**Latihan Inheritance Java**

**File Kendaraan.java**

| package Kendaraan; |
| --- |
| **Abstract Class Kendaraan** |
| // Kelas abstrak Kendaraan  abstract class Kendaraan {  // Atribut  protected int nomor;  protected String merk;  protected int tahun\_keluaran;  // Default constructor  public Kendaraan() {  this.nomor = 0;  this.merk = "";  this.tahun\_keluaran = 0;  }  // User-defined constructor  public Kendaraan(int nomor, String merk, int tahun\_keluaran) {  this.nomor = nomor;  this.merk = merk;  this.tahun\_keluaran = tahun\_keluaran;  }  // Copy constructor  public Kendaraan(Kendaraan k) {  this.nomor = k.nomor;  this.merk = k.merk;  this.tahun\_keluaran = k.tahun\_keluaran;  }  // Method untuk print informasi kendaraan  public void printInfo() {  System.out.println("Nomor: " + nomor);  System.out.println("Merk: " + merk);  System.out.println("Tahun Keluaran: " + tahun\_keluaran);  }  // Method abstrak untuk menghitung biaya sewa  abstract int biayaSewa(int lamaSewa);  } |

| **Class Bus sebagai Turunan dari Class Kendaraan** |
| --- |
| // Kelas Bus yang merupakan turunan dari Kendaraan  public class Bus extends Kendaraan {  // Atribut  private int kapasitas;  // Default constructor  public Bus() {  super(); // Memanggil constructor default dari kelas induk yaitu Kendaraan  this.kapasitas = 0; // Set kapasitas default = 0  }  // User-defined constructor  public Bus(int nomor, String merk, int tahun\_keluaran, int kapasitas) {  super(nomor, merk, tahun\_keluaran); // Memanggil constructor user-defined dari kelas induk yaitu Kendaraan  this.kapasitas = kapasitas;  }  // Copy constructor  public Bus(Bus b) {  super(b); // Memanggil copy constructor dari kelas induk yaitu Kendaraan  this.kapasitas = b.kapasitas;  }  // Method untuk print informasi kendaraan  public void printInfo() {  super.printInfo(); // Memanggil method printInfo dari kelas induk yaitu Kendaraan  System.out.println("Kategori: Bus"); // Print kategori Bus  System.out.println("Kapasitas: " + kapasitas); // Print kapasitas bus  }    // Method untuk menghitung biaya sewa kelas Bus  public int biayaSewa(int lamaSewa) {  return lamaSewa \* 1000000;  }  } |

| **Class MiniBus sebagai Turunan dari Class Kendaraan** |
| --- |
| // Kelas MiniBus yang merupakan turunan dari Kendaraan  public class MiniBus extends Kendaraan {  // Default constructor  public MiniBus() {  super(); // Memanggil constructor default dari kelas induk yaitu Kendaraan  }  // User-defined constructor  public MiniBus(int nomor, String merk, int tahun\_keluaran) {  super(nomor, merk, tahun\_keluaran); // Memanggil constructor user-defined dari kelas induk yaitu Kendaraan  }  // Copy constructor  public MiniBus(MiniBus m) {  super(m); // Memanggil copy constructor dari kelas induk yaitu Kendaraan  }  // Method untuk print informasi kendaraan  public void printInfo() {  super.printInfo(); // Memanggil method printInfo dari kelas induk yaitu Kendaraan  System.out.println("Kategori: MiniBus"); // Print kategori MiniBus  }  // Method untuk menghitung biaya sewa kelas MiniBus  public int biayaSewa(int lamaSewa) {  if (lamaSewa <= 5) {  return 5000000;  }  else {  return (5000000 + 500000 \* (lamaSewa - 5));  }  }  } |

| **Class Mobil sebagai Turunan dari Class Kendaraan** |
| --- |
| // Kelas Mobil yang merupakan turunan dari Kendaraan  public class Mobil extends Kendaraan {  // Atribut  private String supir;  // Default constructor  public Mobil() {  super(); // Memanggil constructor default dari kelas induk yaitu Kendaraan  this.supir = ""; // Set supir default = ""  }  // User-defined constructor  public Mobil(int nomor, String merk, int tahun\_keluaran, String supir) {  super(nomor, merk, tahun\_keluaran); // Memanggil constructor user-defined dari kelas induk yaitu Kendaraan  this.supir = supir;  }  // Copy constructor  public Mobil(Mobil m) {  super(m);  this.supir = m.supir;  }  // Method untuk print informasi kendaraan  public void printInfo() {  super.printInfo(); // Memanggil method printInfo dari kelas induk yaitu Kendaraan  System.out.println("Kategori: Mobil"); // Print kategori Mobil  System.out.println("Supir: " + supir); // Print nama supir  }  // Method untuk menghitung biaya sewa kelas Mobil  public int biayaSewa(int lamaSewa) {  return lamaSewa \* 500000;  }  } |

| **Class KoleksiKendaraan** |
| --- |
| // Kelas KoleksiKendaraan  public class KoleksiKendaraan {  // Atribut  private int size;  private int Neff;  private Kendaraan koleksi[];  // Default constructor  public KoleksiKendaraan() {  this.size = 10;  this.Neff = 0;  this.koleksi = new Kendaraan[this.size];  }  // User-defined constructor  public KoleksiKendaraan(int size) {  this.size = size;  this.Neff = 0;  this.koleksi = new Kendaraan[this.size];  }  // Copy constructor  public KoleksiKendaraan(KoleksiKendaraan kd) {  this.size = kd.size;  this.Neff = kd.Neff;  this.koleksi = new Kendaraan[this.size];  for (int i = 0; i < this.Neff; i++) {  this.koleksi[i] = kd.koleksi[i];  }  }  // Method untuk print semua informasi dari kendaraan pada koleksi kendaraan  public void printAll() {  for (int i = 0; i < this.Neff; i++) {  this.koleksi[i].printInfo();  }  }  // Method untuk menambah kendaraan ke dalam koleksi kendaraan  public void addKendaraan(Kendaraan k) {  if (this.Neff == this.size) {  Kendaraan k1[] = new Kendaraan[this.size \* 2]; // Mengextend size array  for (int i = 0; i < this.Neff; i++) {  k1[i] = this.koleksi[i];  }  this.koleksi = k1;  this.size \*= 2;  }  this.koleksi[this.Neff] = k;  this.Neff++;  }  // Method untuk menambah koleksi kendaraan ke dalam koleksi kendaraan  public void addKoleksiKendaraan(KoleksiKendaraan kd) {  if (this.Neff + kd.Neff > this.size) {  Kendaraan k1[] = new Kendaraan[this.size \* 2]; // Mengextend size array  for (int i = 0; i < this.Neff; i++) {  k1[i] = this.koleksi[i];  }  this.koleksi = k1;  this.size \*= 2;  }  for (int i = 0; i < kd.Neff; i++) {  this.koleksi[this.Neff] = kd.koleksi[i];  this.Neff++;  }  }  } |

**Pengetesan pada file Main.java**

| package Kendaraan;  public class Main {  public static void main(String[] args) {  // Membuat objek koleksi kendaraan  KoleksiKendaraan koleksi = new KoleksiKendaraan();  // Membuat beberapa objek kendaraan  Kendaraan bus = new Bus(123, "Mercedes", 2020, 50);  Kendaraan miniBus = new MiniBus(456, "Toyota", 2018);  Kendaraan mobil = new Mobil(789, "Honda", 2019, "Dudung");  // Menambahkan kendaraan ke dalam koleksi  koleksi.addKendaraan(bus);  koleksi.addKendaraan(miniBus);  koleksi.addKendaraan(mobil);  // Mencetak informasi semua kendaraan dalam koleksi  koleksi.printAll();  }  } |
| --- |