Data Types in js

1) what will be the output?

```
let x = 5;
let y = x;
x=10;
console.log(x);  // what is the value of x?
console.log(y); // what is the value of y and why?
```

Outputs:

1. **console.log(x);**

   Output: 10

   Explanation: You changed the value of x to 10 after initially setting it to 5.

2. **console.log(y);**

   Output: 5

   Explanation: The variable y was assigned the value of x when x was 5. After that, changing x to 10 does not affect y, as y holds its own copy of the value 5.

2) what will be the output?

```
let obj1= {name: "Alice" };

let obj2=obj1;

obj1.name="Bob";

console.log(obj1.name); //what is the output?

console.log(obj2.name); //what is the output and why?
```

Outputs:

1. **console.log(obj1.name);**

   Output: "Bob"

   Explanation: You changed the name property of obj1 from "Alice" to "Bob".

2. **console.log(obj2.name);**

Output: "Bob"

Explanation: obj2 is a reference to obj1, so when you modify obj1.name, it also affects obj2.name. Both obj1 and obj2 point to the same object in memory.

3)let a= "hello";

let b=47;

let c=true;

let d={key: "value"};

let e= null;

let f=undefined;

console.log(typeof a); // What will this output?

console.log(typeof b); // What will this output?

console.log(typeof c); //What will this output?

console.log(typeof d); // What will this output?

console.log(typeof e); // What will this output and why

console.log(typeof f); //what will this output and why?

Outputs:

1. **typeof a:** "string"

   Explanation: a is a string.

2. **typeof b:** "number"

   Explanation: b is a number.

3. **typeof c:** "boolean"

   Explanation: c is a boolean value (true).

4. **typeof d:** "object"

   Explanation: d is an object (specifically, an object literal).

5. **typeof e:** "object"

Explanation: This is a known quirk in JavaScript. null is technically an object, which can be confusing.

6. **typeof f:** "undefined"

   Explanation: f is explicitly set to undefined.

4) let numbers= [10, 20, 30, 40, 50];

console.log(numbers[2]); // What will this output?

console.log(numbers[0]); // What will this output?

console.log(numbers [numbers.length-1]); // // What will this output and Why?

Outputs:

1. **console.log(numbers[2]);**

   Output: 30

   Explanation: Arrays are zero-indexed, so numbers[2] accesses the third element, which is 30.

2. **console.log(numbers[0]);**

   Output: 10

   Explanation: numbers[0] refers to the first element of the array, which is 10.

3. **console.log(numbers[numbers.length - 1]);**

   Output: 50

   Explanation: numbers.length gives the total number of elements in the array (5 in this case). Therefore, numbers.length - 1 gives the index of the last element, which is 50.

5) let fruits=["apple", "banana", "mango"];

fruits[1]="orange";

console.log(fruits); // What will this output?

Output:

- **console.log(fruits);**: ["apple", "orange","mango"]

Explanation:

- You initially create an array fruits containing three elements: "apple", "banana", and "mango".
- The line fruits[1] = "orange"; changes the second element (index 1) from "banana" to "orange".
- When you log the fruits array, it reflects this change, resulting in the new array ["apple", "orange", "mango"].

6) Let matrix=[[1,2,3], [4,5,6],[7,8,9]];

console.log(matrix[1][2]); // What will this output?

console.log(matrix[2][0]); // What will this output?

Outputs:

1. **console.log(matrix[1][2]);**

   Output: 6

   Explanation: matrix[1] accesses the second row of the matrix, which is [4, 5, 6]. The element at index 2 of that row is 6.

2. **console.log(matrix[2][0]);**

   Output: 7

   Explanation: matrix[2] accesses the third row of the matrix, which is [7, 8, 9]. The element at index 0 of that row is 7.

7) Let person={name: ”John”, age:25, city: “New York”};

console.log(person.name); // What will this output?

console.log(person.age); // What will this output?

Outputs:

1. **console.log(person.name);**

   Output: "John"

   Explanation: Accessing the name property of the person object returns "John".

2. **console.log(person.age);**

   Output: 25

Explanation: Accessing the age property of the person object returns 25.

8) Let car={make: ”Toyota”, model: “Corolla”, year: 2021};

console.log(car[“make”]); // What will this output?

console.log(car[“model”]); // What will this output?

Outputs:

1. **console.log(car["make"]);**

   Output: "Toyota"

   Explanation: Accessing the make property of the car object returns "Toyota".

2. **console.log(car["model"]);**

   Output: "Corolla"

   Explanation: Accessing the model property of the car object returns "Corolla".

9) Let book={title: “The Great Gatsby”, author: “F.Scott Fitzgerald”};

book.author=”Anonymous”;

console.log(book.author); // What will this output?

Output:

- **console.log(book.author);**

  Output: "Anonymous"

  Explanation: The author property of the book object is changed from "F. Scott Fitzgerald" to "Anonymous", so when you log book.author, it outputs "Anonymous".

10) Ley student={name: “Alice”, grade: “A”};

student.age=20;

console.log(student); // What will this output?

Output:

- **console.log(student);**

Output: { name: "Alice", grade: "A", age: 20 }

Explanation: The student object initially has name and grade properties. When you add student.age = 20;, you add a new property age to the object. The output reflects all three properties: name, grade, and age.