Class: Person

Attribute:

-nama: String

-umur: int

-alamat: String

Method:

+Person(nama: String, umur: int, alamat: String): void

+getNama(): String

+getUmur(): int

+getAlamat(): String

+setNama(nama: String): void

+setUmur(umur: int): void

+setAlamat(alamat: String): void

Source code:

public class Person {

private String nama;

private int umur;

private String alamat;

public Person(String nama, int umur, String alamat) {

this.nama = nama;

this.umur = umur;

this.alamat = alamat;

}

public String getName() {

return nama;

}

public void setName(String nama) {

if (nama == null || nama.trim().isEmpty()) {

throw new IllegalArgumentException("Name cannot be null or empty");

}

this.nama = nama;

}

public int getAge() {

return umur;

}

public void setAge(int umur) {

if (umur < 0) {

throw new IllegalArgumentException("Age cannot be negative");

}

this.umur = umur;

}

public String getAddress() {

return alamat;

}

public void setAddress(String alamat) {

if (alamat == null || alamat.trim().isEmpty()) {

throw new IllegalArgumentException("Address cannot be null or empty");

}

this.alamat = alamat;

}

@Override

public String toString() {

return "Person{" +

"nama='" + nama + '\'' +

", umur=" + umur +

", alamat='" + alamat + '\'' +

'}';

}

public static void main(String[] args) {

Person person = new Person("Jhony", 25, "asumta blok d 12");

System.out.println(person);

person.setName("gibran");

person.setAge(30);

person.setAddress("mutiara blok f 02");

System.out.println(person);

}

}

Analisa:

Kode Java di atas menjelaskan kelas Person yang memiliki tiga atribut privat: nama, umur, dan alamat. Kelas ini juga memiliki konstruktor dengan tiga parameter untuk menginisialisasi atribut-atribut tersebut.

Kelas Person memiliki beberapa method getter dan setter untuk mengambil dan mengubah nilai atribut. Method setter memeriksa nilai yang akan diset dan akan me throw exception jika nilai yang diset tidak memenuhi syarat. Misalnya, method setName akan me throw IllegalArgumentException jika nilai yang diset kosong atau null.

Kelas Person juga memiliki method toString yang akan dipanggil saat object Person dicetak. Method ini akan mengembalikan string yang berisi informasi tentang object Person.

Di dalam method main, kita membuat object Person dengan nama "Jhony", umur 25, dan alamat "asumta blok d 12". Kita mencetak object Person tersebut dan melihat bahwa informasi yang tersimpan dalam object tercetak juga.

Kita kemudian mengubah nilai atribut Person tersebut dengan menggunakan method setter dan mencetak object Person lagi. Hasilnya, informasi yang tersimpan dalam object Person telah berubah menjadi nama "gibran", umur 30, dan alamat "mutiara blok f 02".

2.

Class: Persegi Panjang

Attribute:

-lebar: int

-tinggi: int

Method:

+PersegiPanjang(lebar: int, tinggi: int): void

+lebar(): int

+tinggi(): int

+setLebar(lebar: int): void

+setTinggi(tinggi: int): void

+getLuas(): int

+getKeliling(): int

Source code:

public class Rectangle {

private int lebar;

private int tinggi;

public Rectangle(int lebar, int tinggi) {

this.lebar = lebar;

this.tinggi = tinggi;

}

public int getWidth() {

return lebar;

}

public int getHeight() {

return tinggi;

}

public int getArea() {

return lebar \* tinggi;

}

public int getPerimeter() {

return 2 \* (lebar + tinggi);

}

public void setWidth(int lebar) {

this.lebar = lebar;

}

public void setHeight(int tinggi) {

this.tinggi = tinggi;

}

public static void main(String[] args) {

Rectangle rectangle = new Rectangle(5, 10);

System.out.println("lebar: " + rectangle.getWidth());

System.out.println("tinggi: " + rectangle.getHeight());

System.out.println("luas: " + rectangle.getArea());

System.out.println("keliling: " + rectangle.getPerimeter());

rectangle.setWidth(10);

rectangle.setHeight(5);

System.out.println("Updated lebar: " + rectangle.getWidth());

System.out.println("Updated tinggi: " + rectangle.getHeight());

System.out.println("Updated luas: " + rectangle.getArea());

System.out.println("Updated keliling: " + rectangle.getPerimeter());

}

}

Analisa:

Kode di atas merupakan implementasi kelas Rectangle (persegi panjang) dalam bahasa pemrograman Java. Kelas ini memiliki dua atribut privat, yaitu lebar dan tinggi, yang digunakan untuk menyimpan dimensi persegi panjang.

Kelas Rectangle memiliki konstruktor dengan dua parameter untuk menginisialisasi nilai atribut lebar dan tinggi. Kelas ini juga memiliki beberapa method getter dan setter untuk mengambil dan mengubah nilai atribut.

Kelas Rectangle juga memiliki dua method getArea dan getPerimeter yang digunakan untuk menghitung luas dan keliling persegi panjang. Method getArea mengalikan nilai lebar dan tinggi untuk menghitung luas, sedangkan method getPerimeter mengalikan dua dengan jumlah lebar dan tinggi untuk menghitung keliling.

Di dalam method main, kita membuat object Rectangle dengan lebar 5 dan tinggi 10. Kita mencetak beberapa informasi tentang object Rectangle tersebut, seperti lebar, tinggi, luas, dan keliling.

Setelah itu, kita mengubah nilai atribut lebar dan tinggi dengan menggunakan method setter dan mencetak kembali informasi tersebut. Hasilnya, informasi yang tersimpan dalam object Rectangle telah berubah menjadi lebar 10 dan tinggi 5.

Selain itu, kode di atas juga memiliki beberapa komentar yang menjelaskan tentang fungsi dari kode tersebut.

3.

Class: Akun Bank

Attribute:

nomorRekening: String

saldo: double

Method:

AkunBank(nomorRekening: String, saldo: double): void

getNomorRekening(): String

getSaldo(): double

simpan(jumlah: double): void

ambil(jumlah: double): void

transfer(jumlah: double, targetAkun: AkunBank): void

public class RekeningBank {

private String nomorRekening;

private double saldo;

public RekeningBank(String nomorRekening, double saldo) {

this.nomorRekening = nomorRekening;

this.saldo = saldo;

}

public String getNomorRekening() {

return nomorRekening;

}

public double getSaldo() {

return saldo;

}

public void deposit(double jumlah) {

saldo += jumlah;

}

public boolean withdraw(double jumlah) {

if (saldo >= jumlah) {

saldo -= jumlah;

return true;

} else {

return false;

}

}

public boolean transfer(double jumlah, RekeningBank rekeningTujuan) {

if (withdraw(jumlah)) {

rekeningTujuan.deposit(jumlah);

return true;

} else {

return false;

}

}

public static void main(String[] args) {

RekeningBank rekening1 = new RekeningBank("123456", 1000.00);

RekeningBank rekening2 = new RekeningBank("789101", 500.00);

System.out.println("Saldo rekening 1: " + rekening1.getSaldo());

System.out.println("Saldo rekening 2: " + rekening2.getSaldo());

rekening1.withdraw(200.00);

rekening2.deposit(200.00);

System.out.println("Saldo rekening 1 setelah penarikan: " + rekening1.getSaldo());

System.out.println("Saldo rekening 2 setelah penyetoran: " + rekening2.getSaldo());

rekening1.transfer(500.00, rekening2);

System.out.println("Saldo rekening 1 setelah transfer: " + rekening1.getSaldo());

System.out.println("Saldo rekening 2 setelah transfer: " + rekening2.getSaldo());

}

}

Analisa:

Kode di atas merupakan implementasi kelas RekeningBank (Rekening Bank) dalam bahasa pemrograman Java. Kelas ini memiliki dua atribut privat, yaitu nomorRekening dan saldo, yang digunakan untuk menyimpan nomor rekening dan saldo rekening.

Kelas RekeningBank memiliki konstruktor dengan dua parameter untuk menginisialisasi nilai atribut nomorRekening dan saldo. Kelas ini juga memiliki beberapa method getter dan setter untuk mengambil dan mengubah nilai atribut.

Kelas RekeningBank juga memiliki beberapa method untuk melakukan transaksi, yaitu deposit, withdraw, dan transfer. Method deposit digunakan untuk menambahkan jumlah uang ke saldo rekening, method withdraw digunakan untuk mengurangi jumlah uang dari saldo rekening (jika saldo cukup), dan method transfer digunakan untuk mentransfer jumlah uang tertentu ke rekening lain.

Di dalam method main, kita membuat dua object RekeningBank dengan nomor rekening dan saldo yang berbeda. Kita mencetak beberapa informasi tentang object RekeningBank tersebut, seperti nomor rekening dan saldo.

Setelah itu, kita melakukan penarikan uang dari rekening pertama dan setoran uang ke rekening kedua dengan menggunakan method withdraw dan deposit. Kita mencetak kembali informasi tersebut.

Selanjutnya, kita melakukan transfer uang dari rekening pertama ke rekening kedua dengan menggunakan method transfer. Hasilnya, uang akan dipindahkan dari rekening pertama ke rekening kedua.

Selain itu, kode di atas juga memiliki beberapa komentar yang menjelaskan tentang fungsi dari kode tersebut.