1. How do you copy by value a composite data type?

Ans: We can use some functions like slice, concat etc which makes to store same array with different address.

```
Ex: var arr=[1,2,3];
var arr2=arr.slice(0);
//var arr2=arr.concat();
console.log(arr===arr2); // false
arr2[2]=6;
console.log(arr,arr2);// [1,2,3],[1,2,6];
```

2. Why there is a difference in behavior for copying contents in primitive and non primitive type?

Ans: While in copying or assigning reference values for primitive types they get assigned values which are defined to variables but in non primitive types like arrays or objects they get assigned to address but not value of variable which is referred. So when we change any values in primitive type the value which is assigned get change but in non primitive type the value of all same addresses will get change.

3. Use typeof in all the datatypes and check the result

* **typeof(1)** -- number
* **typeof(1.1)** -- number
* **typeof("1.1")** -- string
* **typeof(true)** -- boolean
* **typeof(null)** -- object
* **typeof(undefined)** -- undefined
* **typeof([])** -- object
* **typeof({})** -- object

4. Write a blog about objects and its internal representation in Javascript ?

Ans:

Objects are the most important composite data types in JavaScript. They are quite different from primitive data types where they store single values but in composite data types they store different properties along with values.

In sense of real life examples objects can be compared with clothes for instance take a shirt as an object with properties like color, size, design, type of material etc. In the same way javascript objects have properties which define their characteristics.

Objects are more complex and each object may contain any combination of these primitive data-types as well as reference data-types.An object, is a reference data type. Variables that are assigned a reference value are given a reference or a pointer to that value. That reference or pointer points to the location in memory where the object is stored. The variables don't actually store the value in composite data types.

Objects in JavaScript may be defined as an unordered collection of related data, of primitive or reference types, in the form of "key: value" pairs. These keys can be variables or functions and are called properties and methods, respectively, in the context of an object.
An object can be created with figure brackets {…} with an optional list of properties. A property is a "key: value" pair, where a key is a string (also called a "property name"), and value can be anything.

Example:

```
let shirt = {
brand : “U.S.POLO”,
size : “Medium”,
color : “blue”,
}
```

In the above example “brand”, “size”, “color” are all “keys” and “U.S.POLO”, “Medium” and blue are values of these keys respectively.

We can access the properties of objects as with dot notation-- shirt.brand/shirt.size/shirt.color or with square brackets –shirt[‘size’]/ shirt[‘brand’]/ shirt[‘color’].

Each of these keys is referred to as properties of the object. An object in JavaScript may also have a function as a value for its property, in which case it will be known as a method of that object.

Let us see such an example :

```
// javascript code demonstrating a simple object
var shirt = {brand : 'U.S.POLO',

    size : 'Medium',

    color : 'blue',

    displayInfo : function(){

        console.log(“This is”+ shirt.brand + “shirt with size”+ shirt.size+ “and color” +shirt.color)

    }

 }

 console.log(shirt.displayInfo())
```

Output:

In the above example, “displayinfo” is a method of the shirt object that is being used to work with the object’s data, stored in its properties. It results as “This is U.S.POLO shirt with size Medium and color blue”.

We can add properties to existing object with keys and values as shown in below example:

```
shirt[‘design’]=’round neck’;  or shirt.design=’round neck’
```

It adds design property and its value to the object shirt and displays as follows:

```
console.log(shirt);

{ brand: 'U.S.POLO',
  size: 'Medium',
  color: 'blue',
  displayInfo: [Function: displayInfo],
  design: ‘round neck’ }
```

5. execute and see at least 15 cli commands. like mkdir, ls etc.

Ans: ls—shows all files in the folder

cd pathname—Change directory to folder with that pathname.
cd.. --Move one level up in folder in the file system.
cp – copy a file to another folder.
mv – Move a file to another foler.
mkdir – creates a new directory.
rmdir – removes a directory.
clear – clears the CLI window.
exit – closes the CLI window.
path – displays or sets a search path for files.
date – displays a date.
npm – is used to install softwares.
help – shows the manual for given command.
type – type of file.
vol – displays disk volume label and serial number

6. What is the difference between window, screen, and document in Javascript?

Ans: Window is the execution context and global object for that contexts javascript where as document is the property of window and contains DOM, initialised by parsing HTML and Screen is also a property of window and describes the physical properties like width and height of display.