1.what will be the output ?

```
Var x=5;
Var y=x;
x=10;
console.log(x);
console.log(y);
```

o/p 10,5
Reason: we are initialising x variable with 5 value then we are initialising y with x value after we are assigning x value with 10 so the y value won't change, only value is going to be reassigned, as interpreter follows line by line execution x value is reassigned after initialising y variable .

2.what will be the output ?

```
Var obj1={name:"Alice"};
Var obj2=obj1;
obj1.name="bob";
console.log(obj1.name);
console.log(obj2.name);
```

o/p: bob bob
Because it is a nonprimitive data type which is mutable so the address of obj1 obj2 is the same if you change value in any one of it,it will affect overall.

3.what will be output?

```
Var a="hello";
Var b=42;
Var c=true;
Var d={  key:"value" };
Var e=null;
Var f=undefined;
console.log(typeof a);
console.log(typeof b);
console.log(typeof c);
console.log(typeof d);
console.log(typeof e);
console.log(typeof f);
```

o/p: string
number
boolean
object
object
undefined

typeof keyword is used to determine what type of data type is assigned to a variable ,string is represented in double quotation.boolean value may be true or false,typeof null is object type of key value pair is represented as object,typeof undefined is undefined.

4.var numbers = [10,20,30,40,50];
console.log(numbers[2]);
console.log(numbers[0]);
console.log(numbers[numbers.length - 1]);

o/p: 30,10,50

Reason: In array starting index value is 0 ,so when we are accessing the value of the reference then the particular value is going to be printed as output.the length of array is determined by number of values presented in it if there are six elements then length is going to be 6.to access the last element we need to subtract length of array with 1.

5.var fruits = [“apple”,”banana”,”mango”];
fruits[1] = “orange”;
console.log(fruits);

o/p: [“apple”, “orange”, “mango”].
Reason: when initialising the fruits the index of 1 is assigned with banana. In the next step the reference of 1 is reassigned with orange so that output is going to change.

6.var matrix = [[1,2,3],[4,5,6],[7,8,9]];
console.log(matrix[1][2]);
console.log(matrix[2][0]);

o/p: 6,7
reason : in the above matrix the values are stored in index based so when we are trying to access the matrix[1][2] then at second index 3 value is going to print as output which is 6.in the second case the matrix[2][0] at third index first value is going to be printed.

7.var person = { name:”john”, age:25, city:”new york”};
console.log(person.name);
console.log(person.age);

o/p : john,25
Reason : the above data type is object because the data is stored in key value pair.each value is called property separated by ,. When we are trying to access the key then the associated value is going to print as output.

8.var car={ make:”toyota”,model:”corolla”,year:2021};
console.log(car[“make”]);
console.log(car[“model”]);

o/p : toyota,corolla

Reason : in object data type we can access the value by .(dot)form and in brackets mentioned in double quotes so that we are trying to access the key then the associated value is going to print as output.

9.var book = { title: “the great gatsby”, author:”F.Scott Fitzergerald”};
book.author = “Anonymous”;
console.log(book.author);

o/p : Anonymous
Reason : we are reassigning author value with Anonymous so that ,when we perform console operation then Anonymous is going to print.

10.var student = { name:”alice”, grade:”A” };
student.age = 20;
console.log(student);

o/p: { name: 'alice', grade: 'A', age: 20 }

Reason: we are assign 20 to the age variable then by .(dot) we are assign to add to which object we have student object we are assigning to