**Adv. javascript theory**

1. **Write a JavaScript program to convert a given string into an array of words.**

**string\_to\_array = function (str) {**

**return str.trim().split(" ");**

**};**

**document.write(string\_to\_array("Dhruv"));**

**2. Write a JavaScript program to test a value, x, against a predicate function. If true, return fn(x). Else, return x.**

**const when = (pred, whenTrue) => x => (pred(x) ? whenTrue(x) : x);**

**const doubleEvenNumbers = when(x => x % 2 === 0, x => x \* 2);**

**console.log(doubleEvenNumbers(2));**

**console.log(doubleEvenNumbers(1));**

**3. Write a JavaScript program to reduce a given Array-like into a value hash (keyed data store).**

**const toHash = (object, key) =>**

**Array.prototype.reduce.call(**

**object,**

**(acc, data, index) => ((acc[!key ? index : data[key]] = data), acc),**

**{}**

**);**

**toHash([4, 3, 2, 1]); // { 0: 4, 1: 3, 2: 2, 1: 1 }**

**toHash([{ a: 'label' }], 'a'); // { label: { a: 'label' } }**

**// A more in depth example:**

**let users = [{ id: 1, first: 'Jon' }, { id: 2, first: 'Joe' }, { id: 3, first: 'Moe' }];**

**let managers = [{ manager: 1, employees: [2, 3] }];**

**// We use function here because we want a bindable reference, but a closure referencing the hash would work, too.**

**managers.forEach(**

**manager =>**

**(manager.employees = manager.employees.map(function(id) {**

**return this[id];**

**}, toHash(users, 'id')))**

**);**

**managers; // [ { manager:1, employees: [ { id: 2, first: "Joe" }, { id: 3, first: "Moe" } ] } ]**

**console.log(managers);**

**4.Write a JavaScript program to convert a float-point arithmetic to the Decimal mark form and It will make a comma separated string from a number.**

**const toDecimalMark = num => num.toLocaleString('en-US');**

**console.log(toDecimalMark(12305030388.9087));**

**console.log(toDecimalMark(123.2264))**

**console.log(toDecimalMark(-100.10))**

**5.Write a JavaScript program to create an object with the same keys as the provided object and values generated by running the provided function for each value.**