  
if (num.toFixed(2).length > 4) {

// Code to handle the condition

}  
  
  
4. Assign a number with many decimal places to a variable. Code an alert that displays the number rounded to 2 decimal places and converted to a string.  
  
Answer: To assign a number with many decimal places to a variable and then display the number rounded to 2 decimal places and converted to a string using an alert, you can do this:  
var myNumber = 123.456789;

var roundedNumber = myNumber.toFixed(2);

alert(roundedNumber); // This will display "123.46"  
  
  
Chapter 31 - 34 (Date & Time)  
  
1. Code a statement that creates a new Date object and assigns it to dObj, which hasn't been declared beforehand.  
  
Answer: To create a new Date object and assign it to a variable **dObj**, you can do this:  
var dObj = new Date();  
  
2. Code a statement that creates a new Date object, converts it to a string, and assigns the string to dStr, which hasn't been declared beforehand.  
  
Answer: To create a new Date object, convert it to a string, and assign the string to a variable **dStr**, you can do this:  
var d = new Date();

var dStr = d.toString();  
  
3. Code a statement that extracts the day of the week from a Date object represented by d and assigns it to day, which hasn't been declared beforehand.  
  
Answer: To extract the day of the week from a Date object represented by **d** and assign it to a variable **day**, you can do this:  
var day = d.getDay();  
  
4. The day has been extracted from the Date object and assigned to d. The names of the days of the week have been assigned to the array dayNames. Alert the current day with array index.  
  
Answer: To alert the current day using the array **dayNames** based on the index from the **getDay()** method:  
var dayNames = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];

var dayIndex = d.getDay();

var currentDay = dayNames[dayIndex];

alert(currentDay);  
  
5. Extract all parts of the Date and Time and assign it to the variable which has not been declared beforehand.  
  
Answer: To extract all parts of the Date and Time and assign them to a variable that hasn't been declared beforehand, you can do this:  
var currentDate = new Date();  
  
6. Code a statement that creates a Date object for the last day of the last month of 2020 and assigns it to later, which hasn't been declared beforehand.  
  
Answer: To create a Date object for the last day of the last month of 2020 and assign it to a variable **later**, you can do this:  
var later = new Date(2020, 11, 31); // Months are zero-based, so 11 represents December.  
  
7. Create a Date object for the second day of the second month of 1992 and assign it to a variable that hasn't been declared beforehand.  
  
Answer: To create a Date object for the second day of the second month of 1992 and assign it to a variable, you can do this:  
var specificDate = new Date(1992, 1, 2); // Months are zero-based, so 1 represents February.  
  
8. Code a single statement that displays in an alert the milliseconds that elapsed between the reference date and the beginning of 1980.  
  
Answer: To display in an alert the milliseconds that elapsed between the reference date and the beginning of 1980, you can do this:  
var referenceDate = new Date(1980, 0, 1); // January is month 0.

var elapsedTime = referenceDate.getTime();

alert(elapsedTime);  
  
9. How do you change the year of a date in JavaScript?  
  
Answer: To change the year of a date in JavaScript, you can use the **setFullYear()** method:  
var myDate = new Date();

myDate.setFullYear(2022); // Changes the year to 2022  
  
10. Write a JavaScript function to change the month of a given date to January.  
  
Answer: To change the month of a given date to January, you can use the **setMonth()** method:  
var myDate = new Date();

myDate.setMonth(0); // Sets the month to January (0-indexed)  
  
11. What is the method to change the day of the week for a specific date in JavaScript?  
  
Answer: You cannot directly change the day of the week for a specific date in JavaScript, as the day of the week is determined by the date itself. It's not something you can manipulate independently.  
  
12. Write a JavaScript function to change the minutes of a given time to a specific value. (Value by prompt)  
  
Answer: To change the minutes of a given time to a specific value, you can use the **setMinutes()** method:  
var myDate = new Date();

myDate.setMinutes(30); // Sets the minutes to 30  
  
13. Write a JavaScript function to add a specific number of hours to a given time.  
  
Answer: To add a specific number of hours to a given time, you can use the **setHours()** method:  
var myDate = new Date();

myDate.setHours(myDate.getHours() + 3); // Adds 3 hours to the current time  
  
14. You have to create a age calculator in JavaScript.  
  
Answer: To create an age calculator in JavaScript, you can create a function that takes a birthdate as input and calculates the age based on the current date. Here's a simple example:  
function calculateAge(birthdate) {

var currentDate = new Date();

var birthDate = new Date(birthdate);

var age = currentDate.getFullYear() - birthDate.getFullYear();

// Adjust age if the birthdate hasn't occurred yet this year

if (currentDate.getMonth() < birthDate.getMonth() || (currentDate.getMonth() === birthDate.getMonth() && currentDate.getDate() < birthDate.getDate())) {

age--;

}

return age;

}

// Usage:

var birthdate = "1990-06-15"; // Replace with the birthdate in the format "YYYY-MM-DD"

var age = calculateAge(birthdate);

console.log("Age: " + age + " years");

This function calculates the age by comparing the birthdate with the current date, taking into account whether the birthday has occurred this year or not.  
  
  
Chapter 35 - 37 (Functions)  
  
  
1. Code the first line of a function displayAlert.  
  
Answer: To code the first line of a function named **displayAlert**, you can do this:  
function displayAlert() {  
  
2. Code a function named askName that prompts the user to "Enter name" and assigns the answer to userName, which hasn't been declared beforehand.  
  
Answer: To code a function named **askName** that prompts the user to "Enter name" and assigns the answer to **userName**, you can do this:  
function askName() {

var userName = prompt("Enter name");

}  
  
  
3. Code a function that calls 2 other functions.  
  
Answer: To code a function that calls two other functions, you can do this:  
function callTwoFunctions() {

function1();

function2();

}  
  
  
4. Code a function that displays a prompt, "Enter name" and then displays the name in an alert. Call the function.  
  
Answer: To code a function that displays a prompt "Enter name" and then displays the name in an alert and call the function, you can do this:  
function displayName() {

var name = prompt("Enter name");

alert("Hello, " + name);

}

displayName(); // Call the function  
  
  
5. Code the first line of a function named concat that has 3 parameters, the first three letters of the alphabet. Call that takes a variable, a string, and a number as arguments.  
  
Answer: To code the first line of a function named **concat** that has three parameters, you can do this:  
function concat(param1, param2, param3) {  
  
  
6. Code a function that has 2 parameters. Concatenate them and assign the result to a variable that hasn't been declared beforehand.  
  
Answer: To code a function that concatenates two parameters and assigns the result to a variable, you can do this:  
function concatenateStrings(str1, str2) {

var result = str1 + str2;

return result;

}  
  
  
7. Code a function that has three parameters. Multiply them and assign them to a variable that hasn't been declared yet.  
  
Answer: To code a function that has three parameters, multiplies them, and assigns the result to a variable, you can do this:  
function multiplyThreeNumbers(num1, num2, num3) {

var result = num1 \* num2 \* num3;

return result;

}  
  
  
8. Write a JavaScript function that takes an array of numbers as input and returns the average of those numbers.  
  
Answer: To write a JavaScript function that takes an array of numbers as input and returns the average of those numbers, you can do this:  
function calculateAverage(numbers) {

var sum = 0;

for (var i = 0; i < numbers.length; i++) {

sum += numbers[i];

}

var average = sum / numbers.length;

return average;

}  
  
  
9. Write a JavaScript function that takes two parameters and returns their sum.  
  
Answer: To write a JavaScript function that takes two parameters and returns their sum, you can do this:  
function addNumbers(num1, num2) {

return num1 + num2;

}  
  
  
10. Write a JavaScript function that takes an array of numbers as input and returns the average of those numbers.  
  
Answer: To write a JavaScript function that takes an array of numbers as input and returns the average of those numbers, you can reuse the **calculateAverage** function from question 8.  
  
11. You have to capture the returned value from a function and store it in a variable?  
  
Answer: To capture the returned value from a function and store it in a variable, you can do this:  
var result = someFunction(); // Store the returned value in the 'result' variable  
  
12. Write a function which tells letter counts of (word).  
  
Answer: To write a function that tells the letter counts of a word, you can do this:  
function letterCount(word) {

return word.length;

}  
  
  
13. Write a function to set (year) in date object.  
  
Answer: To write a function to set the year in a date object, you can do this:  
function setYear(dateObject, year) {

dateObject.setFullYear(year);

}  
  
  
14. Write a function which tells the age of a person who Born on (dateOfBirth)  
  
Answer: To write a function that tells the age of a person who was born on a given date of birth, you can do this:  
function calculateAge(dateOfBirth) {

var birthDate = new Date(dateOfBirth);

var currentDate = new Date();

var age = currentDate.getFullYear() - birthDate.getFullYear();

// Adjust age if the birthday hasn't occurred yet this year

if (currentDate.getMonth() < birthDate.getMonth() || (currentDate.getMonth() === birthDate.getMonth() && currentDate.getDate() < birthDate.getDate())) {

age--;

}

return age;

}  
  
  
15. Write a function which tells the presense of (word) in an array = ['zaid','haris','raza','abubakar','hassan','hussain','fatima'] result should be in true or false  
  
Answer: To write a function that tells the presence of a word in an array and returns **true** or **false**, you can do this:  
function checkWordPresence(word, array) {

return array.includes(word);

}  
  
  
16. Write a function which repeat (letter) 10 times. Hint: "abcde" str.repeat(10)  
  
Answer: To write a function that repeats a letter 10 times, you can do this:  
function repeatLetter(letter) {

return letter.repeat(10);

}  
  
  
17. write a function which reverse array = ['a','b','c','d','e'] Hint: arr.reverse()  
  
Answer: To write a function that reverses an array, you can use the **reverse()** method:  
function reverseArray(arr) {

return arr.reverse();

}  
  
Remember to call these functions with appropriate arguments as needed in your code.  
  
  
  
Chapter 38 (Local vs. Global Variables)  
  
1. Write a JavaScript function that demonstrates the usage of a local variable.  
  
Answer: To demonstrate the usage of a local variable in a JavaScript function, you can define a variable inside the function, which will only be accessible within that function. Here's an example:  
function myFunction() {

var localVar = "I'm a local variable";

console.log(localVar);

}

myFunction(); // Call the function to demonstrate the local variable  
  
In this example, **localVar** is a local variable defined within the **myFunction** function. It cannot be accessed outside of that function.  
  
  
2. How can you access a global variable inside a function in JavaScript?  
  
Answer: You can access a global variable inside a function in JavaScript directly by referencing it with its name. Global variables are accessible from anywhere in your code, including inside functions. Here's an example:  
  
var globalVar = "I'm a global variable"; // This is a global variable

function myFunction() {

console.log(globalVar); // Accessing the global variable inside the function

}

myFunction(); // Call the function to access the global variable  
  
  
In this example, **globalVar** is a global variable defined outside the function **myFunction**. It can be accessed inside the function without any issues.  
  
  
  
Chapter 39, 40 (Switch Statements)  
  
1. Write a JavaScript switch statement that checks the value of a variable and performs different actions based on different cases.  
  
Answer: A switch statement that checks the value of a variable and performs different actions based on different cases:  
var dayOfWeek = "Monday";

switch (dayOfWeek) {

case "Monday":

console.log("It's the first day of the week.");

break;

case "Tuesday":

console.log("It's the second day of the week.");

break;

case "Wednesday":

console.log("It's the middle of the week.");

break;

default:

console.log("It's not a recognized day.");

}  
  
In this example, the **switch** statement checks the value of the **dayOfWeek** variable and performs different actions based on the value.  
  
  
2. Write a JavaScript switch statement that check whether cityName given by user check the cityName if match alert the user and break the statement, if not default message will be shown to user.  
  
Answer: A switch statement that checks whether the **cityName** given by the user matches a specific city and alerts the user accordingly:  
  
var cityName = prompt("Enter a city name:");

switch (cityName) {

case "Paris":

alert("You entered Paris!");

break;

case "London":

alert("You entered London!");

break;

case "New York":

alert("You entered New York!");

break;

default:

alert("City not recognized.");

}  
  
In this example, the **switch** statement checks the value of the **cityName** variable based on user input and shows an alert message depending on the input.  
  
  
  
Chapter 41, 42 (while loops, do…while loops)  
(All Questions must be done with both loops)  
  
  
1. Write a Code that runs twice and does nothing. The counter i has already been declared and assigned 0.  
  
Answer: Code that runs twice and does nothing:  
// Using a while loop

let i = 0;

while (i < 2) {

// Do nothing

i++;

}

// Using a do...while loop

i = 0;

do {

// Do nothing

i++;

} while (i < 2);  
  
  
2. Code that looks for "pig" in the array. When it finds it, an alert displays saying, "Found it!" Use the length of the array as the loop limiter. Break out of the loop when it's found. var animals=["horse", "ox", "cow", "pig", "duck"];  
  
Answer: Code that looks for "pig" in the array and displays an alert when found:  
var animals = ["horse", "ox", "cow", "pig", "duck"];

let i = 0;

// Using a while loop

while (i < animals.length) {

if (animals[i] === "pig") {

alert("Found it!");

break; // Exit the loop when "pig" is found

}

i++;

}

// Using a do...while loop

i = 0;

do {

if (animals[i] === "pig") {

alert("Found it!");

break; // Exit the loop when "pig" is found

}

i++;

} while (i < animals.length);  
  
  
3. Code to use a while & do while loop to print the numbers from 1 to 10.  
  
Answer: Code to print the numbers from 1 to 10 using both while and do...while loops:  
// Using a while loop

let i = 1;

while (i <= 10) {

console.log(i);

i++;

}

// Using a do...while loop

i = 1;

do {

console.log(i);

i++;

} while (i <= 10);  
  
  
4. To use a while loop to ask the user for a number and then print that number back to them.  
  
Answer: Using a while loop to ask the user for a number and then print that number back to them:  
let userInput;

let isValid = false;

// Using a while loop

while (!isValid) {

userInput = prompt("Enter a number:");

if (!isNaN(userInput)) {

isValid = true;

console.log("You entered: " + userInput);

} else {

alert("Please enter a valid number.");

}

}  
  
  
5. To use a while loop to check if a number is even or odd.  
  
Answer: Using a while loop to check if a number is even or odd:  
let number = 7; // Replace with the number you want to check

// Using a while loop

while (number > 0) {

if (number % 2 === 0) {

console.log(number + " is even.");

} else {

console.log(number + " is odd.");

}

number--;

}  
  
  
6. Create a guessing game where the user has to guess a number between 1 and 100.  
  
Answer: Creating a guessing game where the user has to guess a number between 1 and 100:  
const secretNumber = Math.floor(Math.random() \* 100) + 1;

let guess;

let attempts = 0;

// Using a while loop

while (guess !== secretNumber) {

guess = parseInt(prompt("Guess the number (1-100):"));

attempts++;

if (guess === secretNumber) {

console.log("Congratulations! You guessed the correct number: " + secretNumber);

console.log("Number of attempts: " + attempts);

} else if (guess < secretNumber) {

alert("Try a higher number.");

} else {

alert("Try a lower number.");

}

}  
  
  
7. Use a while & do-while loop to create a countdown timer?  
  
Answer: Using both a while and do-while loop to create a countdown timer:  
// Using a while loop

let countdown = 10;

while (countdown >= 0) {

console.log(countdown);

countdown--;

}

// Using a do...while loop

countdown = 10;

do {

console.log(countdown);

countdown--;

} while (countdown >= 0);  
  
These examples cover various scenarios using **while** and **do...while** loops in JavaScript. You can adapt and modify them as needed for your specific requirements.  
  
  
  
Chapter 46, 48 (Events)  
  
  
1. Create a simple event that prints a message when the user clicks on an element.  
  
Answer: <button id="myButton">Click me</button>

<script>

document.getElementById("myButton").addEventListener("click", function() {

console.log("Button clicked!");

});

</script>  
  
  
2. Create an event that hides an element when the user clicks on a different element  
  
Answer: <button id="hideButton">Hide Element</button>

<div id="elementToHide">This will be hidden.</div>

<script>

document.getElementById("hideButton").addEventListener("click", function() {

document.getElementById("elementToHide").style.display = "none";

});

</script>  
  
  
3. Create a link that changes color when the user hovers over it and then back to its original color when the user moves the mouse away.  
  
Answer: <a href="#" id="colorChangeLink">Hover over me</a>

<script>

const link = document.getElementById("colorChangeLink");

link.addEventListener("mouseover", function() {

link.style.color = "red";

});

link.addEventListener("mouseout", function() {

link.style.color = "blue";

});

</script>  
  
  
4. Create an event that scrolls to the top of the page when the user clicks on a link.  
  
Answer: <a href="#" id="scrollToTopLink">Scroll to Top</a>

<script>

const scrollToTopLink = document.getElementById("scrollToTopLink");

scrollToTopLink.addEventListener("click", function() {

window.scrollTo(0, 0);

});

</script>  
  
  
5. Create a link that plays a sound when the user clicks on it.  
  
Answer: <a href="#" id="playSoundLink">Play Sound</a>

<audio id="myAudio">

<source src="sound.mp3" type="audio/mp3">

</audio>

<script>

const playSoundLink = document.getElementById("playSoundLink");

const audio = document.getElementById("myAudio");

playSoundLink.addEventListener("click", function() {

audio.play();

});

</script>  
  
  
6. Create a simple link that opens a new window when clicked.  
  
Answer: <a href="https://www.example.com" target="\_blank">Open New Window</a>  
  
7. Create a simple button that alert a message when clicked.  
  
Answer: <button id="alertButton">Click me for an alert</button>

<script>

document.getElementById("alertButton").addEventListener("click", function() {

alert("Button clicked!");

});

</script>  
  
  
8. Create a button that changes color when the user hovers over it.  
  
Answer: <button id="colorChangeButton">Hover over me</button>

<script>

const colorChangeButton = document.getElementById("colorChangeButton");

colorChangeButton.addEventListener("mouseover", function() {

colorChangeButton.style.backgroundColor = "red";

});

colorChangeButton.addEventListener("mouseout", function() {

colorChangeButton.style.backgroundColor = "blue";

});

</script>  
  
  
9. Create a button that plays a sound when the user clicks on it.  
  
Answer: Create a button that plays a sound when the user clicks on it (similar to example 5).  
  
10. Create an event that alert a message when the user moves the mouse over an element.  
  
Answer: <div id="hoverElement">Hover over me</div>

<script>

const hoverElement = document.getElementById("hoverElement");

hoverElement.addEventListener("mouseover", function() {

alert("Mouse over the element!");

});

</script>  
  
  
11. Create an event that hides an element when the user moves the mouse out of it.  
  
Answer: <div id="hideOnMouseOut">Move the mouse out of me</div>

<script>

const hideOnMouseOut = document.getElementById("hideOnMouseOut");

hideOnMouseOut.addEventListener("mouseout", function() {

hideOnMouseOut.style.display = "none";

});

</script>  
  
  
12. Create a link that opens a new window when the user clicks on it and then closes the window when the user clicks on a close button in the new window. The size of the new window should be determined by the user's mouse position.  
  
Answer: Create a link that opens a new window when clicked and closes the window when the user clicks on a close button in the new window. The size of the new window should be determined by the user's mouse position (requires more complex JavaScript and is browser-dependent).  
  
  
  
Chapter 49, 50 (Reading and Setting field values)  
  
1. Read the value of a user Input and print it to the console.  
  
Answer: <input type="text" id="userInput" placeholder="Enter something">

<button onclick="readInputValue()">Read Input Value</button>

<script>

function readInputValue() {

const userInput = document.getElementById("userInput").value;

console.log("User input value: " + userInput);

}

</script>  
  
  
2. Read the value of a checkbox and print it to the console  
  
Answer: <input type="checkbox" id="checkBox"> Checkbox

<button onclick="readCheckboxValue()">Read Checkbox Value</button>

<script>

function readCheckboxValue() {

const checkBox = document.getElementById("checkBox");

console.log("Checkbox checked: " + checkBox.checked);

}

</script>  
  
  
3. Read the value of a select box and print it to the console.  
  
Answer: <select id="selectBox">

<option value="option1">Option 1</option>

<option value="option2">Option 2</option>

<option value="option3">Option 3</option>

</select>

<button onclick="readSelectValue()">Read Select Value</button>

<script>

function readSelectValue() {

const selectBox = document.getElementById("selectBox");

const selectedValue = selectBox.options[selectBox.selectedIndex].value;

console.log("Selected value: " + selectedValue);

}

</script>  
  
  
4. Place some text in a field. Make up an id for the field.  
  
Answer: <input type="text" id="customField">

<button onclick="placeTextInField()">Place Text</button>

<script>

function placeTextInField() {

document.getElementById("customField").value = "This is some text";

}

</script>  
  
  
5. Ask the user about Martial Status. If the variable married is false, place the value "available" in the field with the id "status"  
  
Answer: <input type="text" id="status">

<button onclick="setStatus()">Set Status</button>

<script>

let married = false; // You can set this variable based on user input

function setStatus() {

const statusField = document.getElementById("status");

statusField.value = married ? "Married" : "Available";

}

</script>  
  
  
6. If a input is empty, fill it with a string.  
  
Answer: <input type="text" id="emptyField" placeholder="Enter text">

<button onclick="fillEmptyField()">Fill Empty Field</button>

<script>

function fillEmptyField() {

const emptyField = document.getElementById("emptyField");

if (emptyField.value.trim() === "") {

emptyField.value = "Default Text";

}

}

</script>  
  
  
7. Create a Registeration Form with Validation.  
  
Answer: <form id="registrationForm" onsubmit="return validateForm()">

<input type="text" id="username" placeholder="Username">

<input type="password" id="password" placeholder="Password">

<input type="password" id="confirmPassword" placeholder="Confirm Password">

<button type="submit">Register</button>

</form>

<script>

function validateForm() {

const username = document.getElementById("username").value;

const password = document.getElementById("password").value;

const confirmPassword = document.getElementById("confirmPassword").value;

if (username === "" || password === "" || confirmPassword === "") {

alert("Please fill in all fields.");

return false;

}

if (password !== confirmPassword) {

alert("Passwords do not match.");

return false;

}

// Additional validation logic can be added here

alert("Registration successful!");

return true;

}

</script>  
  
  
This is a basic registration form with client-side validation. You can add more validation rules as needed for your specific requirements.  
  
  
  
Chapter 51 (Reading and Setting paragraph text)  
  
1. How can I create a paragraph that expands when the user clicks on a Show More?  
  
Answer: <p id="expandableParagraph">

This is some text that can be expanded.

</p>

<button onclick="expandParagraph()">Show More</button>

<script>

function expandParagraph() {

const paragraph = document.getElementById("expandableParagraph");

paragraph.style.overflow = "visible";

paragraph.style.height = "auto";

}

</script>  
  
  
2. Read the text of a paragraph and use it to search for a specific word or phrase.  
  
Answer: <p id="searchParagraph">

This is a sample paragraph to search within.

</p>

<button onclick="searchText('sample')">Search</button>

<script>

function searchText(searchTerm) {

const paragraphText = document.getElementById("searchParagraph").textContent;

if (paragraphText.includes(searchTerm)) {

alert(`Found "${searchTerm}" in the paragraph.`);

} else {

alert(`"${searchTerm}" not found in the paragraph.`);

}

}

</script>  
  
  
3. Read the text of a paragraph and use it to find the longest word.  
  
Answer: <p id="longestWordParagraph">

This is a sentence with some long words like "supercalifragilisticexpialidocious."

</p>

<button onclick="findLongestWord()">Find Longest Word</button>

<script>

function findLongestWord() {

const paragraphText = document.getElementById("longestWordParagraph").textContent;

const words = paragraphText.split(/\s+/);

let longestWord = "";

for (const word of words) {

if (word.length > longestWord.length) {

longestWord = word;

}

}

alert(`The longest word in the paragraph is "${longestWord}".`);

}

</script>  
  
  
4. Set the text of a paragraph to the value of a text field, but only if the value is not empty.  
  
Answer: <input type="text" id="textField" placeholder="Enter text">

<button onclick="setTextFromField()">Set Text</button>

<p id="textParagraph">Original text</p>

<script>

function setTextFromField() {

const textFieldValue = document.getElementById("textField").value;

if (textFieldValue !== "") {

document.getElementById("textParagraph").textContent = textFieldValue;

}

}

</script>  
  
  
5. Set the text of a paragraph to the value of a text field, but only if the value is equal to a specific word or phrase.  
  
Answer: <input type="text" id="textField" placeholder="Enter text">

<button onclick="setTextFromField()">Set Text</button>

<p id="textParagraph">Original text</p>

<script>

function setTextFromField() {

const textFieldValue = document.getElementById("textField").value;

if (textFieldValue === "specific\_word\_or\_phrase") {

document.getElementById("textParagraph").textContent = textFieldValue;

}

}

</script>  
  
  
6. Set the text of a paragraph to the value of a text field, but only if the value is less than a certain length.  
  
Answer: <input type="text" id="textField" placeholder="Enter text">

<button onclick="setTextFromField()">Set Text</button>

<p id="textParagraph">Original text</p>

<script>

function setTextFromField() {

const textFieldValue = document.getElementById("textField").value;

if (textFieldValue.length < 10) {

document.getElementById("textParagraph").textContent = textFieldValue;

}

}

</script>  
  
These examples demonstrate how to work with paragraph text in JavaScript to achieve various tasks. You can customize them further to meet your specific needs.  
  
  
  
  
Chapter 52 (Manipulating Images and Text)  
  
  
1. When the user clicks on the sentence, it turns bold.  
  
Answer: <p id="boldText" onclick="makeBold()">Click me to make bold!</p>

<script>

function makeBold() {

const text = document.getElementById("boldText");

text.style.fontWeight = "bold";

}

</script>  
   
  
2. How can I make an image disappear when the user hovers over it and show text instead? When the user hovers off the image, the original image should reappear.  
  
Answer: <div id="imageContainer" onmouseover="showText()" onmouseout="showImage()">

<img src="your-image.jpg" alt="Image">

<p id="imageText" style="display: none;">Hover over the image to see text.</p>

</div>

<script>

function showText() {

document.getElementById("imageText").style.display = "block";

document.querySelector("#imageContainer img").style.display = "none";

}

function showImage() {

document.getElementById("imageText").style.display = "none";

document.querySelector("#imageContainer img").style.display = "block";

}

</script>  
  
  
3. When the user mouses over the heading, its background turns black, but it retains its original light gray color  
  
Answer: <h1 id="hoverHeading" onmouseover="changeBackground(true)" onmouseout="changeBackground(false)">Hover me!</h1>

<style>

#hoverHeading {

background-color: lightgray;

transition: background-color 0.3s;

}

#hoverHeading.hovered {

background-color: black;

}

</style>

<script>

function changeBackground(isHovered) {

const heading = document.getElementById("hoverHeading");

if (isHovered) {

heading.classList.add("hovered");

} else {

heading.classList.remove("hovered");

}

}

</script>  
  
  
4. When user hover to a paragraph, Add a underline style to text.  
  
Answer: <p id="hoverParagraph" onmouseover="underlineText(true)" onmouseout="underlineText(false)">Hover over me to underline!</p>

<script>

function underlineText(shouldUnderline) {

const paragraph = document.getElementById("hoverParagraph");

if (shouldUnderline) {

paragraph.style.textDecoration = "underline";

} else {

paragraph.style.textDecoration = "none";

}

}

</script>  
  
  
These examples demonstrate how to manipulate images and text in response to user actions using JavaScript and CSS. You can adapt and style them further to fit your specific design and interaction requirements.