\

Advance JavaScript

MODULE: 1 (Introduction and Code Quality)

From: Shivani Gohil

1. Write a program to Show an alert

Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/Alert_box.html>

1. What will be the result for these expressions?

1. 5 > 4

2. "apple" > "pineapple"

3. "2" > "12"

4. undefined == null

5. undefined === null

6. null == "\n0\n"

7. 7. null === +"\n0\n"

Ans. The result of 5 > 4 is true. This is because 5 is greater than 4.

The result of "apple" > "pineapple" is false. This is because when comparing strings, JavaScript compares the Unicode code points of the characters in the strings from left to right. In this case, the first characters of each string are compared, and since "a" comes before "p" in the Unicode table, "apple" is considered less than "pineapple".

The result of "2" > "12" is true. This is because when comparing strings, JavaScript compares the characters in the strings from left to right. In this case, the first characters of each string are compared, and since "2" is greater than "1", the result is true.

The result of undefined == null is true. This is because undefined and null are considered equal when using the == operator (but not the === operator, as we'll see in the next example).

The result of undefined === null is false. This is because undefined and null are not the same type of value (even though they are both considered "falsy" values in JavaScript).

The result of null == "\n0\n" is false. This is because when comparing null to a string, the string is converted to a number (which in this case is 0). Since null is not equal to 0, the result is false.

The result of null === +"\n0\n" is false. This is because the + operator converts the string "\n0\n" to a number (which in this case is 0). Since null is not the same type of value as a number, the result is false.

<https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/practical.html>

1. Will alert be shown? if ("0") { alert( 'Hello'); }

Ans. When you run this program in a web browser, it will check the condition **"0"** and find that it is a truthy value, so it will execute the code inside the curly braces **{}**. This code calls the **alert()** function with the argument **"Hello"**, which will cause an alert box to be shown with the message "Hello".

<https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/practical1.html>

1. What is the code below going to output? alert( null || 2 || undefined );

Ans. When this program is run, the **alert()** function is called with the expression **null || 2 || undefined**. This expression uses the logical OR operator **||** to return the first truthy value it encounters from left to right. In this case, **null** is falsy, but **2** is truthy, so the expression evaluates to **2**. The **alert()** function then shows an alert box with the value **2**.

<https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/practical2.html>

5. The following function returns true if the parameter age is greater than

18. Otherwise it asks for a confirmation and returns its result:

function

checkAge(age)

{

if (age> 18) { return true; }

else {

// ...return confirm (‘did parents allow you?');

}

}

Ans.

5. The following function returns true if the parameter age is greater than

18. Otherwise it asks for a confirmation and returns its result:

function

checkAge(age)

{

if (age> 18) { return true; }

else {

// ...return confirm (‘did parents allow you?');

}

}

Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/practical3.html>

6. Replace Function Expressions with arrow functions in the code below:

Function

ask(question, yes, no)

{ if (confirm(question))yes();

else

no();

}

ask("Do you agree?", function()

{ alert("You agreed."); },

function() {

alert("You canceled the execution."); }

}Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/Advance%20js%20practical/practical4.html>

MODULE: 2 (Data Types and Objects)

1. Write the code, one line for each action:

a) Create an empty object user.

b) Add the property name with the value John.

c) Add the property surname with the value Smith.

d) Change the value of the name to Pete.

e) Remove the property name from the object.

Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul_2(1).html>

1. Is array copied?

let fruits = ["Apples", "Pear", "Orange"]; // push a new value into the "copy" let

shoppingCart = fruits; shoppingCart.push("Banana"); // what's in fruits?

alert( fruits.length ); // ?

Ans. the shoppingCart variable is assigned to the fruits array using assignment by reference. This means that shoppingCart and fruits refer to the same underlying array in memory, and any changes made to one will be reflected in the other.

Therefore, after the Banana element is added to the shoppingCart array using the push() method, the fruits array will also have the Banana element added to it.

Link <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul2(2).html>

3. Map to names

let john = { name: "John", age: 25 }; let pete = { name: "Pete", age: 30 }; let mary =

{ name: "Mary", age: 28 }; let users = [ john, pete, mary ]; let names = /\* ... your

code \*/ alert( names ); // John, Pete, Mary

Ans. In this code, we define three objects named john, pete, and mary, each with a name and an age property. We then create an array named users and fill it with these objects.

To get an array of names from the users array, we call the map() method on it and pass in an arrow function that accesses the name property of each object. The resulting array of names is then assigned to the names variable.

Finally, we use the alert() function to display the resulting array of names in an alert dialog box.

Link <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul2(3).html>

4. Map to objects

let john = { name: "John", surname: "Smith", id: 1 }; let pete = { name: "Pete",

surname: "Hunt", id: 2 }; let mary = { name: "Mary", surname: "Key", id: 3 }; let

users = [ john, pete, mary ]; let usersMapped = /\* ... your code ... \*/

/\*

usersMapped = [

{ fullName: "John Smith", id: 1 },

{ fullName: "Pete Hunt", id: 2 },

{ fullName: "Mary Key", id: 3 }

]

\*/ alert( usersMapped[0].id ) // 1 alert( usersMapped[0].fullName ) // John Smith

Ans. To map the users array to an array of objects containing fullName and id properties, we can use the map() method with an arrow function that creates a new object with the desired properties for each object in the original array.

Link <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul2(4).html>

5. Sum the properties There is a salaries object with arbitrary number of salaries. Write

the function sumSalaries(salaries) that returns the sum of all salaries using

Object.values and the for..of loop.If salaries is empty, then the result must be 0.

let salaries = {

"John": 100,

"Pete": 300,

"Mary": 250

};

alert( sumSalaries(salaries) ); // 650

Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul2(5).html>

6. Destructuring assignment We have an object: Write the Destructuring assignment that

reads:

a) Name property into the variable name.

b) Year’s property into the variable age.

c) isAdmin property into the variable isAdmin (false, if no such property)

d) let user = { name: "John", years: 30};

ans. we first define the object user with name and years properties. We then use destructuring assignment to extract the name and years properties into separate variables name and age. We also extract the isAdmin property into the isAdmin variable with a default value of false if the isAdmin property doesn't exist in the user object.

Ans. [https://github.com/Shivani081/Advance-JS- practical/blob/main/modul2(6).html](https://github.com/Shivani081/Advance-JS-%20%20%20%20%20%20%20%20%20%20%20%20practical/blob/main/modul2(6).html)

7. Turn the object into JSON and back Turn the user into JSON and then read it back

into another variable.

user = { name: "John Smith", age: 35};

MODULE: 3 (Document, Event and Controls

Ans. To turn the user object into JSON and back, we can use the built-in JSON.stringify() and JSON.parse() methods in JavaScript.

Ans. <https://github.com/Shivani081/Advance-JS-practical/blob/main/modul2(7).html>

**Thank You**