Mata Kuliah

Pemrograman Berorientasi Objek

Pertemuan 5. Tugas5

” Javascript Class, Method , Polymorphism”

Dosen Pengampu:

Willdan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh:

Habibirrohim

2300149

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

PENJELASAN

1. **Kelas Induk: Kapal**

`class Kapal {`
deklarasi kelas Kapal. Kelas ini akan menjadi dasar untuk semua jenis kapal lainnya.

`constructor(nama, jenis) {`
Constructor adalah metode khusus yang dijalankan saat objek baru dibuat.
Parameter ‘nama’ dan ‘jenis’ digunakan untuk menginisialisasi properti objek.

`this.nama = nama;`
‘this.nama’ adalah properti yang menyimpan nama kapal.
Nilai ‘nama’ yang diberikan saat pembuatan objek akan disimpan di sini.

`this.jenis = jenis;`
‘this.jenis’ adalah properti yang menyimpan jenis kapal.
Nilai ‘jenis’ yang diberikan saat pembuatan objek akan disimpan di sini.

`} infoKapal() {`
Metode ini memberikan informasi dasar tentang kapal.

`return \`${this.nama}`** adalah kapal jenis **`${this.jenis}.\`;`
Mengembalikan string yang berisi nama dan jenis kapal.
Menggunakan template literal untuk memudahkan penyisipan nilai variabel.

```
    operasi() {
    // Metode ini memberikan informasi umum tentang operasi kapal.
   return `Kapal ${this.nama} sedang beroperasi.`;
    // Mengembalikan string yang menunjukkan bahwa kapal sedang beroperasi.
  }

    muatanInfo() {
    // Ini adalah metode abstrak yang harus diimplementasikan oleh kelas turunan.

      throw new Error("Metode muatanInfo() harus
diimplementasikan");
    // Melempar error jika metode ini dipanggil tanpa diimplementasikan di kelas
turunan.
    // Ini memastikan bahwa setiap jenis kapal memiliki implementasi sendiri untuk
informasi muatan.
  }
}
```

2. **Kelas Turunan: KapalPenumpang**

```
// Mendefinisikan kelas KapalPenumpang yang mewarisi dari Kapal
class KapalPenumpang extends Kapal {
  // 'extends Kapal' menunjukkan bahwa KapalPenumpang adalah turunan dari kelas Kapal.
  // Ini berarti KapalPenumpang mewarisi semua properti dan metode dari Kapal.

    constructor(nama, kapasitasPenumpang) {
    // Constructor untuk KapalPenumpang, menerima nama dan kapasitas penumpang.

      super(nama, "Penumpang");
    // 'super()' memanggil constructor dari kelas induk (Kapal).
    // Mengirim 'nama' dan jenis "Penumpang" ke constructor Kapal.

      this.kapasitasPenumpang = kapasitasPenumpang;
    // Menyimpan kapasitas maksimum penumpang kapal.
      this.penumpangSaatIni = 0;
    // Menginisialisasi jumlah penumpang saat ini dengan 0.
  }

    naikkanPenumpang(jumlah) {
    // Metode untuk menambah penumpang ke kapal.

      if (this.penumpangSaatIni + jumlah <= this.kapasitasPenumpang) {
    // Memeriksa apakah penambahan penumpang tidak melebihi kapasitas.

        this.penumpangSaatIni += jumlah;
    // Jika tidak melebihi, tambahkan jumlah penumpang.

        return `${jumlah} penumpang naik. Total penumpang: ${this.penumpangSaatIni}`;
    // Mengembalikan informasi tentang penumpang yang naik dan total saat ini.
   }
      return "Kapasitas penuh, tidak bisa menambah penumpang.";
    // Jika kapasitas penuh, kembalikan pesan bahwa tidak bisa menambah penumpang.
  }

    operasi() {
    // Override metode operasi dari kelas induk.
```

```
      return `Kapal penumpang ${this.nama} sedang berlayar
dengan ${this.penumpangSaatIni} penumpang.`;
    // Memberikan informasi spesifik tentang operasi kapal penumpang.
  }

    muatanInfo() {
    // Implementasi metode muatanInfo untuk KapalPenumpang.
   return `Jumlah penumpang: ${this.penumpangSaatIni}/${this.kapasitasPenumpang}`;
    // Mengembalikan informasi tentang jumlah penumpang saat ini dan kapasitas.
  }
}
```

3. Kelas Turunan: KapalKargo

```
// Mendefinisikan kelas KapalKargo yang mewarisi dari Kapal
class KapalKargo extends Kapal {
  // KapalKargo adalah turunan dari Kapal, mewarisi properti dan metode dasarnya.

    constructor(nama, kapasitasMuatan) {
    // Constructor untuk KapalKargo, menerima nama dan kapasitas muatan.
      super(nama, "Kargo");
    // Memanggil constructor Kapal dengan nama dan jenis "Kargo".
      this.kapasitasMuatan = kapasitasMuatan;
    // Menyimpan kapasitas maksimum muatan dalam ton.
      this.muatanSaatIni = 0;
    // Menginisialisasi berat muatan saat ini dengan 0 ton.
  }
    muatBarang(berat) {
    // Metode untuk menambah muatan ke kapal kargo.

      if (this.muatanSaatIni + berat <=
this.kapasitasMuatan) {
    // Memeriksa apakah penambahan muatan tidak melebihi kapasitas.
```

```
            this.muatanSaatIni += berat;
```
// Jika tidak melebihi, tambahkan berat muatan.
```
            return `${berat} ton barang dimuat. Total muatan:
${this.muatanSaatIni} ton`;
```
// Mengembalikan informasi tentang muatan yang ditambahkan dan total saat ini.

}
```
        return "Kapasitas penuh, tidak bisa menambah
muatan.";
```
// Jika kapasitas penuh, kembalikan pesan bahwa tidak bisa menambah muatan.

}
```
    operasi() {
```
// Override metode operasi dari kelas induk.
```
   return `Kapal kargo ${this.nama} sedang mengangkut
${this.muatanSaatIni} ton barang.`;
```
// Memberikan informasi spesifik tentang operasi kapal kargo.

}
```
    muatanInfo() {
```
// Implementasi metode muatanInfo untuk KapalKargo.
```
        return `Muatan:
${this.muatanSaatIni}/${this.kapasitasMuatan} ton`;
```
// Mengembalikan informasi tentang muatan saat ini dan kapasitas dalam ton.

}

}

4. Kelas Turunan: KapalTanker

// Mendefinisikan kelas KapalTanker yang mewarisi dari Kapal
```
class KapalTanker extends Kapal {
```
// KapalTanker adalah turunan dari Kapal, fokus pada pengangkutan cairan.
```
    constructor(nama, kapasitasTangki) {
```
// Constructor untuk KapalTanker, menerima nama dan kapasitas tangki.
```
        super(nama, "Tanker");
```

```
    // Memanggil constructor Kapal dengan nama dan jenis "Tanker".
        this.kapasitasTangki = kapasitasTangki;
    // Menyimpan kapasitas maksimum tangki dalam liter.
        this.isiTangki = 0;
    // Menginisialisasi volume cairan saat ini dengan 0 liter.
  }

    isiMuatan(volume) {
    // Metode untuk menambah muatan cairan ke kapal tanker.
        if (this.isiTangki + volume <= this.kapasitasTangki)
{
    // Memeriksa apakah penambahan volume tidak melebihi kapasitas tangki.
            this.isiTangki += volume;
    // Jika tidak melebihi, tambahkan volume cairan.
            return `${volume} liter cairan ditambahkan. Total
isi tangki: ${this.isiTangki} liter`;
    // Mengembalikan informasi tentang volume yang ditambahkan dan total saat ini.
    }
        return "Tangki penuh, tidak bisa menambah cairan.";
   // Jika tangki penuh, kembalikan pesan bahwa tidak bisa menambah cairan.
  }

    operasi() {
    // Override metode operasi dari kelas induk.
        return `Kapal tanker ${this.nama} sedang mengangkut
${this.isiTangki} liter cairan.`;
    // Memberikan informasi spesifik tentang operasi kapal tanker.
  }

    muatanInfo() {
    // Implementasi metode muatanInfo untuk KapalTanker.
        return `Isi tangki:
${this.isiTangki}/${this.kapasitasTangki} liter`;
    // Mengembalikan informasi tentang isi tangki saat ini dan kapasitas dalam liter.
  }
}
```

5. **Kelas Turunan: KapalPenangkapIkan**

```
// Mendefinisikan kelas KapalPenangkapIkan yang mewarisi dari Kapal
class KapalPenangkapIkan extends Kapal {
  // KapalPenangkapIkan adalah turunan dari Kapal, khusus untuk menangkap ikan.

    constructor(nama, kapasitasJaring) {
    // Constructor untuk KapalPenangkapIkan, menerima nama dan kapasitas jaring.

      super(nama, "Penangkap Ikan");
    // Memanggil constructor Kapal dengan nama dan jenis "Penangkap Ikan".

      this.kapasitasJaring = kapasitasJaring;
    // Menyimpan kapasitas maksimum jaring dalam kg.

      this.tangkapan = 0;
    // Menginisialisasi berat tangkapan saat ini dengan 0 kg.
  }

    tangkapIkan(jumlah) {
    // Metode untuk menambah tangkapan ikan.

      if (this.tangkapan + jumlah <= this.kapasitasJaring)
{
    // Memeriksa apakah penambahan tangkapan tidak melebihi kapasitas jaring.

        this.tangkapan += jumlah;
    // Jika tidak melebihi, tambahkan jumlah tangkapan.

        return `${jumlah} kg ikan ditangkap. Total
tangkapan: ${this.tangkapan} kg`;
    // Mengembalikan informasi tentang ikan yang ditangkap dan total saat ini.
   }

      return "Jaring penuh, tidak bisa menambah
tangkapan.";
   // Jika jaring penuh, kembalikan pesan bahwa tidak bisa menambah tangkapan.
  }

    operasi() {
    // Override metode operasi dari kelas induk.
```

```
        return `Kapal penangkap ikan ${this.nama} sedang
melaut dengan ${this.tangkapan} kg ikan.`;
      // Memberikan informasi spesifik tentang operasi kapal penangkap ikan.
    }

     muatanInfo() {
      // Implementasi metode muatanInfo untuk KapalPenangkapIkan.

        return `Tangkapan:
${this.tangkapan}/${this.kapasitasJaring} kg`;
      // Mengembalikan informasi tentang tangkapan saat ini dan kapasitas jaring dalam
kg.
    }
}
```

6. **Fungsi Demonstrasi dan Penggunaan**

```
// Fungsi untuk mendemonstrasikan polimorfisme

function demonstrasiOperasiKapal(kapal) {
   // Fungsi ini menerima objek kapal sebagai parameter.
   // Fungsi ini mendemonstrasikan polimorfisme karena dapat bekerja dengan berbagai
jenis kapal.

      console.log(kapal.infoKapal());
   // Memanggil dan mencetak hasil dari metode infoKapal() objek kapal.

      console.log(kapal.operasi());
   // Memanggil dan mencetak hasil dari metode operasi() objek kapal.

      console.log(kapal.muatanInfo());
   // Memanggil dan mencetak hasil dari metode muatanInfo() objek kapal.
}
// Membuat instance dari setiap kelas

const kapalPenumpang = new KapalPenumpang("Nusantara",
500);
// Membuat objek KapalPenumpang bernama "Nusantara" dengan kapasitas 500
penumpang.

const kapalKargo = new KapalKargo("Gojali", 10000);
// Membuat objek KapalKargo bernama "Gojali" dengan kapasitas 10000 ton.
```

```
const kapalTanker = new KapalTanker("Pertamina", 50000);
```
// Membuat objek KapalTanker bernama "Pertamina" dengan kapasitas 50000 liter.

```
const kapalIkan = new KapalPenangkapIkan("Nelayan Jaya",
5000);
```
// Membuat objek KapalPenangkapIkan bernama "Nelayan Jaya" dengan kapasitas jaring 5000 kg.
// Mendemonstrasikan polimorfisme

```
console.log("Kapal Penumpang:");
```
// Mencetak judul untuk demonstrasi KapalPenumpang.

```
kapalPenumpang.naikkanPenumpang(300);
```
// Menaikkan 300 penumpang ke kapalPenumpang.

```
demonstrasiOperasiKapal(kapalPenumpang);
```
// Memanggil fungsi demonstrasi untuk kapalPenumpang.

```
console.log("\nKapal Kargo:");
```
// Mencetak judul untuk demonstrasi KapalKargo.
kapalK