**LAPORAN RESMI**

**MODUL II**

**(Constructor dan Keyword Static)**

**PEMROGRAMAN BERBASIS OBJEK**



**NAMA : DONY EKA OCTAVIAN PUTRA**

**N.R.P : 230441100041**

**DOSEN : FIRMANSYAH ADIPUTRA, S.T., M.Cs.**

**ASISTEN : PUTRI NIHAYATUL HUSNA**

**TGL PRAKTIKUM : 26 MARET 2024**

**Disetujui :04. APRIL 2024**

**Asisten**

**PUTRI NIHAYATUL HUSNA 21.04.411.00009**

**LABORATORIUM BISNIS INTELIJEN SISTEM**

**PRODI SISTEM INFORMASI**

**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS TRUNOJOYO MADURA**

# **BAB I PENDAHULUAN**

## **Latar Belakang**

Praktikum Pemrograman Berbasis Objek, yang membahas "Constructor dan Keyword Static", merupakan lanjutan dari pemahaman konsep dasar pemrograman berbasis objek. Konstruktor, sebagai metode khusus dalam sebuah kelas, menjadi kunci utama dalam menginisialisasi objek. Melalui konstruktor, objek dapat diatur dengan nilai awal yang sesuai, memudahkan penggunaan objek dalam program. Penggunaan konstruktor penting karena memungkinkan definisi nilai default dari atribut-atribut objek, sehingga membantu efisiensi dalam pengembangan program

Selain konstruktor, modul ini juga membahas penggunaan kata kunci static. Kata kunci ini digunakan untuk mendefinisikan atribut atau metode yang bersifat global dalam suatu kelas. Dengan kata kunci static, nilai atribut atau metode dapat diakses langsung dari kelas tanpa perlu menginstansiasi objek kelas tersebut. Hal ini mempermudah penggunaan atribut atau metode secara langsung, tanpa harus membuat objek baru.

Diharapkan melalui pemahaman yang mendalam tentang konstruktor dan kata kunci static, mahasiswa dapat lebih menguasai konsep dasar pemrograman berbasis objek. Kemampuan ini akan membantu mereka dalam mengimplementasikan konsep-konsep tersebut dalam pembuatan program-program yang lebih kompleks dan efisien.

## **Tujuan**

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

# **BAB II DASAR TEORI**

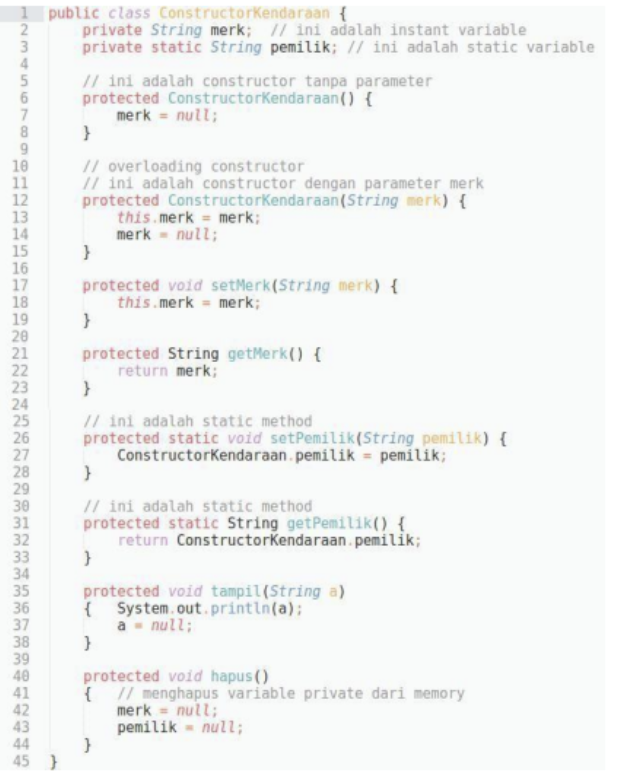
## **Constructor**

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 halnnya dengan method, constructor dapat memiliki satu atau banyak parameter maupun tanpa parameter. Hal mendasar yang perlu diperhatikan, yaitu :

1. Nama Constructor sama dengan nama Class.
2. Tidak ada return type yang diberikan kedalam Constructor Signature.
3. Tidak ada return statement, di dalam tubuh constructor.

**Contoh Program**

* 1. **Constructor Kendaraan.java**

****

* 1. **ConstructorMotor.java**



* 1. **MainConstructorMotor.java**

****



## **2,2 Keyword Static**

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.

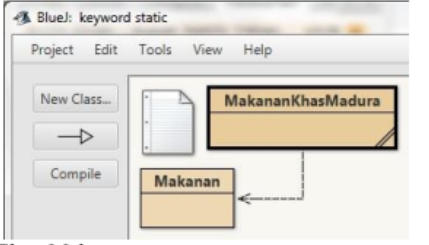
Bentuk Umum:

static void namaMethodStatic(){

Method yang dideklarasikan sebagai static memiliki aturan sebagai berikut

* 1. Hanya dapat dipanggil oleh method lain yang juga adalah static method.
  2. Hanya dapat mengakses atribut static.
  3. Tidak dapat menggunakan keyword this dan super, karena kedua keyword inimenunjuk ke suatu instance tertentu, bukan pada sebuah object.

**Contoh Program**

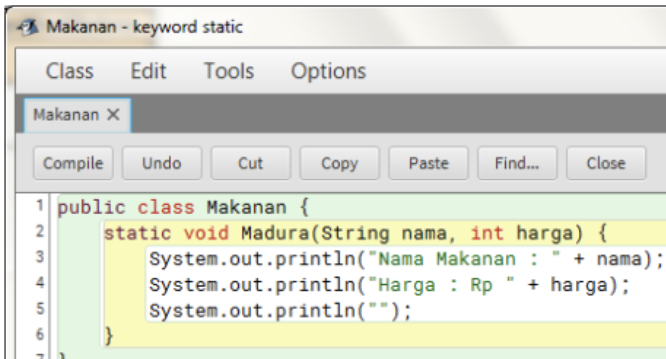
****

* 1. Class Makanan
  2. Class MakananKhasMadura

**Tampilan Class**

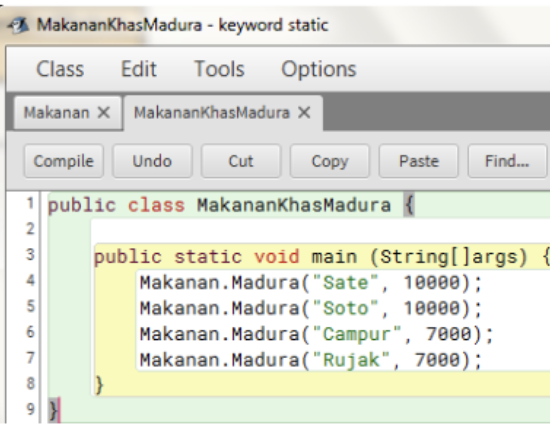
1. **Class Makanan**

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



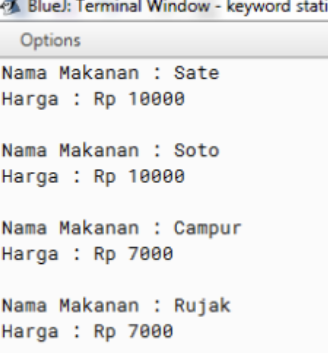
1. **Class MakananKhasMadura**

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

****

**Running Program**

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



**Penjelasan Program**

1. **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.

1. **Class Makanan Khas Madura**

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 IV IMPLEMENTASI**

## **Source Code**

Buatlah aplikasi Data Mahasiswa sebagai berikut:

No 1

1. Masukkan Nama Universitas! Buat static variable dan static method setter getter-

nya!

1. Masukkan data mahasiswa

1. NIM

2. NAMA

3. ALAMAT

1. JURUSAN :

1. 41 = TEKNIK INFORMATIKA

2. 42 = TEKNIK INDUSTRI,

3. 43 = TEKNIK ELEKTRO

4. 44 = SISTEM INFORMASI

5. 48 = TEKNIK MESIN

6. 49 = TEKNIK MEKATRONIKA

1. Apakah Anda ingin memasukkan data lagi? (Y) Ya; (T)Tidak

Jika user memasukkan “Y”, maka lanjut untuk memasukkan data mahasiswa lagi danjika user memasukkan data “T”,maka aplikasi berhenti. Desainlah aplikasi Data Mahasiswa tersebut dengan konsep constructor dan inheritance(tentukan parent class dan child class). Setelah itu, implementasikan class class yang telah didesain dengan membuat program sederhana yang memiliki fasilitas entri data Mahasiswa dan melihat daftar Mahasiswa yang telah dimasukkan.

Jawaban

1.

public class Kampus {

static String Universitas ;

static void setUniversitas (String Kampus){

Universitas=Kampus;

}

static String getUniversitas(){

return Universitas ;

}

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

// TODO code application logic here

}

}

2.

public class Mahasiswa extends Kampus{

String nim, nama, alamat, jurusan;

Mahasiswa(String nama, String nim, String alamat, String jurusan) {

this.nim = nim;

this.nama = nama;

this.alamat = alamat;

this.jurusan = jurusan;

}

}

3.

package com.mycompany.kampus;

import java.util.Scanner;

/\*\*

\*

\* @author Asus

\*/

public class proses {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

String Ulang;

System.out.println("Masukkan Nama Universitas:");

String namaUniversitas = scan.nextLine();

Kampus.setUniversitas(namaUniversitas);

do {

System.out.println("Silahkan Input");

System.out.println("NIM :");

String nim = scan.nextLine();

System.out.println("Nama :");

String nama = scan.nextLine();

System.out.println("Alamat :");

String alamat = scan.nextLine();

System.out.println("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.println("Pilih Kode Jurusan :");

String kodeJurusan = scan.nextLine();

String jurusan ="";

jurusan = switch (kodeJurusan) {

case "41" -> "TEKNIK INFORMATIKA";

case "42" -> "TEKNIK INDUSTRI";

case "43" -> "TEKNIK ELEKTRO";

case "44" -> "SISTEM INFORMASI";

case "48" -> "TEKNIK MESIN";

case "49" -> "TEKNIK MEKATRONIKA";

default -> "Tidak Tersedia";

};

Mahasiswa pertama = new Mahasiswa(nama, nim, alamat, jurusan);

System.out.println("Nama :" + pertama.nama);

System.out.println("NIM :" + pertama.nim);

System.out.println("Alamat :" + pertama.alamat);

System.out.println("Jurusan :" + pertama.jurusan);

System.out.print("APAKAH ANDA INGIN MEMASUKKAN DATA LAGI ? Y/T:");

Ulang = scan.nextLine();

} while (Ulang.equalsIgnoreCase("Y"));

}

}

3.

package com.mycompany.kampus;

import java.util.Scanner;

/\*\*

\*

\* @author Asus

\*/

public class proses {

public static void main(String[] args) {

Scanner scan = new Scanner(System.in);

String Ulang;

System.out.println("Masukkan Nama Universitas:");

String namaUniversitas = scan.nextLine();

Kampus.setUniversitas(namaUniversitas);/\*set univ dari krlas kampus, untuk mengatur nama\*/

do {

System.out.println("Silahkan Input");

System.out.println("NIM :");

String nim = scan.nextLine();

System.out.println("Nama :");

String nama = scan.nextLine();

System.out.println("Alamat :");

String alamat = scan.nextLine();

System.out.println("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.println("Pilih Kode Jurusan :");

String kodeJurusan = scan.nextLine();

String jurusan ="";

jurusan = switch (kodeJurusan) {/\*menentukan jurusan berdasarkan kode\*/

case "41" -> "TEKNIK INFORMATIKA";

case "42" -> "TEKNIK INDUSTRI";

case "43" -> "TEKNIK ELEKTRO";

case "44" -> "SISTEM INFORMASI";

case "48" -> "TEKNIK MESIN";

case "49" -> "TEKNIK MEKATRONIKA";

default -> "Tidak Tersedia";

};

Mahasiswa pertama = new Mahasiswa(nama, nim, alamat, jurusan);

System.out.println("Nama :" + pertama.nama);

System.out.println("NIM :" + pertama.nim);

System.out.println("Alamat :" + pertama.alamat);

System.out.println("Jurusan :" + pertama.jurusan);

System.out.print("APAKAH ANDA INGIN MEMASUKKAN DATA LAGI ? Y/T:");

Ulang = scan.nextLine();

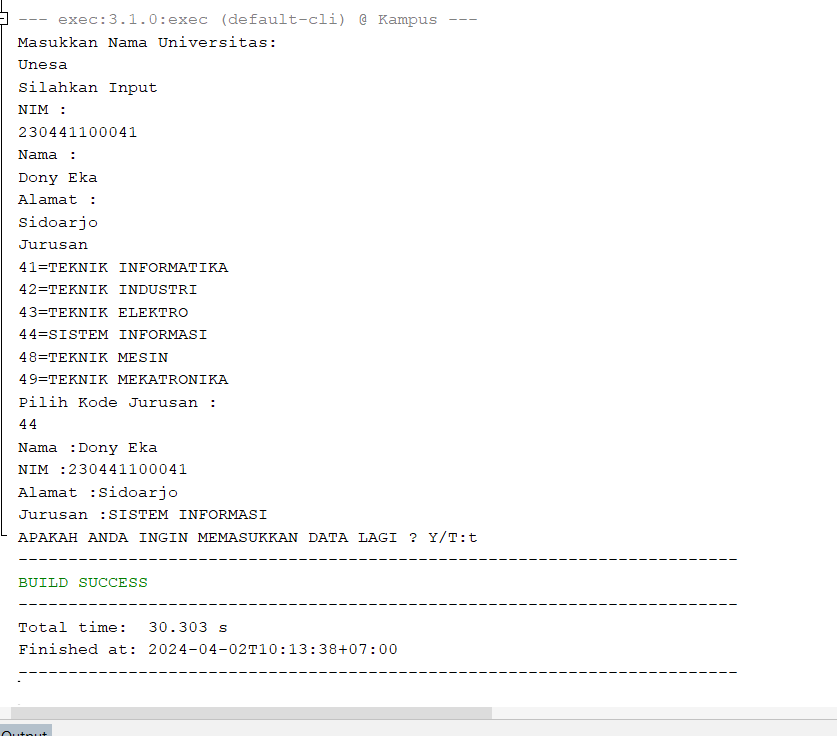
} while (Ulang.equalsIgnoreCase("Y"));

}

}

## **Hasil**

1.



## **Penjelasan**

1.

Package `com.mycompany.kampus` merupakan sebuah package yang berisi kelas `Kampus`. Kode ini menjelaskan tentang sebuah kelas `Kampus` yang memiliki dua metode: `setUniversitas` dan `getUniversitas`. Metode `setUniversitas` digunakan untuk mengatur atau mengatur nilai dari variabel `Universitas` dengan mengirimkan parameter `Kampus` sebagai argumen. Metode `getUniversitas` digunakan untuk mengambil nilai dari variabel `Universitas`. Kode ini juga menjelaskan tentang sebuah metode `main` yang tidak dilakukan apapun.

2.

Kode di atas merupakan sebuah class Java yang diberi nama Mahasiswa yang merupakan subclass dari class Kampus. Kata kunci extends digunakan dalam kode tersebut untuk membuat class Mahasiswa menjadi turunan dari class Kampus. Dengan kata lain, class Mahasiswa mewarisi semua hal yang dimiliki oleh class Kampus, seperti atribut dan metode.

Dengan menggunakan extends, class Mahasiswa bisa menggunakan semua fitur yang telah didefinisikan di class Kampus tanpa perlu menulis ulang. Jadi, bisa dibilang bahwa class Mahasiswa "adalah" tipe dari class Kampus, atau dalam kata lain, seorang mahasiswa adalah jenis khusus dari entitas kampus.

Class Mahasiswa memiliki beberapa atribut yaitu nim, nama, alamat, dan jurusan yang merupakan data-data yang terkait dengan seorang mahasiswa.

Selain atribut, class Mahasiswa juga memiliki sebuah constructor (konstruktor) yang digunakan untuk membuat objek Mahasiswa dengan 4 parameter yaitu nama, nim, alamat, dan jurusan. Di dalam constructor tersebut, nilai-nilai dari parameter-parameter tersebut akan diset ke dalam atribut-atribut yang sesuai di objek Mahasiswa menggunakan kata kunci this.

3.

Pertama-tama saya membuat class proses kemudian saya membuat inputan nama universitas dan menyimpan di dalam kelas kampus, lalu saya membuat perulangan do while dan membuat inputan nim,nama,alamat,dan jurusan, kemudiaan saya membuat objek mahasiswa baru dengan menggunakan konstruktor mahasiswa, lalu saya membuat tampilan nim,nama,alamat,jurusan, dan memanggil fariabel nya

# **BAB V PENUTUP**

## **Analisa**

Pada praktikum Pemrograman Berorientasi Objek (PBO) yang membahas tentang Constructor dan Keyword Static, mahasiswa akan mendapatkan pemahaman yang mendalam mengenai dua konsep penting dalam pemrograman berbasis objek. Constructor, sebagai metode khusus dalam sebuah kelas, digunakan untuk menginisialisasi objek dari kelas tersebut dengan nilai awal yang sesuai. Dengan constructor, mahasiswa dapat belajar bagaimana menginisialisasi objek secara konsisten dan aman, serta memastikan objek dibuat dalam keadaan yang valid. Sementara itu, penggunaan keyword static dalam praktikum akan memberikan pemahaman tentang penggunaan variabel atau metode yang dimiliki oleh kelas itu sendiri, bukan oleh objek individu. Variabel statis memungkinkan akses tanpa membuat objek kelas, serta dapat digunakan untuk menyimpan informasi global di dalam kelas. Dengan penerapan konsep Constructor dan Keyword Static dalam praktikum PBO, mahasiswa dapat memperluas pemahaman mereka tentang konsep pemrograman berorientasi objek dan bagaimana konsep-konsep ini dapat diterapkan dalam pengembangan perangkat lunak yang lebih kompleks.

## **Kesimpulan**

Dalam praktikum PBO tentang Constructor dan Keyword Static, mahasiswa dapat memahami penggunaan Constructor untuk inisialisasi objek dan Keyword Static untuk variabel/method kelas. Dengan pemahaman ini, mahasiswa siap menghadapi pengembangan perangkat lunak kompleks secara efisien. Praktikum ini memberikan landasan yang kokoh dalam pemahaman prinsip dasar PBO.