



MODUL

## PEMROGRAMAN BERORIENTASI OBJEK 1



**2024**

Sri Anardani, S.Kom., M.T  
Yoga Prisma Yuda, S.Kom., M.Kom



PROGRAM STUDI S1 TEKNIK INFORMATIKA  
UNIVERSITAS PGRI MADIUN  
JL.AURI no 14-16 Madiun

---

**MODUL PRAKTIKUM**  
**PEMROGRAMAN BERORIENTASI OBJEK 1**  
**(Semester Genap 2023/2024)**

**Tim Penyusun:**

1. Sri Anardani, S.Kom.,MT
2. Yoga Prisma Yuda, S.Kom, M.Kom

Program Studi Teknik Informatika

UNIVERSITAS PGRI MADIUN

2024

## **DAFTAR ISI**

	Hal
Kata Pengantar-----	2
Tata Tertib Praktikum-----	3
1 Object dan Class-----	4
2 Constroktur dan Method-----	8
3 Inheritance -----	12
4 Encapsulation-----	15
5 Polymorphism -----	18
Daftar Pustaka-----	22

## KATA PENGANTAR

Pemrograman Java dalam Pemrograman Berorientasi Objek (PBO) memiliki kemampuan berjalan pada berbagai platform dan memiliki fitur yang sesuai kebutuhan para pengembang perangkat lunak merupakan keunggulan yang layak untuk dipelajari. Dengan memahami teori dasar serta penerapan dari PBO, mahasiswa diharapkan mampu mengembangkan aplikasi dari nol (awal) atau bisa menggunakan framework di berbagai platform (dekstop, mobile, web) yang dikembangkan di beberapa bahasa pemrograman.

Dalam rangka melengkapi materi kuliah Pemrograman Berorientasi Objek 1 maka modul praktikum ini disusun untuk membahas materi Java mulai dari dasar hingga penerapannya dalam bentuk sederhana. Diharapkan modul ini bisa membantu mahasiswa untuk memahami materi Java.

Penyusunan modul ini telah melalui tahapan dari penelitian, namun tentu saja masih ada ketidaksempurnaan. Pengembangan terhadap modul ini akan terus dilakukan ke depannya. Penulis ucapan terima kasih atas dukungan yg telah diberikan oleh seluruh civitas akademika Universitas PGRI Madiun.

Madiun, 02 Februari 2024

Tim Penulis

## **TATA TERTIB PRAKTIKUM**

- 1. Praktikan wajib membawa Modul Praktikum, bagi yang tidak membawa tidak diperkenankan mengikuti praktikum.**
- 2. Praktikan mengenakan pakaian yang sesuai dengan ketentuan (tidak diperkenankan menggunakan kaos/t-shirt oblong)**
- 3. Wajib datang tepat pada waktunya.**
- 4. Selama berada di dalam Laboratorium Komputer, praktikan wajib mentaati Dosen Pengampu, Laboran, Asisten Lab. Dan Tata Tertib Praktikum.**
- 5. Praktikan wajib mengganti peralatan di Laboratorium Komputer jika kedapatan merusak atau menghilangkan peralatan tersebut (baik sengaja maupun tidak sengaja)**

# PENGENALAN CLASS DAN OBJECT

1

## TUJUAN:

1. Mahasiswa mengenal class dan object
2. Mahasiswa mampu membuat class dan object

## PERSIAPAN:

1. Perangkat lunak Java JDK17
2. Apache Netbeans IDE 14

## DASAR TEORI:

**Class** merupakan rancangan yang mendefinisikan variable dan method pada objek tertentu. Class berfungsi menampung isi dari program yang akan dijalankan, dimana program berisi attribute/type data serta method untuk menjalankan program.

**Object** merupakan benda nyata yang dibuat berdasarkan rancangan yang di definisikan di dalam class. Object merupakan instance dari class. Contoh Class Buah memiliki object mangga, apel, anggur, dll

- Struktur penulisan Java adalah sebagai berikut:

```
-----  
| public class namakelas  
| {  
|     public static void main (String args[])  
|     {  
|         Kode program  
|     }  
| }
```

- Struktur penyimpanan file Java :*namakelas.java*
- Contoh Penulisan Java:

```
-----  
| public class Buah {  
|     public static void main (String args[])  
|     {  
|  
|         System.out.println("Nama' Buah: Mangga");  
|     }  
| }
```

- **Hasil Run:**

```
run:  
  Nama Buah: Mangga  
  BUILD SUCCESSFUL (total time: 0 seconds)
```

**PRAKTIKUM 1:**

1. Buka Apache Netbenas IDE 14

2. Buatlah object Dosen yang berisi attribute dosen: nik, nama, prodi

3. Ketik source code berikut ini:

```
-----  
  public class Dosen  
  {  
    public String nik, nama, prodi;  
  }  
-----
```

**Keterangan:**

a. Nama file tersebut adalah **Dosen.java** nama kelas adalah **Dosen**

b. Kode program berisi attribute yang dimiliki mahasiswa yaitu: **nik, nama, prodi.**

4. Kode diatas belum bisa langsung dijalankan karena belum memiliki **Class Main** yang digunakan untuk mengeksekusi kode program pada class Dosen. Maka buat 1 file simpan dengan nama **MainDosen.java** kemudian ketik source code berikut ini:

```
public class MainDosen
{
    public static void main(String[] args)
    {
        Dosen dsn1 = new Dosen();
        dsn1.nama = "Bu Dani";
        dsn1.prodi = "Informatika";
        Dosen dsn2 = new Dosen();
        dsn2.nama = "Pak yoga";
        dsn2.prodi = "Informatika";

        System.out.println("Object dsn1");
        System.out.println("Nama Dosen: "+ dsn1.nama);
        System.out.println("Nama Prodi: "+ dsn1.prodi);

        System.out.println("-----");

        System.out.println("Object dsn2");
        System.out.println("Nama Dosen: "+ dsn2.nama);
        System.out.println("Nama Prodi: "+ dsn2.prodi);
    }
}
```

**Keterangan:**

- a. Pada source code diatas terdapat 2 object dosen dari **Class Dosen**, yaitu dsn1 dan dsn2.
- b. Setiap object memiliki attribute sesuai dengan **Class Dosen**

**Hasil Output:**

```
run:
Object dsn1
Nama Dosen: Bu Dani
Nama Prodi: Informatika
-----
Object dsn2
Nama Dosen: Pak yoga
Nama Prodi: Informatika
```

**TUGAS PRAKTIKUM 1**

1. Buatlah program untuk menampilkan attribute class Mobil sehingga menghasilkan output seperti berikut ini:

```
run:  
Mobil saya :kuning  
Tahun Produksi Mobil saya :2009  
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Buatlah program untuk menghitung Luas Persegi Panjang dengan attribute panjang dan lebar. Nilai parameter dalam bentuk inputan pengguna sehingga method dapat bersifat dinamis. Output yang dihasilkan adalah sebagai berikut:

```
W run:  
W Masukkan Panjang :12  
W Masukkan Lebar :10  
Luas Persegi Panjang=120.0  
BUILD SUCCESSFUL (total time: 11 seconds)
```

TUJUAN:

1. Mahasiswa mengenal Constructor dan Method
2. Mahasiswa mampu membuat Constructor dan Method

PERSIAPAN:

1. Perangkat lunak Java JDK17
2. Apache Netbeans IDE 14

DASAR TEORI:

**1. CONSTRUCTOR**

Constructor merupakan method khusus yang akan dieksekusi pada saat pembuatan objek (instance). Method ini digunakan untuk inisialisasi atau mempersiapkan data untuk objek. Maka kita harus membuat parameter sebagai inputan untuk constructor. Namun Constructor dapat dibuat dengan atau tanpa menggunakan parameter.

**2. METHOD**

Merupakan kumpulan baris kode yang dikelompokkan untuk menjalankan tugas tertentu. Sebuah method dapat mengembalikan nilai tertentu (memiliki *return value*), bisa juga tidak (*void*). Method dapat kita panggil berulang-ulang dari mana saja dalam program kita. Dengan method kita tidak perlu menulis kode program yang melakukan hal sama berkali-kali

PRAKTIKUM 2

• **Contoh Constructor:**

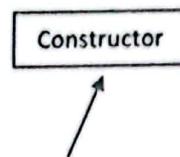
```
Membuat Class SiswaP2 dengan attribute nama, kelas  
-----  
public class SiswaP2 {  
    public String nama, kelas;  
    public SiswaP2(String nama, String kelas)  
    {  
        this.nama = nama;  
        this.kelas = kelas;  
    }  
-----
```

- Kemudian pada Class MainSiswa2 kita akan menggunakan Constructor untuk membuat object dengan mendefinisikan nilai setiap attribute

```

public class MainSiswa2 {
    public static void main(String[] args) {
        Siswa siswa1 = new Siswa("Bu Dani", "anggrek");
        Siswa siswa2 = new Siswa("Pak Yoga", "pinguin");
        siswa1.BiayaTrip(400000);
    }
}

```



- Pada source code diatas Class SiswaP2 memiliki 2 object yaitu siswa1 dan siswa2. Masing-masing object memiliki attribute nama dan kelas. Memiliki Nilai “Bayu” dan “Anggrek”

- Hasil Output:

```

$ run:
$ Object siswa1
  Nama: Bu Dani
  kelas: anggrek
-----
  Object siswa2
  Nama: Pak Yoga
  kelas: pinguin
BUILD SUCCESSFUL (total time: 0 seconds)

```

- **Contoh Method:**

Membuat method pada class Siswa

```
-- public class SiswaP2 {
    public String nama, kelas;
    public SiswaP2(String nama, String kelas)
    {
        this.nama = nama;
        this.kelas = kelas;
    }
```

```
public void BiayaTrip(Integer biaya)
{
```

```
    Integer biaya_trip = 300000;
    if(biaya >= 300000)
    {
        System.out.println("Biaya Trip Lunas");
    }
    else
    {
        System.out.println("Biaya Trip Maksimal");
        "+biaya+" Rupiah";
    }
}
```

- Pada source code diatas **Method BiayaTrip** memiliki parameter integer **biaya** yang digunakan pada oleh method sesuai dengan kebutuhan. Method **BiayaTrip** tidak memiliki nilai return. Output dikeluarkan dengan fungsi **System.out.println**.
- Source Code pada **Class MainSiswa2** digunakan untuk memanggil method dari **class Siswa**

```
-- public class MainSiswa2 {
    public static void main(String[] args)
    {
        SiswaP2 siswa1 = new SiswaP2("Bu Dani", "anggrek");
        SiswaP2 siswa2 = new SiswaP2("Pak Yoga", "pinguin");
        siswa1.BiayaTrip(400000);
    }
}
```

- **Output:**

```
↳ run:
↳ Biaya Trip Lunas
↳ BUILD SUCCESSFUL (total time: 0 seconds)
```

## TUGAS PRAKTIKUM 2

1. Buatlah Constructor untuk Class dengan Method sesuai dengan instruksi berikut:

**Class:** Mobil

**Attribute:** Warna dan Merk

**Method:** Maju dan Mundur

**Output:**

- ▷ run:
- ▷ Sedan Toyota warna Kuning bergerak maju
- ▷ Minibus Toyota warna Kuning bergerak mundur
- ▷ BUILD SUCCESSFUL (total time: 0 seconds)

**TUJUAN:**

1. Mahasiswa mengenal Inheritance
2. Mahasiswa mampu membuat Inheritance

**PERSIAPAN:**

1. Perangkat lunak Java JDK17
2. Apache Netbeans IDE 14

**DASAR TEORI:**

Merupakan konsep pemrograman dimana sebuah class dapat menurunkan attribute dan method untuk diwariskan kepada class lain. Class yang diturunkan disebut “parent class” atau “super class”. Sedangkan class yang menerima warisan disebut “child class”.

**PRAKTIKUM 3:**

1. Ketik source code untuk class Animal berikut ini:

```
-----  
| public class Animal {  
| // field dan method parent class  
|     String name;  
|     public void eat() {  
|         System.out.println("I can eat");  
|     }  
| }  
  
| // inherit dari Animal  
| class Dog extends Animal {  
  
|     // method baru di subclass  
|     public void display() {  
|         System.out.println("My name is " + name);  
|     }  
| }  
-----
```

**Keterangan:**

- Class Animal merupakan Parent Class yang memiliki attribute nama
- Pada Class Animal memiliki method eat.
- Class Dog merupakan Child Class yang memiliki method display

2. Langkah berikutnya adalah membuat **main class** yang bertujuan untuk membuat **object** dari **Child class Dog**. Ketik source code dari **main class** seperti berikut ini:

```
-----  
public class MainAnimal {  
    public static void main(String[] args) {  
  
        // membuat object dari child class  
        Dog doggy = new Dog();  
  
        // akses field dari parent class  
        doggy.name = "Snoopy";  
        doggy.display();  
  
        // memanggil method dari parent class  
        // menggunakan object dari child class  
        doggy.eat();  
  
    }  
}
```

-----

**Keterangan:**

- Pada Class Main **MainAnimal** berfungsi untuk membuat **object** dari **child class Dog**.
- **Object** dari **class Dog** yaitu **doggy** masing-masing memiliki nilai “**Snoopy**” untuk **attribute name**.
- Setelah **object Dog** dibuat maka **object tersebut** memanggil **method display** dari **child class dog**
- Selain itu **Object Dog** juga dapat memanggil **method dari parent class** yaitu **eat**.
- **Output:**

```
▷ run:  
▷ My name is Snoopy  
▷ I can eat  
▷ BUILD SUCCESSFUL (total time: 0 seconds)
```

### TUGAS PRAKTIKUM 3

1. Buatlah Class Parent Karyawan yang memiliki method TampilNama, TampilJabatan dan sesuaikan attributnya

**Contoh output:** "Karyawan dengan nama Cinta"

"Karyawan dengan jabatan Direktur"

2. Tambahkan Child Class Pimpinan dengan attribute NIK, tunjangan

3. Pada Child Class Pimpinan tambahkan method detailpimpinan

**Contoh Output:** "Cinta adalah pimpinan personalia"

"Cinta memiliki NIK 07223456"

**TUJUAN:**

1. Mahasiswa mengenal Encapsualtion
2. Mahasiswa mampu membuat Encaptulation

**PERSIAPAN:**

1. Perangkat lunak Java JDK17
2. Apache Netbeans IDE 14

**DASAR TEORI:**

Merupakan salah satu prinsip dasar pada Object Oriented Programming yang bertujuan untuk membungkus data dalam satu unit. Maksud dari enkapsulasi adalah untuk menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau di intervensi oleh program lain. Konsep enkapsulasi dilakukan untuk menjaga kebutuhan program agar dapat diakses sewaktu-waktu. Di dalam Java, pengapsulan dapat dilakukan dengan pembentukan kelas-kelas menggunakan keyword class. Penyembunyian informasi dilakukan dengan penerapan keyword **private** dan **protected** pada elemen data. Dengan demikian meskipun Parent

Class (orang tua) dapat mewarikan attribute dan methodnya ke Child Class melalui Inheritance tetapi tidak untuk attribute dan method yang memiliki access modifier **private**.

**PRAKTIKUM 4:**

1. Ketik source code untuk membuat class Lensa berikut ini:

```
-----  
public class Lensa {  
    public String nama, diagnosa;  
    private Integer harga;  
    public Integer getHarga()  
    {  
        return harga;  
    }  
    public void setHarga(Integer harga) {  
        this.harga = harga;  
    }  
    protected void tampilHarga(){  
        System.out.println(this.nama + " memiliki harga " + getHarga());  
    }  
}
```

**Keterangan:**

- Class **Lensa** memiliki attribute yang bersifat **public** yaitu **nama**, **diagnosa**. Sedangkan attribute **harga** memiliki sifat **private**, maka hal ini menunjukkan bahwa attribute **harga** tidak bisa langsung diakses oleh **class turunan (child class)**.
- Class **Lensa** memiliki method **setHarga** yang bersifat **public** dan method **tampilHarga** yang bersifat **protected**.

2. Kemudian tambahkan class baru yang merupakan turunan dari class **lensa**, yaitu class **Optik**:

```
-----  
public class Optik extends Lensa {  
  
    public Integer paket;  
  
    public Optik(String nama, String diagnosa, Integer paket, Integer harga)  
    {  
        super.nama = nama;  
        super.diagnosa = diagnosa;  
        this.paket = paket;  
        setHarga(harga);  
    }  
  
    public void punyaPaket(){  
        System.out.println(this.nama + "punya paket"+ this.paket);  
    }  
}
```

-----

**Keterangan:**

- Class **Mobil** memiliki attribute **paket**
- Pada bagian **Constructor** terdapat parameter dan attribute **nama** dan **diagnosa** yang merupakan attribute dari Parent Class **Lensa**
- Pada **Constructor** juga terdapat parameter **paket** untuk mengatur nilai dari attribute **paket** dari Class **Optik**.
- Pada bagian akhir **Constructor** terdapat pemanggilan method **setHarga** dengan parameter **harga optik**. Method tersebut berasal dari Class **Lensa** yang berfungsi untuk mengatur nilai dari attribute **Lensa**.
- Pada Class **Optik** memiliki method **punyaPaket** yang menampilkan **nama** dari **Optik** serta **jumlah paket**.

3. Langkah berikutnya buatlah Class Main yang digunakan untuk membuat object dari Class Optik, sekaligus digunakan untuk memanggil method pada Class Parent Lensa.

```
public class MainLensa {  
    public static void main(String[] args) {  
        Optik optik1 = new Optik("Andi", "Minus", 4, 100000000);  
        optik1.punyaPaket();  
        optik1.tampilHarga();  
    }  
}
```

**Keterangan:**

- Pada kode diatas dijelaskan bahwa object dari optik adalah optik1 dengan nilai yang di isi melalui konstruktor yaitu ("Andi", "Minus", 4, 100000000)
- Object optik1 memanggil method punyaPaket dan tampilHarga
- Hasil Outputnya sebagai berikut:

```
run:  
Andi punya paket 4  
Andi memiliki harga 100000000  
BUILD SUCCESSFUL (total time: 0 seconds)
```

**Tugas Praktikum Modul 4**

1. Buatlah Class Parent:Manusia yang memiliki method TampilNama dan TampilAlamat.  
Output TampilNama: "Manusia dengan nama Pak Yoga"  
Output TampilAlamat: "Manusia dengan nama Pak Yoga alamat di Ponorogo"
2. Tambahkan Child Class Dosen dengan attribute NIDN, jenis kelamin dan Keahlian
3. Pada Class Dosen tambahkan method DetailDosen  
Output DetailDosen: "Pak Yoga Dosen laki-laki dengan NIDN 0722233399 adalah seorang Dosen dengan Keahlian Multimedia"
4. Attribute Alamat dan jenis kelamin bersifat private

**TUJUAN:**

1. Mahasiswa mengenal Polymorphism
2. Mahasiswa mampu membuat Polymorphism

**PERSIAPAN:**

1. Perangkat lunak Java JDK 17
2. Perangkat lunak Apache Netbeans 14

**DASAR TEORI:**

Dalam ilmu sains Polymorphism adalah sebuah prinsip dimana suatu organisme atau spesies memiliki banyak bentuk serta tahapan (*stages*). Pada konsep Pemrograman Berorientasi Objek Polymorphism merupakan class yang memiliki banyak “bentuk” method yang berbeda walaupun namanya sama. Polymorphism merupakan cara bagaimana sebuah Class mampu berubah menjadi bentuk object dari referensi Class yang lain. Ada 2 jenis Polymorphism yaitu:

1. **Method overloading (statis)** terjadi pada sebuah *class* yang memiliki nama method yang sama tapi memiliki parameter dan tipe data yang berbeda.
2. **Method overriding (dinamis)** terjadi saat kita menggunakan pewarisan (*inheritance*) dan implementasi *interface*. Class anak akan memiliki nama method yang sama, tapi nanti isi dan parameternya bisa berbeda dari *class* induk.

**PRAKTIKUM 5**

1. Ketik source code untuk membuat *class* Kendaraan berikut

```
-----  
public class Kendaraan  
{  
    public String nama, merek;  
  
    public void punyaRoda()  
    {  
        System.out.println("Semua kendaraan punya roda");  
    }  
}
```

2. Tambahkan source code class baru yang merupakan turunan dari **class Kendaraan** yaitu **class Mobil**

```
-----  
public class Mobil extends Kendaraan  
{  
    public Integer roda;  
    public Mobil(String nama, String merek)  
    {  
        super.nama = nama;  
        super.merek = merek;  
    }  
    @Override  
    public void punyaRoda()  
    {  
        System.out.println(this.nama + " adalah kendaraan mobil  
yang punya roda 4");  
    }  
}
```

-----

**Keterangan:**

Pada Class Mobil memiliki method punyaRoda yang menampilkan nama dari mobil serta jumlah rodanya. **Method punyaRoda** merupakan **Override** dari method yang sudah dideklarasikan di Parent Class Kendaraan

3. Tambahkan Class baru yang merupakan Class Turunan dari **Kendaraan** yaitu **Class Motor**

```
-----  
public class Motor extends Kendaraan  
{  
    public Integer roda;  
    public Motor(String nama, String merek)  
    {  
        super.nama = nama;  
        super.merek = merek;  
    }  
  
    @Override  
    public void punyaRoda()  
    {  
        System.out.println(this.nama + ": kendaraan mobil yang  
punya roda 2");  
    }  
}
```

-----

4. Buat source code Class Main untuk membuat object dari Class Mobil yang digunakan untuk memanggil method dari Parent Class Kendaraan

```
public class Main
{
    public static void main(String[] args)
    {
        Kendaraan kendaraan1 = new Kendaraan();
        kendaraan1.punyaRoda();

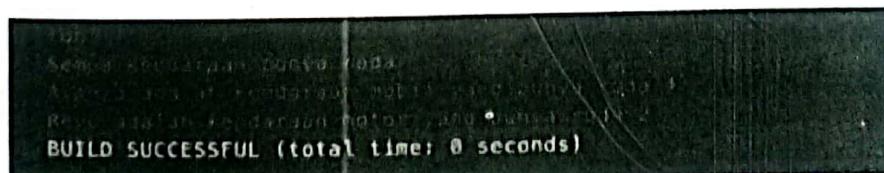
        kendaraan1 = new Mobil("Avanza", "Toyota");
        kendaraan1.punyaRoda();

        kendaraan1 = new Motor("Revo", "Honda");
        kendaraan1.punyaRoda();
    }
}
```

Keterangan:

- Object kendaraan1 memanggil method punyaRoda dari Class Kendaraan
- Object kendaraan1 juga digunakan sebagai object dari Class Mobil dan memanggil method punyaRoda. Method punyaRoda berasal dari Class Mobil.
- Object kendaraan1 juga digunakan sebagai object dari Class Motor dan memanggil method punyaRoda. Method punyaRoda berasal dari Class Motor.
- Maka bisa diketahui bahwa kendaraan1 yang semula adalah object dari Class Kendaraan bisa menjadi object Class Mobil dan Class Motor

5. Hasil Output:



A screenshot of a terminal window displaying Java compilation output. The text shows the compilation of a Java file named 'Main.java' using the 'javac' command. It includes the package declaration 'package kendaraan;', imports for 'java.util.\*', and the definition of the 'Main' class with its constructor and the 'main' method. The output ends with 'BUILD SUCCESSFUL (total time: 0 seconds)'.

```
javac -d . kendaraan/Main.java
package kendaraan;
import java.util.*;
public class Main
{
    public static void main(String[] args)
    {
        kendaraan.Kendaraan kendaraan1 = new kendaraan.Kendaraan();
        kendaraan1.punyaRoda();

        kendaraan1 = new kendaraan.Mobil("Avanza", "Toyota");
        kendaraan1.punyaRoda();

        kendaraan1 = new kendaraan.Motor("Revo", "Honda");
        kendaraan1.punyaRoda();
    }
}
BUILD SUCCESSFUL (total time: 0 seconds)
```

### Tugas Praktikum Modul 5

1. Buat Class Parent: ShapeRuang dengan method deskripsiShape  
Output: "All shapes have an area and a perimeter"
2. Tambahkan Child Class  
**Class:Segitiga; Attribute:Alas,Tinggi; Method:LuasSegitiga**  
**Class:Lingkaran; Attribute:jari-jari; Method:LuasLingkaran**
3. Terapkan Polymorphism pada case study diatas

#### **DAFTAR PUSTAKA**

- 1 Sianipar, 2013, Teori Dan Implementasi JAVA, Penerbit Informatika Bandung
- 2 Bachtiar, A.D dan Fakhrul, F.N., 2018, Pemrograman Berorientasi Objek Menggunakan JAVA, Penerbit Informatika Bandung
- 3 Hendra Kurniawan, Eri Mardiani, Nur Rahmansyah, 2013, Aplikasi Inventory Menggunakan Java NetBeans, XAMMP, dan iReport, Penerbit PT Elex Media Komputindo.
- 4 Abdul Kadir, 2023, Logika Pemrograman Java (Update Version), Penerbit PT Elex Media Komputindo.
- 5 [https://www.kodingindonesia.com/belajar-pemrograman-visual-java-menggunakan-netbeans-bagian-1/#google\\_vignette](https://www.kodingindonesia.com/belajar-pemrograman-visual-java-menggunakan-netbeans-bagian-1/#google_vignette)

**MODUL**  
**PEMROGRAMAN BERDIRENTASI OBJEK 1**