```
// Inheritance Example
class Animal {
    constructor(name, age) {
        this.name = name;
        this.age = age;
    }

    speak() {
        return "Animal sound";
    }
}

class Dog extends Animal {
    speak() {
        return "Woof";
    }
}
class Cat extends Animal {
    speak() {
        return "Meow";
    }
}

const dog = new Dog("Buddy", 5);
const cat = new Cat("Riskers", 3);

console.log(`${dog.name} says: ${dog.speak()}`);
console.log(`${cat.name} says: ${cat.speak()}`);

//encapsulation example
class BankAccount {
    #balance
```

```
    constructor(accountNumber, initialBalance = 0) {
        this.accountNumber = accountNumber;
        this.#balance = initialBalance;
    }

    deposit(amount) {
        if (amount > 0) {
            this.#balance += amount;
        } else {
            console.log("Deposit amount must be
positive.");
        }
    }
    withdraw(amount) {
        if (amount > 0) {
            if (this.#balance >= amount) {
                this.#balance -= amount;
            } else {
                console.log("Insufficient funds.");
            }
        } else {
            console.log("Withdrawal amount must be
positive.");
        }
    }

    getBalance() {
        return this.#balance;
    }
}

const account = new BankAccount("123456", 1000);
```

```
console.log("Initial Balance:", account.getBalance());

account.deposit(500);
console.log("After Deposit:", account.getBalance());

account.withdraw(200);
console.log("After Withdrawal:", account.getBalance());

account.withdraw(1500);
console.log("You Trying to Overdraw:",
account.getBalance());

//Polymoriphism
class Shape{
    calculateArea(){
        return 0;
    }
    getName(){
        return "Shape";
    }
}

class Rectangle extends Shape {
        constructor(width, height) {
            super();
            this.width = width;
            this.height = height;
        }
        calculateArea(){
            return this.width * this.height;
        }
```

```
        getName(){
            return "Rectangle";
        }
}

class Circle extends Shape {
    static PI = 3.14; //class object variable

    constructor(radius) {
        super();
        this.radius = radius;
    }

    calculateArea() {
        return Circle.PI * this.radius * this.radius;
    }
    getName(){
        return "Circle";
    }
}
const shapes = [
    new Rectangle(5, 10),
    new Circle(3),
    new Rectangle(7, 4),
    new Circle(5)
];
for (i in shapes) {
    const shape =shapes[i]
    console.log(`Area of ${shape.getName()}:
${shape.calculateArea().toFixed(2)}`);
}
```