🧪 JavaScript OOP Exercises

🧪 Exercise 1 — Basic Class & Constructor

Task: Buat class Person dengan:

- property: `name`, `age`
- constructor menerima `name` dan `age`

Expected Input:

```
const p1 = new Person("Budi", 25);
console.log(p1);
```

Expected Output:

```
Person { name: "Budi", age: 25 }
```

🧪 Exercise 2 — Default Parameter

Task: Buat class Car dengan:

- property: `brand`
- property: `year` (default = 2020)

Expected Input:

```
const car1 = new Car("Toyota");
const car2 = new Car("Honda", 2023);
console.log(car1);
console.log(car2);
```

Expected Output:

```
Car { brand: "Toyota", year: 2020 }
Car { brand: "Honda", year: 2023 }
```

🧪 Exercise 3 — Public & Private Property

Task: Buat class Account:

- public `owner`
- private `balance`
- Tambahkan method `showBalance()` untuk menampilkan balance.

Expected Input:

```
const acc1 = new Account("Nadhif", 100000);
console.log(acc1.owner);
console.log(acc1.showBalance());
```

Expected Output:

```
Nadhif
Balance: 100000
```

🧪 Exercise 4 — Getter

Task: Buat class Student dengan:

- private `_score`
- getter `score`

Expected Input:

```
const s1 = new Student("Andi", 90);
console.log(s1.score);
```

Expected Output:

```
Nilai Andi adalah 90
```

🧪 Exercise 5 — Setter Validation

Task: Tambahkan setter `score`:

- jika < 0 → throw Error "Score cannot be negative"

Expected Input:

```
const s2 = new Student("Siti", 80);
s2.score = 95;
console.log(s2.score);
```

Expected Output:

```
Nilai Siti adalah 95
```

🧪 Exercise 6 — Encapsulation (Bank Account)

Task: Buat class BankAccount:

- private `balance`
- method `deposit(amount)`
- method `getBalance()`

Expected Input:

```
const acc = new BankAccount();
acc.deposit(50000);
acc.deposit(25000);
console.log(acc.getBalance());
```

Expected Output:

```
75000
```

🧪 Exercise 7 — Inheritance (Animal)

Task: Buat:

- class `Animal` → property `name`
- class `Cat` extends `Animal` → property `color`

Expected Input:

```
const cat1 = new Cat("Milo", "Orange");
console.log(cat1.name);
console.log(cat1.color);
```

Expected Output:

```
Milo
Orange
```

🧪 Exercise 8 — Method Inheritance

Task: Tambahkan method `speak()` di Animal:

- `speak() { return "Animal makes sound"; }`

Override di Cat:

- `speak() { return "Meow"; }`

Expected Input:

```
const animal = new Animal("Unknown");
const cat = new Cat("Kitty", "White");
console.log(animal.speak());
console.log(cat.speak());
```

Expected Output:

```
Animal makes sound
```

🧪 Exercise 9 — Product with Getter & Setter

Task: Buat class Product:

- private `_price`
- getter `price`
- setter `price` (tidak boleh < 0)

Expected Input:

```
const p = new Product("Laptop", 10000000);
p.price = 12000000;
console.log(p.price);
```

Expected Output:

```
Harga produk adalah 12000000
```

🧪 Exercise 10 — Mini Real Case (User & Role)

Task: Buat class User:

- `name`
- `role` (default = "user")
- method `getInfo()`

Expected Input:

```
const u1 = new User("Budi");
const u2 = new User("Admin", "admin");
console.log(u1.getInfo());
console.log(u2.getInfo());
```

Expected Output:

```
User Budi memiliki role user
User Admin memiliki role admin
```