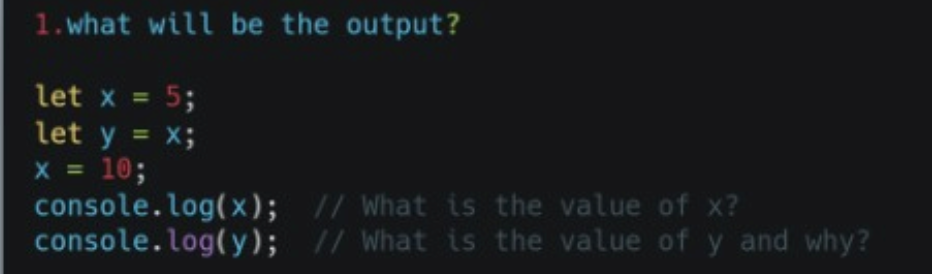
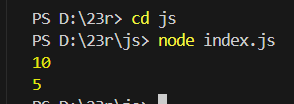
**TASK-2(Java Script)**

**1.**



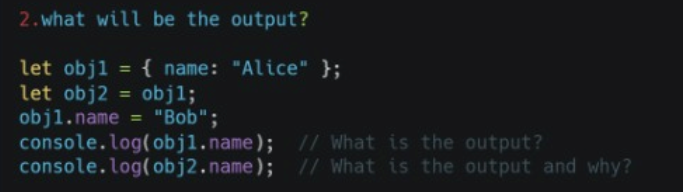
**Output:**



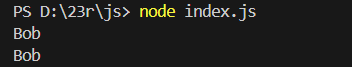
**Explanation:**

1. x is set to 5, and then y is given the value of x (which is 5).
2. x is changed to 10, but y still holds the old value (5) because y was assigned before x changed.
3. When console.log(x) runs, it shows 10, and console.log(y) shows 5.

**2.**



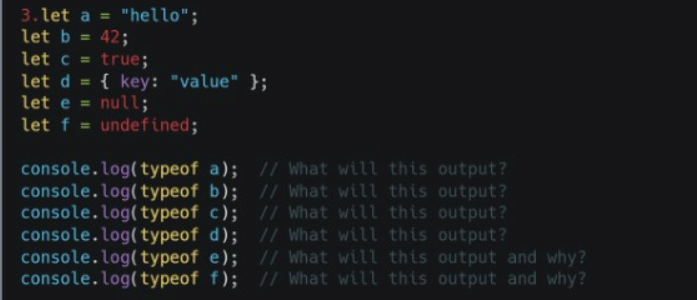
**Output:**

****

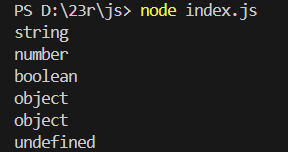
**Explanation:**

1. obj1 and obj2 both refer to the same object, so any changes to obj1 affect obj2 as well.
2. When obj1.name is changed to "Bob", obj2 also sees this change since they share the same object.
3. Both console.log(Obj1.name) and console.log(Obj2.name) will output "Bob".

**3.**



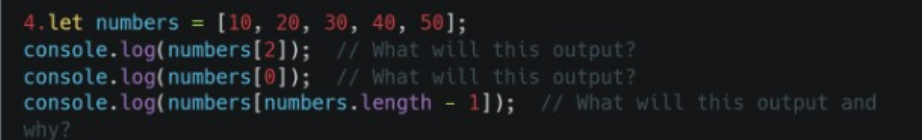
**Output:**

****

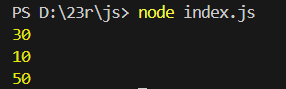
**Explanation:**

* "string" for a because it holds text.
* "number", "boolean", "object","object" for b, c, d, e respectively as per their types.
* null returns "object" (an exception in JavaScript), and f returns "undefined".

**4.**



**Output:**

****

**Explanation:**

* numbers[2] gives 30 because it accesses the element at index 2.
* numbers[0] gives 10 as it's the first element of the array.
* numbers[numbers.length - 1] gives 50, accessing the last element by calculating the length minus 1.

**5.**



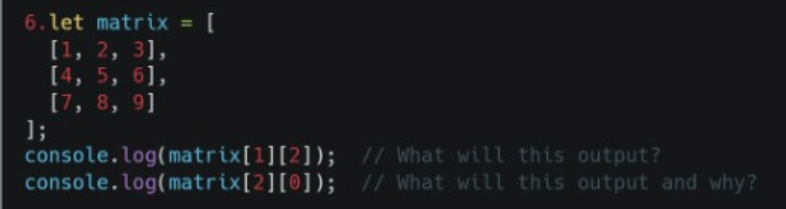
**Output:**

****

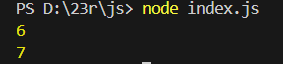
**Explanation:**

In the given JavaScript code, an array fruits is initialized with the values "apple", "banana", and "mango". The value at index 1 (originally "banana") is then replaced with "orange". The final output of console.log(fruits) will display the updated array: ["apple", "orange", "mango"].

**6.**



**Output:**

****

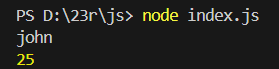
**Explanation:**

* matrix[1][2] refers to the element at row 1, column 2, which is 6. So, console.log(matrix[1][2]) outputs 6.
* matrix[2][0] refers to the element at row 2, column 0, which is 7. So, console.log(matrix[2][0]) outputs 7.

**7.**



**Output:**

****

**Explanation:**

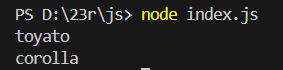
An object person is defined with properties name, age, and city.

* console.log(person.name) outputs "john" because it accesses the name property of the person object.
* console.log(person.age) outputs 25 because it retrieves the age property from the object.

**8.**



**Output:**

****

**Explanation:**

An object car is created with properties make, model, and year.

* console.log(car["make"]) outputs "toyato", accessing the make property of the car object.
* console.log(car["model"]) outputs "corolla", retrieving the model property of the object.

**9.**



**Output:**

****

**Explanation:**

In this JavaScript code, an object book is created with properties title and author.

* The author property is initially set to "F. Scott Fitzgerald", but is later updated to "Anonymous".
* console.log(book.author) will output "Anonymous", reflecting the updated value of the author property.

**10.**



**Output:**

****

**Explanation:**

In this JavaScript code, an object student is created with properties name and grade.

* A new property age is added to the student object with the value 20.
* console.log(student) outputs the entire object: {name: "alice", grade: "A", age: 20}.