Contoh Program Polymorfism

Contoh 1

// Kelas dasar

class Hewan {

// Metode yang akan di-override oleh kelas turunan

void suara() {

System.out.println("Hewan membuat suara");

}

}

// Kelas turunan

class Kucing extends Hewan {

// Override metode suara untuk kucing

@Override

void suara() {

System.out.println("Kucing mengeong");

}

}

class Anjing extends Hewan {

// Override metode suara untuk anjing

@Override

void suara() {

System.out.println("Anjing menggonggong");

}

}

public class Main {

public static void main(String[] args) {

// Membuat array dari objek Hewan

Hewan[] hewanPeliharaan = new Hewan[2];

// Mengisi array dengan objek Kucing dan Anjing

hewanPeliharaan[0] = new Kucing();

hewanPeliharaan[1] = new Anjing();

// Menggunakan loop untuk memanggil metode suara

for (Hewan hewan : hewanPeliharaan) {

hewan.suara(); // Polymorphism dalam aksi

}

}

}

Output :

Kucing mengeong

Anjing menggonggong

Contoh 2

// Kelas dasar

abstract class Karyawan {

String nama;

double gajiDasar;

Karyawan(String nama, double gajiDasar) {

this.nama = nama;

this.gajiDasar = gajiDasar;

}

// Metode abstrak untuk menghitung gaji, akan diimplementasikan oleh subclass

abstract double hitungGaji();

}

// Kelas Manajer

class Manajer extends Karyawan {

double bonus;

Manajer(String nama, double gajiDasar, double bonus) {

super(nama, gajiDasar);

this.bonus = bonus;

}

@Override

double hitungGaji() {

return gajiDasar + bonus;

}

}

// Kelas Programmer

class Programmer extends Karyawan {

int jamLembur;

double tarifLembur;

Programmer(String nama, double gajiDasar, int jamLembur, double tarifLembur) {

super(nama, gajiDasar);

this.jamLembur = jamLembur;

this.tarifLembur = tarifLembur;

}

@Override

double hitungGaji() {

return gajiDasar + (jamLembur \* tarifLembur);

}

}

// Kelas Desainer

class Desainer extends Karyawan {

double insentif;

Desainer(String nama, double gajiDasar, double insentif) {

super(nama, gajiDasar);

this.insentif = insentif;

}

@Override

double hitungGaji() {

return gajiDasar + insentif;

}

}

public class Main {

public static void main(String[] args) {

// Membuat array dari objek Karyawan

Karyawan[] karyawan = new Karyawan[3];

// Mengisi array dengan objek Manajer, Programmer, dan Desainer

karyawan[0] = new Manajer("Alice", 5000000, 2000000);

karyawan[1] = new Programmer("Bob", 4000000, 50, 100000);

karyawan[2] = new Desainer("Charlie", 3000000, 1500000);

// Menggunakan loop untuk menghitung dan menampilkan gaji

for (Karyawan k : karyawan) {

System.out.println("Nama: " + k.nama + ", Gaji: " + k.hitungGaji());

}

}

}

Output :

Nama: Alice, Gaji: 7000000.0

Nama: Bob, Gaji: 9000000.0

Nama: Charlie, Gaji: 4500000.0

Contoh 3

// Kelas dasar Buku

abstract class Buku {

String judul;

String pengarang;

Buku(String judul, String pengarang) {

this.judul = judul;

this.pengarang = pengarang;

}

// Metode abstrak untuk mendapatkan deskripsi buku, akan diimplementasikan oleh subclass

abstract String dapatkanDeskripsi();

}

// Kelas Fiksi

class Fiksi extends Buku {

String genre;

Fiksi(String judul, String pengarang, String genre) {

super(judul, pengarang);

this.genre = genre;

}

@Override

String dapatkanDeskripsi() {

return "Fiksi: " + judul + " oleh " + pengarang + ", Genre: " + genre;

}

}

// Kelas NonFiksi

class NonFiksi extends Buku {

String subjek;

NonFiksi(String judul, String pengarang, String subjek) {

super(judul, pengarang);

this.subjek = subjek;

}

@Override

String dapatkanDeskripsi() {

return "Non-Fiksi: " + judul + " oleh " + pengarang + ", Subjek: " + subjek;

}

}

// Kelas Ensiklopedia

class Ensiklopedia extends Buku {

int jumlahVolume;

Ensiklopedia(String judul, String pengarang, int jumlahVolume) {

super(judul, pengarang);

this.jumlahVolume = jumlahVolume;

}

@Override

String dapatkanDeskripsi() {

return "Ensiklopedia: " + judul + " oleh " + pengarang + ", Jumlah Volume: " + jumlahVolume;

}

}

public class Main {

public static void main(String[] args) {

// Membuat array dari objek Buku

Buku[] koleksiBuku = new Buku[3];

// Menambahkan beberapa buku ke koleksi

koleksiBuku[0] = new Fiksi("Harry Potter", "J.K. Rowling", "Fantasy");

koleksiBuku[1] = new NonFiksi("Sapiens", "Yuval Noah Harari", "Sejarah");

koleksiBuku[2] = new Ensiklopedia("Britannica", "Encyclopædia Britannica, Inc.", 32);

// Menggunakan loop untuk menampilkan deskripsi setiap buku

for (Buku buku : koleksiBuku) {

System.out.println(buku.dapatkanDeskripsi());

}

}

}

Output :

Fiksi: Harry Potter oleh J.K. Rowling, Genre: Fantasy

Non-Fiksi: Sapiens oleh Yuval Noah Harari, Subjek: Sejarah

Ensiklopedia: Britannica oleh Encyclopædia Britannica, Inc., Jumlah Volume: 32