

# **MODUL PRAKTIKUM MATEMATIKA INFORMATIKA 2**

## **"CENGKARENG"**



WRITTEN BY : JOHAN

GITHUB :<https://github.com/jolabti/matificengkareng.git>  
(dijinkan untuk mengimprove modul ini bagi setiap PJ)

**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2016**

## **Materi – Materi**

- 1. Mengenal Java**
- 2. Tipe Data pada Java / Identifier**
- 3. Operator / Operasi Java**
- 4. Percabangan / Perulangan Java**
- 5. Konsep OOP Java**
- 6. Array 1 / 2 Dimensi Java**
- 7. Penjumlahan dan Perkalian Matriks**

Target Dasar : Praktikan mampu memahami penggunaan Java sebagai sarana pemrosesan angka.

Target Lebih : Memantapkan potensi OOP pada Program Java kepada Praktikan.

# 1. MENGENAL JAVA

**Java** adalah **bahasa pemrograman** yang dapat dijalankan di berbagai **komputer** termasuk **telepon genggam**. Bahasa ini awalnya dibuat oleh **James Gosling** saat masih bergabung di **Sun Microsystems** saat ini merupakan bagian dari **Oracle** dan dirilis tahun **1995**. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada **C** dan **C++** namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam **p-code** (*bytecode*) dan dapat dijalankan pada berbagai **Mesin Virtual Java (JVM)**. Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform **sistem operasi** yang berbeda, java dikenal pula dengan slogannya, "***Tulis sekali, jalankan di mana pun***". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi.

Contoh Aplikasi yang menggunakan Java :

1. Desktop Application
2. Android
3. Website
4. Microcontroller

Pada Java, library yang dimilikinya bisa dikatakan sangat lengkap. Terlebih pengembangan bahasa ini selalu berjalan secara konstan hingga terbit versi yang paling terbaru JDK (Java Development Kit) Versi 8.

Daftar IDE (Integrated Development Environment) Java :

- Netbeans V6-->ke atas Stabil
- Eclipse

## 2. TIPE DATA JAVA / IDENTIFIER

JAVA merupakan bahasa pemrograman berbasis objek. Apa yang dimaksud dengan objek ? Objek adalah sebuah komponen yang memiliki berbagai atribut khusus di dalamnya. Jika di analogikan objek itu seorang “**manusia**” maka dia dapat dipastikan memiliki *nama, umur, alamat, pekerjaan, serta aktifitas yang bisa dikerjakan*. Namun setiap atribut yang dimiliki tersebut memiliki tipe datanya tersendiri.

### Tipe Data dalam Java

Tipe data merupakan suatu kelas dari objek data dengan kumpulan operasi untuk membentuk dan memanipulasinya. Setiap bahasa pemrograman mempunyai kumpulan tipe data sederhana yang sudah terpaket di dalamnya. Tetapi juga dimungkinkan untuk dapat mendefinisikan tipe data baru.

Terdapat 8 tipe data sederhana dalam Java, yaitu :

1. **byte** dengan jangkauan : -128 sampai 127
2. **short** dengan jangkauan : -32.768 sampai 32.767
3. **int** dengan jangkauan : -2.147.483.648 sampai -2.147.483.647
4. **long** dengan jangkauan : -9.223.372.036.854.775.808 sampai 9.223.372.036.854.775.807
5. **float** dengan jangkauan : 3.4e-038 sampai 3.4e+038
6. **double** dengan jangkauan : 1.7e-308 sampai 1.7e+308
7. **char**
8. **boolean**
9. **String**

Dari 8 tipe data sederhana tersebut dapat dikelompokkan dalam 4 macam tipe data, yaitu :

1. Kelompok tipe data Integer, digunakan untuk merepresentasikan data dengan tipe bilangan bulat. Tipe data yang termasuk dalam kelompok ini adalah byte, short, int dan long.
2. Kelompok tipe data floating point, digunakan untuk merepresentasikan data dengan tipe bilangan riil/pecahan. Tipe data yang termasuk dalam kelompok ini adalah float dan double.
3. Kelompok tipe data karakter, digunakan untuk merepresentasikan data dengan tipe karakter alfanumerik. Tipe data yang termasuk dalam kelompok ini adalah char.
4. Kelompok tipe data boolean, digunakan untuk merepresentasikan data dengan tipe logika (benar/salah). Tipe data yang termasuk dalam kelompok ini adalah boolean.

Reserve word (Kata Kunci Java)

**Keyword** (kunci) adalah kata-kata khusus dalam Java yang digunakan untuk pemrograman.

Berikut adalah keyword yang digunakan dalam bahasa pemrograman Java:

```
abstract  boolean  break  byte
case      catch   char   class
const     continue default do
double    else    extends final
finally   float   for    future
goto      if      implements import
instanceof int    interface long
native     new     package  private
protected public  return   short
static     super   switch   synchronized
this       throw   throws   transient
```

```
/* try void volatile while
```

```
*/ To change this license header, choose License Headers in Project Properties.
```

```
*/ To change this template file, choose Tools | Templates
```

```
*/ and open the template in the editor.
```

```
*/
```

```
package matif;
```

```
/**
```

```
 *
```

```
 * @author johan
```

```
 */
```

```
public class Matif {
```

```
 /**
```

```
 * @param args the command line arguments
```

```
 */
```

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

```
    byte a = -128;
```

```
    short b = 32000;
```

```
    int c = 2500000, d;
```

```
    d = a + b + c;
```

```
    System.out.println(
```

```
        "hasilnya = " + d
```

```
    );
```

```
    Penggunaan Identifier :
```

```
 }
```

## 1. Tanpa Keyword

```
=====
Hasilnya :2531872
```

```
BUILD SUCCESSFUL (total time: 0 seconds)
```

Contoh dibawah ini bila tidak menggunakan keyword :

```
class Labti
{
int nama;
}
```

Hanya kode-kode yang terdapat dalam KB01 dan classclass lain yang dideklarasikan dalam package dimana MyClass dideklarasikan yang dapat mengakses variable nama.

## 2. Private

```
class Mahasiswa
{
private double ipk
}
```

Hanya kode-kode yang terdapat dalam class Mahasiswa yang dapat mengakses variable ipk.

## 3. Public

```
public class Pegawai
{
public String nama;
}
```

Kode-kode yang terdapat didalam class Pegawai dan class yang terdapat di dalam package lain dapat mengakses variable nama (class Pegawai harus dideklarasikan public juga agar dapat diakses class-class dalam package lain).

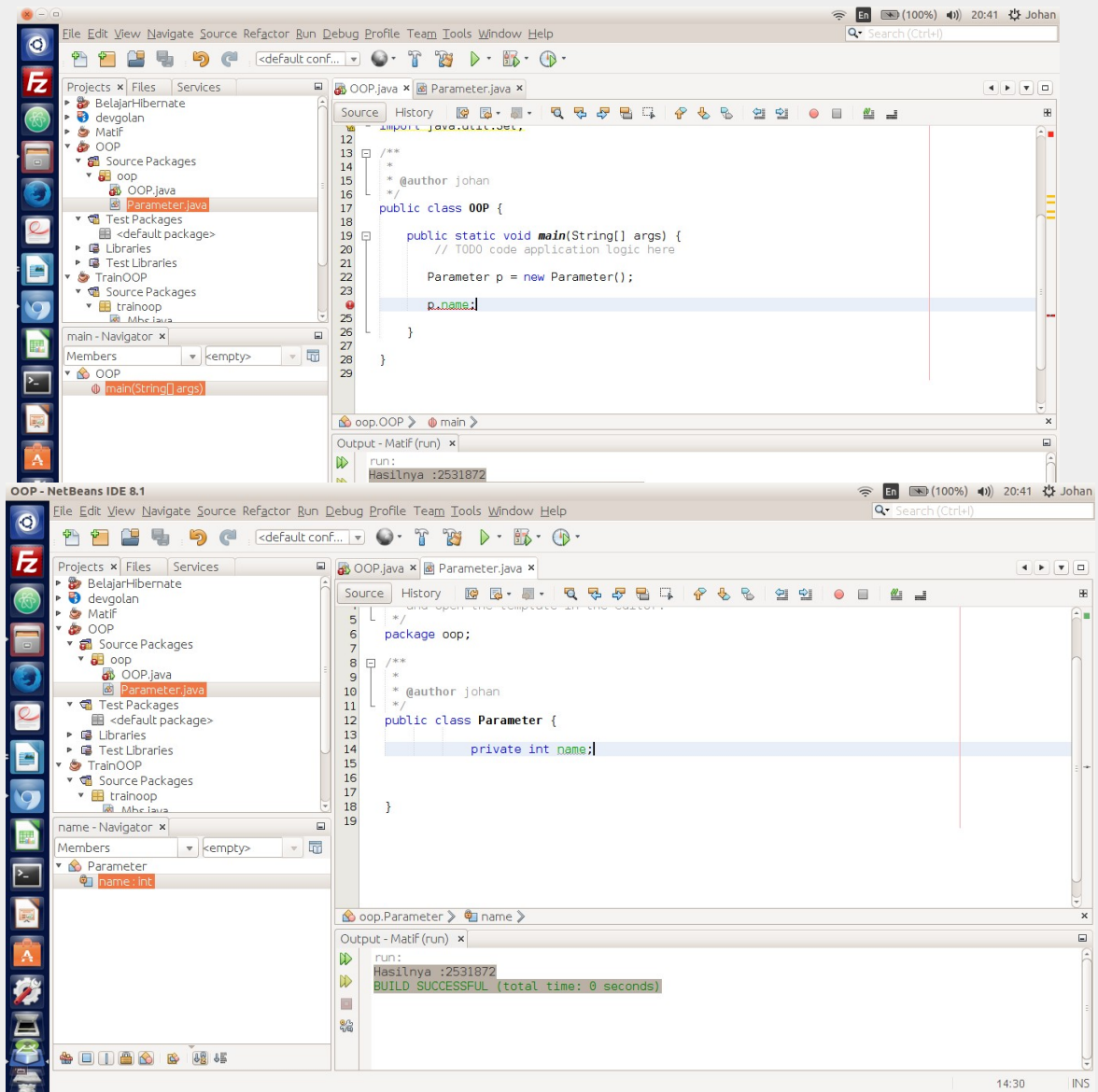
## 4. Protected

```
public class Pegawai
{
protected String nama;
}
```

Hanya kode-kode yang terdapat dalam class Pegawai dan class-class lain dalam satu package yang sama dengan class Pegawai dan seluruh sub class dari class Pegawai (yang

dideklarasikan dalam package lain) dapat mengakses variable nama.

### Contoh Kasus :



### Program Kalkulator :

#### Class : OOP.java

/\*

- \* To change this license header, choose License Headers in Project Properties.
- \* To change this template file, choose Tools | Templates
- \* and open the template in the editor.

\*/

package oop;

import java.util.Scanner;

```

/**
 *
 * @author johan
 */
public class OOP {

    public static void main(String[] args) {
        // TODO code application logic here
        int inputA;
        int inputB;

        int inputPilihan;

        Aksi ak = new Aksi();

        Scanner sc = new Scanner(System.in);

        System.out.println("Aplikasi Kalkulator Console Java \n");

        System.out.println("=====2016\n");
        System.out.println("1. Tambah");
        System.out.println("2. Kurang");
        System.out.println("3. Kali");
        System.out.println("4. Bagi");

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

        System.out.println("Masukkan Input A :.....");

        inputA = sc.nextInt();

        System.out.println("Masukkan Input B :.....");

        inputB = sc.nextInt();

        System.out.println("Masukkan Pilihan :.....");
    }
}

```



```
inputPilihan = sc.nextInt();

switch (inputPilihan) {

    case 1:

        ak.prosesTambah(inputA, inputB);

        break;
    case 2:

        ak.prosesKurang(inputA, inputB);

        break;

    case 3:

        ak.prosesKali(inputA, inputB);
        break;

    case 4:
        ak.prosesBagi(inputA, inputB);

    default:

        System.out.println("No choices");

}

}

}
```

Class : Aksi.java

/\*

\* To change this license header, choose License Headers in Project Properties.

```
* To change this template file, choose Tools | Templates
* and open the template in the editor.
```

```
*/
```

```
package oop;
```

```
/**
```

```
*
```

```
* @author johan
```

```
*/
```

```
public class Aksi {
```

```
    public int prosesTambah(int a, int b){
```

```
        int hasil=a +b;
```

```
        System.out.println("Hasil Penambahan :" + hasil);
```

```
        return hasil;
```

```
    }
```

```
    public int prosesKurang(int a, int b){
```

```
        int hasil=a - b;
```

```
        System.out.println("Hasil Pengurangan :" + hasil);
```

```
        return hasil;
```

```
    }
```

```
    public double prosesKali (double a,double b){
```

```
        double hasil = a * b ;
```

```
        System.out.println("Hasil Perkalian:" + hasil);
```

```
        return hasil;
```

```
    }
```

```
    public double prosesBagi (double a,double b){
```

```
        double hasil = a / b ;
```

```
        System.out.println("Hasil Pembagian:" + hasil);  
        return hasil;  
    }  
  
}
```

Kasus Tambahan untuk Percabangan :

#Tambahkan percabangan yang mengidentifikasi bahwa hasil dari operasi tersebut ganjil atau genap

#Tambahkan program cetak bintang sesuai dengan hasil yang telah di hasilkan oleh operasi penjumlahan atau pengurangan.

## 5. KONSEP OOP (OBJECT ORIENTED PROGRAMMING)

Secara umum, OOP terdiri dari Inheritance, Polymorphism, Encapsulation. Untuk penjelesan lebih lanjut tentang konsep – konsep tersebut terdapat pada item di bawah ini:

### 1. Inheritance

Sebuah metode berbasis objek yang memiliki tujuan dan ciri untuk melakukan aktifitas penurunan sifat dari class Parent(induk) nya. Sebagai analogi, “seorang anak pasti akan dapat menurunkan sifat atau karakter dari pribadi sang ayah.”

Contoh implementasi dalam kodingan :

**Simpan dengan nama Inheritance1.java**

```
class Inheritance1
{
    private String na;
    private String nama;
    public void setna (String van)
    {
        na=van;
    }
    public void setnama (String vnama)
    {
        nama=vnama;
    }
    public String getna()
    {
        return (na);
    }
    public String getnama()
    {
        return (nama);
    }
    public void display()
    {
        System.out.println("NPM"+getna());
        System.out.println("Nama"+getnama());
    }
}
```

```
}
```

Simpan dengan nama Inheritance2.java

```
class Inheritance2 extends Inheritance1
```

```
{
```

```
private double ip;
```

```
public void setip (double vip)
```

```
{
```

```
ip=vip;
```

```
}
```

```
public double getip()
```

```
{
```

```
return(ip);
```

```
}
```

```
public void display()
```

```
{
```

```
System.out.println("IP : "+getip());
```

```
System.out.println("Nama : "+getnama());
```

```
System.out.println("NPM : "+getna());
```

```
}
```

```
}
```

Simpan dengan nama InheritanceCoba.java

```
public class InheritanceCoba
```

```
{
```

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

```
{
```

```
Inheritance2 ap1=new Inheritance2();
```

```
ap1.setna("56410268 & 55410416");
```

```
ap1.setnama("Rudy Eriyanto & Pulung Bagaskoro");
```

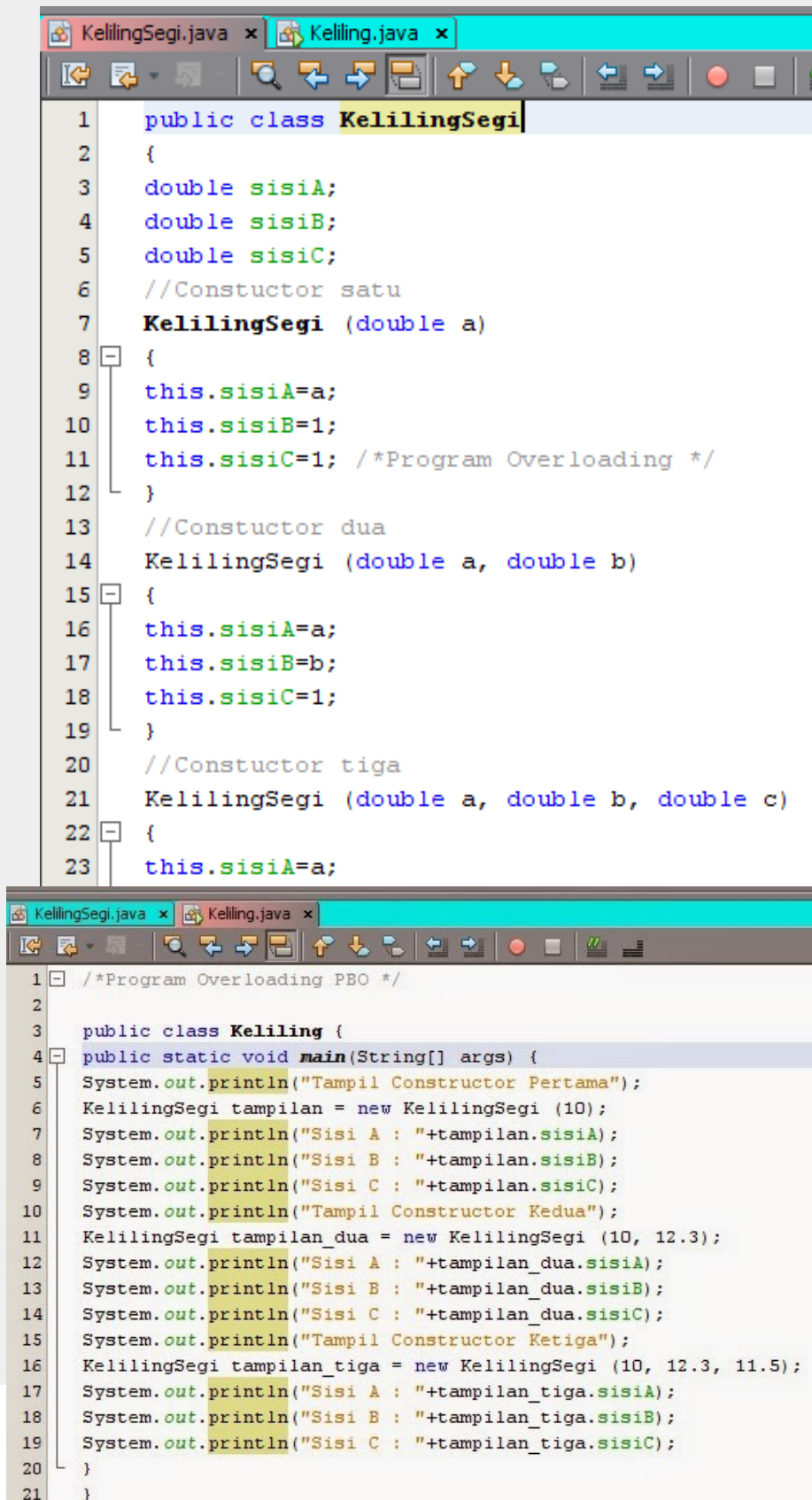
```
ap1.setip(3.5);
```

```
ap1.display();
```

```
}
```

```
}
```

## 2. Polimorphism



```
KelilingSegi.java x Keliling.java x
1 public class KelilingSegi
2 {
3     double sisiA;
4     double sisiB;
5     double sisiC;
6     //Constuctor satu
7     KelilingSegi (double a)
8     {
9         this.sisiA=a;
10        this.sisiB=1;
11        this.sisiC=1; /*Program Overloading */
12    }
13    //Constuctor dua
14    KelilingSegi (double a, double b)
15    {
16        this.sisiA=a;
17        this.sisiB=b;
18        this.sisiC=1;
19    }
20    //Constuctor tiga
21    KelilingSegi (double a, double b, double c)
22    {
23        this.sisiA=a;
24    }
25 }

KelilingSegi.java x Keliling.java x
1 /*Program Overloading PBO */
2
3 public class Keliling {
4     public static void main(String[] args) {
5         System.out.println("Tampil Constructor Pertama");
6         KelilingSegi tampilan = new KelilingSegi (10);
7         System.out.println("Sisi A : "+tampilan.sisiA);
8         System.out.println("Sisi B : "+tampilan.sisiB);
9         System.out.println("Sisi C : "+tampilan.sisiC);
10        System.out.println("Tampil Constructor Kedua");
11        KelilingSegi tampilan_dua = new KelilingSegi (10, 12.3);
12        System.out.println("Sisi A : "+tampilan_dua.sisiA);
13        System.out.println("Sisi B : "+tampilan_dua.sisiB);
14        System.out.println("Sisi C : "+tampilan_dua.sisiC);
15        System.out.println("Tampil Constructor Ketiga");
16        KelilingSegi tampilan_tiga = new KelilingSegi (10, 12.3, 11.5);
17        System.out.println("Sisi A : "+tampilan_tiga.sisiA);
18        System.out.println("Sisi B : "+tampilan_tiga.sisiB);
19        System.out.println("Sisi C : "+tampilan_tiga.sisiC);
20    }
21 }
```

### 3. Encapsulation

```
public class EncapsulationDemo{
    private int ssn;
    private String empName;
    private int empAge;

    //Getter and Setter methods
    public int getEmpSSN(){
        return ssn;
    }

    public String getEmpName(){
        return empName;
    }

    public int getEmpAge(){
        return empAge;
    }

    public void setEmpAge(int newValue){
        empAge = newValue;
    }

    public void setEmpName(String newValue){
        empName = newValue;
    }

    public void setEmpSSN(int newValue){
        ssn = newValue;
    }
}

public class EncapsTest{
    public static void main(String args[]){
        EncapsulationDemo obj = new EncapsulationDemo();
        obj.setEmpName("Mario");
        obj.setEmpAge(32);
        obj.setEmpSSN(112233);
        System.out.println("Employee Name: " + obj.getEmpName());
        System.out.println("Employee SSN: " + obj.getEmpSSN());
        System.out.println("Employee Age: " + obj.getEmpAge());
    }
}
```

## 6. ARRAY 1 / 2 DIMENSI JAVA

Array pada Java merupakan kumpulan atau deretan data yang memiliki tipe data yang sejenis.

Contoh : Array String, char, Integer, Double.

Indeks pada array dimulai dari angka 0.

```
String[] kumpulanString = {"perkenalkan","nama","saya","adalah","lahardi","alkawero"};
```

```
int[] kumpulanInt = {121,2324,300,343,12,50};
```

```
#Cetaklah data dalam kumpulanString
```

```
#Cetaklah data dalam kumpulanInt
```



## 7. Penjumlahan dan Perkalian Matrik

Dua buah matriks A dan B dapat dijumlah dan dikurang jika ordo keduanya sama hasil penjumlahan dan pengurangan matriks A dan B didapat dengan cara menjumlahkan atau mengurangkan unsur-unsur yang seletak.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} + \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a+e & b+f \\ c+g & d+h \end{bmatrix}$$

### Sifat-sifat penjumlahan dan pengurangan Matriks :

(1) Pengurangan dua matriks merupakan penjumlahan dengan matriks lawannya.

$$\text{atau } A - B = A + (-B)$$

(2) Misalkan A, B dan C adalah tiga matriks yang ordonya sama, maka berlaku :

$$A + B = B + A$$

(3) Perkalian suatu bilangan real k dengan matriks A adalah suatu matriks kA yang didapat dengan cara mengalikan setiap unsur matriks A dengan k

$$(A + B) + C = A + (B + C)$$

(4) Matriks nol adalah matriks yang semua elemennya adalah nol (dilambangkan dengan O). Matriks ini adalah matriks identitas penjumlahan, sehingga

$$A + O = O + A = A \quad (b) \quad A + (-A) = O$$

### Sifat-sifat Perkalian Matriks

Operasi perkalian matriks berbeda dengan operasi penjumlahan/pengurangan matriks yang cukup sederhana. Operasi perkalian matriks mempunyai metode tersendiri. Dua matriks dapat dioperasikan dengan perkalian jika banyak kolom matriks pertama sama dengan banyak baris matriks kedua, sedangkan hasil perkalian matriksnya akan memiliki baris yang sama banyak dengan baris matriks pertama dan memiliki kolom yang sama banyak dengan kolom matriks kedua, dapat ditulis sebagai berikut :

$$A_{m \times n} \times B_{n \times r} = (AB)_{m \times r}$$

Metode perkalian dua matriks adalah memasang baris pada matriks pertama dengan kolom pada matriks kedua. Perhatikan metode perkalian matriks berikut ini.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

Perhatikan matriks hasil perkaliannya. Baris1 pada matriks pertama adalah [a b] dan kolom1 pada matriks kedua adalah [e g]. Pasangan ini akan mengisi baris1-kolom1 pada matriks hasil perkaliannya. Memasangkannya adalah dengan menjumlahkan hasil perkalian masing-masing komponen secara berurutan, yaitu menjumlahkan ae dengan bg, ditulis ae+bg. Dengan cara yang sama, akan didapat komponen-komponen lainnya.

Class : Matriks.java

//Try this

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package matriks;

import java.util.Scanner;

/\*\*

\*

\* @author johan

\*/

public class Matriks {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

int A[][] = new int[2][2];

int B[][] = new int[2][2];

int C[][] = new int[2][2];

System.out.println("Masukkan Matriks A");

for (int i = 0; i < 2; i++) {

for (int j = 0; j < 2; j++) {

System.out.print "[" + (i + 1) + "][" + (j + 1) + "]:");

A[i][j] = input.nextInt();

}

}

System.out.println("Masukkan Matriks B");

```

for (int k = 0; k < 2; k++) {
    for (int l = 0; l < 2; l++) {
        System.out.print "[" + (k + 1) + "]" + (l + 1) + ": ";
        B[k][l] = input.nextInt();
    }
}
System.out.println("Matriks A");
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
        System.out.print +(A[i][j]) + " ";
    }
    System.out.println(" ");
}
System.out.println("Matriks B");
for (int k = 0; k < 2; k++) {
    for (int l = 0; l < 2; l++) {
        System.out.print +(B[k][l]) + " ";
    }
    System.out.println(" ");
}
for (int x = 0; x < 2; x++) {
    for (int y = 0; y < 2; y++) {
        C[x][y] = A[x][y] + B[x][y];
    }
}
System.out.println("Hasil penjumlahan Matriks A dan Matriks B");
for (int x = 0; x < 2; x++) {
    for (int y = 0; y < 2; y++) {
        System.out.print +(C[x][y]) + " ";
    }
    System.out.println(" ");
}
}
}

```