

**PRAKTIKUM**  
**PEMROGRAMAN BERORIENTASI OBJEK**



Nama : Muhammad Fahmi

Nim : 13020190019

Kelas : A1

**PROGRAM STUDI TEKNIK INFORMATIKA**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS MUSLIM INDONESIA**  
**MAKASSAR**  
**2021**

## **KATA PENGANTAR**

*Bismillahir rohmani rohiim.*

*Assalamu Alaikum Warahmatullahi Wabarakatuh.*

Segala puji bagi Allah yang telah memberikan kemudahan sehingga dapat menyelesaikan laporan praktikum ini. Tanpa pertolongan-Nya mungkin saya tidak akan sanggup menyelesaikan laporan praktikum ini dengan baik. Shalawat dan salam semoga terlimpah curahkan kepada baginda tercinta kita yakni Nabi Muhammad SAW.

Dalam penyusunan laporan praktikum ini, saya mengucapkan banyak terima kasih kepada Ibu Mardiyah Hasnawi, S.Kom., M.T. selaku dosen mata kuliah pemrograman berorientasi objek dan juga kepada Rifqatul Mukarramah selaku asisten laboratorium yang telah memberi bimbingan dan kepercayaan. Sehingga, laporan ini dapat saya susun dengan baik.

Demikianlah yang dapat saya tuliskan disini apabila terdapat kekurangan ataupun kesalahan dalam makalah ini saya berharap maklumi karena saya masih dalam pembelajaran.

*Wassalamu Alaikum Wr. Wb.*

**Makassar, 21 April 2021**

**Muhammad Fahmi Frek. PBO - 1**

## DAFTAR ISI

<b>Kata Pengantar .....</b>	<b>i</b>
<b>Daftar Isi .....</b>	<b>ii</b>
<b>BAB I. PENDAHULUAN.....</b>	<b>1</b>
<b>1.1 Latar Belakang .....</b>	<b>1</b>
<b>1.2 Rumusan Masalah .....</b>	<b>2</b>
<b>1.3 Batasan Masalah.....</b>	<b>2</b>
<b>1.4 Tujuan.....</b>	<b>3</b>
<b>1.5 Manfaat.....</b>	<b>3</b>
<b>BAB II. LANDASAN TEORI .....</b>	<b>4</b>
<b>2.1 Pemrograman Java.....</b>	<b>4</b>
<b>2.2 Tools Pemrograman Java .....</b>	<b>4</b>
<b>2.3 Konsep Pemrograman Java.....</b>	<b>4</b>
<b>2.4 Class, Objek dan Fungsi Main Pada Pemrograman Java .....</b>	<b>5</b>
<b>2.5 Tipe Data .....</b>	<b>9</b>
<b>2.6 Atribut .....</b>	<b>9</b>
<b>2.7 Konversi Tipe Data.....</b>	<b>10</b>
<b>2.8 Method.....</b>	<b>11</b>
<b>2.9 Package &amp; Import.....</b>	<b>13</b>
<b>2.10 Enkapsulasi .....</b>	<b>14</b>

2.11 Penginputan .....	15
2.12 Exception .....	16
2.13 Menangani Exception .....	16
2.14 Melempar Exception .....	17
2.15 Inheritance.....	17
2.16 Overriding .....	18
2.17 Polimorfisme .....	18
2.18 Abstract Class .....	19
2.19 Interface.....	20
2.20 ArrayList .....	20
2.21 HashMap .....	21
<b>BAB III. IMPLEMENTASI &amp; PEMBAHASAN .....</b>	<b>22</b>
3.1 Implementasi.....	22
3.2 Pembahasan .....	35
<b>BAB IV. PENUTUP .....</b>	<b>54</b>
4.1 Kesimpulan .....	54
4.2 Saran.....	55
<b>Daftar Pustaka.....</b>	<b>56</b>

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Pesatnya kemajuan teknologi informasi saat ini menuntut untuk selalu belajar mengenai komputer. Ditambah lagi saat ini, telah memasuki zaman yang dinamakan revolusi industri 4.0. komputerisasi dan automasi sebenarnya telah ada saat revolusi industri 3.0, namun hal tersebut belum begitu diterapkan oleh masyarakat Indonesia pada umumnya. Padahal pada saat itu, segala bentuk administrasi di negara ini telah berevolusi menjadi komputerisasi. Untuk itu, pemerintah Indonesia di era revolusi industri 4.0 ini, menegaskan kembali kepada kaum terpelajar untuk meningkatkan diri mengikuti perkembangan teknologi informasi dibidang komputerisasi.

Bagi sebagian orang yang bisa mengoperasikan komputer pada saat ini, bahasa pemrograman komputer masih merupakan bahasa yang sulit untuk dimengerti sehingga sebagian orang tersebut enggan atau malas untuk mempelajari lebih jauh mengenai bahasa pemrograman komputer tersebut. Perlu diketahui bahwa program yang ditulis dengan menggunakan bahasa Java mempunyai ciri yang terstruktur sehingga mudah dipahami maupun dikembangkan oleh pemrogram.

Hal yang lainnya adalah berupa lengkapnya fasilitas yang disediakan, sehingga bahasa *java* dapat dipakai untuk memecahkan masalah dari masalah yang banyak memerlukan perhitungan sampai masalah implementasi permainan. Oleh karena itu melihat seberapa pentingnya peranan bahasa *java*

dalam dunia pemrograman maupun dalam pendidikan, maka ditulis laporan praktikum pemrograman berorientasi objek ini. Pada laporan praktikum ini akan dibahas pada modul 1 sampai 8 tentang pemrograman java, Atribut & Method, Package, Import, dan Enkapsulasi, Penginputan & Exception, Inheritance, Polimorfisme, Abstract Class dan Interface, dan yang terakhir adalah ArrayList & HashMap. Tidak hanya itu salah satu latar belakang penulisan makalah ini adalah sebagai tugas praktikum yang diberikan oleh asisten laboratorium pada mata kuliah pemrograman berorientasi objek.

## **1.2 Rumusan Masalah**

Adapun rumusan masalah pada modul 1 sampai modul 8, yaitu:

1. Bagaimana konsep dasar, atribut dan method pada pemrograman java?
2. Bagaimana konsep package, import dan enkapsulasi pada pemrograman java?
3. Bagaimana konsep scanner, bufferedreader, joptionpane, try-catch, throw, dan throws pada pemrograman java?
4. Bagaimana konsep inheritance, polimorfisme, abstract class, interface, arraylist dan hashmap pada pemrograman java?

## **1.3 Batasan Masalah**

Adapun batasan masalah pada modul 1 sampai modul 8 ini, yaitu:

1. Praktikum dilakukan dalam 4 pertemuan pada tanggal, 31 Maret dan 1 April 2021 di Laboratorium Jaringan dan 7, 8 April 2021 di Laboratorium Multimedia, Fakultas Ilmu Komputer, Universitas Muslim Indonesia.

2. Praktikum dilakukan dengan menggunakan software IDE NetBeans pada OS Windows 10.
3. Praktikum modul 1 sampai modul 8 ini membahas tentang konsep pemrograman java mulai dari mengenal bentuk deklarasi class hingga dengan mengimplementasikan arraylist dan hashmap dalam pemrograman Java.

#### **1.4 Tujuan**

Adapun tujuan praktikum pada modul 1 sampai modul 8 ini, yaitu:

1. Praktikan mampu menjelaskan dan mengimplementasikan konsep, atribut dan method pada pemrograman java.
2. Praktikan mampu menjelaskan dan mengimplementasikan konsep package, import dan enkapsulasi pada pemrograman java.
3. Praktikan mampu menjelaskan dan mengimplementasikan konsep scanner, bufferedreader, joptionpane, try-catch, throw, dan throws pada pemrograman java.
4. Praktikan mampu menjelaskan dan mengimplementasikan konsep inheritance, polimorfisme, abstract class, interface, arraylist dan hashmap pada pemrograman java.

#### **1.5 Manfaat**

1. Menambah wawasan mengenai pemrograman bahasa java.
2. Memberikan pengetahuan mengenai struktur dan sintaks bahasa java.
3. Memberikan pengetahuan mengenai bagaimana mengaplikasikan bahasa java dalam pembuatan berbagai program.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Pemrograman Java**

Java merupakan salah satu bahasa pemrograman yang sudah ada dari era 1990-an. Bahasa Pemrograman Java adalah bahasa pemrograman yang mengadopsi sintaks dari bahasa C dan C++. Salah satu sifat pemrograman Java yang menjadi keuntungan dalam menggunakan bahasa pemrograman ini adalah bersifat *multiplatform* yaitu Java dapat dijalankan pada banyak platform atau sistem informasi.

#### **2.2 Tools Pemrograman Java**

Untuk membuat program atau aplikasi menggunakan Java ada beberapa tools yang dibutuhkan. Hal yang sangat penting dan wajib adalah compiler bahasa pemrograman Java yaitu JDK (Java Development Kit). Kemudian Text Editor untuk menuliskan source code pemrograman Java. Namun, sekarang sudah ada IDE (Integrated Development Environment – Lingkungan Pengembangan Terintegrasi) untuk bahasa pemrograman Java seperti Netbeans, Eclipse, IntelliJIDEA.

#### **2.3 Konsep Pemrograman Java**

Konsep yang digunakan dalam pemrograman Java adalah OOP (Object Oriented Programming) atau Pemrograman Berorientasi Objek. OOP adalah sebuah metodologi atau cara berpikir dalam melakukan pemrograman dimana pendefinisian tipe data disertai dengan pendefinisian fungsi. Struktur data yang seperti ini disebut dengan istilah object. Paradigma pemrograman OOP dapat



dilihat sebagai interaksi sebuah object dalam melakukan tugasnya. OOP ini terdiri dari 3 konsep yaitu :

- a) Enkapsulasi, yaitu pemberian hak akses oleh akses modifier (public, private, protected, default)
- b) Inheritance yaitu konsep pewarisan. Konsep ini akan dibahas di modul selanjutnya.
- c) Polimorphisme yang artinya banyak bentuk. Konsep ini akan dibahas di modul selanjutnya.

Untuk dapat mengetahui dan memahami konsep OOP tersebut, maka hal yang mesti dipahami adalah Class, Objek, dan Fungsi Main pada pemrograman Java.

## 2.4 Class, Objek dan Fungsi Main Pada Pemrograman Java

### 2.4.1 Class atau Kelas

Class atau kelas adalah kelompok objek-objek yang memiliki karakteristik yang sama (yang sejenis). Dalam ruang lingkup pemrograman, class ini sering dianalogikan sebagai sebuah cetakan dalam kehidupan nyata, dimana dari sebuah cetakan kita bisa membuat banyak kue dan kue ini bisa kita sebut sebagai object.

#### Deklarasi Class Pada Java

```
Akses_Modifier  class  NamaClass {
...
}
```

**Contoh :**

```
public class Rumah {
    ...
}
```

Pada pemrograman java, Setiap Class akan disimpan dalam 1 file yang berekstensi “ .java ” dengan nama file harus sesuai dengan nama class yang sudah dibentuk. Untuk contoh diatas, untuk penyimpanannya adalah “Rumah.java”.

#### 2.4.2 Konstruktor

Konstruktor adalah sebuah tipe khusus dari method yang digunakan untuk membuat dan menginisialisasi sebuah object baru. Berikut adalah property dari konstruktor :

- a) Konstruktor memiliki nama yang sama dengan nama class.
- b) Konstruktor tidak memiliki tipe data dan tidak mengembalikan nilai.
- c) Konstruktor tidak dapat dipanggil secara langsung, namun harus dipanggil dengan menggunakan operator *new* pada pembentukan sebuah *class*.

**Deklarasi Konstruktor Pada Java**

```
AksesModifier    NamaKelas (parameter) {
    statement
}
```

<b>Contoh :</b>
<pre>public class <b>Rumah</b> {     ... }</pre> <p>Untuk Class Diatas, Untuk Pembuatan Konstruktornya adalah : public</p> <pre><b>Rumah</b>() {     ... }</pre>

### 2.4.3 Objek Class

Setiap class pasti memiliki objek, satu kelas ini dapat terbentuk banyak objek. Objek ini digunakan umumnya untuk berelasi dengan objek lainnya.

<b>Deklarasi Objek Pada Java</b>
<b>NamaClass NamaObjek = new NamaClass();</b>
<b>Contoh :</b>
<pre>public class <b>Rumah</b> {     ... }</pre> <p>Untuk Class Diatas, Untuk Pembuatan Objeknya adalah :</p> <pre><b>Rumahr</b> = new <b>Rumah</b>();</pre>

Pada pemrograman Java, tidak menutup kemungkinan dalam satu program dibutuhkan lebih dari 1 Class yang mana artinya dalam 1 program terdapat banyak Class dan 1 fungsi utama. Untuk memberi dan mengambil nilai dari suatu atribut atau method dari Class lain maka dibutuhkan objek dari Class tempat pendeklarasian atribut atau method tersebut.

**Contoh :**

```
public class Alamat {
    String alamat=" ";
}
```

```
public class Rumah {
    public static void main( String [] args){
        //Untuk menggunakan atribut/variabel alamat pada Class
        Alamat maka dibutuhkan objek dari Class Alamat.
        Alamat al = new Alamat();
        al.alamat = "Batua Raya X ";
    }
}
```

## 2.4.4 Fungsi Utama atau Fungsi Main.

Fungsi utama adalah fungsi yang pertama kali dijalankan. Didalam fungsi utama diperlukan objek untuk menggunakan variable atau procedure dari luar fungsi utama.

**Deklarasi Objek Pada Java**

```
AksesModifier static void main (String [ ] args) {
    ...
}
```

**Contoh :**

```
public class Rumah {

    public static void main ( String [ ] args ) {

        System.out.println ("Hello Word");

    }

}
```

**2.5 Tipe Data**

Tipe data adalah jenis data yang dapat ditampung dalam suatu variable. Pada pemrograman java tipe data terbagi dua yaitu tipe data primitive dan tipe data referensi. Tipe data primitive adalah tipe data yang memiliki nilai dan ukuran tertentu dan bukan referensi dari objek atau class seperti int, char, float, double, boolean, dll. Sedangkan tipe data referensi atau tipe data class adalah tipe data yang merupakan referensi dari objek atau class seperti Integer, String, Boolean, dll.

**2.6 Atribut**

Atribut atau variabel merupakan satuan dasar penyimpanan dalam program Java. Atribut/variabel ini ada 3 macam yaitu sebagai berikut.

- a) Variabel Global adalah variabel yang dideklarasikan diluar fungsi/prosedur sehingga dapat digunakan oleh semua fungsi dan prosedur yang ada dalam suatu class.
- b) Variabel Lokal adalah variabel yang dideklarasikan didalam suatu fungsi/prosedur sehingga hanya dapat digunakan oleh fungsi/prosedur tempat pendeklarasiannya.

- c) Variabel Static yaitu variabel yang pada saat dideklarasikan menggunakan keyword 'static', variabel static ini artinya variabel yang bersifat instan maksudnya dalam pemanggilannya tidak dibutuhkan referensi dari suatu objek classnya.

Deklarasi Atribut / Variabel Pada Java	
TipeData	NamaVariabel ;
Contoh :	
int nilai; → Contoh Variabel Menggunakan Tipe Data Primitive	
String nama1, → Contoh Variabel Menggunakan Tipe Data Referensi	

## 2.7 Konversi Tipe Data

Konversi tipe data adalah merubah tipe data suatu variabel. Berikut beberapa cara konversi pada pemrograman java.

- a) Mengubah data string ke numerik integer

```
int    varInteger    =    Integer.parseInt ( strVarInteger );
```

- b) Mengubah data String ke numerik double

```
double varDouble    =    Double.parseDouble ( strVarDouble );
```

- c) Mengubah data dari string ke short

```
short  varShort     =    Short.parseShort ( strVarShort );
```

- d) Mengubah data dari string ke long

```
long   varlong      =    Long.parseLong ( strVarLong );
```

e) Mengubah data dari string ke long

```
float    varFloat    =    Float.parseFloat ( strVarFloat );
```

Terdapat banyak jenis konversi bilangan mulai dari konversi tipe data nonnumerik (text) ke tipe data numerik ataupun sebaliknya.

## 2.8 Method

Method adalah kumpulan kode perintah untuk melakukan sebuah proses yang diberi nama yang disebut sebagai nama method. Nama method sebagai rujukan untuk sekumpulan kode tersebut. Method pada pemrograman java adalah istilah untuk prosedur dan fungsi (*function*) pada pemrograman yang lain. Method pada pemrograman java ada banyak jenisnya yaitu sebagai berikut.

a) Konstruktor

Konstruktor adalah sebuah fungsi yang dijalankan pertama kali pada sebuah class. Konstruktor ini berperan dalam inisialisasi objek pada suatu class. Konstruktor digunakan untuk memberi nilai awal pada suatu class. Konstruktor tidak dapat mengembalikan nilai balik. Deklarasi konstruktor harus memiliki nama sesuai dengan nama classnya.

### Deklarasi Konstruktor Pada Java

```
AksesModifier  NamaClass () {  
  
    //statement  
  
}
```

**Contoh :**

```
public class Modul4 { public Modul4(){
    //statement
}
}
```

## b) Procedure dan Function

Procedure adalah method yang bertipe data **void** sehingga tidak mengembalikan nilai.

**Deklarasi Method Tidak Mengembalikan Nilai**

```
AksesModifier void NamaMethod (){
    //statement
}
```

**Contoh :**

```
public void Biodata (){
    //statement
}
```



Function adalah method yang bertipe data selain dari **void** seperti int, String, float, dll. Method ini mengembalikan nilai balik sesuai dengan tipe data method.

<b>Deklarasi Method Tidak Mengembalikan Nilai</b>
<pre> <b>AksesModifier TipeData NamaMethod ()</b>{      <b>return Nilai;</b>    <i>//nilai yang dikembalikan harus bertipe data sesuai tipe data method</i>  } </pre>
<b>Contoh :</b>
<pre> public int nilai(){      return 100;  } </pre>

#### c) Method Overloading

Overloading adalah kondisi dimana terdapat dua atau lebih method yang memiliki nama method yang sama tetapi parameter yang berbeda. Konsep overloading ini juga berlaku pada konstrukto

#### d) Method Overriding

Overriding adalah kondisi dimana terdapat dua atau lebih method dengan nama yang sama yang berada dalam konsep pewarisan / inheritance.

## 2.9 Package & Import

Package adalah : suatu cara untuk memanage atau mengelompokkan class-class berdasarkan kesamaan atau kemiripan fungsi. Dalam versi Java ada 6 package besar yaitu :

- a) java.io
- b) java.lang
- c) java.applet
- d) java.awt
- e) java.net
- f) java.util

Secara fisik, package dapat berupa folder yang berisi file/class, interface/enum lain yang kegunaannya hampir sama sehingga patut untuk dikelompokkan. Package mempengaruhi hak akses ke class lainnya.

#### 2.9.1 Menggunakan Package

Ada 2 cara menggunakan suatu package yaitu:

- a) Apabila Class yang berhubungan dengan Class lain berada pada package sama maka tidak diperlukan import package.
- b) Apabila Class yang berhubungan dengan Class lain berada pada package yang berbeda maka diperlukan import package.

#### 2.9.2 Mengimport Package

```
Import namaPackage>NamaClass;    atau    Import namaPackage.*;
```

### 2.10 Enkapsulasi

Enkapsulasi artinya pembungkusan, pembungkusan yang dimaksud adalah membungkus suatu class, atribut, atau method menggunakan access modifier untuk memberikan hak akses. Ada 4 macam modifier yaitu :

a) Default

Modifier default merupakan akses untuk class yang hanya bisa diakses oleh class dalam package yang sama.

b) Public

Modifier public yaitu hak akses yang dapat di akses dari manapun, baik itu dari class yang berbeda maupun dari package yang berbeda.

c) Protected

Hak akses untuk tipe protected hampir sama dengan public akan tetapi tidak bisa diakses dari paket yang berbeda kecuali merupakan turunan.

d) Private

Modifier private yaitu hak akses yang hanya bisa di akses oleh kelas itu sendiri. Untuk mengambil atau memberikan nilai pada atribut tersebut digunakan method setter dan getter.

## **2.11 Penginputan**

a) Scanner

Scanner pada java merupakan sebuah class yang menyediakan fungsi-fungsi untuk membaca inputan pengguna. Class Scanner ini didapatkan dari package java.util. Scanner dapat membaca inputan pengguna dalam beberapa tipe data.

b) JOptionPane

JOptionPane merupakan sebuah class yang menyediakan fungsi-fungsi dalam menerima inputan dan menampilkan output pengguna dengan

memunculkan dialog box. Class ini berada pada package `javax.swing.JOptionPane`.

#### c) `BufferedReader`

`BufferedReader` adalah salah satu class library di dalam pemrograman java (*`java.io.BufferedReader`*) sejak JDK Versi 1, yang fungsinya untuk membaca text dengan menggunakan Buffering dengan kemampuan membaca dari skala besar dalam satu waktu. `BufferedReader` akan membungkus dan membaca karakter input stream menggunakan objek dari class `InputStreamReader` (*`java.io.InputStreamReader`*) sebagai argument.

### 2.12 Exception

Exception adalah sebuah peristiwa yang menjalankan alur proses normal pada program. Peristiwa ini biasanya berupa kesalahan(error) dari beberapa bentuk. Beberapa contoh dari exception yaitu *`ArrayIndexOutOfBoundsException`*, yang terjadi jika kita mencoba mengakses elemen array yang tidak ada, atau *`NumberFormatException`*, yang terjadi ketika kita mencoba melalui parameter bukan angka dalam method `Integer.parseInt`.

### 2.13 Menangani Exception

Untuk menangani exception dalam Java, digunakan blok Try Catch. Try Catch ini adalah salah satu fitur dari Java untuk menangkap eksepsi atau kesalahan.

#### Bentuk Umum Try Catch

```
try
{<blok_programs>; } catch(<parameter>)
{<blok_yang_dijalankan_jika_terjadi_eksepsi>;} finally
{<blok_yang_dijalankan_terakhir>;}
```

## 2.14 Melempar Exception

Disamping menangkap exception, Java juga mengizinkan seorang user untuk melempar sebuah exception. Syntax pelemparan exception cukup sederhana.

```
throw <exception object>;
```

Jika sebuah method dapat menyebabkan sebuah exception namun tidak menangkapnya, maka digunakan keyword *throws*. Aturan ini hanya berlaku pada checked exception.

Berikut ini penulisan syntax menggunakan keyword *throws* :

```
<type> <methodName> (<parameterList>) throws  
<exceptionList>
```

Sebuah method perlu untuk menangkap ataupun mendaftarkan seluruh exceptions yang mungkin terjadi, namun hal itu dapat menghilangkan tipe Error, RuntimeException, ataupun subclass-nya.

## 2.15 Inheritance

Inheritance merupakan pewarisan atribut dan method pada sebuah class yang diperoleh dari class yang telah terdefiniskan. Class yang diwariskan disebut dengan subclass (class anak), sedangkan class yang mewariskan disebut dengan superclass (class induk). Salah satu keuntungan dari inheritance adalah Subclass dapat merubah atau memodifikasi apa yang telah diwariskan oleh Superclass. Untuk membuat class anak diperlukan deklarasi class menggunakan *extends*.

```
class <nama-class> extends <nama-  
superclass>{  
    //isi program
```

Super ini digunakan untuk memanggil konstruktor dari superclass atau menjadi variabel yang mengacu pada superclass.

## **2.16 Overriding**

Method subclass override terhadap method superclass ketika subclass mendeklarasikan method yang signaturenya serupa ke method dalam superclass. Signature dari method hanyalah informasi yang ditemukan dalam definisi method bagian atas. Signature mengikuti tipe return, nama dan daftar parameter method tetapi itu tidak termasuk acces modifier dan tipe yang lain dari kata kunci seperti final dan static. Inilah perbedaan dari method overloading. Method overloading secara singkat didiskusikan dalam sub bagian pada kata kunci this. Method ini hanya terjadi jika merupakan implementasi dari pewarisan.

## **2.17 Polimorfisme**

Polimorfisme berarti banyak bentuk. Ada beberapa definisi berbeda tentang polimorfisme yang berkaitan dengan pemrograman berorientasi obyek.

### **a) Method Overloading**

Overloading adalah mendefinisikan dua atau lebih method di dalam kelas yang sama, dengan nama yang sama, namun dengan deklarasi parameter yang berbeda.

### **b) Konstruktor Overloading**

Konstruktor overloading merupakan situasi dimana terdapat lebih dari satu konstruktor dengan tipe data parameter yang berbeda.

### c) Method Overriding

Method subclass override terhadap method superclass ketika subclass mendeklarasikan method yang signaturenya serupa ke method dalam superclass. Method ini hanya terjadi jika merupakan implementasi dari pewarisan.

## 2.18 Abstract Class

Abstract class adalah class yang terletak pada posisi tertinggi pada hirarki class. Class ini digunakan sebagai basis bagi penurunan kelas lainnya, sehingga abstract class tidak dapat dinstansiasi secara langsung menjadi object. Suatu abstract class dapat mengandung method konkrit dan/atau method abstract. Abstract method adalah suatu method yang tidak mempunyai badan method, hanya berupa nama method dan parameter inputan method. Ketika suatu class abstract diturunkan, maka subclass harus mengimplementasikan semua method abstract pada superclass. Jika tidak mengimplementasikan semua method abstract maka subclass harus dideklarasikan sebagai abstract class.

Deklarasi Abstract Class
<pre> <b>AksesModifier</b> <b>abstract class</b>   <b>NamaClass</b> {      <b>//statement</b>  } </pre>
Contoh :
<pre> <b>public abstract class</b> Hitung{      <b>//statement</b>  } </pre>

<b>Deklarasi Abstract Method</b>
<b>AksesModifier abstract TipeData NamaMethod(parameter)</b>
<b>Contoh :</b>
public abstract void HasilHitung ();

## 2.19 Interface

Interface adalah kumpulan method yang hanya memuat deklarasi dan struktur method tanpa detail implementasinya. Sedangkan detail dari method berada pada class yang mengimplementasikan interface tersebut. Interface digunakan bila anda ingin mengaplikasikan suatu method yang spesifik, yang tidak diperoleh dari proses inheritance. Method yang dapat dibuat pada interface adalah method abstract.

## 2.20 ArrayList

ArrayList merupakan collection yang menjadi bagian dari Java.Util.ArrayList. Berbeda dengan array biasa, ArrayList memiliki sejumlah operasi yang lebih lengkap dan mudah digunakan dibandingkan dengan array biasa. ArrayList dibentuk untuk menutupi kekurangan yang ada pada Array, contohnya dalam Array sudah diberikan batasan tampungan data sedangkan ArrayList tidak memiliki batasan tampungan dimana dapat menambah data baru secara dinamis tanpa harus menentukan ukurannya di awal. Berikut ini merupakan bentuk dasar listing dari ArrayList :

```
ArrayList< Object > nama ArrayList = new ArrayList<>();
```

Contoh:

```
ArrayList< Integer > data = new ArrayList<>();
```



## 2.21 HashMap

Class HashMap merupakan class turunan dari class AbstractMap dan implementasi dari interface Map. HashMap adalah sebuah class yang berisi sekumpulan pasangan nilai (value) dan kunci (key). Pada penjelasan HashMap, dapat disimpulkan bahwa HashMap memiliki 2 atribut yaitu value dan key. value bisa dalam bentuk string, integer, boolean, float, double, dan objek. Sedangkan untuk key biasanya dalam bentuk string dan integer. Operasi yang dapat dilakukan terhadap HashMap antara lain :

- a) Put(), untuk memberikan nilai
- b) Get(), untuk mengambil nilai
- c) Remove(), untuk menghapus salah satu nilai
- d) Clear(), untuk menghapus semua nilai
- e) Replace(), untuk mengubah nilai

Library yang digunakan untuk HashMap berada pada Java.Util.HashMap. Ketika ingin menggunakan HashMap, maka kita harus mengimportnya terlebih dahulu. Berikut ini merupakan bentuk dasar kodingan HashMap :

Contoh :

```
HashMap< TD _key , TD_value > Nama_HashMap = new HashMap< TD_key
```

```
HashMap< Integer , Strnig > Hari = new HashMap< Integer , String
```

## BAB III

### IMPLEMENTASI & PEMBAHASAN

#### 3.1 Implementasi

##### 3.1.1 Modul 1

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada modul 1 berupa *listing program*, serta *outputnya*(IntelliJ):

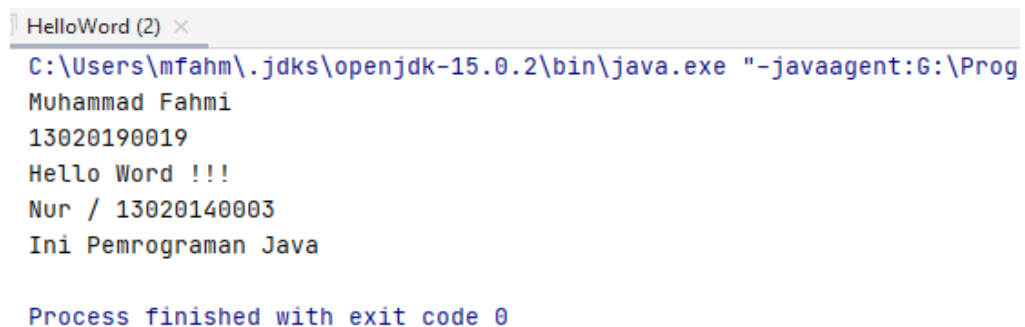
```
package Fahmi.Mod1;

public class HelloWorld { //spertinya modul typo
    Word(Wor"l"d)
    public HelloWorld() {
        System.out.println("Muhammad Fahmi");
        System.out.println("13020190019");
    }

    /*private class HelloWorld { //Akan error karna private
        public HelloWorld() {
            System.out.println("Muhammad Fahmi");
            System.out.println("13020190019");
        }
    }*/

    public static void main(String[] args) {
        HelloWorld hw = new HelloWorld();
        System.out.println("Hello Word !!!");
        System.out.println("Nur / 13020140003");
        System.out.println("Ini Pemrograman Java");
        // jika println diubah jd print, maka tdk ada line new
    }
}
```

**Listing Program Modul 1**



```
C:\Users\mfahm\jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\Prog
Muhammad Fahmi
13020190019
Hello Word !!!
Nur / 13020140003
Ini Pemrograman Java

Process finished with exit code 0
```

**Gambar 3.1 Output Program Modul 1**

### 3.1.2 Modul 2

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 2 berupa *listing program*, serta *outputnya*(IntelliJ):

```
package Fahmi.Mod2;

public class Hitung {
    String nama, stb;

    public Hitung() {
        nama = "Ismail";
        stb = "13020100101";
    }

    public float tambah(int nilai1, int nilai2) {
        return nilai1 + nilai2;
    }

    public float selisih(int nilai1, int nilai2) {
        return nilai2 - nilai1;
    }

    public float kali(int nilai1, int nilai2) {
        return nilai1 * nilai2;
    }

    public float bagi(int nilai1, int nilai2) {
        return nilai1 / nilai2;
    }

    public void tampilNama() {
        System.out.println("Nama : " + nama);
        System.out.println("Stb : " + stb);
    }

    public static void main(String[] args) {
        Hitung h = new Hitung();
        int nilai1, nilai2;
        nilai1 = 10;
        nilai2 = 5;
        String pilihan = "tambah";
        if (pilihan.equals("tambah")) {
            System.out.println(nilai1 + " + " + nilai2 + " = " +
h.tambah(nilai1, nilai2));
        } else if (pilihan.equals("selisih")) {
            System.out.println(nilai2 + " - " + nilai1 + " = " +
h.selisih(nilai2, nilai1));
        } else if (pilihan.equals("kali")) {
            System.out.println(nilai1 + " * " + nilai2 + " = " +
h.kali(nilai1, nilai2));
        } else if (pilihan.equals("bagi")) {
            System.out.println(nilai1 + " / " + nilai2 + " = " +
h.bagi(nilai1, nilai2));
        } else {
            System.out.println("Tidak ada pilihan !");
        }
        System.out.println("-----");
        h.tampilNama();
    }
}
```

**Listing Program Modul 2**

```

Hitung x
C:\Users\mfahm\.jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\Program
10 + 5 = 15.0
-----
Nama : Ismail
Stb  : 13020100101

Process finished with exit code 0

```

**Gambar 3.2 Output Program Modul 2**

### 3.1.3 Modul 3

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 3 berupa *listing program*, serta *outputnya*(IntelliJ):

```

package Mahasiswa;

public class Mahasiswa {
    private String stb, nama;

    public String getStb() {
        return stb;
    }

    public void setStb(String stb) {
        this.stb = stb;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }
}

```

**Listing Program Modul 3 Class Mahasiswa**

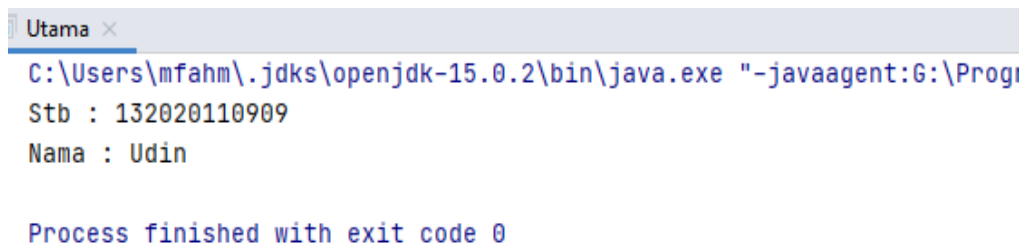
```

package Mahasiswa;

public class Utama {
    public static void main(String[] args) {
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNama("Udin");
        mhs.setStb("132020110909");
        System.out.println("Stb : " + mhs.getStb());
        System.out.println("Nama : " + mhs.getNama());
    }
}

```

**Listing Program Modul 3 Class Utama**



```

Utama x
C:\Users\mfahm\.jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\Progr
Stb : 132020110909
Nama : Udin

Process finished with exit code 0

```

**Gambar 3.3 Output Program Modul 3**

### 3.1.4 Modul 4

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada

Modul 4 berupa *listing program*, serta *outputnya*(IntelliJ):

```

package ContohScanner;

import java.util.Scanner;

public class LuasSegitiga {

    public double luas(int alas, int tinggi) {
        return (0.5 * alas * tinggi);
    }

    public static void main(String[] args) {
        LuasSegitiga ls = new LuasSegitiga();
        int alas, tinggi;
        Scanner sc = new Scanner(System.in);
        System.out.print("Masukkan Alas Segitiga : ");
        alas = sc.nextInt();
        System.out.print("Masukkan Tinggi Segitiga : ");
        tinggi = sc.nextInt();
        System.out.println("Luas Segitiga : " + ls.luas(alas,
tinggi));
    }
}

```

**Listing Program Modul 4 Import Scanner**

```

package ContohBufferedReader;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class LuasSegitiga {
    public double luas(int alas, int tinggi) {
        return (0.5 * alas * tinggi);
    }

    public static void main(String[] args) throws IOException {
        LuasSegitiga ls = new LuasSegitiga();
        BufferedReader bf = new BufferedReader(new
InputStreamReader(System.in));
        String a, t;
        int alas, tinggi;
        System.out.print("Masukkan Alas Segitiga : ");
        a = bf.readLine();
        System.out.print("Masukkan Tinggi Segitiga : ");
        t = bf.readLine();
        alas = Integer.parseInt(a);
        tinggi = Integer.parseInt(t);
        System.out.println("Luas Segitiga : " + ls.luas(alas, tinggi));
    }
}

```

**Listing Program Modul 4 Import Scanner**

```

package ContohBufferedReader;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class LuasSegitiga {
    public double luas(int alas, int tinggi) {
        return (0.5 * alas * tinggi);
    }

    public static void main(String[] args) throws IOException {
        LuasSegitiga ls = new LuasSegitiga();
        BufferedReader bf = new BufferedReader(new
InputStreamReader(System.in));
        String a, t;
        int alas, tinggi;
        System.out.print("Masukkan Alas Segitiga : ");
        a = bf.readLine();
        System.out.print("Masukkan Tinggi Segitiga : ");
        t = bf.readLine();
        alas = Integer.parseInt(a);
        tinggi = Integer.parseInt(t);
        System.out.println("Luas Segitiga : " + ls.luas(alas, tinggi));
    }
}

```

**Listing Program Modul 4 Import BufferedReader**

```

LuasSegitiga (2) x
C:\Users\mfahm\.jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\Prog
Masukkan Alas Segitiga : 69
Masukkan Tinggi Segitiga : 2
Luas Segitiga :69.0

Process finished with exit code 0

```

**Gambar 3.4 Output Program Modul 4**

### 3.1.5 Modul 5

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 5 berupa *listing program*, serta *outputnya*(IntelliJ):

```

public class BangunDatar {
    int jumlahSisi;
    double luas, keliling;
    String bangunDatar;

    void Luas() {
        System.out.println("Luas " + bangunDatar + " = " +
        luas);
    }

    void Keliling() {
        System.out.println("Keliling " + bangunDatar + " = "
        + keliling);
    }

    void Info() {
        System.out.println("Ini Adalah Bangun Datar. Ada
        Banyak Jenis-jenis Bangun Datar.");
    }
}

```

**Listing Program Modul 5 Class BangunDatar**

```

public class Segitiga extends BangunDatar {
    int alas, tinggi;

    void Info() {
        System.out.println("Ini Adalah Bangun Datar Berupa
        Segitiga");
    }
}

```

**Listing Program Modul 5 Class Segitiga**

```

public class Persegi extends BangunDatar {
    int sisi;

    void Info() {
        System.out.println("Ini Adalah Bangun Datar Berupa
Persegi");
    }
}

```

**Listing Program Modul 5 Class Persegi**

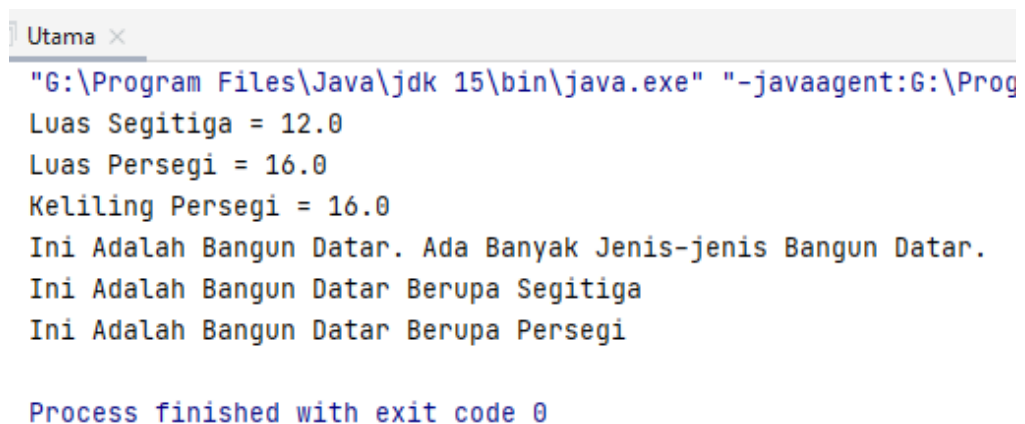
```

public class Utama {
    public static void main(String[] args) {
        Segitiga se = new Segitiga();
        se.bangundatar = "Segitiga";
        se.jumlahsisi = 3;
        se.alas = 6;
        se.tinggi = 4;
        se.luas = (0.5 * se.alas * se.tinggi);
        se.Luas();
        Persegi pe = new Persegi();
        pe.bangundatar = "Persegi";
        pe.sisi = 4;
        pe.luas = pe.sisi * pe.sisi;
        pe.keliling = 4 * pe.sisi;
        pe.Luas();
        pe.Keliling();

        BangunDatar Bangun = new BangunDatar();
        Bangun.Info();
        se.Info();
        pe.Info();
    }
}

```

**Listing Program Modul 5 Class Utama**



```

"G:\Program Files\Java\jdk 15\bin\java.exe" "-javaagent:G:\Prog
Luas Segitiga = 12.0
Luas Persegi = 16.0
Keliling Persegi = 16.0
Ini Adalah Bangun Datar. Ada Banyak Jenis-jenis Bangun Datar.
Ini Adalah Bangun Datar Berupa Segitiga
Ini Adalah Bangun Datar Berupa Persegi

Process finished with exit code 0

```

**Gambar 3.5 Output Program Modul 5**



### 3.1.6 Modul 6

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 6 berupa *listing program*, serta *outputnya*(IntelliJ):

```
public class BangunDatar {
    private int jumlahhisi;
    private double luas, keliling;
    private String bangundatar;

    protected void info() {
        System.out.println("Ini bangun datar");
    }

    protected void Luas() {
        System.out.println("Luas " + bangundatar + " = " + luas);
    }

    protected void Keliling() {
        System.out.println("Keliling " + bangundatar + " = " + keliling);
    }

    protected void Luas(int sisi) {
        luas = sisi * sisi;
        System.out.println("Luas persegi : " + luas);
    }

    protected void Luas(int a, int t) {
        luas = a * t * 0.5;
        System.out.println("Luas Segitiga : " + luas);
    }

    public void setBangundatar(String bangundatar) {
        this.bangundatar = bangundatar;
    }

    public void setJumlahhisi(int jumlahhisi) {
        this.jumlahhisi = jumlahhisi;
    }

    public void setKeliling(double keliling) {
        this.keliling = keliling;
    }

    public void setLuas(double luas) {
        this.luas = luas;
    }

    public int getJumlahhisi() {
        return jumlahhisi;
    }

    public double getLuas() {
        return luas;
    }

    public String getBangundatar() {
        return bangundatar;
    }

    public double getKeliling() {
        return keliling;
    }
}
```

**Listing Program Modul 6 Class BangunDatar**

```

public class Persegi extends BangunDatar {
    private int sisi;

    public void setSisi(int sisi) {
        this.sisi = sisi;
    }

    public int getSisi() {
        return sisi;
    }

    protected void info() {
        System.out.println("Ini bangun persegi");
    }
}

```

**Listing Program Modul 6 Class Persegi**

```

public class Segitiga extends BangunDatar {
    int alas, tinggi;
    private int jumlahsisi;

    public void setAlas(int alas) {
        this.alas = alas;
    }

    public void setTinggi(int tinggi) {
        this.tinggi = tinggi;
    }

    public int getJumlahhisi() {
        return super.getJumlahhisi();
    }

    public int getAlas() {
        return alas;
    }

    public int getTinggi() {
        return tinggi;
    }

    public int getJumlahsisi() {
        return jumlahsisi;
    }

    protected void info() {
        System.out.println("ini bangun segitiga");
    }
}

```

**Listing Program Modul 6 Class Segitiga**

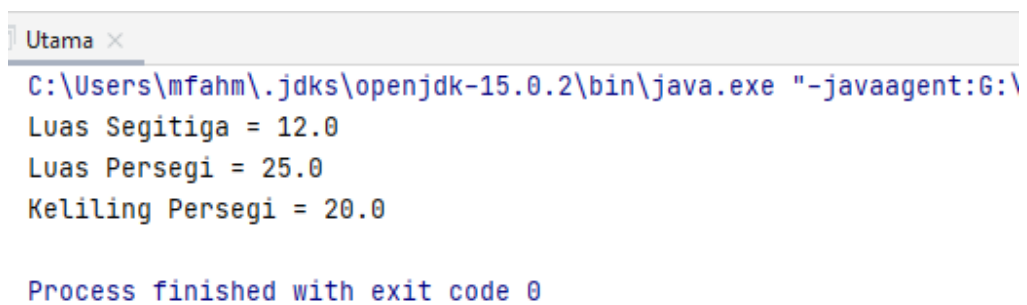
```

public class Utama {
    public static void main(String[] args) {
        Segitiga se = new Segitiga();
        se.setBangundatar("Segitiga");
        se.setJumlahisi(3);
        se.setAlas(6);
        se.setTinggi(4);
        se.setLuas(0.5 * se.alas * se.tinggi);
        se.Luas();

        Persegi pe = new Persegi();
        pe.setBangundatar("Persegi");
        pe.setSisi(5);
        pe.setLuas(pe.getSisi() * pe.getSisi());
        pe.setKeliling(4 * pe.getSisi());
        pe.Luas();
        pe.Keliling();
    }
}

```

**Listing Program Modul 6 Class Utama**



```

C:\Users\mfahm\.jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\
Luas Segitiga = 12.0
Luas Persegi = 25.0
Keliling Persegi = 20.0

Process finished with exit code 0

```

**Gambar 3.6 Output Program Modul 6**

### 3.1.7 Modul 7

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 7 berupa *listing program*, serta *outputnya*(IntelliJ):

```

package Interface;

public interface DataPraktikan {
    public void InputNamaStb();

    public void InputPraktikum();
}

```

**Listing Program Modul 7 Interface DataPraktikan**

```
package Interface;

public interface ProsesNilaiPraktikum {
    public float nilaiTugas1();

    public float nilaiTugas2();

    public float nilaiAkhir(float tugas1, float tugas2,
float mid, float uas);
}
```

**Listing Program Modul 7 Interface ProsesNilaiPraktikum**

```
package Abstarct;

public abstract class InputNilai {
    public int acc1, acc2, acc3, acc4, mid, uas;

    public abstract void inputNilaiAcc();

    public abstract void inputNilaiUjian();

    public void tampilNilai() {
        System.out.println("Nilai Acc1 : " + acc1);
        System.out.println("Nilai Acc2 : " + acc2);
        System.out.println("Nilai Acc3 : " + acc3);
        System.out.println("Nilai Acc4 : " + acc4);
        System.out.println("Nilai MID : " + mid);
        System.out.println("Nilai Final : " + uas);
    }
}
```

**Listing Program Modul 7 Abstract Class InputNilai**

```
package Utama;

import java.util.Scanner;

import Interface.DataPraktikan;
import Abstarct.InputNilai;
import Interface.ProsesNilaiPraktikum;

import java.time.temporal.TemporalAdjusters;

public class Utama extends InputNilai implements
DataPraktikan, ProsesNilaiPraktikum {
    String nama, stb, praktikum;
    Scanner sc;

    public Utama() {
        sc = new Scanner(System.in);
    }
}
```

```

@Override
    public void inputNilaiAcc() {
        acc1 = sc.nextInt();
        acc2 = sc.nextInt();
        acc3 = sc.nextInt();
        acc4 = sc.nextInt();
    }

    @Override
    public void inputNilaiUjian() {
        mid = sc.nextInt();
        uas = sc.nextInt();
    }

    @Override
    public void InputNamaStb() {
        nama = sc.next();
        stb = sc.nextLine();
        System.out.println(nama + stb);
    }

    @Override
    public void InputPraktikum() {
        praktikum = sc.next();
    }

    @Override
    public float nilaiTugas1() {
        return (acc1 + acc2) / 2;
    }

    @Override
    public float nilaiTugas2() {
        return (acc3 + acc4) / 2;
    }

    @Override
    public float nilaiAkhir(float tugas1, float tugas2, float mid,
float uas) {
        return (((tugas1 + tugas2) / 2) * 0.3f) + (mid * 0.3f) +
(uas * 0.4f);
    }

    public static void main(String[] args) {
        Utama ut = new Utama();

        ut.InputNamaStb();
        ut.InputPraktikum();
        ut.inputNilaiAcc();
        ut.inputNilaiUjian();
        float tugas1 = ut.nilaiTugas1();
        float tugas2 = ut.nilaiTugas2();
        float na = ut.nilaiAkhir(tugas1, tugas2, ut.mid, ut.uas);
        System.out.println("Nilai Akhir : " + na);

    }
}

```

**Listing Program Modul 7 Class Utama**

```

Utama x
C:\Users\mfahm\.jdk\openjdk-15.0.2\bin\java.exe "-javaagent:G:\Program Files\
Muhammad Fahmi 13020190019
Muhammad Fahmi 13020190019
7
5 6 7 8
9 8
Nilai Akhir : 7.7

Process finished with exit code 0

```

**Gambar 3.7 Output Program Modul 7**

### 3.1.8 Modul 8

Berikut dibawah ini adalah implementasi dari kegiatan praktikum pada Modul 8 berupa *listing program*, serta *outputnya*(IntelliJ):

```

package Mahasiswa;

import javax.swing.*;
import java.util.HashMap;
import java.util.Scanner;

public class Mahasiswa {

    public static void main(String[] args) {
        HashMap<Integer, String> name = new HashMap<Integer,
String>();
        /*Scanner sc = new Scanner(System.in);
        name.put(sc.nextInt(), sc.next());
        name.put(2, " Aarim");*/

        name.put(1, " Petra");
        name.put(2, " Aarim");
        System.out.println("Isi objek name : " + name);
        System.out.println("Asisten urut ke-2 : " +
name.get(2));
        // JOptionPane.showInputDialog("Masukkan Nilai");
    }
}

```

**Listing Program Modul 8 Class Mahasiswa**

```

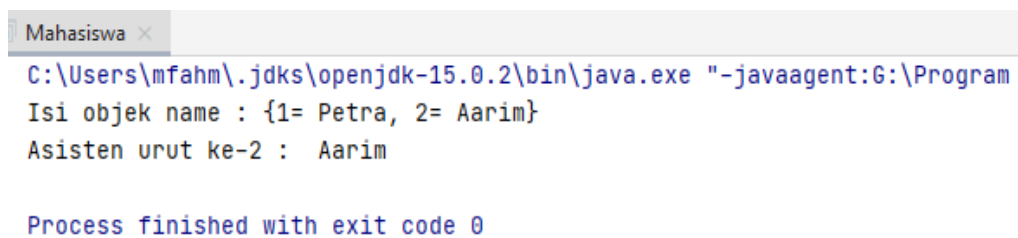
package Mahasiswa;

import java.util.ArrayList;

public class Gudang {
    public static void main(String[] args) {
        ArrayList<String> barang = new ArrayList<>();
        barang.add("Monitor");
        barang.add("Keyboard");
        barang.add("Meja");
        barang.remove("Meja");
        System.out.println(barang);
        System.out.println("isi Gudang Sebanyak " +
        barang.size() + " item");
    }
}

```

**Listing Program Modul 8 Class Gudang**



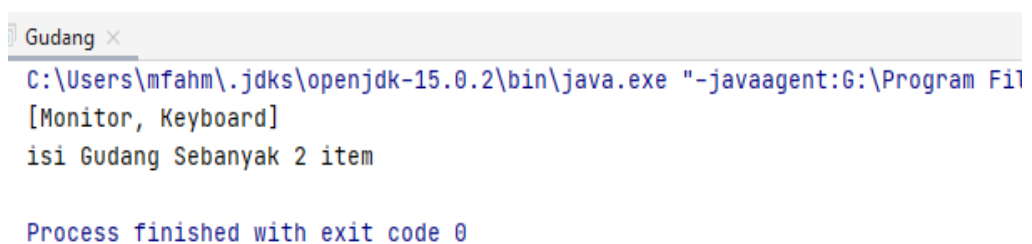
```

C:\Users\mfahm\jdk\openjdk-15.0.2\bin\java.exe -javaagent:G:\Program
Isi objek name : {1= Petra, 2= Aarim}
Asisten urut ke-2 : Aarim

Process finished with exit code 0

```

**Gambar 3.8 Output Program Modul 8 Class Mahasiswa**



```

C:\Users\mfahm\jdk\openjdk-15.0.2\bin\java.exe -javaagent:G:\Program Fil
[Monitor, Keyboard]
isi Gudang Sebanyak 2 item

Process finished with exit code 0

```

**Gambar 3.9 Output Program Modul 8 Class Gudang**

### 3.2 Pembahasan

Adapun pembahasan tiap-tiap hasil implementasi untuk setiap modul mulai dari modul 1 hingga modul 8.

### 3.2.1 Modul 1

Membuat sebuah Class dengan nama class “ HelloWorld” dengan akses modifier class “public”.

```
package Fahmi.Mod1;

public class HelloWorld {
}
```

Menambahkan konstruktor pada class HelloWorld dengan isi konstruktor menampilkan nama dan stb.

```
public HelloWorld() {
    System.out.println("Muhammad Fahmi");
    System.out.println("13020190019");
}
```

Menambahkan fungsi utama pada class HelloWorld.

```
package Fahmi.Mod1;

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

Membuat program untuk menghasilkan output berupa text “Hello Word !!!”

```
package Fahmi.Mod1;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello Word !!!");
    }
}
```

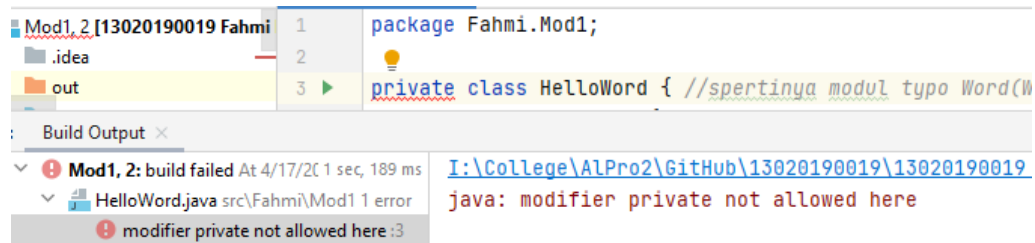
Lalu melengkapi program hingga menghasilkan output berupa text “Nur / 13020140003 “ dan “Ini Pemrograman Java”

```
package Fahmi.Mod1;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello Word !!!");
        System.out.println("Nur / 13020140003");
        System.out.println("Ini Pemrograman Java");
    }
}
```

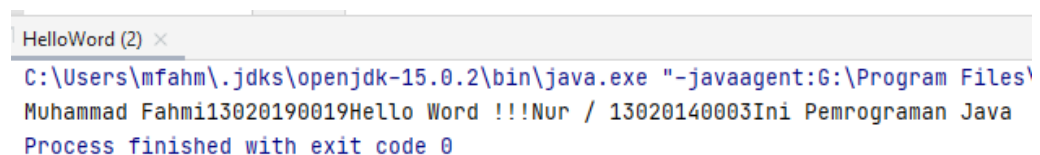


Kemudian ketika mengubah akses modifier class dari public menjadi private. Maka akan error seperti dibawah ini:



**Gambar 3.10 Public ke Private**

Kemudian ketika mengubah keyword println menjadi print. Maka akan tidak ada new line pada outputnya.



**Gambar 3.11 Println ke Print**

### 3.2.2 Modul 2

Membuat sebuah Class dengan nama class “ Hitung” dengan akses modifier class “public” dgn deklarasi atribut nama dan stb dgn tipe data String

```
package Fahmi.Mod2;

public class Hitung {
    String nama, stb;
}
```

Menambahkan method pada class tampilNama dengan isi untuk menampilkan nama dan stb.

```
public void tampilNama() {
    System.out.println("Nama : " + nama);
    System.out.println("Stb : " + stb);
}
```

Menambahkan fungsi utama pada class Hitung. Yang dimana didalamnya terdapat objek Hitung bernama h, dan deklarasi atribut nilai1 bernilai 10 dan nilai2 bernilai 5. Juga memanggil method tampilNama

```
public static void main(String[] args) {
    Hitung h = new Hitung();
    int nilai1, nilai2;
    nilai1 = 10;
    nilai2 = 5;
    h.tampilNama();
}
```

Membuat method tambah, selisih, kali dan bagi didalam kelas Hitung yang dimana nilainya akan dikembalikan sesuai rumus seperti untuk contoh tambah jd “return nilai1 + nilai2;” dst.

```
package Fahmi.Mod2;

public class Hitung {
    public float tambah(int nilai1, int nilai2) {
        return nilai1 + nilai2;
    }

    public float selisih(int nilai1, int nilai2) {
        return nilai2 - nilai1;
    }

    public float kali(int nilai1, int nilai2) {
        return nilai1 * nilai2;
    }

    public float bagi(int nilai1, int nilai2) {
        return nilai1 / nilai2;
    }
}
```

Lalu terakhir membuat pilihan menggunakan yang operator yang mana dengan Java Conditions and If Statements, seperti dibawah ini:

```
public class Hitung {
    public static void main(String[] args) {
        if (pilihan.equals("tambah")) {
            System.out.println(nilai1 + " + " + nilai2 + " = " + h.tambah(nilai1,
                nilai2));
        } else if (pilihan.equals("selisih")) {
            System.out.println(nilai2 + " - " + nilai1 + " = " + h.selisih(nilai2,
                nilai1));
        } else if (pilihan.equals("kali")) {
            System.out.println(nilai1 + " * " + nilai2 + " = " + h.kali(nilai1,
                nilai2));
        } else if (pilihan.equals("bagi")) {
            System.out.println(nilai1 + " / " + nilai2 + " = " + h.bagi(nilai1,
                nilai2));
        } else {
            System.out.println("Tidak ada pilihan !");
        }
    }
}
```

### 3.2.3 Modul 3

Membuat sebuah Class dengan nama class “Mahasiswa” dengan akses modifier class “public” dgn deklarasi atribut nama dan stb dgn tipe data String. Dengan akses modifier class “private”, disini akan menggunakan getter dan setter agar bisa terbaca oleh class Utama.

```
package Mahasiswa;

public class Mahasiswa {
    private String stb, nama;

    public String getStb() {
        return stb;
    }

    public void setStb(String stb) {
        this.stb = stb;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }
}
```

Membuat Class dengan nama “Utama”. Didalamnya terdapat objek Mahasiswa dengan nama mhs dan juga memanggil method setNama dan setStb dengan parameter Udin dan 132020110909. Lalu melengkapi program hingga menghasilkan output berupa text “Stb “ dan “Nama” yang dimana akan menampilkan sesuai apa yang ditulis pada parameter setNama dan setStb.

```
package Mahasiswa;

public class Utama {
    public static void main(String[] args) {
        Mahasiswa mhs = new Mahasiswa();
        mhs.setNama("Udin");
        mhs.setStb("132020110909");
        System.out.println("Stb : " + mhs.getStb());
        System.out.println("Nama : " + mhs.getNama());
    }
}
```

### 3.2.4 Modul 4

Mengimport Scanner sebagai penginputan oleh user atau pengguna.

Membuat sebuah Class dengan nama class “LuasSegitiga” dengan akses modifier class “public” yang dimana didalamnya terdapat method luas bertipe double berparameter alas dan tinggi. Akan mengembalikan nilai 0.5 dikali alas dikali tinggi.

```
package ContohScanner;

import java.util.Scanner;

public class LuasSegitiga {

    public double luas(int alas, int tinggi) {
        return (0.5 * alas * tinggi);
    }
}
```

Membuat function main didalam kelas LuasSegitiga. Didalamnya terdapat objek LuasSegitiga dengan nama ls dan juga Scanner untuk penginputan nilai dengan nama objek sc. Juga mendeklarasi atribut alas dan tinggi dengan tipe int. Lalu melengkapi program hingga menghasilkan output berupa text "Masukkan Alas Segitiga : " , "Masukkan Tinggi Segitiga : " dan "Luas Segitiga : " yang dimana akan

menampilkan sesuai apa yang diinput pada cmd dengan yang tadinya menggunakan Objek sc Scanner.

```
public static void main(String[] args) {
    LuasSegitiga ls = new LuasSegitiga();
    int alas, tinggi;
    Scanner sc = new Scanner(System.in);
    System.out.print("Masukkan Alas Segitiga : ");
    alas = sc.nextInt();
    System.out.print("Masukkan Tinggi Segitiga : ");
    tinggi = sc.nextInt();
    System.out.println("Luas Segitiga : " + ls.luas(alas,
tinggi));
}
```

Begitupun juga berlaku pada Import BufferedReader dan Joption, ketiganya mempunyai konsep yang sama yaitu untuk kebutuhan penginputan hanya saja dengan penulisan program yang sedikit berbeda.

### 3.2.5 Modul 5

Membuat sebuah SuperClass dengan nama class “BangunDatar” dengan akses modifier class “public” yang dimana didalamnya deklarasi jumlah sisi tipe int. Luas dan keliling tipe double dan bangun datar tipe String. Juga terdapat method Luas, Keliling, dan Info yang tidak

```
public class BangunDatar {
    int jumlahSisi;
    double luas, keliling;
    String bangunDatar;

    void Luas() {
        System.out.println("Luas " + bangunDatar + " = " +
luas);
    }

    void Keliling() {
        System.out.println("Keliling " + bangunDatar + " = "
+ keliling);
    }

    void Info() {
        System.out.println("Ini Adalah Bangun Datar. Ada
Banyak Jenis-jenis Bangun Datar.");
    }
}
```

mengembalikan nilai karena void. Didalam ketiga method tersebut akan menampilkan sesuai apa yang diperintahkan dibawah:

Membuat sebuah SubClass Segitiga dari SuperClass “BangunDatar” dengan akses modifier class “public” yang dimana didalamnya terdapat hanya method Info yang tidak mengembalikan nilai dan akan menampilkan "Ini Adalah Bangun Datar Berupa Segitiga". Juga terdapat deklarasi alas dan tinggi bertipe int pada SubClass Segitiga.

```
public class Segitiga extends BangunDatar {
    int alas, tinggi;

    void Info() {
        System.out.println("Ini Adalah Bangun Datar Berupa
Segitiga");
    }
}
```

Sama pada penjelasan diatas, hanya saja beda SubClass bernama Persegi yang akan menampilkan "Ini Adalah Bangun Datar Berupa Persegi".

```
public class Persegi extends BangunDatar {
    int sisi;

    void Info() {
        System.out.println("Ini Adalah Bangun Datar Berupa
Persegi");
    }
}
```

Membuat class utama yang dimana terdapat function main. Dalamnya berisi deklarasi method Segitiga bernama se. Lalu men-set nilai bangundatar dengan Segitiga, nilai jumlahtinggi dengan 3, nilai alas dengan 6, nilai tinggi dengan 4, nilai luas dengan 0.5 dikali se.alas dikali se.tinggi. Terakhir memanggil objek se.Luas().

```

public class Utama {
    public static void main(String[] args) {
        Segitiga se = new Segitiga();
        se.bangundatar = "Segitiga";
        se.jumlahsisi = 3;
        se.alas = 6;
        se.tinggi = 4;
        se.luas = (0.5 * se.alas * se.tinggi);
        se.Luas();
    }
}

```

Sama seperti lanjutan diatas, berisi deklarasi method Persegi bernama pe. Lalu men-set nilai pe.bangundatar dengan Persegi, nilai pe.sisi dengan 4, nilai pe.luas dengan pe.sisi dikali pe.sisi, nilai pe.keliling dengan 4 dikali pe.sisi. Kemudian memanggil objek pe.Luas(), pe.Keliling(), se.Info() dan pe.Info(). Terakhir membuat objek BangunDatar dengan nama Bangun dan memanggil Bangun.Info().

```

public class Utama {
    public static void main(String[] args) {
        Persegi pe = new Persegi();
        pe.bangundatar = "Persegi";
        pe.sisi = 4;
        pe.luas = pe.sisi * pe.sisi;
        pe.keliling = 4 * pe.sisi;
        pe.Luas();
        pe.Keliling();
        se.Info();
        pe.Info();

        BangunDatar Bangun = new BangunDatar();
        Bangun.Info();
    }
}

```

### 3.2.6 Modul 6

Membuat class BangunDatar dengan atribut jumlah sisi tipe int. Luas dan keliling bertipe double. Bangundatar bertipe String. Keempatnya mempunyai akses modifier private, maka dari itu diperlukannya setter dan getter.

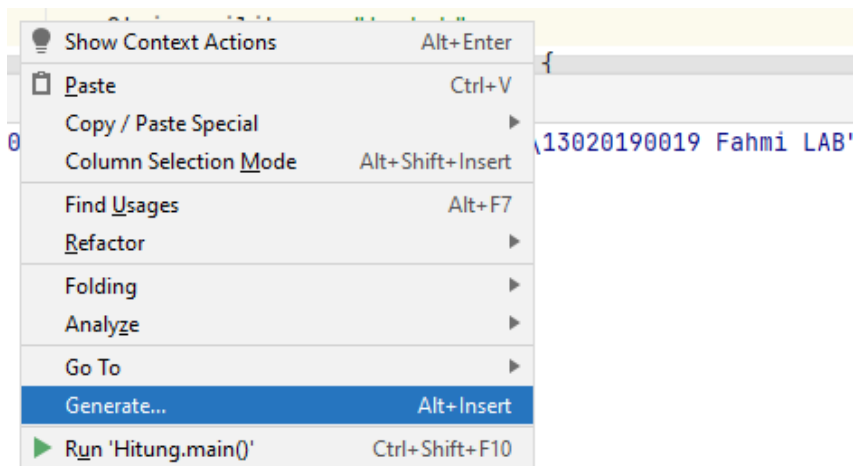
```
public class BangunDatar {
    private int jumlahSisi;
    private double luas, keliling;
    private String bangunDatar;

    protected void info() {
        System.out.println("Ini bangun datar");
    }
    protected void Luas() {
        System.out.println("Luas " + bangunDatar + " = " + luas);
    }
    protected void Keliling() {
        System.out.println("Keliling " + bangunDatar + " = " + keliling);
    }
    protected void Luas(int sisi) {
        luas = sisi * sisi;
        System.out.println("Luas persegi : " + luas);
    }
    protected void Luas(int a, int t) {
        luas = a * t * 0.5;
        System.out.println("Luas Segitiga : " + luas);
    }
    public void setBangundatar(String bangunDatar) {
        this.bangunDatar = bangunDatar;
    }
    public void setJumlahSisi(int jumlahSisi) {
        this.jumlahSisi = jumlahSisi;
    }
    public void setKeliling(double keliling) {
        this.keliling = keliling;
    }
    public void setLuas(double luas) {
        this.luas = luas;
    }
    public int getJumlahSisi() {
        return jumlahSisi;
    }
    public double getLuas() {
        return luas;
    }
    public String getBangundatar() {
        return bangunDatar;
    }
    public double getKeliling() {
        return keliling;
    }
}
}
```



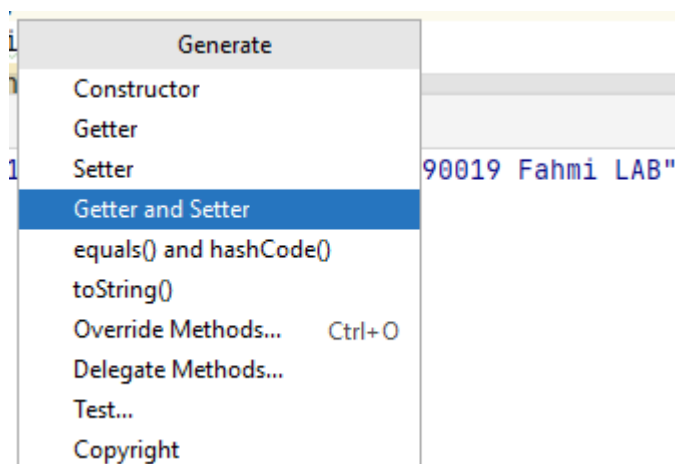
Disini ada cara cepat untuk membuat setter dan getter secara otomatis pada IntelliJ, berikut caranya:

- a) Klik kanan pada window IntelliJ lalu pilih generate... atau bisa menggunakan shorcut Alt+Insert



**Gambar 3.12 Right-Click Menu pada IntelliJ**

- b) Setelah itu pilih Setter and Getter lalu dengan sendirinya akan men-generate setter dan getter di programnya.



**Gambar 3.13 Right-Click Menu pada Generate di IntelliJ**

Membuat sebuah SubClass Persegi dari SuperClass “BangunDatar” dengan akses modifier class “public” yang dimana didalamnya terdapat deklarasi atribut sisi bertipe int dengan akses modifier private, jadi harus menggunakan setter dan getter agar bisa terbaca oleh class yang lainnya. Juga terdapat deklarasi method info yang dimana akan menampilkan "Ini bangun persegi".

```
public class Persegi extends BangunDatar {
    private int sisi;

    public void setSisi(int sisi) {
        this.sisi = sisi;
    }

    public int getSisi() {
        return sisi;
    }

    protected void info() {
        System.out.println("Ini bangun persegi");
    }
}
```

Membuat sebuah SubClass Segitiga dari SuperClass “BangunDatar” dengan akses modifier class “public” yang dimana didalamnya terdapat deklarasi atribut alas dan tinggi bertipe int dengan akses modifier default dan jumlahsisi dengan akses modifier private, jadi harus menggunakan setter dan getter agar bisa terbaca oleh class yang lainnya. Juga terdapat deklarasi method info yang dimana akan menampilkan "Ini bangun segitiga".

```
public class Segitiga extends BangunDatar {  
    int alas, tinggi;  
    private int jumlahsisi;  
  
    public void setAlas(int alas) {  
        this.alas = alas;  
    }  
  
    public void setTinggi(int tinggi) {  
        this.tinggi = tinggi;  
    }  
  
    public int getJumlahhisi() {  
        return super.getJumlahhisi();  
    }  
  
    public int getAlas() {  
        return alas;  
    }  
  
    public int getTinggi() {  
        return tinggi;  
    }  
  
    public int getJumlahsisi() {  
        return jumlahsisi;  
    }  
  
    protected void info() {  
        System.out.println("ini bangun segitiga");  
    }  
}
```

Membuat class utama yang dimana terdapat function main. Dalamnya berisi deklarasi method Segitiga bernama se. Lalu men-set nilai se.setBangundatar dengan Segitiga, nilai se.setJumlahisi dengan 3, nilai se.setAlas dengan 6, nilai se.setTinggi dengan 4, nilai se.setLuas dengan 0.5 dikali se.alas dikali se.tinggi. Terakhir memanggil objek se.Luas(). Selanjutnya berisi deklarasi method Persegi bernama pe. Lalu men-set nilai pe. pe.setBangundatar dengan Persegi, nilai pe.setSisi dengan 5, nilai pe.setLuas dengan pe.getSisi() dikali pe.getSisi(), nilai pe.setKeliling dengan 4 dikali pe.getSisi(). Kemudian memanggil objek pe.Luas(), pe.Keliling().

```
public class Utama {
    public static void main(String[] args) {
        Segitiga se = new Segitiga();
        se.setBangundatar("Segitiga");
        se.setJumlahisi(3);
        se.setAlas(6);
        se.setTinggi(4);
        se.setLuas(0.5 * se.alas * se.tinggi);
        se.Luas();

        Persegi pe = new Persegi();
        pe.setBangundatar("Persegi");
        pe.setSisi(5);
        pe.setLuas(pe.getSisi() * pe.getSisi());
        pe.setKeliling(4 * pe.getSisi());
        pe.Luas();
        pe.Keliling();
    }
}
```

### 3.2.7 Modul 7

Membuat interface DataPraktikan, dalamnya terdapat method tidak mengembalikan nilai atau void InputNamaStb() dan InputPraktikum().

```
public interface DataPraktikan {
    public void InputNamaStb();
    public void InputPraktikum();
}
```

Selanjutnya, Membuat interface ProsesNilaiPraktikum, dalamnya terdapat method nilaiTugas1() dan nilaiTugas2(), dan nilaiAkhir berparameter (float tugas1, float tugas2, float mid, float uas).

```
package Interface;

public interface ProsesNilaiPraktikum {
    public float nilaiTugas1();

    public float nilaiTugas2();

    public float nilaiAkhir(float tugas1, float tugas2,
float mid, float uas);
}
```

Membuat abstract class InputNilai dengan deklarasi atribut acc1, acc2, acc3, acc4, mid, uas bertipe int dan method abstract void inputNilaiAcc() dan abstract void inputNilaiUjian(). Serta tampilNilai untuk menampilkan apa yang ditulis di listing program dibawah:

```
package Abstarct;

public abstract class InputNilai {
    public int acc1, acc2, acc3, acc4, mid, uas;

    public abstract void inputNilaiAcc();

    public abstract void inputNilaiUjian();

    public void tampilNilai() {
        System.out.println("Nilai Acc1 : " + acc1);
        System.out.println("Nilai Acc2 : " + acc2);
        System.out.println("Nilai Acc3 : " + acc3);
        System.out.println("Nilai Acc4 : " + acc4);
        System.out.println("Nilai MID : " + mid);
        System.out.println("Nilai Final : " + uas);
    }
}
```

Membuat class utama yang dimana terdapat function main. Menggunakan import Scanner sebagai penginputan dan abstract untuk memanggil Abstarct.InputNilai serta Interface untuk memanggil Interface.ProsesNilaiPraktikum. Mendeklarasi nama, stb, praktikum bertipe String modifier default dan sc Scanner. Juga membuat method/konstruktor utama untuk pemanggilan Scanner agar bisa diinput oleh user pada cmd.

```
package Utama;

import java.util.Scanner;

import Interface.DataPraktikan;
import Abstarct.InputNilai;
import Interface.ProsesNilaiPraktikum;

import java.time.temporal.TemporalAdjusters;

public class Utama extends InputNilai implements
DataPraktikan, ProsesNilaiPraktikum {
    String nama, stb, praktikum;
    Scanner sc;

    public Utama() {
        sc = new Scanner(System.in);
    }
}
```

Kemudian lanjutan dari listing program diatas membuat method inputNilaiAcc(), inputNilaiUjian(), InputNamaStb(), InputPraktikum(), nilaiTugas1(), nilaiTugas2(), nilaiAkhir(float tugas1, float tugas2, float mid, float uas) dan nextInt() sebagai metode pengiputan.

```

@Override
public void inputNilaiAcc() {
    acc1 = sc.nextInt();
    acc2 = sc.nextInt();
    acc3 = sc.nextInt();
    acc4 = sc.nextInt();
}

@Override
public void inputNilaiUjian() {
    mid = sc.nextInt();
    uas = sc.nextInt();
}

@Override
public void InputNamaStb() {
    nama = sc.next();
    stb = sc.nextLine();
    System.out.println(nama + stb);
}

@Override
public void InputPraktikum() {
    praktikum = sc.next();
}

@Override
public float nilaiTugas1() {
    return (acc1 + acc2) / 2;
}

@Override
public float nilaiTugas2() {
    return (acc3 + acc4) / 2;
}

@Override
public float nilaiAkhir(float tugas1, float tugas2, float mid,
float uas) {
    return (((tugas1 + tugas2) / 2) * 0.3f) + (mid * 0.3f) +
(uas * 0.4f);
}

```

Terakhir membuat function main pada class Utama. Pertama membuat objek Utama bernama ut, dan memanggil method ut.InputNamaStb(), ut.InputPraktikum(), ut.inputNilaiAcc(), ut.inputNilaiUjian(). Dan menampilkan "Nilai Akhir : " + na, dmn na adalah method ut.nilaiAkhir(tugas1, tugas2, ut.mid, ut.uas).

```

public static void main(String[] args) {
    Utama ut = new Utama();

    ut.InputNamaStb();
    ut.InputPraktikum();
    ut.inputNilaiAcc();
    ut.inputNilaiUjian();
    float tugas1 = ut.nilaiTugas1();
    float tugas2 = ut.nilaiTugas2();
    float na = ut.nilaiAkhir(tugas1, tugas2, ut.mid,
ut.uas);
    System.out.println("Nilai Akhir : " + na);

}
}

```

### 3.2.8 Modul 8

Membuat class Mahasiswa dengan menggunakan import HasMap dan Scanner sebagai penginputan. Membuat function main, dalamnya membuat Method `HashMap<Integer, String> name = new HashMap<Integer, String>()`. Lalu mendeklarasi HashMap Petra sebagai 1 dan Aarim sebagai 2. Terakhir akan menampilkan sesuai apa yang ditulis pada HashMap.

```

package Mahasiswa;

import javax.swing.*;
import java.util.HashMap;
import java.util.Scanner;

public class Mahasiswa {

    public static void main(String[] args) {
        HashMap<Integer, String> name = new HashMap<Integer,
String>();
        /*Scanner sc = new Scanner(System.in);
        name.put(sc.nextInt(), sc.next());
        name.put(2, " Aarim");*/

        name.put(1, " Petra");
        name.put(2, " Aarim");
        System.out.println("Isi objek name : " + name);
        System.out.println("Asisten urut ke-2 : " + name.get(2));
        // JOptionPane.showInputDialog("Masukkan Nilai");
    }
}

```



Membuat class Gudang dengan menggunakan import ArrayList. Membuat function main, dalamnya membuat ArrayList<String> barang = new ArrayList<>(). Lalu menambahkan Monitor, Keyboard, Meja dan menghapus Meja. Terakhir akan menampilkan apa apa saja list dari barang dan jumlah item barang dengan barang.size().

```
package Mahasiswa;

import java.util.ArrayList;

public class Gudang {
    public static void main(String[] args) {
        ArrayList<String> barang = new
ArrayList<>();
        barang.add("Monitor");
        barang.add("Keyboard");
        barang.add("Meja");
        barang.remove("Meja");
        System.out.println(barang);
        System.out.println("isi Gudang Sebanyak "
+ barang.size() + " item");
    }
}
```

## BAB IV

### KESIMPULAN & SARAN

#### 4.1 Kesimpulan

Setelah melakukan praktikum, praktikan bisa membuat program *Java* berdasarkan percobaan yang telah dilakukan. Beberapa hasil praktikum yang bisa kita dapat dapatkan, yaitu:

1. *Java* merupakan salah satu bahasa pemrograman yang sudah ada dari era 1990-an. Bahasa Pemrograman *Java* adalah bahasa pemrograman yang mengadopsi sintaks dari bahasa C dan C++. Salah satu sifat pemrograman *Java* yang menjadi keuntungan dalam menggunakan bahasa pemrograman ini adalah bersifat *multiplatform* yaitu *Java* dapat dijalankan pada banyak *platform* atau sistem informasi.
2. Class atau kelas adalah kelompok objek-objek yang memiliki karakteristik yang sama (yang sejenis). Setiap class pasti memiliki objek, satu kelas ini dapat terbentuk banyak objek. Untuk membuat dan menginisialisasi sebuah *object* baru diperlukan konstruktor yang merupakan sebuah tipe khusus dari *method* yang digunakan. *Method* adalah kumpulan kode perintah untuk melakukan sebuah proses yang diberi nama yang disebut sebagai nama *method*.
3. Package adalah suatu cara untuk manage atau mengelompokkan *class-class* berdasarkan kesamaan atau kemiripan fungsi. Selain itu untuk membaca inputan pengguna diperlukannya import *Scanner*, *JOptionPane*, atau *BufferedReader*.

4. Konsep yang digunakan dalam pemrograman *Java* adalah OOP (*Object Oriented Programming*) yang dimana terdiri dari 3 konsep yaitu:

- a) Enkapsulasi, yaitu pemberian hak akses oleh akses *modifier* (*public, private, protected, default*)
- b) *Inheritance* yaitu konsep pewarisan. Konsep ini akan dibahas di modul selanjutnya.
- c) *Polimorphisme* yang artinya banyak bentuk. Konsep ini akan dibahas di modul selanjutnya.

5. *ArrayList* merupakan *collection* yang menjadi bagian dari *Java.Util.ArrayList*. Berbeda dengan *array* biasa, *ArrayList* memiliki sejumlah operasi yang lebih lengkap dan mudah digunakan dibandingkan dengan *array* biasa. Sedangkan *HashMap* merupakan class turunan dari class *AbstractMap* dan implementasi dari *interface Map*. *HashMap* adalah sebuah class yang berisi sekumpulan pasangan nilai (*value*) dan kunci (*key*).

## 4.2 Saran

Adapun saran dari praktikum ini, yaitu:

- 1. Sebelum praktikum, pastikan aplikasi IDE *Java* seperti *IntelliJ* atau *NetBeans* telah terinstal dengan baik agar tidak *error* saat dijalankan.
- 2. Sebaiknya praktikan lebih teliti dalam membuat program dan dapat menghindari kesalahan dalam pemakaian simbol-simbol dalam *Java* agar program yang dibuat dapat dijalankan.

*Daftar Pustaka*

*Laboratorium Terpadu Universitas Muslim Indonesia, 2018, Modul Praktikum Pemrograman Berorientasi Objek, Universitas Muslim Indonesia.*

*Rizki Muliono, <http://rizkimuliono.blog.uma.ac.id/pemrograman-berorientasi-objek/>, Diakses Tanggal 05 April 2021.*