**2) Explain the following phenomena in JavaScript:**

**You can access a ‘var’ variable before even declaring it and you don’t get any reference error, why is this the case?**

* Because of phenomena called “hoisting” in JavaScript.

**If you use ‘let’, ‘const’ instead of ‘var’, you can’t access those variables before declaration and get a reference error, why?**

* Let and Const cannot be accessed without first adding value because they are not in global object like “var”, whereas they are in separate memory space.

**If you use a first-class function instead of a regular function, you cannot access the first-class function before the declaration line, why?**

* First class functions are passed like values and passed as an argument to another functions, it is undefined when declared first, only when variable is assigned it points to the function.

**3) In JS, everything except primitive data types (bool, variables) is inherited from the Object, is it True? If so, show a code demonstration where a prototype ultimately points to the Object.**

* Yes, it is true.
* Code demonstration:

function Object(name, age) {

this.name = name;

this.age = age;

}

Object.prototype = new Obj();

Let Object1 = new Object(“Alex”, 57);

Console.log(Object1);

**4) In a MVC pattern, explain the different roles of Model, View and Controller. For the Booklist app you from the class contents, explain how the different components fit in different sections of the ‘M’, ‘V’ and ‘C’.**

* In MVC, “Controller” takes a request from a user, passes the data logic for the “Model” to perform and gets the data back to send it to the user. It also assigns presentation request from user to the “View” and receives back the presentation. Only, Controller interacts with both Model and View. Model and View do not interact with one another.
* Model handles the data logic and interacts with the database.
* View handles the data presentation.
* In a book list app, the server sends the request to the Controller that handles booklist. That Controller would then ask the Model that handles book list to return the list of all books. The Model would query the database for the list of all books and return that list back to the controller. If the response back from the Model is successful, then the Controller would ask the View associated with books to return a presentation of the list of books. This View would take the list from Controller and render the list into an HTML that could be used by the bowser. The controller would take that presentation and return it back to the user thus ending the request. If the Model returns an error the error is send to the user by Controller.

**5) What is prototype chaining? Explain the concept with a code example.**

* The ability to use prototype of the class after the instantiation of the object is called object prototype chaining.

function Object(name, age) {

this.name = name;

this.age = age;

}

Object.prototype.nameAge = function () {

console.log(this.name, "is ", this.age, " years old.");

};

let Object1 = new Object("Alex", 32);

let Object2 = new Object("Nina", 27);

Object1.nameAge();

Object2.nameAge();