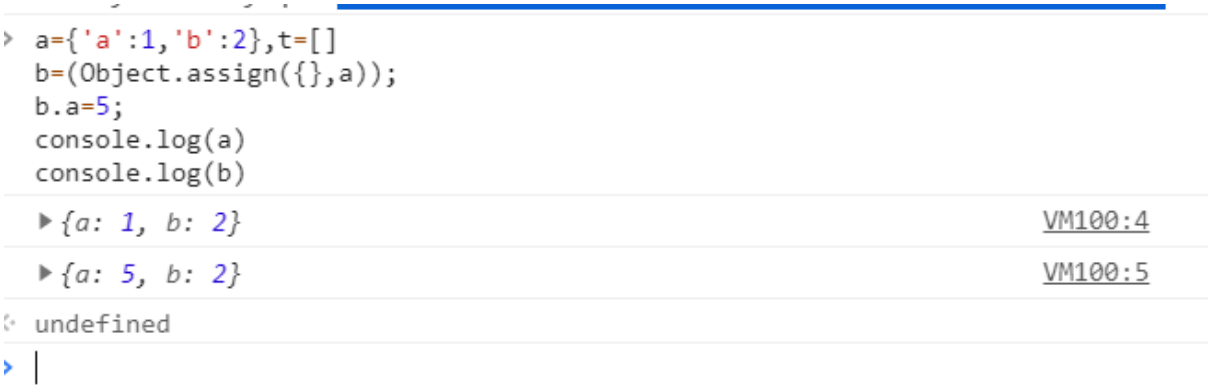
How do you copy by value a composite data type?

Copy by value can be done using loops in a composite data type. Every individual value must be taken and be copied into another array instead of copying as a whole.

The below snippet is an example for copy by value.

In case of Objects, assign function can be implemented.



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

Primitive data types go by copy-by-value mechanism. This is because, when you create a string variable, you tend to allocate memory in accordance with it. It is more like a single box, in which you can store any item. If you want to add a new item to it, you can only replace it by removing the old item and not add anything upon it. But this can be done any number of times.

Non-Primitive data types go by copy-by reference mechanism. This initiates a memory location to the entire set of data. For example, it is like room full of boxes stacked. You can add any number of items to the room if there exist boxes to accommodate them. If you must shift the items, you do it entirely along with the boxes rather than individually shifting the

items. Thus, the location of items does not get altered in any scope. Consequently, copying in such case, involves the copying of reference or address of the element too. Therefore, when you change the value, it affects entirely.

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

Write a blog about objects and its internal representation in JavaScript.

Object is a type of datatype in JavaScript. It holds the properties and is always stored in key value pairs.

An example of the object literal is shown below:
var student={‘id’: 143, ‘name’: ‘John’,}

Here, id and name are called as keys. They hold property values 143 and ‘John’. The key can be represented using string. Whilst the object could be of any datatypes.

An object can be created by multiple methods. An Object has to be first defined like other variables.

var obj = new Object();

Or

var obj={};

These create an empty object. If you want to create an object having values like above, you can directly assign them or you can add properties to an empty object as mentioned below.

obj.color = ’red’;

obj.fruit = ’banana’;

obj.number = 8;

obj.func = function(){console.log(‘Hello’)};

This will create an object of value,

```
{ color: 'red', fruit: 'banana', number: 8, func: [Function] }
```

The same can also be created by using square braces.

Obj[‘color’] = ’red’;

Obj[‘fruit’] = ’banana’;

Obj[‘number’] = 8;

Obj[‘func’] = function(){console.log(‘Hello’)};

To access these objects, we can use either of dot or square brackets.

console.log(obj['number'])

gives

8

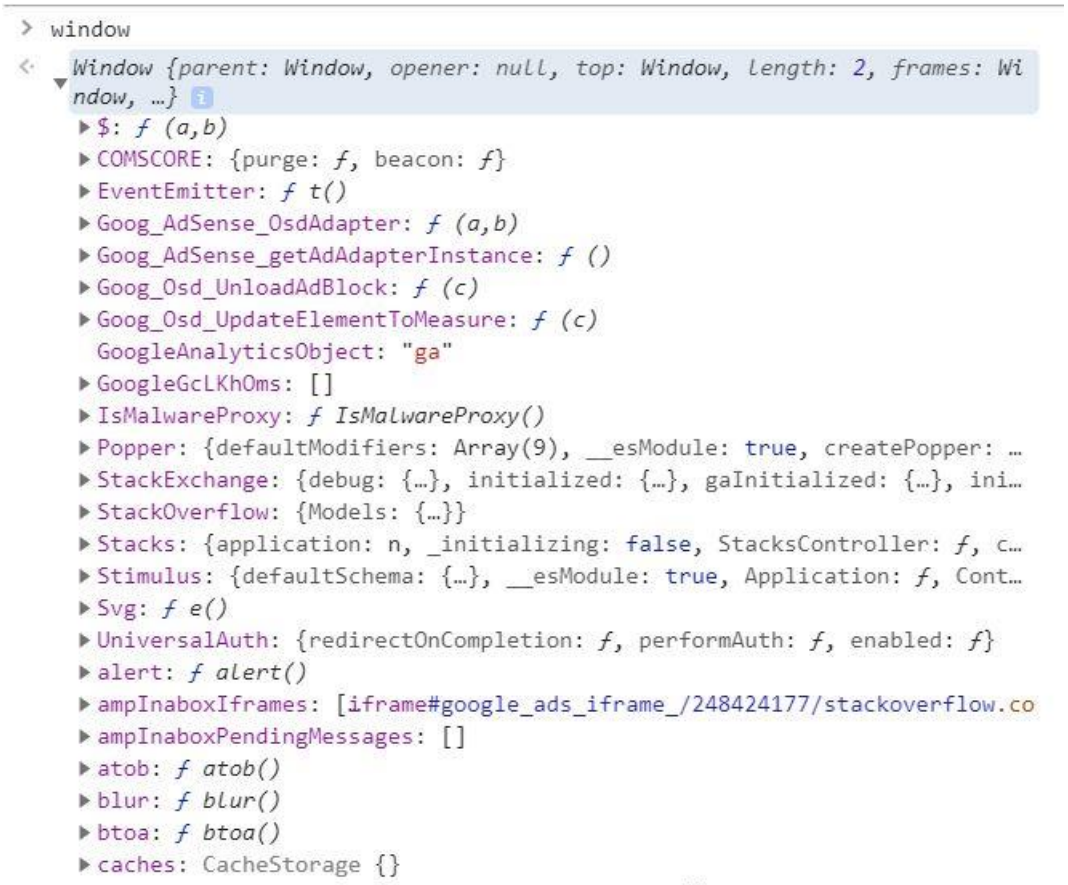
Though JavaScript does not have classes it supports objects and instances of objects. It allows all functionalities as in other class-based languages.

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

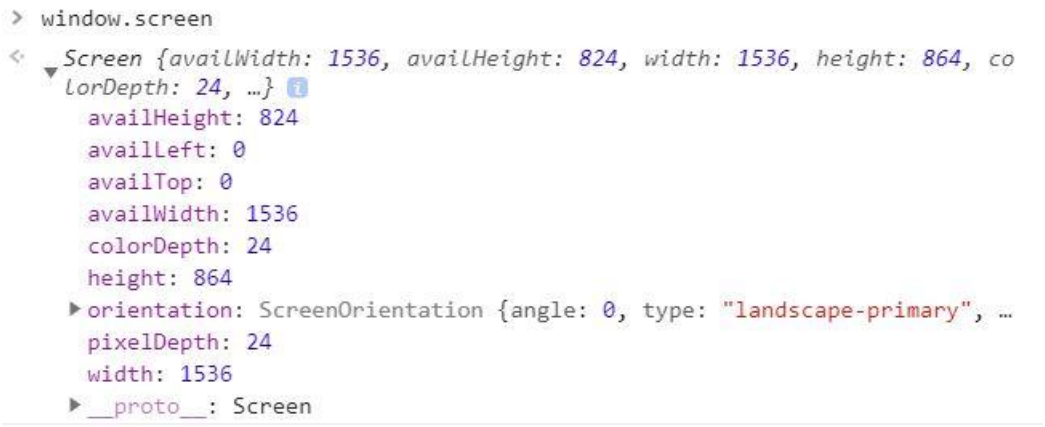
Present in new.txt

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

Window is the like global object for a browser. It is the root of the JavaScript root model. Window is the first thing that gets loaded into the browser. It ahs various properties like name, length, parent, alerts etc.



Screen is a small information object about the physical screen dimensions.



Document comes in lower hierarchy and is the potentially visible object in the Document Object model

Extract and print the flag url of all the countries in console. use the html template.

https://restcountries.eu/rest/v2/all

Code:

```
for(var i in a){
   console.log(a[i].flag)
}
```

```
> for(var i in a){
      console.log(a[i].flag)
  }
```

https://restcountries.eu/data/afg.svg	VM88:2
https://restcountries.eu/data/ala.svg	VM88:2
https://restcountries.eu/data/alb.svg	VM88:2
https://restcountries.eu/data/dza.svg	VM88:2
https://restcountries.eu/data/asm.svg	VM88:2
https://restcountries.eu/data/and.svg	VM88:2
https://restcountries.eu/data/ago.svg	VM88:2
https://restcountries.eu/data/aia.svg	VM88:2
https://restcountries.eu/data/ata.svg	VM88:2
https://restcountries.eu/data/atg.svg	VM88:2
https://restcountries.eu/data/arg.svg	VM88:2
https://restcountries.eu/data/arm.svg	VM88:2
https://restcountries.eu/data/abw.svg	VM88:2
https://restcountries.eu/data/aus.svg	VM88:2
https://restcountries.eu/data/aut.svg	VM88:2
https://restcountries.eu/data/aze.svg	VM88:2
https://restcountries.eu/data/bhs.svg	VM88:2
https://restcountries.eu/data/bhr.svg	VM88:2
https://restcountries.eu/data/bgd.svg	VM88:2
https://restcountries.eu/data/brb.svg	VM88:2
https://restcountries.eu/data/blr.svg	VM88:2
https://restcountries.eu/data/bel.svg	VM88:2

GUVI : Zen Code-Sprint : JavaScript Practice problems in JSON(Objects) and List

PROBLEM 0 : PART A:

```
var a=0, cat = {
 "name": "Fluffy","activities": ["play", "eat cat food"],
 "catFriends": [{"name": "bar","activities": ["be grumpy", "eat bread omblet"],"weight": 8,"furcolor":
"white"}, {"name": "foo","activities": ["sleep", "pre-sleep naps"],"weight": 3}]};
console.log(cat);
cat.height=2.3;                                                    //1
cat.weight=6;                                                      //1
cat.name="Fluffyy";                                                //2
for(i=0;i<cat.catFriends.length;i++){
console.log(cat.catFriends[i].activities.join());                  //3
}
for(i=0;i<cat.catFriends.length;i++){
   console.log(cat.catFriends[i].name)                             //4
}
for(i=0;i<cat.catFriends.length;i++){
```

```
    a+= +(cat.catFriends[i].weight)
}
console.log(a)                                                          //5
console.log(cat.activities.join())
for(i=0;i<cat.catFriends.length;i++){
console.log(cat.catFriends[i].activities.join());                       //5
}
cat.catFriends[0].activities.push("play")                               //6
cat.catFriends[0].activities.push("cuddle")                              //6
cat.catFriends[1].activities.push("bite")                               //6
cat.catFriends[1].activities.push("run")                                //6
cat.catFriends[0].furcolor="black";                                     //7
```

PROBLEM 0 : PART B:

```
var myCar = {
 "make": "Bugatti",
 "model": "Bugatti La Voiture Noire",
 "year": 2019,
 "accidents": [
 {
 "date": "3/15/2019",
 "damage_points": 5000,
 "atFaultForAccident": true
 },
 {
 "date": "7/4/2022",
 "damage_points": 2200,
 "atFaultForAccident": true
 },
 {
 "date": "6/22/2021",
 "damage_points": 7900,
 "atFaultForAccident": true
 }
 ]
}
console.log(myCar)
//Task 1

for(i=0;i<myCar.accidents.length;i++){
    myCar.accidents[i].atFaultForAccident=false;
}
//Task 2

for(i=0;i<myCar.accidents.length;i++){
    console.log(myCar.accidents[i].date)
}
```

PROBLEM 1:

```
var obj = {"name" : "RajiniKanth", "age" : 33, "hasPets" : false};
console.log(printAllValues(obj))
function printAllValues(obj) {
 var a=Object.values(obj)
 return a;
}
```

PROBLEM 2:

```
var obj = {"name" : "RajiniKanth", "age" : 33, "hasPets" : false};
console.log(printAllValues(obj))
function printAllValues(obj) {
 var a=Object.keys(obj)
 return a;
}
```

PROBLEM 3:

```
var a=[],obj = {"name": "ISRO", "age": 35, "role": "Scientist"};
console.log(convertListToObject(obj))
function convertListToObject(obj){
 var a=Object.entries(obj)
 return a;
}
```

PROBLEM 4:

```
var a={}, arr = ["GUVI", "I", "am", "a geek"];
console.log(transformFirstAndLast(arr))
function transformFirstAndLast(arr) {
 var l=arr[arr.length -1];
var f= arr[0];
        a[f]=l;
 return a;
}
```

PROBLEM 5:

```
var a = {},arr = [["make", "Ford"], ["model", "Mustang"], ["year", 1964]];
console.log(fromListToObject(arr))
function fromListToObject(arr) {
   for(i=0;i<arr.length;i++){
     var F=arr[i];
     var f=F[0]
     var l=F[1];
     a[f]=l;
```

```
    }
 return a;
}
```

PROBLEM 6:

```
var t=[],a = {},arr= [[["firstName", "Vasanth"], ["lastName", "Raja"], ["age", 24], ["role",
"JSWizard"]], [["firstName", "Sri"], ["lastName", "Devi"], ["age", 28], ["role", "Coder"]]];
console.log(transformEmployeeData(arr));
function transformEmployeeData(arr) {
    for(i=0;i<arr.length;i++){
        b=arr[i];
        //console.log(b)
        for(j=0;j<b.length;j++){
      var F=b[j];
      var f=F[0]
      var l=F[1];
      a[f]=l; if(j==b.length-1){t.push(Object.assign({},a));}
    }
    }
 return t;
}
```

PROBLEM 7:

```
var expected = {"foo": 5, "bar": 6};
var actual = {"foo": 5, "bar": 6}
assertObjectsEqual(actual, expected, 'detects that two objects are equal')
function assertObjectsEqual(actual, expected, testName){
  var act = JSON.stringify(actual);
  var ex = JSON.stringify(expected);

   if (act === ex) {
      console.log('Passed');
    }  else {
      console.log('FAILED [' + testName + '] Expected ' + ex + ', but got ' + act);
    }
}
```

PROBLEM 8:

```
var a=[],r={},val={},status=false,count=0,securityQuestions = [
 {
 "question": "What was your first pet’s name?",
 "expectedAnswer": "FlufferNutter"
 },
 {
 "question": "What was the model year of your first car?",
 "expectedAnswer": "1985"
 },
```

```
 {
 "question": 'What city were you born in?',
 "expectedAnswer": 'NYC'
 }
]
for(i=0;i<securityQuestions.length;i++){
var quest=Object.values(securityQuestions[i])
   f=quest[0];l=quest[1];
   val[f]=l;
   a.push(JSON.stringify(val));
   Object.keys(val).forEach(k => delete val[k]);
}

console.log(chksecurityQuestions("What was your first pet's name?","FlufferNutter"));
function chksecurityQuestions(q,b) {
r[q]=b;
res=JSON.stringify(r);

 for(i=0;i<a.length;i++){

    if(res===a[i]){
       count++;
    }
 }
if(count>0){status=true;}
return status;
}
```

PROBLEM 9:

```
var a=[],students = [
 {
 "name": 'Siddharth Abhimanyu', 'age': 21}, { 'name': 'Malar', 'age': 25},
 {'name': 'Maari','age': 18},{'name': 'Bhallala Deva','age': 17},
 {'name': 'Baahubali','age': 16},{'name': 'AAK chandran','age': 23},   {'name':'Gabbar
Singh','age': 33},{'name': 'Mogambo','age': 53},
 {'name': 'Munnabhai','age': 40},{'name': 'Sher Khan','age': 20},
 {'name': 'Chulbul Pandey','age': 19},{'name': 'Anthony','age': 28},
 {'name': 'Devdas','age': 56}
 ];

 console.log(returnMinors(students))
function returnMinors(arr)
{
   for(i=0;i<students.length;i++){
      if(students[i].age<18){
         a.push(students[i].name)
      }
   }return a;
}
```