***Assignment 26***

**1. Type Conversion**

function convertToNumber(str) {

let num = Number(str);

if (isNaN(num)) {

return "Invalid number";

} else {

return num;

}

}

console.log(convertToNumber("123")); // 123

console.log(convertToNumber("abc")); // Invalid number

**2. Building Robust Functions in JavaScript**

function getPerson(obj) {

try {

if (!obj.hasOwnProperty("name") || !obj.hasOwnProperty("age")) {

throw new Error("Invalid parameter type");

}

return `Name: ${[obj.name](https://obj.name/)}, Age: ${obj.age}`;

} catch (error) {

return error.message;

}

}

console.log(getPerson({ name: "Mithun", age: 20 })); // Name: Mithun, Age: 20

console.log(getPerson({ name: "Mithun" })); // Invalid parameter type

console.log(getPerson(["name", "Mithun"])); // Invalid parameter type

**3. Car Description Class**

class Car {

constructor(company, model, year) {

[this.company](https://this.company/) = company;

this.model = model;

this.year = year;

}

getDescription() {

return `This is a ${this.year} ${[this.company](https://this.company/)} ${this.model}.`;

}

}

const myCar = new Car("Skoda", "Rapid", 2022);

console.log(myCar.getDescription()); // Output: This is a 2022 Skoda Rapid.

**4. Employee Class Challenge**

class Employee {

constructor(name, position, salary) {

[this.name](https://this.name/) = name;

this.position = position;

this.salary = salary;

}

getSalary() {

return this.salary;

}

}

const employee = new Employee("John Doe", "Manager", 80000);

console.log(employee.getSalary()); // Output: 80000

**5. Implementing a Person Class with Default Values**

class Person {

constructor(name = "Unknown", age = 0) {

[this.name](https://this.name/) = name;

this.age = age;

}

getDetails() {

return `Name: ${[this.name](https://this.name/)}, Age: ${this.age}`;

}

}

const person1 = new Person("Mithun", 20);

console.log(person1.getDetails()); // Output: Name: Mithun, Age: 20

const person2 = new Person();

console.log(person2.getDetails()); // Output: Name: Unknown, Age: 0

**6. Using Static Method to Add Two Numbers with Calculator Class**

class Calculator {

static add(num1, num2) {

return num1 + num2;

}

}

const result = Calculator.add(7, 8);

console.log(result); // Output: 15

**7. Password Checker**

class User {

constructor(username, password) {

this.username = username;

this.\_password = password;

}

getPassword() {

return "\*".repeat(this.\_password.length);

}

setPassword(newPassword) {

const regex = /^(?=.\*\d)(?=.\*[A-Z]).{8,}$/;

if (regex.test(newPassword)) {

this.\_password = newPassword;

} else {

console.log(

"Error: Password must be at least 8 characters long and contain at least one number and one uppercase letter."

);

}

}

}

const user = new User("Mithun", "Password123");

console.log(user.getPassword()); // Output: \*\*\*\*\*\*\*\*\*\*\*

user.setPassword("myPassword"); // Error: Password must be at least 8 characters long and contain at least one number and one uppercase letter.

user.setPassword("MyPassword"); // Error: Password must be at least 8 characters long and contain at least one number and one uppercase letter.

user.setPassword("Mypassword123");

console.log(user.getPassword()); // Output: \*\*\*\*\*\*\*\*\*\*\*\*\*

**8. Adding a Method to a Prototype**

function Student(name) {

[this.name](https://this.name/) = name;

}

Student.prototype.printDetails = function() {

console.log(`Hello, my name is ${[this.name](https://this.name/)}`);

}

const student = new Student("Mithun");

student.printDetails(); // Output: Hello, my name is Mithun

**9. Check the presence using closures**

function numberChecker(arr) {

return function(num) {

return arr.includes(num);

}

}

const arr = [1, 2, 3, 4, 5];

const checkNum = numberChecker(arr);

console.log(checkNum(3)); // Output: true

console.log(checkNum(6)); // Output: false

**10. Filter by Category**

function filterByCategory(products) {

return function(category) {

return products.filter(product => product.category === category);

}

}

var products = [

{ name: "Shirt", category: "Clothing" },

{ name: "Pants", category: "Clothing" },

{ name: "Hat", category: "Accessories" },

{ name: "Sunglasses", category: "Accessories" }

];

var clothingProducts = filterByCategory(products)("Clothing");

console.log(clothingProducts); // Output: [{ name: "Shirt", category: "Clothing" }, { name: "Pants", category: "Clothing" }]