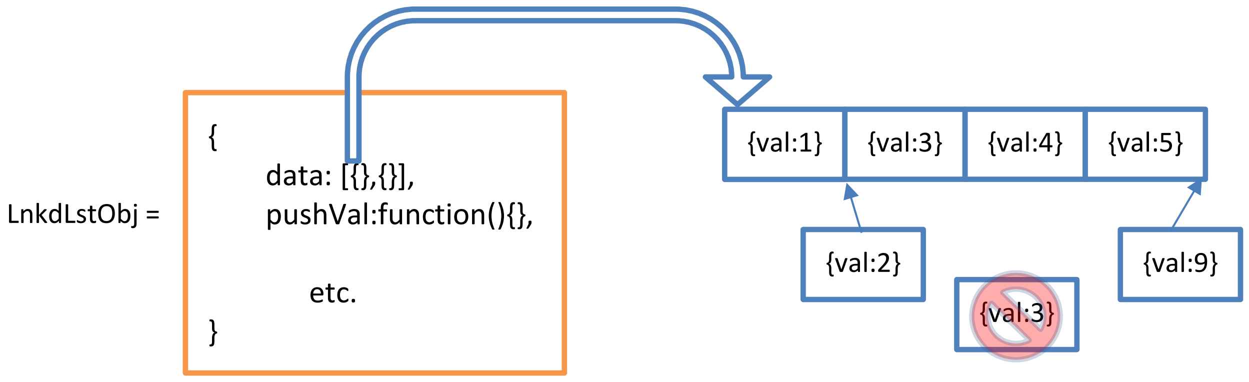
Advanced JavaScript
Lab 1

A. Object Object



A.1. Make your own custom Object that simulates the linked list that accepts objects with a single numeric property value in ascending order. Let your object has the following functionalities

- **Enqueue a value as long as the value is in the sequence otherwise through an exception (push an item at the end of the list with the passed value).**
- **Insert an item in a specific place as long as the value is missing from the sequence otherwise through an exception.**
- **Pop a value (remove an item from the end of the list).**
- **Remove an item from a specific place with the required value, if the value is not added return a message with "data not found".**
- **Dequeue a value (remove an item from the beginning of the list).**
- **Display the content of the list.**
- **Ensure that there is no duplication in your entered values.**
- ~~**All of the properties should be defined using data descriptor, prevent them from being deleted, iterated or being modified.**~~
- **You can add any property you need.**

Note: Use Array Object methods and there is no need to use sort() function.

B. Function Objects

B.1. Write two different functions with two different ways of implementations that takes any number of parameters and returns them as a reversed collection using array's reverse function.
Note: using of any loop is not allowed

B.2. Create your own custom object that has getSetGen as a function value, this function should generate setters and getters for the properties of the caller object
This object may have a description property of string value if needed
Let any other created object can use this function property to generate getters and setters for its own properties
Avoid generating getters or setters for any property of function value

Hint:
if getSetGen() is applied on any other object it should generate getters and setters for all of the applied object properties

i.e. if you have the following object

```
obj = {id: ”SD-10”,
        location: ”SV”,
        addr: ”123 st.”,
        getSetGen: function(){/*should be implemented*/}
        }
```

using of getSetGen() will generate the following getId(), setId(), getLocation(), setLocation(), getAddr(), setAddr().

If you created the following object
var user = {name: “Ali”, age: 10}
When applying getSetGen() on the user object (you can use call or bind or apply), it will result in creating the following: getName(), getAge(), setName(), setAge().

Note: Make your own interface for the above tasks.