

LAPORAN RESMI
MODUL II
(CONSTRUCTOR DAN KEYWORD STATIC)
PEMROGRAMAN BERBASIS OBJEK



NAMA	: ABAS PERMADANI
N.R.P	: 230441100107
DOSEN	: ACHMAD ZAIN NUR, S.KOM., M.T.
ASISTEN	: MUHAMMAD ROSYID MAULANA
TGL PRAKTIKUM	: 30 MARET 2023

Disetujui : 17 MEI 2024
Asisten

MUHAMMAD ROSYID MAULANA
20.04.411.00019



LABORATORIUM BISNIS INTELIJEN SISTEM
PRODI SISTEM INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA

BAB I

PENDAHULUAN

1.1 Latar Belakang

Constructor adalah metode khusus dalam sebuah kelas yang secara otomatis dipanggil saat objek dari kelas tersebut dibuat. Constructor biasanya digunakan untuk melakukan inisialisasi awal objek, seperti menetapkan nilai awal untuk atribut-atribut objek. Kata kunci static digunakan untuk menyatakan bahwa suatu metode, variabel, atau blok kode merupakan milik kelas (bukan objek) dan dapat diakses langsung dari kelas tersebut tanpa perlu membuat objek dari kelas tersebut.

Kedua konsep ini sangat penting dalam pemrograman berorientasi objek, karena constructor digunakan untuk menginisialisasi objek saat pembuatannya, sementara variabel atau metode static digunakan untuk menyimpan informasi bersama atau menyediakan perilaku yang terkait dengan kelas itu sendiri, bukan dengan setiap objek individu yang dibuat dari kelas tersebut.

1.2 Tujuan

- Mahasiswa mampu memahami konsep Constructor dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.
- Mahasiswa mampu memahami penggunaan keyword static dalam Pemrograman Berorientasi Objek serta mampu mengimplementasikannya.

BAB II

DASAR TEORI

2.1 Constructor

A. Pengertian/Konsep

Constructor adalah method yang secara otomatis dipanggil/dijalankan pada saat new dipakai untuk menciptakan instan kelas. Atau dengan kata lain constructor adalah method yang pertama kali dijalankan pada saat sebuah objek pertama kali diciptakan. Jika dalam sebuah class tidak terdapat constructor, maka secara otomatis Java akan membuatkan sebuah default constructor. Sama halnya dengan method, constructor dapat memiliki satu atau banyak parameter maupun tanpa parameter. Hal mendasar yang perlu diperhatikan, yaitu :

- a. Nama Constructor sama dengan nama Class.
- b. Tidak ada return type yang diberikan kedalam Constructor Signature.
- c. Tidak ada return statement, di dalam tubuh constructor.

B. Contoh Program

- a. Constructor Kendaraan.java

```

1 public class ConstructorKendaraan {
2     private String merk; // ini adalah instant variable
3     private static String pemilik; // ini adalah static variable
4
5     // ini adalah constructor tanpa parameter
6     protected ConstructorKendaraan() {
7         merk = null;
8     }
9
10    // overloading constructor
11    // ini adalah constructor dengan parameter merk
12    protected ConstructorKendaraan(String merk) {
13        this.merk = merk;
14        merk = null;
15    }
16
17    protected void setMerk(String merk) {
18        this.merk = merk;
19    }
20
21    protected String getMerk() {
22        return merk;
23    }
24
25    // ini adalah static method
26    protected static void setPemilik(String pemilik) {
27        ConstructorKendaraan.pemilik = pemilik;
28    }
29
30    // ini adalah static method
31    protected static String getPemilik() {
32        return ConstructorKendaraan.pemilik;
33    }
34
35    protected void tampil(String a)
36    { System.out.println(a);
37      a = null;
38    }
39
40    protected void hapus()
41    { // menghapus variable private dari memory
42      merk = null;
43      pemilik = null;
44    }
45 }

```

b. ConstructorMotor.java

```

1 // ConstructorMotor turunan dari class ConstructorKendaraan
2 public class ConstructorMotor extends ConstructorKendaraan {
3     private String warna; // ini adalah instant variable
4
5     // ini adalah constructor dengan parameter merk & warna
6     protected ConstructorMotor(String merk, String warna) {
7         // memanggil constructor parent (class ConstructorKendaraan)
8         // dengan keyword super dan parameter merk
9         super(merk);
10
11        this.warna = warna;
12
13        // menghapus variable parameter dari memory
14        merk = null;
15        warna = null;
16    }
17
18    protected String getWarna() {
19        return warna;
20    }
21
22    protected void hapus()
23    { // menghapus variable private dari memory
24      warna = null;
25      // memanggil method hapus class parent (class ConstructorKendaraan)
26      // dengan keyword super
27      super.hapus();
28    }
29 }

```

c. MainConstructorMotor.java

```
1 public class MainConstructorMotor {
2     public static void main(String args[])
3     {
4         String pemilik = "Ahmad Afif";
5         String merk = "Honda";
6         String warna = "Merah";
7
8         // cara akses static variable dan static method dapat dengan
9         // memanggil class (ConstructorKendaraan) secara langsung
10        ConstructorKendaraan.setPemilik(pemilik);
11        System.out.println("Pemilik Kendaraan = "+ConstructorKendaraan.getPemilik());
12        System.out.println("=====");
13
14        // variable merk menjadi parameter Constructor pada saat
15        // instansiasi/membuat objek baru (ob)
16        ConstructorKendaraan ob = new ConstructorKendaraan(merk);
17        ob.tampil("Merk Kendaraan = "+ob.getMerk());
18        // cara akses static variable dan static method dapat juga dengan
19        // objek (ob) pada method getPemilik()
20        ob.tampil("Pemilik Kendaraan = "+ob.getPemilik());
21        System.out.println("=====");
22
23        // instansiasi/membuat objek baru (ob2) tanpa parameter
24
25        ConstructorKendaraan ob2 = new ConstructorKendaraan();
26        // bandingkan akses untuk menampilkan nilai pada instant variable (merk)
27        // dan static variable (pemilik) melalui method getter setelah membuat
28        // objek baru "ob2", dimana sebelumnya juga sudah membuat objek "ob".
29        // instant variable (merk) nilainya akan hilang,
30        // sedangkan static variable (pemilik) nilainya tidak hilang/berubah
31        ob2.tampil("Merk Kendaraan (instant variable) = "+ob2.getMerk());
32        ob2.tampil("Pemilik Kendaraan (static variable) = "+ob2.getPemilik());
33        System.out.println("=====");
34
35        // variable merk dan warna menjadi parameter Constructor pada saat
36        // instansiasi/membuat objek baru (ob3)
37        ConstructorMotor ob3 = new ConstructorMotor(merk, warna);
38        ob3.tampil("Merk Motor = "+ob3.getMerk());
39        ob3.tampil("Warna Motor = "+ob3.getWarna());
40        ob3.tampil("Pemilik Motor = "+ob3.getPemilik());
41
42        pemilik = null;
43        merk = null;
44        warna = null;
45        ob.hapus();
46        ob = null;
47        ob2 = null;
48        ob3 = null;
49    }
50 }
```

2.2 Keyword Static

A. Pengertian/Konsep

Dalam pemrograman java, keyword static digunakan untuk mengakses variable ataupun method (prosedur atau fungsi) pada class tertentu tanpa harus membuat suatu objek dari class itu. Umumnya untuk mengakses member dari kelas lain kita harus membuat objek kelas, tapi dengan menggunakan keyword static kita dapat langsung menggunakan member kelas lain. Keyword static bisa digunakan untuk variable ataupun method.

Static member adalah variable atau method yang bukan milik dari suatu object, melainkan milik dari class.

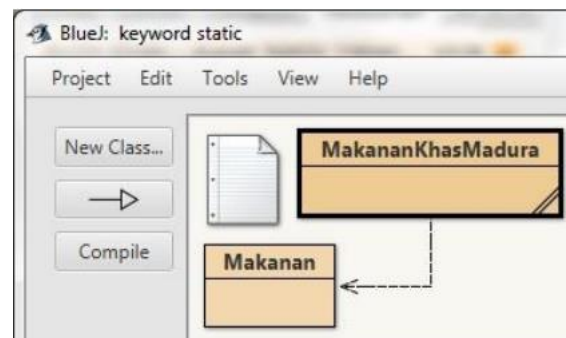
Bentuk Umum:

static void namaMethodStatic(){

Method yang dideklarasikan sebagai static memiliki aturan sebagai berikut :

- a. Hanya dapat dipanggil oleh method lain yang juga adalah static method.
- b. Hanya dapat mengakses atribut static.
- c. Tidak dapat menggunakan keyword this dan super, karena kedua keyword inimenunjuk ke suatu instance tertentu, bukan pada sebuah object.

B. Contoh Program

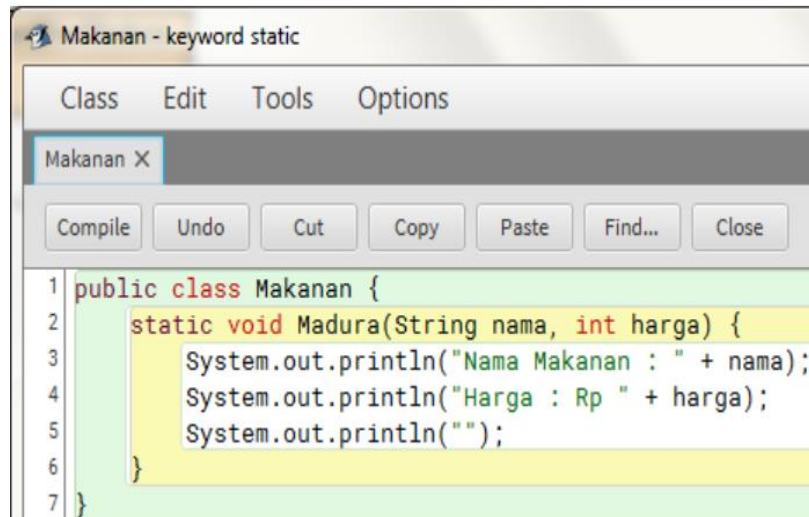


- a. Class Makanan
- b. Class MakananKhasMadura

2.2 Tampilan Class

A. Class Makanan

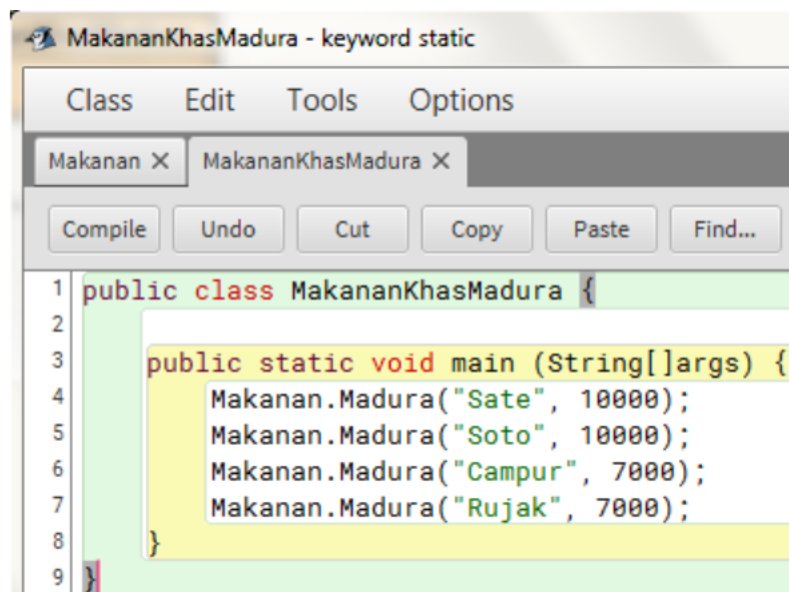
Pada class Makanan, method di set sebagai static dengan parameter makanandan harga.



```
1 public class Makanan {
2     static void Madura(String nama, int harga) {
3         System.out.println("Nama Makanan : " + nama);
4         System.out.println("Harga : Rp " + harga);
5         System.out.println("");
6     }
7 }
```

C. Class MakananKhasMadura

Pada class MakananKhasMadura, method juga bersifat static tanpa membuat instance objek dari kelas Makanan.



```
1 public class MakananKhasMadura {
2
3     public static void main (String[] args) {
4         Makanan.Madura("Sate", 10000);
5         Makanan.Madura("Soto", 10000);
6         Makanan.Madura("Campur", 7000);
7         Makanan.Madura("Rujak", 7000);
8     }
9 }
```

Running Program

Pada saat program di eksekusi, maka akan menampilkan daftar makanan dan harga yang telah diset nilainya. Berikut tampilannya!



```
BlueJ: Terminal Window - keyword static
Options
Nama Makanan : Sate
Harga : Rp 10000

Nama Makanan : Soto
Harga : Rp 10000

Nama Makanan : Campur
Harga : Rp 7000

Nama Makanan : Rujak
Harga : Rp 7000
```

D. Penjelasan Program

a. Class Makanan

Pada class Makanan, terdapat method static dengan parameter makanan dan harga.

- i. Baris 2, keyword static yang berupa method
- ii. Baris 3-4, mencetak nama makanan dan harga.

b. Class MakananKhasMadura

Pada class MakananKhasMadura, method juga bersifat static tanpa membuat instance objek dari kelas Makanan.

- i. Baris 3, method static untuk menjalankan program.
- ii. Baris 4-7, nilai dari parameter yang terdapat pada class Makanan.

BAB III

Tugas Pendahuluan

3.1 soal

1. Jelaskan perbedaan konstruktor dan keyword static!
2. Jelaskan setter dan getter menurut bahasa pemahamanmu!

3.2 jawaban

1. • Constructor adalah method yang secara otomatis dipanggil / dijalankan pada saat new dipakai untuk menciptakan instan kelas. Atau dengan kata lain constructor adalah method yang pertama kali dijalankan pada saat sebuah objek pertama kali diciptakan.
 - keyword static digunakan untuk mengakses variable ataupun method (prosedur atau fungsi) pada class tertentu tanpa harus membuat suatu objek dari class itu.
2. Setter adalah sebuah aksi saat kita memasukkan sebuah nilai atau values kedalam suatu variable atau object, sedangkan Getter adalah sebuah aksi saat kita mengambil sebuah nilai / values dari suatu variable atau object

BAB IV

IMPLEMENTASI

4.1 Soal

1. Buatlah class manusia dengan dengan attribut nama, umur dan alamat serta memiliki method berjalan dan berlari. Buatlah beberapa objek dari class tersebut.
2. Buatlah program java untuk menampilkan nama, nim, jurusan/prodi, dan alamat mahasiswa menggunakan konsep OOP. Nama, nim, jurusan/prodi, dan alamat bersifat dinamis, dengan kata lain menggunakan input dari pengguna.

4.2 Jawaban

- **Source code**

```
package universitas;  
import java.util.ArrayList;  
import java.util.Scanner;
```

```
/**  
 *  
 * @author RIZQY  
 */
```

```
public class UNIVERSITAS{  
    private String nama;  
    private String nim;  
    private String jurusan;  
    private String alamat;
```

```
    public UNIVERSITAS(String nama, String nim, String jurusan, String  
alamat) {
```

```
    this.nama = nama;
    this.nim = nim;
    this.jurusan = jurusan;
    this.alamat = alamat;
}
```

```
//setter untuk memanggil nilai
public void setName(String nama){
    this.nama = nama;
}

//getter untuk menambah nilai
public String getName(){
    return nama ;
}
```

```
//setter untuk nim
public void setNim(String nim){
    this.nim = nim;
}

//getter untuk nim
public String getNim(){
    return nim ;
}
```

```
//setter untuk jurusan
public void setJurusan(String jurusan){
    this.jurusan = jurusan;
}
```

```
//getter untuk jurusan
```

```
public String getJurusan(){  
    return jurusan ;  
}
```

```
//setter untuk alamat
```

```
public void setAlamat(String alamat){  
    this.alamat = alamat;  
}
```

```
//getter untuk alamat
```

```
public String getAlamat(){  
    return alamat ;  
}
```

```
public static void main(String[] args) {
```

```
    Scanner baru = new Scanner(System.in);
```

```
    ArrayList<UNIVERSITAS> daftarMahasiswa = new ArrayList<>();
```

```
    String inputLagi;
```

```
    do {
```

```
        System.out.print("Masukkan NIM: ");
```

```
        String nim = baru.next();
```

```
        baru.nextLine();
```

```
        System.out.print("Masukkan Nama: ");
```

```
        String nama = baru.nextLine();
```

```
        System.out.print("Masukkan Alamat: ");
```

```
String alamat = baru.nextLine();
```

```
System.out.println("Pilihan Jurusan:");
```

```
System.out.println("41 = TEKNIK INFORMATIKA");
```

```
System.out.println("42 = TEKNIK INDUSTRI");
```

```
System.out.println("43 = TEKNIK ELEKTRO");
```

```
System.out.println("44 = SISTEM INFORMASI");
```

```
System.out.println("48 = TEKNIK MESIN");
```

```
System.out.println("49 = TEKNIK MEKATRONIKA");
```

```
System.out.print("Masukkan Kode Jurusan: ");
```

```
String kodeJurusan = baru.next();
```

```
String jurusan;
```

```
// Menggunakan switch-case konvensional
```

```
switch (kodeJurusan) {
```

```
    case "41":
```

```
        jurusan = "TEKNIK INFORMATIKA";
```

```
        break;
```

```
    case "42":
```

```
        jurusan = "TEKNIK INDUSTRI";
```

```
        break;
```

```
    case "43":
```

```
        jurusan = "TEKNIK ELEKTRO";
```

```
        break;
```

```
    case "44":
```

```
        jurusan = "SISTEM INFORMASI";
```

```
        break;
```

```
    case "48":
```

```
        jurusan = "TEKNIK MESIN";
```

```

        break;
    case "49":
        jurusan = "TEKNIK MEKATRONIKA";
        break;
    default:
        jurusan = "jurusan yang anda pilih tidak ada, pilih kode
        jurusan dengan benar";
    }

    UNIVERSITAS mahasiswa = new UNIVERSITAS(nim, nama,
    alamat, jurusan);

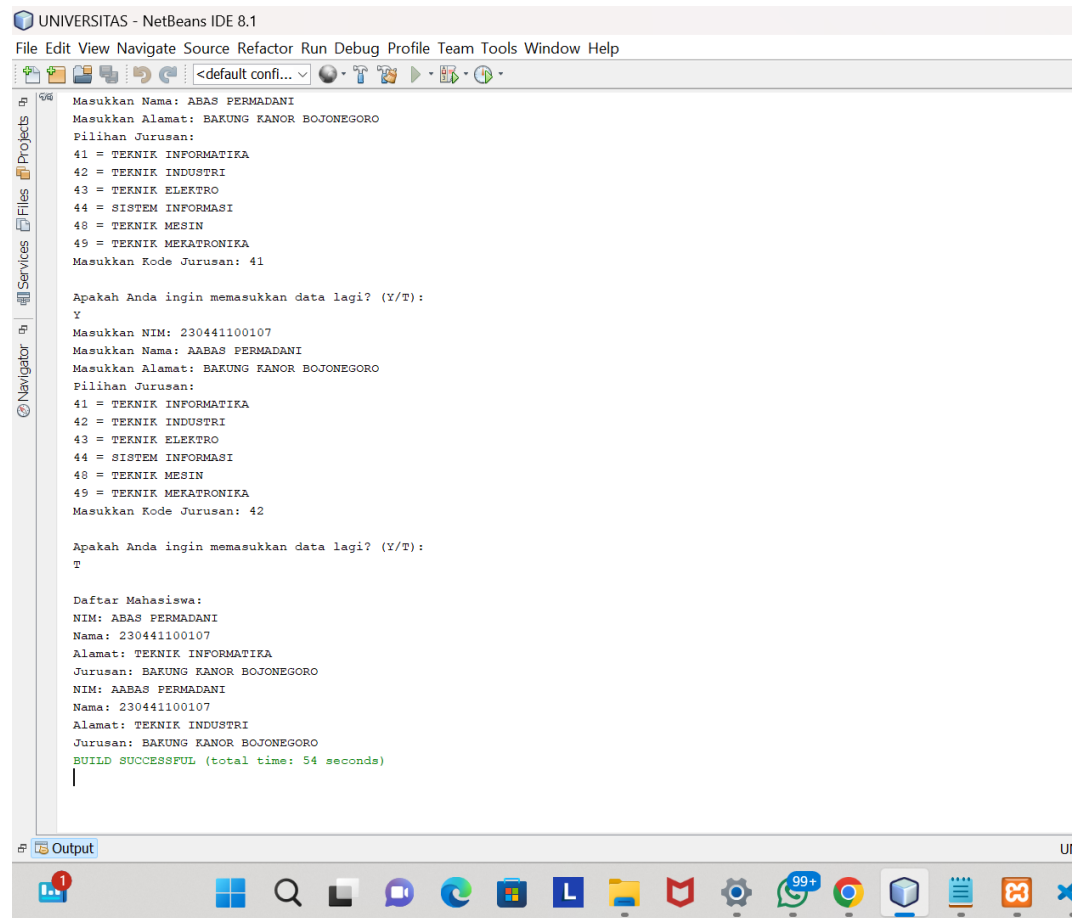
    daftarMahasiswa.add(mahasiswa);

    System.out.println("\nApakah Anda ingin memasukkan data lagi?
    (Y/T): ");
    inputLagi = baru.next();
    } while (inputLagi.equalsIgnoreCase("Y"));

    System.out.println("\nDaftar Mahasiswa:");
    for (UNIVERSITAS mahasiswa1 : daftarMahasiswa) {
        System.out.println("NIM: " + mahasiswa1.getNim());
        System.out.println("Nama: " + mahasiswa1.getNama());
        System.out.println("Alamat: " + mahasiswa1.getAlamat());
        System.out.println("Jurusan: " + mahasiswa1.getJurusan());
    }
}
}
}

```

- **Output**



```
UNIVERSITAS - NetBeans IDE 8.1
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help

Masukkan Nama: ABAS PERMADANI
Masukkan Alamat: BAKUNG KANOR BOJONEGORO
Pilihan Jurusan:
41 = TEKNIK INFORMATIKA
42 = TEKNIK INDUSTRI
43 = TEKNIK ELEKTRO
44 = SISTEM INFORMASI
48 = TEKNIK MESIN
49 = TEKNIK MEKATRONIKA
Masukkan Kode Jurusan: 41

Apakah Anda ingin memasukkan data lagi? (Y/T):
Y
Masukkan NIM: 230441100107
Masukkan Nama: AABAS PERMADANI
Masukkan Alamat: BAKUNG KANOR BOJONEGORO
Pilihan Jurusan:
41 = TEKNIK INFORMATIKA
42 = TEKNIK INDUSTRI
43 = TEKNIK ELEKTRO
44 = SISTEM INFORMASI
48 = TEKNIK MESIN
49 = TEKNIK MEKATRONIKA
Masukkan Kode Jurusan: 42

Apakah Anda ingin memasukkan data lagi? (Y/T):
T

Daftar Mahasiswa:
NIM: ABAS PERMADANI
Nama: 230441100107
Alamat: TEKNIK INFORMATIKA
Jurusan: BAKUNG KANOR BOJONEGORO
NIM: AABAS PERMADANI
Nama: 230441100107
Alamat: TEKNIK INDUSTRI
Jurusan: BAKUNG KANOR BOJONEGORO
BUILD SUCCESSFUL (total time: 54 seconds)
```

- **Penjelasan**

Ini adalah kelas yang merepresentasikan objek mahasiswa. Setiap objek Mahasiswa memiliki atribut seperti nim, nama, alamat, dan jurusan. sebuah constructor yang digunakan untuk membuat objek Mahasiswa

Program meminta pengguna untuk memasukkan data mahasiswa seperti nim, nama, alamat, dan kode jurusan. Setiap data mahasiswa yang dimasukkan oleh pengguna akan dibuatkan objek Mahasiswa baru

Dengan menggunakan program ini, pengguna dapat memasukkan data mahasiswa dan melihat daftar mahasiswa yang telah dimasukkan ke dalam sistem dengan mudah.

BAB V

PENUTUP

5.1 Analisa

Analisis konstruktor dan kata kunci static dalam pemrograman berorientasi objek (OOP) memberikan pemahaman yang mendalam tentang bagaimana kelas bekerja, bagaimana objek dibuat, dan bagaimana data dan perilaku bersama dikelola. Konstruktor bertanggung jawab untuk menginisialisasi objek dari suatu kelas. Ini dilakukan saat objek pertama kali dibuat dengan menggunakan operator new. : Metode static sering digunakan untuk menyediakan fungsi utilitas yang dapat diakses tanpa perlu membuat objek baru. Contohnya adalah metode utilitas matematika, operasi string, dan sebagainya.

Variabel static digunakan untuk menyimpan data bersama yang dibagikan oleh semua objek dari kelas tersebut. Misalnya, hitungan jumlah objek yang telah dibuat dari kelas, atau konstanta yang sama untuk semua objek.

5.2 Kesimpulan

- Konstruktor dan kata kunci static adalah fitur kunci dalam OOP yang menyediakan cara untuk mengelola inisialisasi objek dan data bersama di antara objek-objek dari kelas yang sama.
- Konstruktor memungkinkan inisialisasi objek dengan nilai-nilai awal yang diperlukan, sedangkan static digunakan untuk menyediakan variabel dan metode yang bersama-sama digunakan oleh semua objek dari kelas.
- Dengan menggunakan konstruktor dan kata kunci static dengan bijaksana, pengembang dapat membangun aplikasi yang lebih efisien dan mudah dimengerti dengan memanfaatkan paradigma pemrograman berorientasi objek secara penuh.