MODUL PEMROGRAMAN JAVA 1

CYNTHIA HAYAT SKOM MMSI

UNIVERSITAS KRISTEN KRIDA WACANA
Program Studi Sistem Informasi
TAHUN AKADEMIK GANJIL 2020/2021

Kata Pengantar

Puji Syukur kehadirat Tuhan Yang Maha Esa karena atas berkat dan kasih Karunia-Nya sehingga saya dapat menyelesaikan modul Pemrograman Java 1 ini. Modul ini disusun berdasarkan Kurikulum Program Studi (S1) Sistem Informasi 2017, dibentuk mengacu pada Permenristek Dikti No. 44 Tahun 2015 mengenai Standar Nasional Pendidikan Tinggi (SNPT). Capaian Pembelajaran Lulusan (CPL) yang disusun merujuk pada KKNI (Kualifikasi Kompetensi Nasional Indonesia) level 6, untuk jenjang pendidikan S1.

Modul ini juga dilengkapi dengan latihan soal untuk menguji pemahaman mahasiswa terkait dengan materi yang terdapat pada modul. Terima kasih kepada berbagai pihak yang telah membantu proses penyelesaian modul ini. Semoga modul ini dapat bermanfaat bagi kita semua, khususnya para peserta didik.

Jakarta, 10 Februari 2021

Penyusun

Cynthia Hayat S.Kom., M.MSI

Belajar Java: Memahami Struktur dan Aturan Penulisan Sintaks Java

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

- Setiap bahasa pemrograman memiliki struktur dan aturan penulisan sintaks yang berbeda-beda.
- Java merupakan bahasa pemrograman yang dikembangkan dari bahasa C dan tentunya akan banyak mengikuti gaya penulisan C.

Coba perhatikan program berikut:

```
Package com.petanikode.program; class Program {
public static void main(String args[]){
System.out.println("Hello World"); } }
```

Banyak hal yang kita belum ketahui.
Apa itu package?
Apa itu class?
...dan mengapa ditulis seperti itu?
Oleh sebab itu, kita perlu belajar sintaks dasar dan struktur program Java.
Mari kita mulai...

Struktur Dasar Program Java

Struktur program Java secara umum dibagi menjadi 4 bagian:

1. Deklarasi Package
2. Impor Library
3. Bagian Class
4. Method Main

```java
package com.petanikode.program; //<- 1. deklarasi package

import java.io.File; //<- 2. Impor library

class Program { //<- 3. Bagian class

    public static void main(String args[]){ //<- 4. Method main
        System.out.println("Hello World");
    }

}
```

1. Deklarasi Package

Package merupakan sebuah folder yang berisi sekumpulan program Java.
Deklarasi package biasanya dilakukan saat membuat program atau aplikasi besar.

Contoh deklarasi package:

```
package com.petanikode.program;
```

Biasanya nama package mengikuti nama domain dari sebuah vendor yang mengeluarkan program tersebut.
Pada contoh di atas, com.petanikode adalah nama domain dari petani kode.
Aturannya: nama domain dibalik, lalu diikuti nama programnya.

Bagaimana kalau kita tidak mendeklarasikan *package*? Boleh-boleh saja dan programnya akan tetap bisa jalan. Tapi nanti saat produksi, misalnya saat membuat aplikasi Android. Kita wajib mendeklarasikan *package*.

2. Bagian Impor

Pada bagian ini, kita melakukan impor library yang dibutuhkan pada program.
Library merupakan sekumpulan *class* dan fungsi yang bisa kita gunakan dalam membuat program.

Contoh impor library:

```
import java.util.Scanner;
```

Pada contoh tersebut, kita mengimpor class Scanner dari package java.util

3. Bagian Class

Java merupakan bahasa pemrograman yang menggunakan paradigma OOP *(Object Oriented Programming)*.
Setiap program harus dibungkus di dalam class agar nanti bisa dibuat menjadi objek.
Kalau kamu belum paham apa itu OOP? Cukup pahami class sebagai deklarasi nama program.

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

Ini adalah blok class.
Blok class dibuka dengan tanda kurung kurawal { kemudian ditutup atau diakhiri dengan }.
Di dalam blok class, kita dapat mengisinya dengan method atau fungsi-fungsi dan juga variabel.
Pada contoh di atas, terdapat method main().

4. Method Main

Method `main()` atau fungsi `main()` merupakan blok program yang akan dieksekusi pertama kali.
Ini adalah entri point dari program.
Method `main()` wajib kita buat. Kalau tidak, maka programnya tidak akan bisa dieksekusi.

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

Penulisannya harus seperti ini...
Method `main()` memiliki parameter `args[]`. Parameter ini nanti akan menyimpan sebuah nilai dari argumen di *command line*.
Silahkan baca: Apa Fungsi String[] args pada Pemrograman Java?

Lalu di dalam method main(), terdapat statement atau fungsi:

```
System.out.println("Hello World");
```

Ini adalah fungsi untuk menampilkan teks ke layar monitor.

Statement dan Ekspresi pada Java

Statement dan eksrepsi adalah bagian terkecil dalam program. Setiap statement dan ekspresi di Java, harus diakhiri dengan titik koma (;).

Contoh statemen dan ekspresi:

```
System.out.println("Hello World");
System.out.println("Apa kabar?");
var x = 3;
var y = 8;
var z = x + y;
```

Statemen dan ekspresi akan menjadi instruksi yang akan dikerjakan oleh komputer.
Pada contoh di atas, kita menyuruh komputer untuk menampilkan teks "Hello World", dan "Apa kabar?".
Lalu kita menyuruhnya untuk menghitung nilai x + y.

Blok Program Java

Blok program merupakan kumpulan dari statement dan ekspresi yang dibungkus menjadi satu.
Blok program selalu dibuka dengan kurung kurawal { dan ditutup dengan }.

Contoh blok program:

```
// blok program main
public static void main(String args[]){
    System.out.println("Hello World");
    System.out.println("Hello Kode");

    // blok program if
    if( true ){
        System.out.println('True');
    }

    // blok program for
    for ( int i = 0; i<10; i++){
        System.out.println("Perulangan ke"+i);
    }
}
```

Intinya: jika kamu menemukan kurung { dan }, maka itu adalah sebauh blok program.
Blok program dapat juga berisi blok program yang lain *(nested)*.
Pada contoh di atas, blok program main() berisi blok *if* dan *for*.

Kita akan pelajari blok ini di:

- Percabangan pada Java;
- Perulangan pada Java.

Penulisan Komentar pada Java

Komentar merupakan bagian program yang tidak akan dieksekusi oleh komputer.
Komentar biasanya digunakan untuk:

- Memberi keterangan pada kode program;
- Menonaktifkan fungsi tertentu;
- Membuat dokumentasi;
- dll.

Penulisan komentar pada java, sama seperti pada bahasa C. Yaitu menggunakan:

1. Garis miring ganda (//) untuk komentar satu baris;
2. Garis miring bintang (/*...*/) untuk komentar yang lebih dari satu baris.

Contoh:

```java
public static void main(String args[]){
    // ini adalah komentar satu baris
    System.out.println("Hello World");

    // komentar akan diabaikan oleh komputer
    // berikut ini fungsi yang di-non-aktifkan dengan komentar
    // System.out.println("Hello World");

    /*
    Ini adalah penulisan komentar
    yang lebih dari
    satu baris
    */
}
```

Penulisan String dan Karakter

String merupakan kumpulan dari karakter. Kita sering mengenalnya dengan teks.
Contoh string: "Hello world"

Aturan penulisan string pada Java, harus diapit dengan tanda petik ganda seperti pada contoh di atas.

Apabila diapit dengan tanda petik tunggal, maka akan menjadi sebuah karakter.
Contoh: 'Hello world'.

Jadi harap dibedakan:

- Tanda petik ganda ("...") untuk membuat string;
- Sedangkan tanda petik tunggal ('...') untuk membuat karakter.

Case Sensitive

Java bersifat **Case Sensitive**, artinya huruf besar atau kapital dan huruf kecil dibedakan.

Contoh:

```
String nama = "Petani Kode";
String Nama = "petanikode";
String NAMA = "Petanikode.com";

System.out.println(nama);
System.out.println(Nama);
System.out.println(NAMA);
```

Tiga variabel tersebut merupakan tiga variabel yang bebeda, mesikipun sama-sama bernama nama. Banyak pemula yang sering salah pada hal ini. Karena tidak bisa membedakan mana variabel yang menggunakan huruf besar dan mana yang menggunakan huruf kecil.

Apabila kita membuat variabel seperti ini:

```
String jenisKelamin = "Laki-laki";
```

Maka kita harus memanggilnya seperti ini:

```
System.out.println(jenisKelamin);
```

Bukan seperti ini:

```
System.out.println(jeniskelamin);
```

Perhatikan, huruf `K` adalah huruf kapital.

Gaya Penulisan Case

Gaya penulisan case (case style) yang digunakan oleh Java adalah: **camelCase**, **PascalCase**, dan **ALL UPPER**.
Gaya penulisan *camelCase* digunakan pada nama variabel, nama objek, dan nama method.

Contoh:

```
String namaSaya = "Dian";
```

Lalu untuk *PascalCase* digunakan pada penulisan nama class.

Contoh:

```
class HelloWOrld {
    //...
}
```

Perhatikan nama class-nya, kita menggunakan huruf kapital di awal, dan huruf kapital pada huruf W untuk memisahkan dua suku kata.

Sedangkan *camelCase* huruf depannya menggunakan huruf kecil, dan awalan suku kata berikutnya menggunakan huruf besar.

```
// ini camelCase
belajarJava

// ini PascalCase
BelajarJava
```

Lalu, penulisan *ALL UPPER* atau semaunya kapital digunakan pada pembuatan nama konstanta. Untuk penulisan dua suku kata atau lebih, *ALL UPPER* dipisah dengan garus bawah atau *underscore* (_).

Contoh:

```
public final String DB_NAME = "petanikode";
```

Apakah boleh saya menulis sembarangan?
Misal untuk nama class menggunakan *ALL UPPER*?
Boleh-boleh saja, programnya tidak akan error. Tetapi kode program yang kamu tulis akan terlihat kotor dan keluar dari garis pandu yang sudah ditetapkan.

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Pemrograman Java: Variabel dan Tipe Data

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Variabel adalah tempat menyimpan nilai sementara.
Itu saja.
Serius?
Iya serius, kalau mau yang lebih panjang...

...coba kita lihat pengertiannya dari Wikipedia:
Variabel: (Lat) 1. berubah-ubah, tidak tetap; 2. deklarasi sesuatu yang memiliki variasi nilai 3. berbeda-beda dalam bahasa pemrograman disebut juga simbol yang mewakili nilai tertentu, variabel yang dikenal di sub program disebut variabel lokal. sedang yang di kenal secara umum/utuh dalam satu program disebut variabel global.

Nah, ngerti gak?
Kalau tidak, cukuplah pahami: Variabel sebagai tempat menyimpan nilai sementara.

Lalu, apa itu tipe data?
Tipe data adalah jenis data yang tersimpan dalam variabel.

Macam-macam Tipe Data

Berikut ini macam-macam tipe data pada Java:

- char: Tipe data karakter, contoh Z
- int: angka atau bilangan bulat, contoh 29
- float: bilangan desimal, contoh 2.1
- double: bilangan desimal juga, tapi lebih besar kapasistanya, contoh 2.1
- String: kumpulan dari karakter yang membentuk teks, contoh Hello World!
- boolean: tipe data yang hanya bernilai true dan false

Membuat Variabel

Hal yang perlu diketahui dalam pembuatan variabel di java adalah cara penulisannya.

Formatnya seperti ini:

```
<tipe data> namaVariabel;
```

Contoh:

Membuat variabel kosong bertipe integer:

```
int namaVariabel;
```

Membuat variabel bertipe integer dan langsung diisi nilai:

```
int namaVariabel = 19;
```

Membuat sekumpulan variabel yang tipe datanya sama:

```
int a, b, c;
```

Lalu, dimana itu ditulis?

1. Di dalam fungsi main(): variabel yang ditulis di dalam fungsi main() dan fungsi yang lainnya disebut variabel lokal.
2. Di dalam class: Variabel ini disebut variabel class atau global.

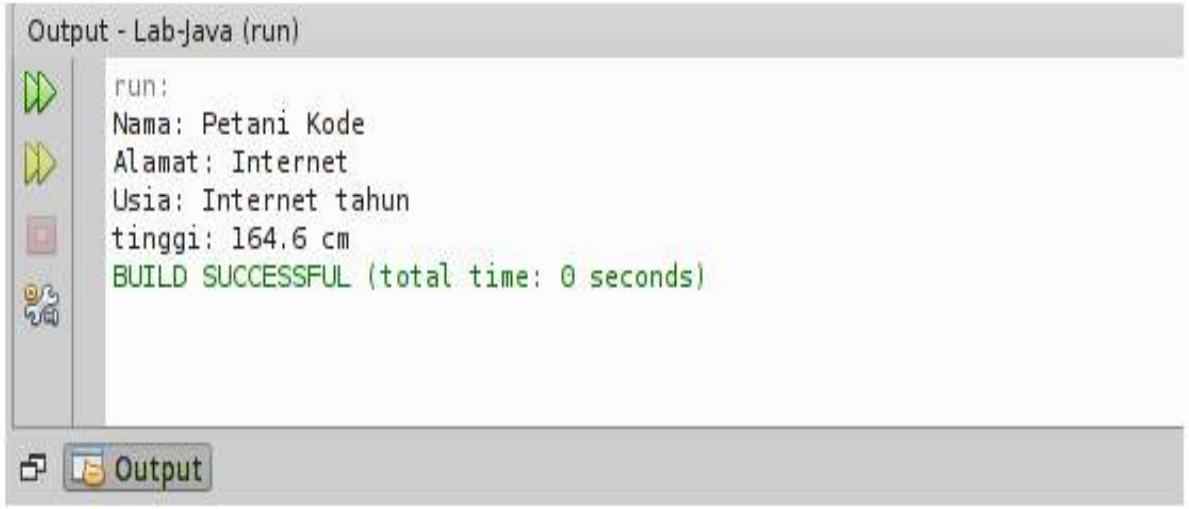
Mari Kita Coba Latihan dengan Membuat Program DataDiri

DataDiri.java

```java
package pertemuan1;

public class DataDiri {

    public static void main(String[] args) {
        // membuat variabel
        String nama, alamat;
        int usia;
        double tinggi;

        // mengisi variabel
        nama = "Petani Kode";
        alamat = "Internet";
        usia = 20;
        tinggi = 164.6;

        // mencetak ke layar isi variabel
        System.out.println("Nama: " + nama);
        System.out.println("Alamat: " + alamat);
        System.out.println("Usia: " + alamat + " tahun");
        System.out.println("tinggi: " + tinggi + " cm");
    }

}
```

pertemuan1.DataDiri > main

Setelah itu, coba jalankan dengan menekan tombol [Shift]+[F6]. Analisa dan pahami maksud kode-kode di atas.



Hasil outputnya sepertinya salah.
Usia kok nilianya alamat, mengapa demikian?
Itu disebabkan karena kita memanggil variabel alamat pada baris *usia*.
Jadi yang akan tampil adalah isi dari variable alamat, bukan isi dari variabel usia.

Perbaikilah kodenya...
Nilai plus untuk yang berhasil pertama kali.

Aturan Penulisan Variabel

Ternyata tidak boleh sembarangan dalam membuat variabel.
Ada aturan yang harus diikuit, diantaranya:

- Nama variabel tidak boleh menggunakan kata kunci dari Java *(reserved word)* seperti if, for, switch, dll.
- Nama variabel boleh menggunakan huruf, angka (0-9), garis bawah *(underscore)*, dan symbol dollar ($), namun penggunaan garis bawah dan symbol lebih baik dihindari.
- Nama variabel harus diawali dengan huruf kecil, karena Java menggunakan gaya CamelCase.
- Apabila nama variabel lebih dari 1 suku kata, maka kata ke-2 dituliskan dengan diawali dengan huruf besar dan seterusnya, contoh namaVariabel.

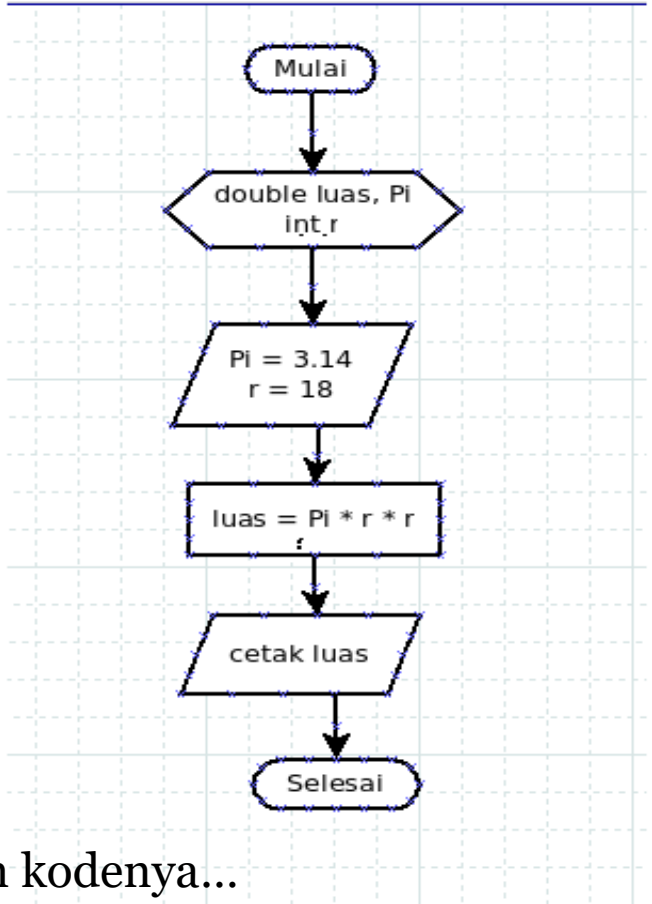
Latihan Ke-2: Program LuasLingkaran

Mari kita mantapkan pemahaman dengan membuat program *LuasLingkaran*.
Program ini fungsinya untuk menghitung luas lingkaran.
Luas lingkaran dapat kita hitung dengan rumus $PI \times r^2$.
Sebelum memulai memrogram, sebaiknya kita pahami dulu algoritma dan flowchart-nya:

Algoritma

```
Deklarasi:
    Double luas, PI
    int r
Deskripsi:
    - Input
        PI = 3.14
        r = 18
    - Proses
        luas = PI * r * r
    - Output
        cetak luas
```

Flow Chart:



Buatlah kodenya...
Nilai plus untuk yang berhasil pertama kali.

Konversi Tipe Data

Konversi artinya merubah ke jenis yang lain.
Kenapa kita perlu konversi tipe data?
Untuk menjawabnya, saya ingin tunjukan ilustrasi disamping :

Air yang bentuknya cair tidak bisa disimpan dalam kardus. Karena itu, air harus konversi dulu menjadi bentuk padat (es) agar bisa disimpan dalam kardus.
Begitu juga dengan variabel.
Tipe data *string* tidak akan bisa disimpan dalam variabel dengan tipe *integer*.

Cara Konversi Tipe Data

Cara 1:

```
Integer.perseInt(variabel);
Integer.perseInt(1.2);
```

Cara 2:

```
Integer.valueOf(2.1);
```

Cara 3:

```
objek.toInt();
```

Cara 4:

```
(int) variabel;
(int) 2.1;
```

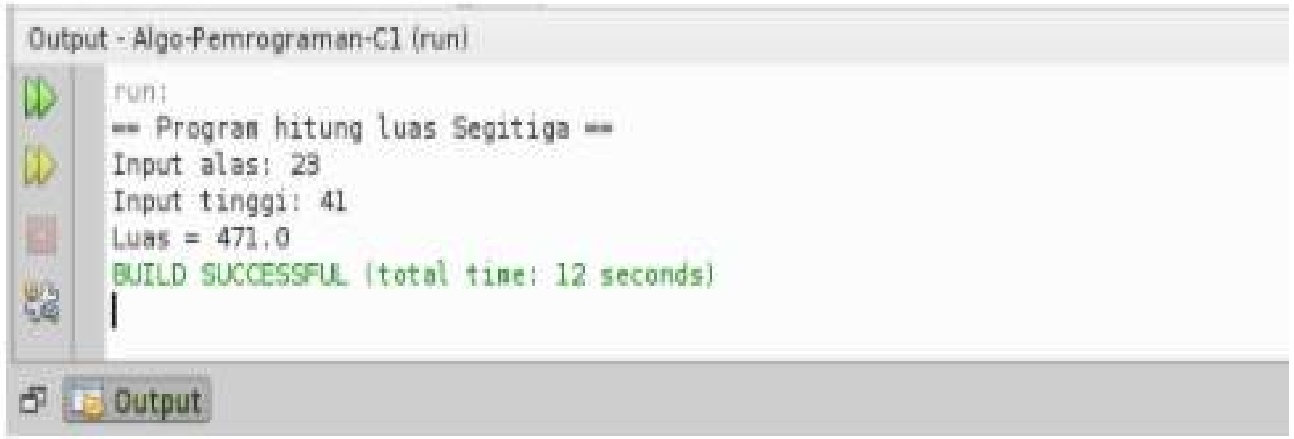
Mari Kita Coba dalam Program...

Kemudian ikuti kode program berikut:

```
package pertemuan2;

import java.util.Scanner;

public class LuasSegitia {

    public static void main(String[] args) {
        // deklarasi
        Double luas;
        int alas, tinggi;

        // mebuat scanner baru
        Scanner baca = new Scanner(System.in);

        // Input
        System.out.println("== Program hitung luas Segitiga ==");
        System.out.print("Input alas: ");
        alas = baca.nextInt();
        System.out.print("Input tinggi: ");
        tinggi = baca.nextInt();

        // proses
        luas = Double.valueOf((alas * tinggi) / 2);

        // output
        System.out.println("Luas = " + luas);
    }

}
```

Variabel luas bertipe data *Double*, berarti nilai yang bisa disimpan adalah *Double*. Sedangkan variabel alas dan tinggi bertipe *Integer*.
Agar hasil operasi data *integer* dapat disimpan dalam variabel bertipe double, maka perlu dikonversi.

Kalau dijalankan, akan menghasilkan:



Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Java: 6 Jenis Operator yang Harus Dipahami

CYNTHIA HAYAT S.KOM., M.MSI

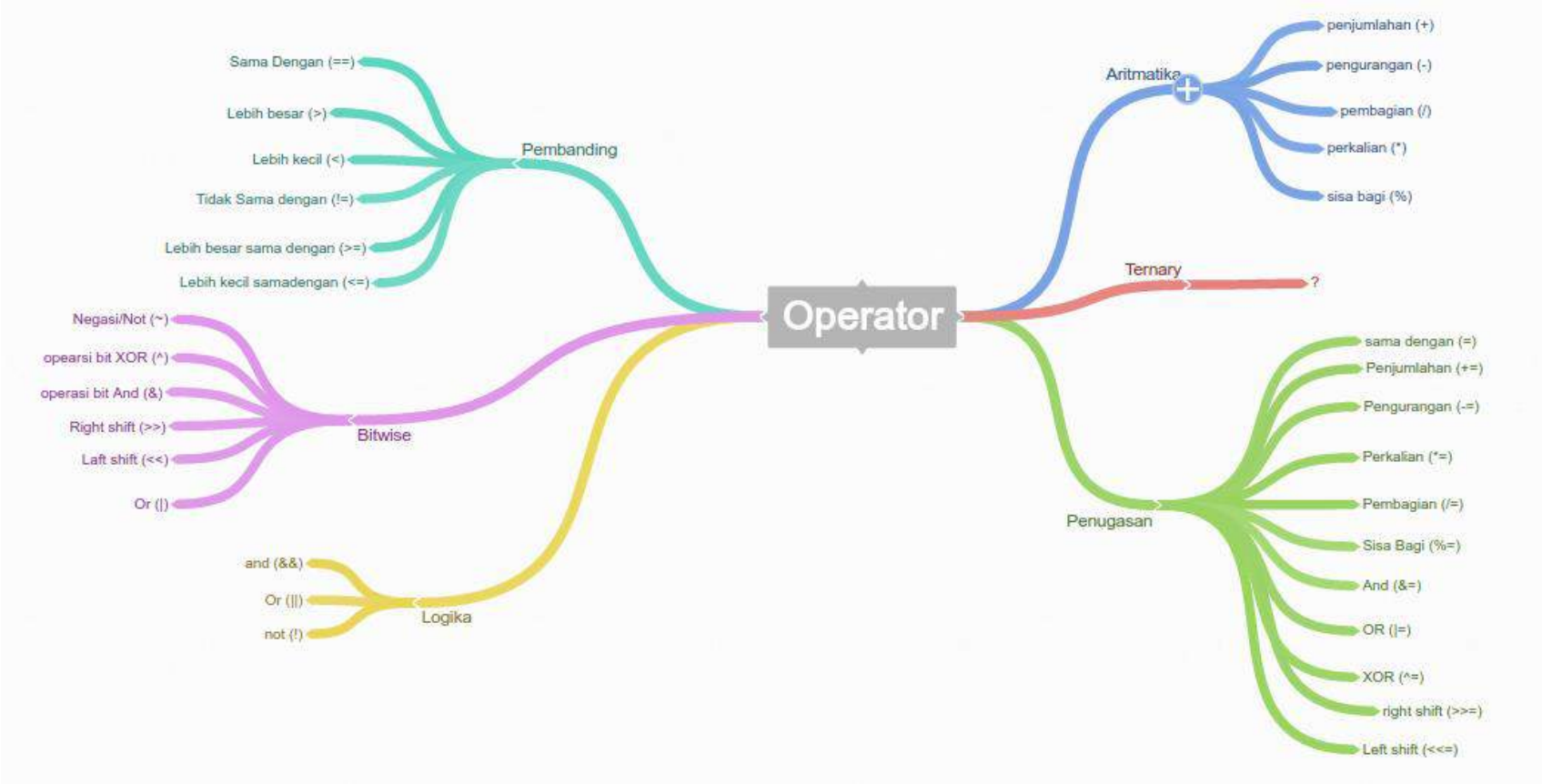
KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Operator dalam pemrograman digunakan untuk melakukan operasi tertentu.
Misalkan kita ingin menjumlahkan nilai dari variabel x dan y, maka kita bisa menggunakan operator penjumlahan (+).

Ada enam jenis kelompok operator dalam pemrograman
Java:

1. Operator Artimatika;
2. Operator Penugasan;
3. Operator Pembanding;
4. Operator Logika;
5. Operator Bitwise;
6. dan Operator Ternary.



2. Operator Aritmatika

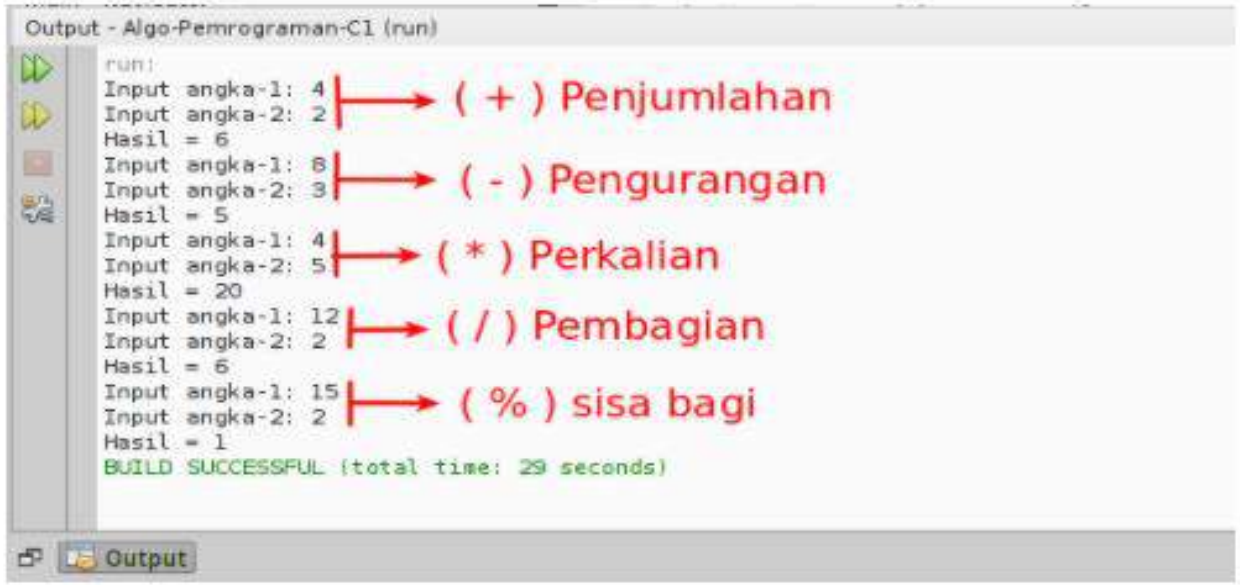
Operator aritmatika digunakan untuk melakukan operasi aritmatika.
Operator ini terdiri dari:

Nama	Simbol
Penjumlahan	+
Pengurangan	-
Perkalian	*
Pembagian	/
Sisa Bagi	%

Exercise 1 :

```java
import java.util.Scanner;

public class OperatorAritmatika {

    public static void main(String[] args) {
        int angka1;
        int angka2;
        int hasil;

        Scanner keyboard = new Scanner(System.in);

        System.out.print("Input angka-1: ");
        angka1 = keyboard.nextInt();
        System.out.print("Input angka-2: ");
        angka2 = keyboard.nextInt();

        // penjumlahan
        hasil = angka1 + angka2;
        System.out.println("Hasil = " + hasil);

        System.out.print("Input angka-1: ");
        angka1 = keyboard.nextInt();
        System.out.print("Input angka-2: ");
        angka2 = keyboard.nextInt();

        // pengurangan
        hasil = angka1 - angka2;
        System.out.println("Hasil = " + hasil);

         System.out.print("Input angka-1: ");
        angka1 = keyboard.nextInt();
        System.out.print("Input angka-2: ");
        angka2 = keyboard.nextInt();

        // perkalian
        hasil = angka1 * angka2;
        System.out.println("Hasil = " + hasil);


        System.out.print("Input angka-1: ");
        angka1 = keyboard.nextInt();
        System.out.print("Input angka-2: ");
        angka2 = keyboard.nextInt();

        // Pembagian
        hasil = angka1 / angka2;
        System.out.println("Hasil = " + hasil);

        System.out.print("Input angka-1: ");
        angka1 = keyboard.nextInt();
        System.out.print("Input angka-2: ");
        angka2 = keyboard.nextInt();

        // Sisa Bagi
        hasil = angka1 % angka2;
        System.out.println("Hasil = " + hasil);

    }

}
```

Silahkan jalankan programnya:



2. Operator Penugasan

Operator penugasan *(Assignment Operator)* fungsinya untuk memberikan tugas pada variabel tertentu. Biasanya untuk mengisi nilai.

Contoh:

```
int a = 10;
```

Variabel `a` ditugaskan untuk menyimpan nilai `10`.

Operator Penugasan terdiri dari:

Nama Operator	Sombol
Pengisian Nilai	`=`
Pengisian dan Penambahan	`+=`
Pengisian dan Pengurangan	`-=`
Pengisian dan Perkalian	`*=`
Pengisian dan Pembagian	`/=`
Pengisian dan Sisa bagi	`%=`

Exercise 2 :

```java
public class OperatorPenugasan {

    public static void main(String[] args) {
        int a;
        int b;

        // Pengisian nilai
        a = 5;
        b = 10;

        // penambahan
        b += a;
        // sekarang b = 15
        System.out.println("Penambahan : " + b);

        // pengurangan
        b -= a;
        // sekarang b = 10 (karena 15-5)
        System.out.println("Pengurangan : " + b);

        // perkalian
        b *= a;
        // sekarang b = 50 (karena 10*5)
        System.out.println("Perkalian : " + b);

        // Pembagian
        b /= a;
        // sekarang b=10
        System.out.println("Pembagian : " + b);

        // Sisa bagi
        b %= a;
        // sekarang b=0
        System.out.println("Sisa Bagi: " + b);

    }

}
```

Hasil outputnya:

3. Operator Pambanding

Sepeti namanya, tugas oprator ini untuk membandingkan.
Operator ini juga dikenal dengan opeartor relasi.
Nilai yang dihasilkan dari operator ini berupa boolean, yaitu: true dan false.

Operator ini terdiri dari:

Nama	Simbol
Lebih Besar	>
Lebih Kecil	<
Sama Dengan	==
Tidak Sama dengan	!=
Lebih Besar Sama dengan	>=
Lebih Kecil Sama dengan	<=

contoh:

```
boolean x = 10 < 12
```

Maka x akan bernilai *true*, karena 10 lebih kecil dari 12.

Exercise 3 :

```java
public class OperatorPembanding {

    public static void main(String[] args) {
        int nilaiA = 12;
        int nilaiB = 4;
        boolean hasil;

        // apakah A lebih besar dari B?
        hasil = nilaiA > nilaiB;
        System.out.println(hasil);

        // apakah A lebih kecil dari B?
        hasil = nilaiA < nilaiB;
        System.out.println(hasil);

        // apakah A lebih besar samadengan B?
        hasil = nilaiA >= nilaiB;
        System.out.println(hasil);

        // apakah A lebih kecil samadengan B?
        hasil = nilaiA <= nilaiB;
        System.out.println(hasil);

        // apakah nilai A sama dengan B?
        hasil = nilaiA == nilaiB;
        System.out.println(hasil);

        // apakah nilai A tidak samadengan B?
        hasil = nilaiA != nilaiB;
        System.out.println(hasil);

    }

}
```

Kode program di atas, akan menghasilkan output seperti ini:

4. Operator Logika

Kalau kamu pernah belajar logika matematika, pasti tidak akan asing dengan operator ini.

Nama	Simbol di Java
Logika AND	&&
Logika OR	||
Negasi/kebalikan	!

Operator Logika digunakan untuk membuat operasi logika.
Misalnya seperti ini:
•Pernyataan 1: Petani Kode seorang programmer
•Pernyattan 2: Petanikode menggunakan Linux

Jika ditanya, apakah Petani Kode programmer yang menggunakan Linux?
Tentu kita akan cek dulu kebenarannya
•Pernyataan 1: Petani Kode seorang programmer = true.
•Pernyataan 2: Petanikode menggunakan Linux = true.

Apa petanikode programmer dan menggunakan Linux?

```
Pertnyataan 1 && Pernyataan 2 = true
```

Bingung?

Coba cek lagi tabel kebenaran untuk logika AND.

Pernyatan 1	Pernytaan 2	Logika AND
true	true	true
true	false	false
false	true	false
false	false	false

Exercise 4 :

Buatlah sebuah kelas baru beranama *OperatorLogika*. Kemudian ikuti kode berikut ini:

```
1  package pertemuan3;
2
3  public class OperatorLogika {
4      public static void main(String[] args) {
5          boolean a = true;
6          boolean b = false;
7          boolean c;
8
9          // konjungsi (dan)
10         c = a && b;
11         System.out.println("true && false = "+c);
12
13         // disjungsi (atau)
14         c = a || b ;
15         System.out.println("true || false = "+c);
16
17         // negasi (bukan)
18         System.out.println("Negasi, !true = " + !a);
19     }
20
21 }
```

Silahkan jalankan dan perhatikan hasilnya:

5. Operator Bitwise

Operator bitwise merupkan operator yang digunakan untuk operasi bit (biner). Operator bitwise terdiri dari:

Operator ini berlaku untuk tipe data int, long, short, char, dan byte.

Operator ini akan menghitung dari bit-ke-bit.
Misalnya, kita punya variabel a = 60 dan b = 13.

Nama	Simbol di Java
AND	&
OR	|
XOR	^
Negasi/kebalikan	~
Left Shift	<<
Right Shift	>>
Left Shift (unsigned)	<<<
Right Shift (unsigned)	>>>

Bila dibuat dalam bentuk biner, akan menjadi seperti ini:

```
a = 00111100
b = 00001101
```

Kemudian, dilakukan operasi bitwise

Operasi AND

```
a = 00111100
b = 00001101
a & b = 00001100
```

Operasi OR

```
a = 00111100
b = 00001101
a | b = 00111101
```

Operasi XOR

```
a = 00111100
b = 00001101
a ^ b = 00110001
```

Opearsi NOT (Negasi/kebalikan)

```
a = 00111100
~a  = 11000011
```

Konsepnya memang hampir sama dengan operator Logika. Bedanya, Bitwise digunakan untuk biner.

Exercise 5 :

Buat kelas baru dengan nama `OperatorBitwise`, kemudian ikuti isinya sebagai berikut.

```java
public class OperatorBitwise {

    public static void main(String[] args) {
        int a = 60;     /* 60 = 0011 1100 */
        int b = 13;     /* 13 = 0000 1101 */
        int c = 0;

        c = a & b;       /* 12 = 0000 1100 */
        System.out.println("a & b = " + c);

        c = a | b;       /* 61 = 0011 1101 */
        System.out.println("a | b = " + c);

        c = a ^ b;       /* 49 = 0011 0001 */
        System.out.println("a ^ b = " + c);

        c = ~a;          /*-61 = 1100 0011 */
        System.out.println("~a = " + c);

        c = a << 2;      /* 240 = 1111 0000 */
        System.out.println("a << 2 = " + c);

        c = a >> 2;      /* 215 = 1111 */
        System.out.println("a >> 2  = " + c);

        c = a >>> 2;     /* 215 = 0000 1111 */
        System.out.println("a >>> 2 = " + c);
    }

}
```

Perhatikanlah hasil outputnya:

6. Operator Ternary

Operator ini unik, seperti membuat pertanyaan.
Simbolnya menggunakan tanda tanya (?) dan titik-dua (:) untuk memisah jawabannya.

Pada contoh di samping, “Kamu suka aku” adalah pertanyaan atau kondisi yang akan diperiksa.
Kalau jawabannya benar, maka iya. Sebaliknya akan tidak.

Exercise 6 :

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

        boolean suka = true;
        String jawaban;

        // menggunakan operator ternary
        jawaban = suka ? "iya" : "tidak";

        // menampilkan jawaban
        System.out.println(jawaban);

    }
}
```

Hasil outputnya:

```
iya
```

Sekarang coba ganti nilai variabel `suka` menjadi `false`, lalu dijalankan lagi.

Pasti akan menghasilkan `tidak`.

Cara lain, dapat juga membuat kondisi seperti ini:

```java
int suka = 8;

String jawaban = (suka > 5) ? "iya" : "tidak";
```

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Java: Memahami 3 Bentuk Percabangan dalam Java

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Misalnya seperti ini:

Kalau kita perhatikan, alur pengeksekusian sebuah kode program dikerjakan satu per satu dari atas sampai ke bawah.
Baris demi baris dibaca, kemudian komputer mengerjakan apa yang diperintahkan.

Alur programnya satu, tidak ada belokan atau percabangan.

Percabangan hanyalah sebuah istilah yang digunakan untuk menyebut alur program yang bercabang.
Percabangan juga dikenal dengan "Control Flow", "Struktur Kondisi", "Struktur IF", "Decision", dsb. Semuanya itu sama.
Pada diagram alur *(Flow Chart)* seperti di atas, alurnya memang satu.
Tapi setelah kita menggunakan percabangan, alurnya akan bertambah menjadi seperti ini.

Lalu bagaimana cara menulis kode percabangan dalam Java?
Caranya: menggunakan kata kunci `if`, `else`, `switch`, dan `case`, dan operator ternary.

Contoh format stuktur IF seperti ini:

```
if( suatu_kondisi ) {
    // lakukan sesuatu kalau kondisi benar
    // Lakukan ini juga
}
```

`suatu_kondisi` hanya bernilai `true`/`false` saja. Kita bisa gunakan operator relasi dan logika di sini.

Untuk lebih jelasnya, nanti akan kita bahas.

Sebelumnya, kamu perlu tahu dulu tiga bentuk percabangan pada Java:

1. Percabangan IF
2. Percabangan IF/ELSE
3. Percabangan IF/ELSE/IF atau SWITCH/CASE

1. Percabangan IF

Percabangan ini hanya memiliki satu pilihan. Artinya, pilihan di dalam IF hanya akan dikerjakan kalau kondisinya benar.

Tapi kalau salah... tidak akan melakukan apa-apa. Alias lanjut eksekusi ke perintah berikutnya.

Contoh:
Pernahkah kalian belanja di toko, kemudian kalau belanja di atas sekian ribu dapat hadiah atau diskon.
Nah! Contoh kasus seperti itu, dapat kita selesaikan dengan menggunakan percabangan ini.
untuk lebih jelasnya, mari kita Latihan..

Exercise 1 :
Mari Kita Membuat Program Hadiah

Misalkan ada sebuah toko buku. Mereka memberikan hadiah berupa perlengkapan sekolah kepada pembeli yang belanja di atas Rp 100.000.

Jalankan programnya dan perhatikanlah hasilnya.

```
run:
Total Belanjaan: Rp 120000
Selamat, anda mendapatkan hadiah!
Terima kasih...
BUILD SUCCESSFUL (total time: 5 seconds)
```

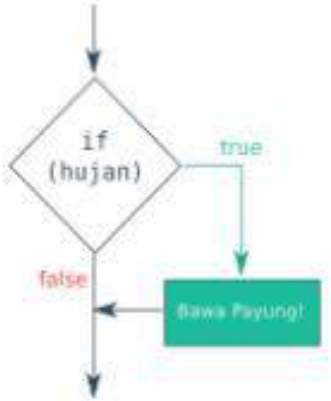
Cobalah untuk memberikan nilai di bawah `100000` dan perhatikan apa akan yang terjadi.

Maka programnya bisa kita buat seperti ini:

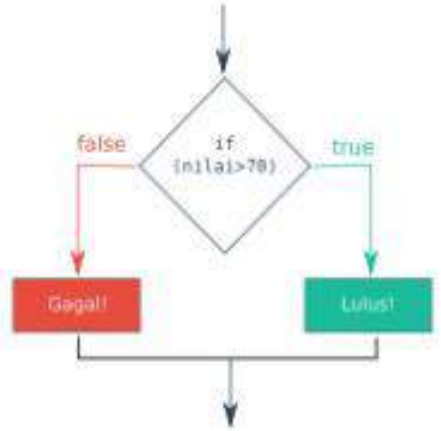
```java
import java.util.Scanner;

public class Hadiah {

    public static void main(String[] args) {

        // membuat variabel belanja dan scanner
        int belanja = 0;
        Scanner scan = new Scanner(System.in);

        // mengambil input
        System.out.print("Total Belanjaan: Rp ");
        belanja = scan.nextInt();

        // cek apakah dia belanja di atas 100000
        if ( belanja > 100000 ) {
            System.out.println("Selamat, anda mendapatkan hadiah!");
        }

        System.out.println("Terima kasih...");

    }

}
```

2. Percabangan IF/ELSE

Sedangkan percabangan IF/ELSE memiliki pilihan alternatif kalau kondisinya salah.
IF: “Jika kondisi benar maka kerjakan ini, kalau tidak silahkan lanjut”
IF/ESLE: “Jika kondisi benar maka kerjakan ini, kalau salah maka kerjakan yang itu, setelah itu lanjut”




```
if (hujan) {
    System.out.println("Bawa Payung!");
}
```




```
if (nilai > 70) {
    System.out.println("Lulus!");
} else {
    System.out.println("Gagal!");
}
```

Exercise 2 :
Program Cek Kelulusan

Misalkan, kalau nilai siswa lebih besar dari 70, maka ia dinyatakan lulus. Kalau tidak, maka dia gagal.

Hasil outputnya:

```
run:
Nama: Petani Kode
Nilai: 89
Selemat Petani Kode, anda lulus!
BUILD SUCCESSFUL (total time: 6 seconds)
```

Cobalah untuk merubah nilai yang dimasukkan dan perhatikan apa yang akan terjadi.

Programnya bisa kita buat seperti ini:

```java
import java.util.Scanner;

public class CekKelulusan {

    public static void main(String[] args) {

        // membuat variabel dan Scanner
        int nilai;
        String nama;
        Scanner scan = new Scanner(System.in);

        // mengambil input
        System.out.print("Nama: ");
        nama = scan.nextLine();
        System.out.print("Nilai: ");
        nilai = scan.nextInt();

        // cek apakah dia lulus atau tidak
        if( nilai >= 70 ) {
            System.out.println("Selemat " + nama + ", anda lulus!");
        } else {
            System.out.println("Maaf " + nama + ", anda gagal");
        }

    }

}
```

Percabangan IF/ELSE dengan Operator Ternary

Selain menggunakan struktur seperti di atas, percahangan ini juga dapat menggunakan operator ternary.
Seperti yang sudah kita pelajari pada pembahasan tentang operator. Operator ternary memiliki konsep yang sama seperti percabganan IF/ELSE.

Exercise 3 :

Contoh programnya:

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

        boolean suka = true;
        String jawaban;

        // menggunakan operator ternary
        jawaban = suka ? "iya" : "tidak";

        // menampilkan jawaban
        System.out.println(jawaban);

    }
}
```

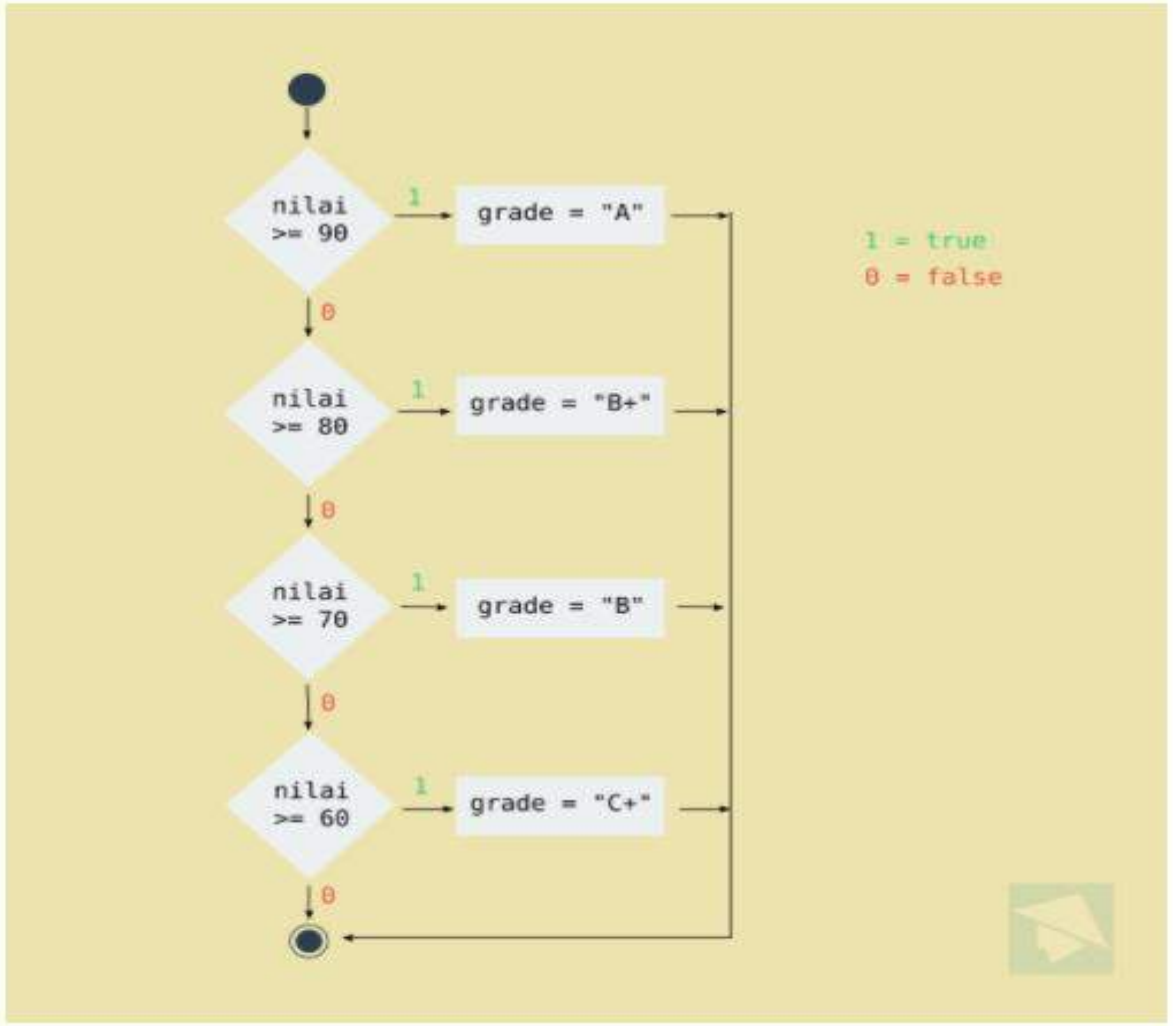
3. Percabangan IF/ELSE/IF dan SWITCH/CASE

Jika percabangan IF/ESLE hanya memiliki dua pilihan saja. Maka percahangan IF/ELSE/IF memiliki lebih dari dua pilihan.

Formatnya seperti ini:

```
if (suatu kondisi) {
    // maka kerjakan ini
    // kerjakan perintah ini juga
    // …
} else if (kondisi lain) {
    // kerjakan ini
    // kerjakan ini juga
    // …
} else if (kondisi yang lain lagi) {
    // kerjakan perintah ini
    // kerjakan ini juga
    // …
} esle {
    // kerjakan ini kalau
    // semua kondisi di atas
    // tidak ada yang benar
    // …
}
```

Coba perhatikan contohnya:



Jika nilainya lebih besar dari 90, maka grade-nya “A”. Sedangkan kalau lebih besar dari 80, maka “B+”. Lebih besar dari 70, maka “B”, dan seterusnya.

Exercise 4 :
Program HitungGrade

Misalkan, kalau nilai siswa lebih besar dari 70, maka ia dinyatakan lulus. Kalau tidak, maka dia gagal.

```java
import java.util.Scanner;

public class HitungGrade {
    public static void main(String[] args) {

        // membuat variabel dan scanner
        int nilai;
        String grade;
        Scanner scan = new Scanner(System.in);

        // mengambil input
        System.out.print("Inputkan nilai: ");
        nilai = scan.nextInt();

        // higung gradenya
        if ( nilai >= 90 ) {
            grade = "A";
        } else if ( nilai >= 80 ){
            grade = "B+";
        } else if ( nilai >= 70 ){
            grade = "B";
        } else if ( nilai >= 60 ){
            grade = "C+";
        } else if ( nilai >= 50 ){
            grade = "C";
        } else if ( nilai >= 40 ){
            grade = "D";
        } else {
            grade = "E";
        }

        // cetak hasilnya
        System.out.println("Grade: " + grade);
    }
}
```

Hasil outputnya:

```
run:
Inputkan nilai: 88
Grade: B+
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Percabangan SWITCH/CASE

Percabangan SWITCH/CASE sebenarnya adalah bentuk lain dari IF/ELSE/IF. Bedanya, percabangan ini menggunakan kata kunci `switch` dan `case`.

Formatnya juga berbeda, tapi cara kerjanya sama.

```
switch(variabel){
    case 1:
        // kerjakan kode ini
        // kode ini juga
        break;
    case 2:
        // kerjakan kode ini
        // kode ini juga
        break;
    case 3:
        // kerjakan kode ini
        // kode ini juga
        break;
    default:
        // kerjakan kode ini
        // kode ini juga
        break;
}
```

Perhatikan: `case 1` artinya nilai `variabel` yang akan dibandingkan, apakah nilainya sama dengan `1` atau tidak.

Kalau iya, maka kerjakan kode yang ada di dalam `case 1`.

Bisa juga betuknya berbeda, misalnya seperti ini:

```
switch (variabel) {
    case 'A':
        // lakukan sesuatu
        break;
    case 'B':
        // lakukan ini
        break;
    default:
        // lakukan ini
}
```

Perlu diperhatikan juga: di sana ada kata kunci break dan default.

- break artinya berhenti. Ini untuk memerintahkan komputer untuk berhenti mengecek case yang lainnya.
- default artinya jika nilai variabel tidak ada yang sama dengan pilihan case di atas, maka kerjakan kode yang ada di dalam default.

Pilihan default bisa juga tidak memiliki break, karena dia adalah pilihan terakhir. Artinya pengecekan akan berakhir di situ.

Exercise 5 :

Contoh program dengan percabangan SWITCH/CASE

```java
import java.util.Scanner;

public class LampuLalulintas {
    public static void main(String[] args) {

        // membuat variabel dan Scanner
        String lampu;
        Scanner scan = new Scanner(System.in);

        // mengambil input
        System.out.print("Inputkan nama warna: ");
        lampu = scan.nextLine();

        switch(lampu){
            case "merah":
                System.out.println("Lampu merah, berhenti!");
                break;
            case "kuning":
                System.out.println("Lampu kuning, harap hati-hati!");
                break;
            case "hijau":
                System.out.println("Lampu hijau, silahkan jalan!");
                break;
            default:
                System.out.println("Warna lampu salah!");
        }
    }
}
```

Hasil outputnya:

```
run:
Inputkan nama warna: hijau
Lampu hijau, silahkan jalan!
BUILD SUCCESSFUL (total time: 3 seconds)
```

Eksperimen: Cobalah untuk menghilangkan `break` di salah satu `case` dan perhatikanlah hasilnya.

Percabangan dalam Percabangan *(Nested)*

Jadi, percabangan itu bisa dibuat di dalam percabangan. Kadang teknik ini disebut juga *nested if*.
Contoh kasus:
Misalnya ada model bisinis seperti ini di sebuah toko. Ketika orang membayar di kasir, biasanya ditanya ada kartu member untuk mendapatkan diskon dan sebagainya.

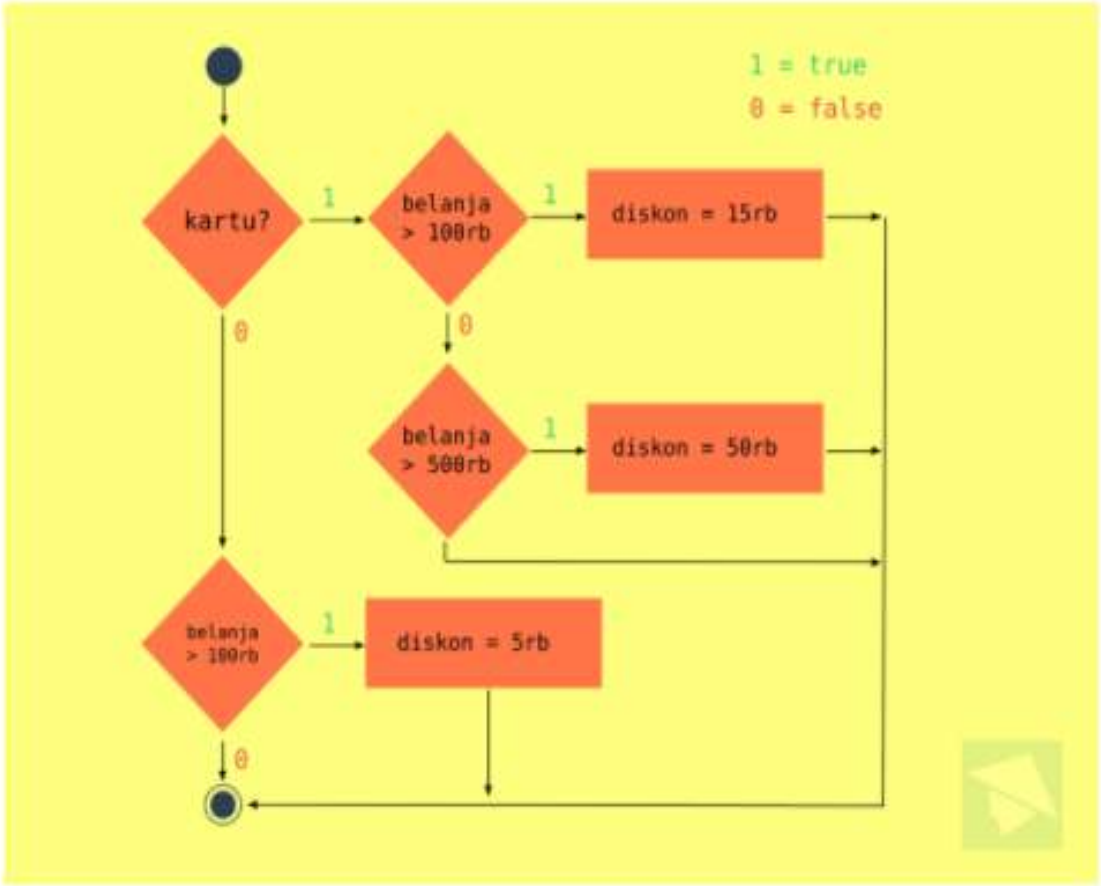
```
Apakah anda punya kartu member?
   - ya
      * Apakah belanjaan anda lebih dari 500rb?
         # ya : mendapatkan diskon 50rb
         # tidak : tidak mendapatkan diskon
      * Apakah belanjaan anda lebih dari 100rb?
         # ya : mendapatkan diskon 15rb
         # tidak: tidak mendapatkan diskon
   - tidak
      * Apakah belanjaan anda lebih dari 100rb?
         # ya : mendapatkan diskon 10rb
         # tidak: tidak mendapatkan diskon
```

Percabangan dalam Percabangan *(Nested)*

Jadi, percabangan itu bisa dibuat di dalam percabangan. Kadang teknik ini disebut juga *nested if*.
Contoh kasus:
Misalnya ada model bisinis seperti ini di sebuah toko. Ketika orang membayar di kasir, biasanya ditanya ada kartu member untuk mendapatkan diskon dan sebagainya.

```
Apakah anda punya kartu member?
   - ya
       * Apakah belanjaan anda lebih dari 500rb?
           # ya : mendapatkan diskon 50rb
           # tidak : tidak mendapatkan diskon
       * Apakah belanjaan anda lebih dari 100rb?
           # ya : mendapatkan diskon 15rb
           # tidak: tidak mendapatkan diskon
   - tidak
       * Apakah belanjaan anda lebih dari 100rb?
           # ya : mendapatkan diskon 10rb
           # tidak: tidak mendapatkan diskon
```

Kalau tidak, coba perhatikan flow chart-nya:



Exercise 6 :

Hasil outputnya:

```
run:
Apakah ada kartu member: ya
Total belanjaan: 334000
Total Bayar: Rp 319000
BUILD SUCCESSFUL (total time: 11 seconds)
```

Cobalah untuk mengubah nilai yang dimasukkan dan perhatikan hasilnya.
Mungkin di sana ada yang perlu diperhatikan:
•Fungsi equalsIgnoreCase("ya") digunakan untuk membandingkan String dengan tidak memperdulikan huruf besar dan kecilnya.
•Ada juga Fungsi equals(), fungsinya sama.
Tapi equals() akan memperhatikan *case* hurufnya.
Kenapa tidak menggunakan operator == atau !=?
Di Java memang seperti itu.
Kalau kita ingin membandingkan nilai String, ya... menggunakan fungsi yang dua tadi.
Tapi, kalau membandingkan selain String, maka bisa pakai operator == atau !=.

```java
import java.util.Scanner;

public class Kasir {
    public static void main(String[] args) {
        // deklarasi variabel dan Scanner
        int belanjaan, diskon, bayar;
        String kartu;
        Scanner scan = new Scanner(System.in);

        // mengambil input
        System.out.print("Apakah ada kartu member: ");
        kartu = scan.nextLine();
        System.out.print("Total belanjaan: ");
        belanjaan = scan.nextInt();

        // proses
        if (kartu.equalsIgnoreCase("ya")) {
            if (belanjaan > 500000) {
                diskon = 50000;
            } else if (belanjaan > 100000) {
                diskon = 15000;
            } else {
                diskon = 0;
            }

        } else {
            if (belanjaan > 100000) {
                diskon = 5000;
            } else {
                diskon = 0;
            }
        }

        // total yang harus dibayar
        bayar = belanjaan - diskon;

        // output
        System.out.println("Total Bayar: Rp " + bayar);
    }
}
```

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Java: Memahami 2 Jenis Perulangan dalam Java

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

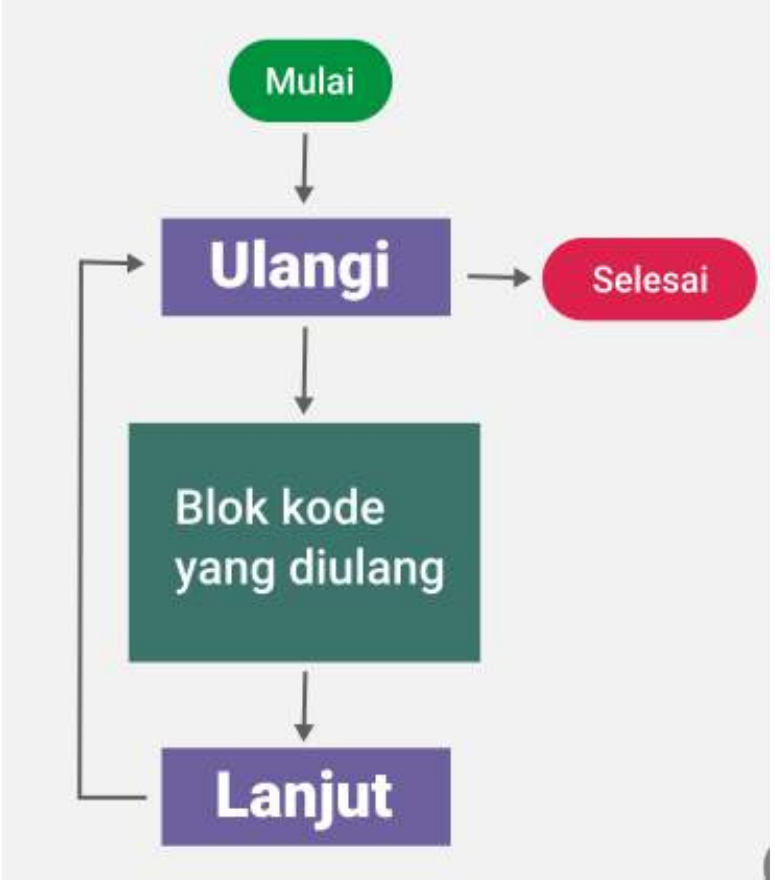
UKRIDA
Universitas Kristen Krida Wacana

Apa yang akan kita lakukan bila ingin menyuruh komputer mengerjakan perintah yang berulang-ulang?
Misalkan kita ingin menyuruh komputer menampilkan teks Petani Kode sebanyak 5x.
Maka kita bisa menyuruhnya seperti ini:

Maka kita bisa menyuruhnya seperti ini:

```
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
System.out.println("Petani Kode");
```

Tapi... bagaimana kalau sebanyak 1000x, apa kita akan mampu mengetik kode sebanyak itu?
Tentunya tidak.
Karena itu, kita harus pakai perulangan.

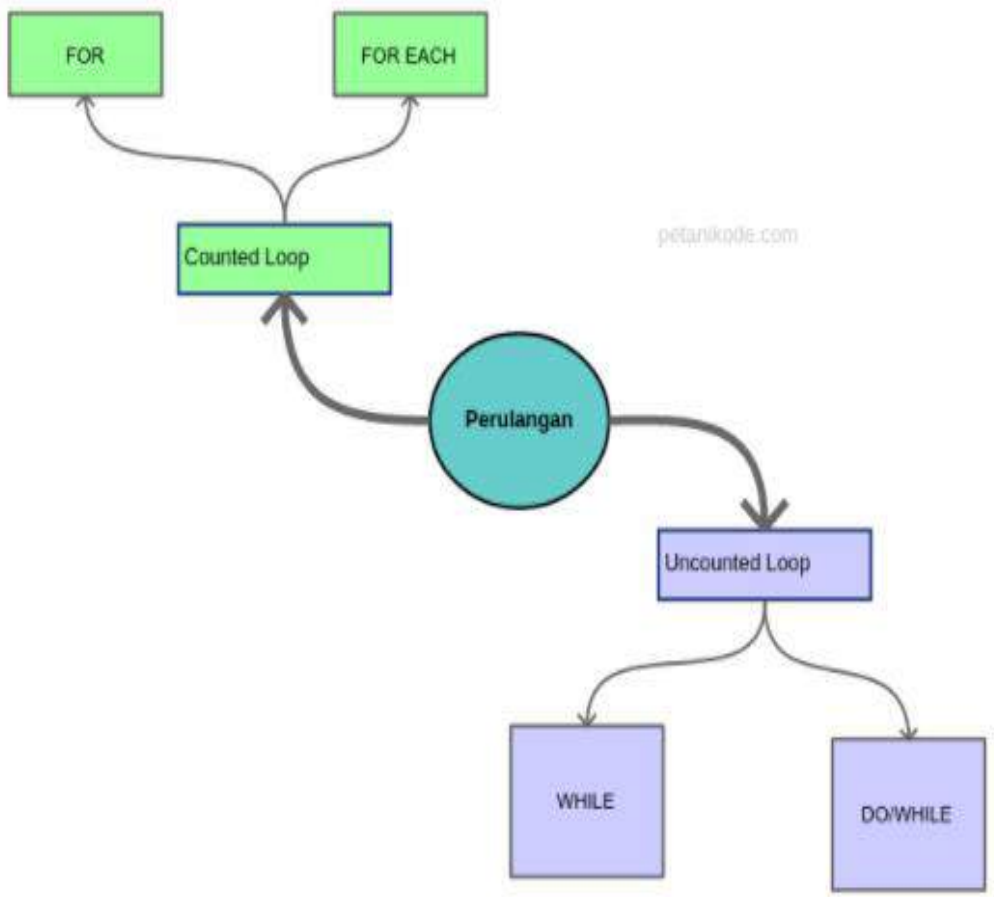


Contoh perulangan:

```
for (int hitungan = 0; hitungan <= 1000; hitungan++) {
    System.out.println("Petani Kode");
}
```

Sebelum masuk ke pembahasan lebih dalam, ada hal yang harus kalian ketahui terlebih dahulu.
Perulangan dalam pemrograman dibagi manjadi dua jenis:

1. **Counted loop**: Perulangan yang jumlah pengulangannya terhitung atau tentu.
2. **Uncounted loop**: Perulangan yang jumlah pengulangannya tidak terhitung atau tidak tentu.



Counted loop terdiri dari perulangan *For* dan *For each*.
Sedangkan *Uncounted loop* terdiri dari perulangan *While* dan *Do/While*

1. Counted Loop

Seperti yang sudah dijelaskan, perulangan ini memiliki jumlah pengulangan yang tentu dan terhitung.
Perulangan ini terdiri dari perulangan *For* dan *For each*.

Perulangan *For*

Format penulisan perulangan *For* di java adalah sebagai berikut:

```
for( int hitungan = 0; hitungan <= 10; hitungan++ ){
    // blok kode yang akan diulang
}
```

Penjelasan:

- variabel hitungan tugasnya untuk menyimpan hitungan pengulangan.
  - ✓ hitungan <= 10 artinya selama nilai hitungannya lebih kecil atau sama dengan 10, maka pengulangan akan terus dilakukan. Dengan kata lain, perualangan ini akan mengulang sebanyak 10 kali.
  - ✓ hitungan++ fungsinya untuk menambah satu (+1) nilai hitungan peda setiap pengulangan.
- Blok kode *For* dimulai dengan tanda '{' dan diakhiri dengan '}'.

Contoh Program Perulangan *For*

Silahkan buat class baru bernama `Bintang`, kemudian ikuti kode berikut:

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

        for(int i=0; i <= 5; i++){
            System.out.println("******");
        }

    }
}
```

Hasil output:

```
******
******
******
******
******
******
```

Buat sebuah program yang menampilkan bilangan ganjil saja.

```
class CetakBilanganGanjil{

    public static void main(String[] argumen){
        for(int i = 1; i <= 20; i += 2){
            System.out.print( i + " ");
        }
    }

}
```

Hasil output:

```
1 3 5 7 9 11 13 15 17 19
```

Perhatikan: di sana kita menggunakan `i += 2`, bukan `i++`.
Apa maksudnya?
Maksudnya, nilai `i` akan ditambah dua (`+2`) di setiap pengulangan.

Perulangan *For Each*

Perulangan ini sebenarnya digunakan untuk menampilkan isi dari *array*.
Apa itu *array*?
Singkatnya, *array* itu variabel yang menyimpan lebih dari satu nilai dan memiliki indeks.
Selengkapnya, nanti bisa di pelajari pada next pertemuan..
Perulangan *For Each* pada Java, dilakukan juga dengan kata kunci *For*.

Contohnya seperti ini:

```
for ( int item : dataArray ) {
    // blok kode yang diulang
}
```

Penjelasan:

- variabel item akan menyimpan nilai dari *array*
- Kita bisa baca seperti ini: “Untuk setiap item dalam dataArray, maka lakukan perulangan”

Contoh Program *For Each*

Buat sebuah class baru bernama `PerulanganForeach`, kemudian ikuti kode berikut.

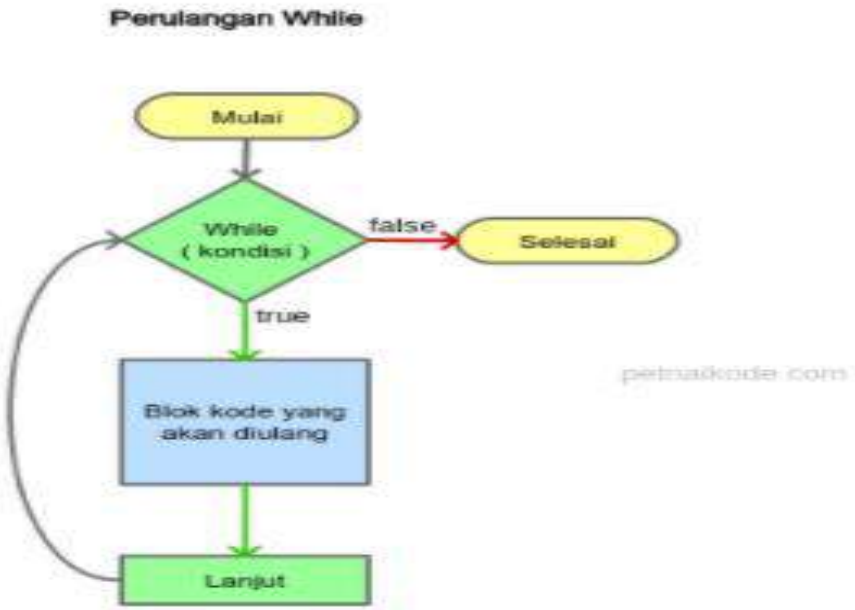
```
public class PerulanganForeach {
    public static void main(String[] args) {

        // membuat array
        int angka[] = {3,1,42,24,12};

        // menggunakan perulangan For each untuk menampilkan angka
        for( int x : angka ){
            System.out.print(x + " ");
        }

    }
}
```

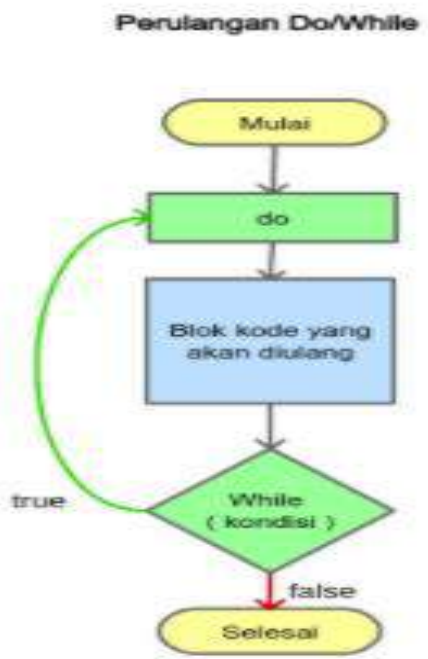
Hasil outputnya:

```
3 1 42 24 12
```

2. Uncounted Loop

Seperti yang sudah dijelaskan di awal tadi, perulangan ini tidak jelas jumlah pengulangannya. Tapi, tidak menutup kemungkinan juga, jumlah pengulangannya dapat ditentukan. Perulangan *uncounted loop* terdirid dari perulangan *While* dan *Do/While*.





Perulangan While

While bisa kita artikan *selama*.
Cara kerja perulangan ini seperti percabangan, ia akan melakukan perulangan selama kondisinya bernilai true.

Struktur penulisan perulangan while:

```
while ( kondisi ) {
    // blok kode yang akan diulang
}
```

Penjelasan:

- *kondisi* bisa kita isi dengan perbandingan maupun variabel boolean. *Kondisi* ini hanya memiliki nilai true dan flase.
- Perulangan while akan berhenti sampai *kondisi* bernilai false.

Contoh Program dengan Perulangan While

```java
import java.util.Scanner;

public class PerulanganWhile {
    public static void main(String[] args) {

        // membuat variabel dan scanner
        boolean running = true;
        int counter = 0;
        String jawab;
        Scanner scan = new Scanner(System.in);

        while( running ) {
            System.out.println("Apakah anda ingin keluar?");
            System.out.print("Jawab [ya/tidak]> ");

            jawab = scan.nextLine();

            // cek jawabnnya, kalau ya maka berhenti mengulang
            if( jawab.equalsIgnoreCase("ya") ){
                running = false;
            }

            counter++;
        }

        System.out.println("Anda sudah melakukan perulangan sebanyak " + cou

    }
}
```

Hasil outputnya:

```
run:
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> tidak
Apakah anda ingin keluar?
Jawab [ya/tidak]> ya
Anda sudah melakukan perulangan sebanyak 4 kali
BUILD SUCCESSFUL (total time: 9 seconds)
```

Di sana telah dilakukan perulangan sebanyak 4 kali. Bisa saja terjadi 10 kali.
Itu targantung dari kondisinya.
Kalau nilai variabel running bernilai false, maka perulangan berhenti.
Contoh kode *while* di atas dapat kita baca seperti ini: “Lakukan perulangan selama nilai running bernilai true.”
Tidak menutup kemungkinan juga, perulangan ini dapat melakukan *counted loop*.

Contohnya seperti ini:

```
int i = 0;

while ( i <= 10 ){
    // blok kode yang akan diulang
    System.out.println('Perulangan ke-' + i);

    // increment nilai i
    i++;
}
```

Hasil outputnya:

```
Perulangan ke-0
Perulangan ke-1
Perulangan ke-2
Perulangan ke-3
Perulangan ke-4
Perulangan ke-5
Perulangan ke-6
Perulangan ke-7
Perulangan ke-8
Perulangan ke-9
Perulangan ke-10
```

Penting: pastikan melakukan *increment* (i++) terhadap variabel *counter*. Karena kalau tidak, perulangannya akan terus-menerus dilakukan sampai komputernya hang.

Perulangan Do/While

Cara kerja perulangan *Do/While* sebenarnya sama seperti perulangan *While*.
Bedanya, *Do/While* melakukan satu kali perulangan dulu. Kemudian mengecek kondisinya.

Struktur penulisannya seperti ini:

```
do {
    // blok kode yang akan diulang
} while (kondisi);
```

Jadi kerjakan dulu (Do), baru di cek kondisinya while(kondisi). Kalau kondisi bernilai ture, maka lanjutkan perulangan.

Contoh Program dengan Perulangan *Do/While*

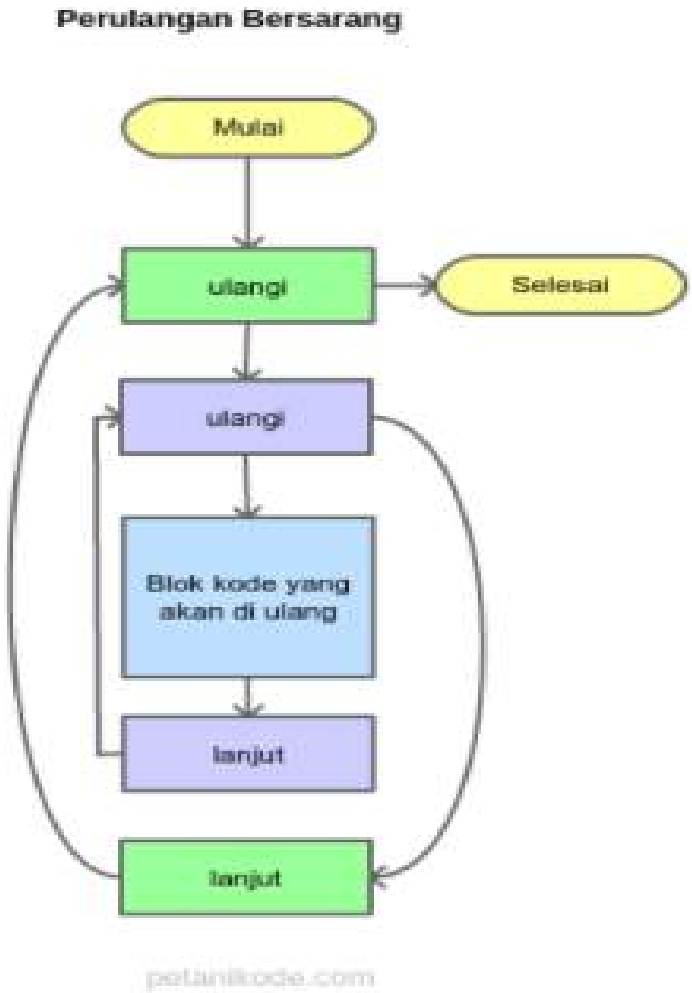
```java
public class PerulanganDoWhile {
    public static void main(String[] args) {

        // membuat variabel
        int i = 0;

        do {
            System.out.println("perulangan ke-" + i);
            i++;
        } while ( i <= 10);

    }
}
```

Hasil outputnya:

```
perulangan ke-0
perulangan ke-1
perulangan ke-2
perulangan ke-3
perulangan ke-4
perulangan ke-5
perulangan ke-6
perulangan ke-7
perulangan ke-8
perulangan ke-9
perulangan ke-10
```

Perulangan Bersarang *(Nested Loop)*

Perulangan juga dapat bersarang. Perulangan bersarang maksudnya, perulangan dalam perulangan atau disebut juga *nested loop*.



Contoh Program Perulangan Bersarang

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

        // membuat variabel
        int x, y;

        // melakukan parulang sebnayan x dan y kali
        for (x = 0; x <= 5; x++){
            for( y = 0; y <= 3; y++){
                System.out.format("Perulangan [x=%d, y=%d] %n", x, y);
            }
        }
    }
}
```

Note :

- Perulangan bersarang sering digunakan pada *array* multi dimensi.
- Jenis perulangan di dalam perulangn bisa berbeda, misalnya di dalam perulangan *while* ada perulangan *for*.

Hasil outputnya:

```
Perulangan [x=0, y=0]
Perulangan [x=0, y=1]
Perulangan [x=0, y=2]
Perulangan [x=0, y=3]
Perulangan [x=1, y=0]
Perulangan [x=1, y=1]
Perulangan [x=1, y=2]
Perulangan [x=1, y=3]
Perulangan [x=2, y=0]
Perulangan [x=2, y=1]
Perulangan [x=2, y=2]
Perulangan [x=2, y=3]
Perulangan [x=3, y=0]
Perulangan [x=3, y=1]
Perulangan [x=3, y=2]
Perulangan [x=3, y=3]
Perulangan [x=4, y=0]
Perulangan [x=4, y=1]
Perulangan [x=4, y=2]
Perulangan [x=4, y=3]
Perulangan [x=5, y=0]
Perulangan [x=5, y=1]
Perulangan [x=5, y=2]
Perulangan [x=5, y=3]
```

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Java: Menggunakan Array untuk Menyimpan Banyak Hal

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Apa yang akan kita lakukan bila memiliki banyak data yang akan disimpan dalam vairabel?
Misalkan kita ingin menyimpan nama-nama teman dalam variabel.

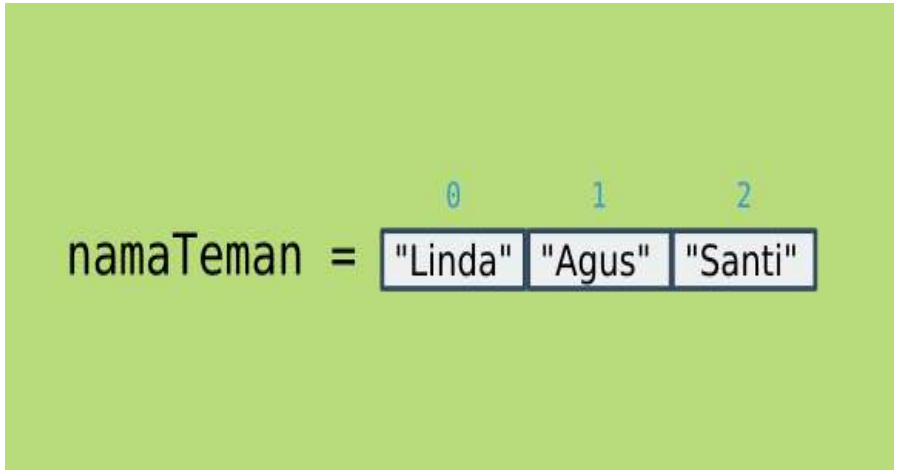
Maka mungkin kita akan melakukannya seperti ini:

```
String namaTeman1 = "Linda";
String namaTeman2 = "Santi";
String namaTeman3 = "Susan";
String namaTeman4 = "Mila";
String namaTeman5 = "Ayu";
```

Hal ini sah-sah saja.
Akan tetapi...
Masalahnya bagaimana kala datanya banyak, misalkan ada 100 data, pastinya capek donk membuat variabel sebanyak itu.
Karena itu, kita bisa menyimpan itu semua dalam Array.

Apa itu Array?

Array adalah sebuah variabel yang bisa menyimpan banyak data dalam satu variabel.
Array menggunakan indeks untuk memudahkan akses terhadap data yang disimpannya.



Indeks array selalu dimulai dari 0...
Dan perlu diketahui juga, indeks tidak selalu dalam bentuk angka. Bisa juga karakter atau teks.

Cara Membuat Array di Java

Perhatikan:

- Kita menggunakan kurung siku `[]` untuk membuat array;
- Kurung siku bisa diletakkan setelah tipe data atau nama array;
- Angka `5` dalam kurung artinya batas atau ukuran array-nya.
- Array yang kosong siap diisi dengan data. Pastikan mengisinya dengan data yang sesuai dengan tipe datanya.

Cara membuat array kosong:

```
// cara pertama
String[] nama;

// cara kedua
String nama[];

// cara ketiga dengan kata kunci new
String[] nama = new String[5];
```

Kita bisa mengisinya seperti ini:

```
nama[0] = "Linda";
nama[1] = "Santi";
nama[2] = "Susan";
nama[3] = "Mila";
nama[4] = "Ayu";
```

Atau kalau tidak mau repot, kita bisa membuat array dan langsung mengisinya.

```
String[] nama = {"Linda", "Santi", "Susan", "Mila", "Ayu"};
```

Mengambil Data dari Array

Seperti yang sudah kita ketahui, array memiliki indeks untuk memudahkan kita mengakses datanya.

Karena itu, kita bisa mengambil datanya dengan cara seperti ini:

```
// membuat array
String[] nama = {"Linda", "Santi", "Susan", "Mila", "Ayu"};

// mengambil data array
System.out.println(teman[2]);
```

Kira-kira apa hasil outputnya?
Yep! benar sekali, hasil outputnya adalah:

```
Susan
```

Karena Susan terletak di indeks ke-2.

Menggunakan Perulangan

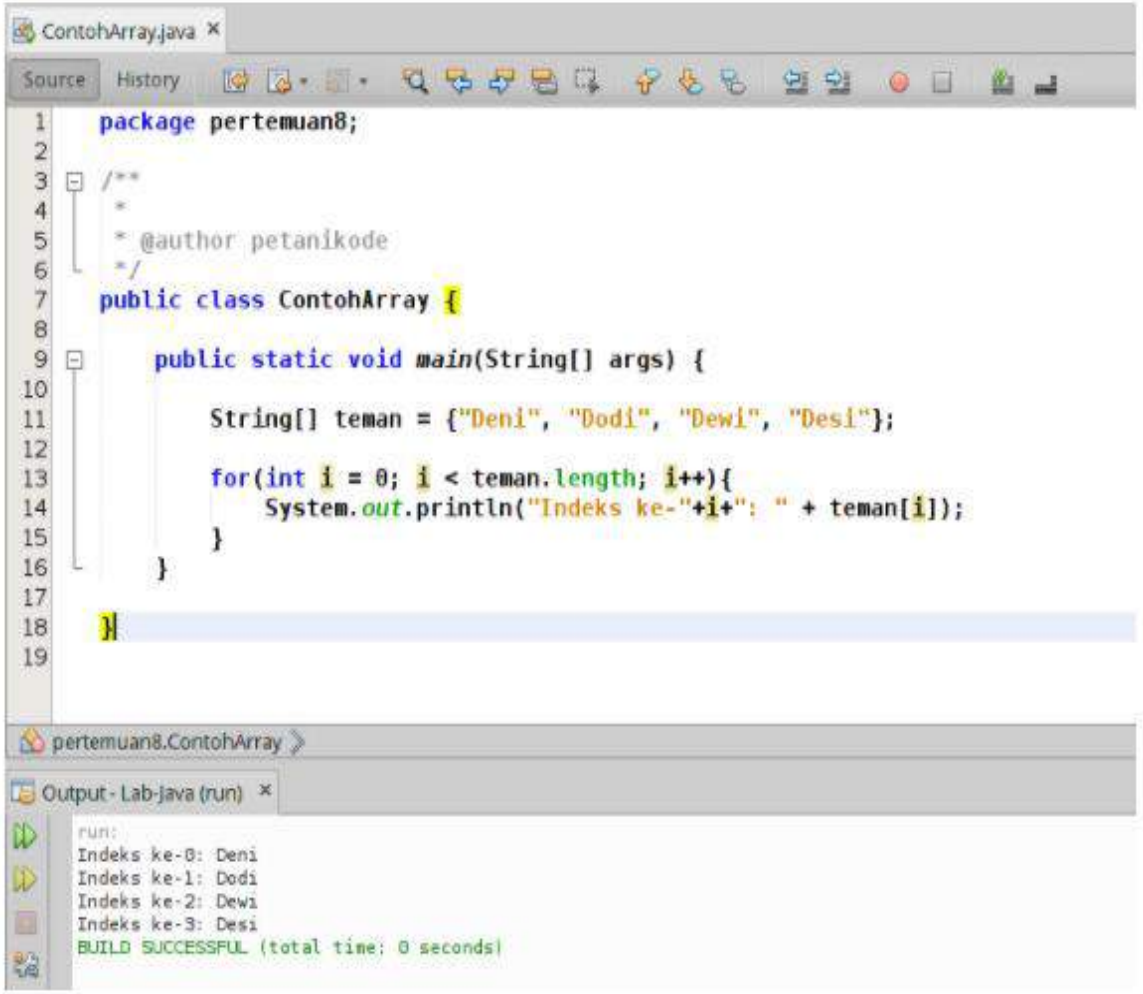
Mengambil data satu per satu dari array mungkin cukup melelahkan, karena kita harus mengtik ulang nama array-nya dengan indeks yang berbeda.

Contoh:

```
System.out.println(teman[0]);
System.out.println(teman[1]);
System.out.println(teman[2]);
System.out.println(teman[3]);
```

Bagaimana kalau data array-nya sampai 1000, maka kita harus mengetik kode sebanyak seribu kali.

Karena itu, disinilah peran perulangan.



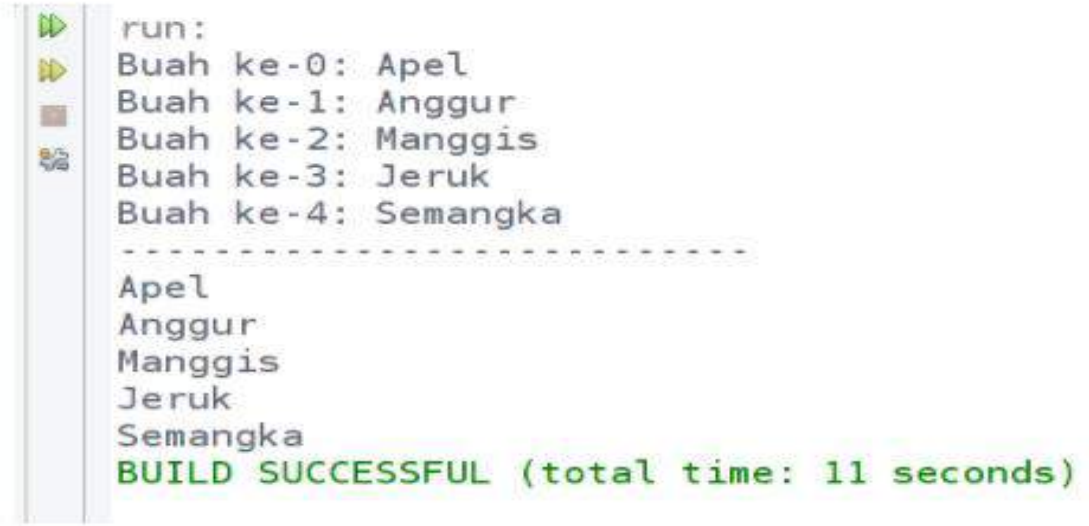
Perhatikan:
Di sana kita menggunakan atribut length untuk mengambil panjang array-nya. Jadi, perulangan akan dilakukan sebanyak isi array-nya.

Sekarang Mari Kita Latihan

Silahkan buat class bernama `Buah`, kemudian ikuti kode berikut:

```java
import java.util.Scanner;

public class Buah {
    public static void main(String[] args) {

        // membuat array buah-buahan
        String[] buah = new String[5];

        // membuat scanner
        Scanner scan = new Scanner(System.in);

        // mengisi data ke array
        for( int i = 0; i < buah.length; i++ ){
            System.out.print("Buah ke-" + i + ": ");
            buah[i] = scan.nextLine();
        }

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

        // menampilkan semua isi array
        for( String b : buah ){
            System.out.println(b);
        }

    }
}
```

Hasil outputnya:



Perhatikan:
Di sana kita menggunakan perulangan `foreach` untuk menampilkan isi array. Seperti yang sudah kita pelajari pada materi Perulangan di Java, perulangan ini dapat kita gunakan untuk menampilkan isi array.

Array Multi Dimensi

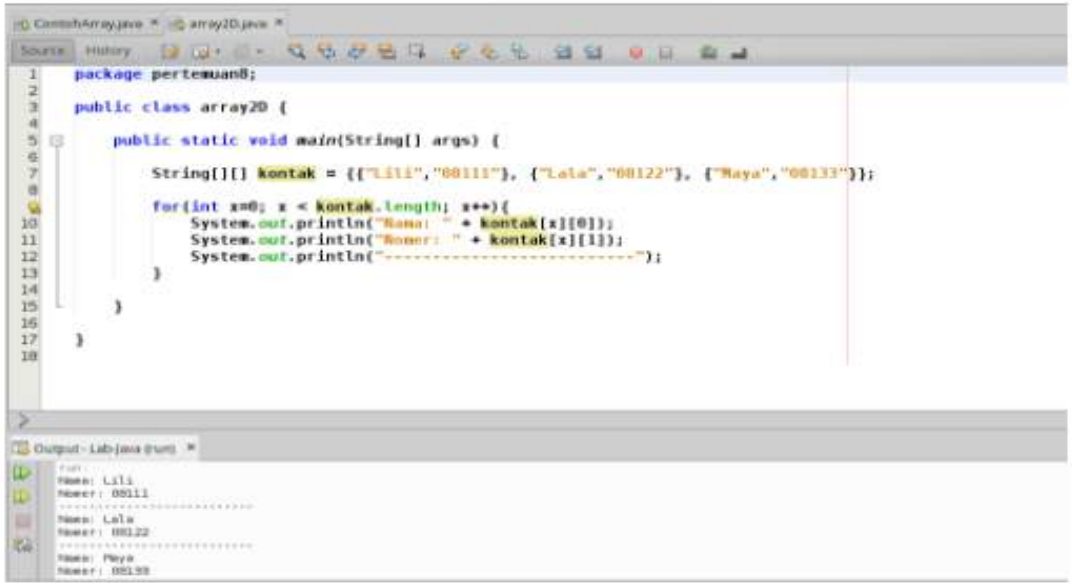
Array multi dimensi artinya array yang memiliki lebih dari satu dimensi.
Atau kita bisa sebut, array di dalam array.
Jumlah dimensinya tidak terbatas, tergantung kita mampunya sampai berapa

Contoh berikut ini adalah array dua dimensi:

```
String[][] kontak = {
    {"Lili","08111"},
    {"Lala","08122"},
    {"Maya","08133"}
};
```

Indek ke-0 pada array kontak berisi array {"lili","08111"}.

Contoh cara mengakses data dari array dua dimensi:



Contoh Program Array Multi Dimensi

Silahkan buat class baru bernama `RuangKelas` kemudian ikuti kode berikut:

```java
import java.util.Scanner;

public class RuangKelas {
    public static void main(String[] args) {

        // Membuat Array dan Scanner
        String[][] meja = new String[2][3];
        Scanner scan = new Scanner(System.in);

        // mengisi setiap meja
        for(int bar = 0; bar < meja.length; bar++){
            for(int kol = 0; kol < meja[bar].length; kol++){
                System.out.format("Siapa yang akan duduk di meja (%d,%d): ",
                meja[bar][kol] = scan.nextLine();
            }
        }

        // menampilkan isi Array
        System.out.println("-------------------------");
        for(int bar = 0; bar < meja.length; bar++){
            for(int kol = 0; kol < meja[bar].length; kol++){
                System.out.format("| %s | \t", meja[bar][kol]);
            }
            System.out.println("");
        }
        System.out.println("-------------------------");
    }
}
```

Hasil outputnya:

```
run:
Siapa yang akan duduk di meja (0,0): Dian
Siapa yang akan duduk di meja (0,1): Anggun
Siapa yang akan duduk di meja (0,2): Fitri
Siapa yang akan duduk di meja (1,0): Adam
Siapa yang akan duduk di meja (1,1): Ayuni
Siapa yang akan duduk di meja (1,2): Selly
-------------------------
| Dian |        | Anggun |      | Fitri |
| Adam |        | Ayuni |       | Selly |
-------------------------
BUILD SUCCESSFUL (total time: 53 seconds)
```

Pada program tersebut, kita menggunakan perulangan bersarang untuk mengambil input dan menampilkan outputnya.
Karena array dua dimensi mirip seperti tabel, maka kita harus melakukan perulangan terhadap baris dan kolomnya.
Lalu bagaimana dengan array 3D, 4D, 5D, dan seterusnya?
Tentu saja kita harus membuat perulangan bersarang sebanyak dimensinya.
Kalau tiga, ya buat tiga perulangan.

%c	Simbol untuk menampilkan nilai karakter
%s	Simbol untuk menampilkan nilai string
%d, %i	Simbol untuk menampilkan nilai angka, desimal
%f	Simbol untuk menampilkan nilai float
/n	Simbol untuk membuat baris baru
/t	Untuk membuat tab

Array List

Array yang kita bahas di atas sebenarnya memiliki beberapa kekurangan, seperti:

- Tidak mampu menyimpan data dengan tipe yang berbeda.
- Ukurannya tidak dinamis.
- Maka dari itu, ada Array List yang menutupi kekurangan tersebut.

Array list merupakan sebuah class yang memungkinkan kita membuat sebuah objek untuk menampung apapun.

Untuk menggunakan Array List, kita harus mengimpornya terlebih dahulu.

```
import java.util.ArrayList;
```

Setelah itu, baru kita bisa membuat sebuah objek Array List seperti ini:

```
ArrayList al = new ArrayList();
```

Contoh Program dengan Array List

Silahkan membuat class dengan nama Doraemon, kemudian ikuti kode berikut:

```
import java.util.ArrayList;

public class Doraemon {
    public static void main(String[] args) {

        // membuat objek array list
        ArrayList kantongAjaib = new ArrayList();

        // Mengisi kantong ajaib dengan 5 benda
        kantongAjaib.add("Senter Pembesar");
        kantongAjaib.add(532);
        kantongAjaib.add("tikus");
        kantongAjaib.add(1231234.132);
        kantongAjaib.add(true);

        // menghapus tikus dari kantong ajaib
        kantongAjaib.remove("tikus");

        // Menampilkan isi kantong ajaib
        System.out.println(kantongAjaib);

        // menampilkan banyak isi kantong ajaib
        System.out.println("Kantong ajaib berisi "+ kantongAjaib.size() +" i
    }
}
```

Hasil outputnya:

```
run:
[Senter Pembesar, 532, 1231234.132, true]
Kantong ajaib berisi 4 item
BUILD SUCCESSFUL (total time: 0 seconds)
```

Karena array list (kantongAjaib) merupakan sebuah objek yang terbuat dari class Array List, maka dia punya *method* (fungsi) untuk melakukan sesuatu.

- Fungsi add() untuk menambahkan sesuatu ke dalam Array List;
- Fungsi remove() untuk menghapus sesuatu ke dalam Array List;
- Fungsi size() untuk mengambil ukuran Array List;
- Fungsi get(id) untuk mengambil item dalam Array List berdasarkan id atau indeks tertentu.
- isEmpty (), memeriksa apakah array kosong
- Set (), menimpa nilai pada indeks tertentu
- Contains(), memeriksa suatu nilai apakah ada dalam array list

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Belajar Java: Menggunakan Prosedur dan Fungsi untuk Membuat Sub-program

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Pada contoh program di pembahasan sebelumnya, kita hanya menulis kode intruksi pada fungsi `main()` saja. Fungsi `main()` adalah fungsi utama dalam program Java. Semua kode yang kita tulis di dalamnya, akan langsung dieksekusi.

Tapi masalahnya sekarang:
"Bagaimana kalau kita membuat program yang cukup besar, apakah kita masih bisa menulis semua kodenya di dalam fungsi `main()`?"
Bisa-bisa saja, tapi kurang efektif dan akan menghabiskan banyak tenaga untuk mengetik kodenya.
Belum lagi kalau ada error…

"Lalu solusinya bagaimana?"
Solusinya menggunakan prosedur/fungsi.
Prosedur/fungsi dapat memecah program menjadi sub-sub program, sehingga kita bisa membuat program lebih efisien.

Penggunaan prosedur/fungsi dapat mengurangi pengetikan kode yang berulang-ulang.

Pengertian Prosedur, Fungsi, dan Method

Jangan bingung...karena ketiga-tiganya sama.
Prosedur, Fungsi, dan Method itu sama.

Prosedur adalah **sebutan** untuk fungsi yang tidak mengembalikan nilai. Fungsi ini biasanya ditandai dengan kata kunci void.

Fungsi adalah **sebutan** untuk fungsi yang mengembalikan nilai.

Method adalah **fungsi** yang berada di dalam Class. Sebutan ini, biasanya digunakan pada OOP.

Untuk memudahkan, mari kita sebut semuanya **fungsi**.

Cara Membuat Fungsi di Java

Fungsi harus dibuat atau ditulis di dalam *class*.

Struktur dasarnya seperti ini:

```
static TypeDataKembalian namaFungsi(){
    // statemen atau kode fungsi
}
```

Penjelasan:

- Kata kunci static, artinya kita membuat fungsi yang dapat dipanggil tanpa harus membuat instansiasi objek.
  - TypeDataKembalian adalah tipe data dari nilai yang dikembalikan setelah fungsi dieksekusi.
  - namaFungsi() adalah nama fungsinya. Biasanya ditulis dengan huruf kecil di awalnya. Lalu, kalau terdapat lebih dari satu suku kata, huruf awal di kata kedua ditulis kapital.

Contoh:

```
static void ucapSalam(){
    System.out.println("Selamat Pagi");
}
```

Tipe data void artinya kosong, fungsi tersebut tidak mengebalikan nilai apa-apa.

Cara Memanggil/Eksekusi Fungsi

Setelah kita membuat fungsi, selanjutnya kita akan mengeksekusi fungsinya.
Fungsi dapat dipanggil dari fungsi `main` atau dari fungsi yang lainnya.

Contoh pemanggilan fungsi dalam dalam funsgi `main`:

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

Maka akan menghasilkan output:

```
Selamat Pagi
```

Kode lengkapnya, silahkan dicoba sendiri:

```
class BelajarFungsi {

    // membuat fungsi ucapSalam()
    static void ucapSalam(){
        System.out.println("Selamat Pagi");
    }

    // membuat fungsi main()
    public static void main(String[] args){
        // memanggil/eksekusi fungsi ucapSalam()
        ucapSalam();
    }
}
```

Fungsi dengan Parameter

Parameter adalah variabel yang menampung nilai untuk diproses di dalam fungsi. Parameter berperan sebagai input untuk fungsi.

Struktur dasarnya seperti ini:

```
static TipeData namaFungsi(TipeData namaParameter, TipeData namaParameterLai
    // kode fungsi
}
```

Penjelasan:

- Parameter ditulis di antara tanda kurung (...);
- Parameter harus diberikan tipe data;
- Bila terdapat lebih dari satu parameter, maka dipisah dengan tanda koma.

Contoh fungsi yang memiliki parameter:

```
static void ucapin(String ucapan){
    System.out.println(ucapan);
}
```

Pada contoh tersebut, kita membuat parameter bernama `ucapan` dengan tipe `String`. Sehingga kita bisa menggunakan variabel `ucapan` di dalam fungsi.

Cara pemanggilan fungsi yang memiliki parameter:

```
ucapin("Hallo!");
ucapin("Selamat datang di pemrograman Java");
ucapin("Saya kira ini bagian terakhir");
ucapin("Sampai jumpa lagi, ya!");
```

Hasil outputnya:

```
Hallo!
Selamat datang di pemrograman Java
Saya kira ini bagian terakhir
Sampai jumpa lagi, ya!
```

Fungsi yang Mengembalikan Nilai

Setelah fungsi memproses data yang diinputkan melalui parameter, selanjutnya fungsi harus mengembalikan nilai agar dapat diolah pada proses berikutnya.
Pengembalian nilai pada fungsi menggunakan kata kunci return.

Contoh:

```
static int luasPersegi(int sisi){
    int luas = sisi * sisi;
    return luas;
}
```

Pada contoh tersebut, kita membuat sebuah parameter bernama `sisi`. Kemudian fungsi akan mengembalikan nilai dengan tipe `int` *(integer)* dari variabel `luas`.

Contoh pemanggilanya:

```
System.out.println("Luas Persegi dengan panjang sisi 5 adalah " + luasPerseg
```

Hasil Output:

```
Luas Persegi dengan panjang sisi 5 adalah 25
```

Pemanggilan Fungsi di Fungsi Lain

Fungsi-fungsi dapat saling memanggil untuk memproses data.
Contoh, sebuah program Kalkulator Bangun Ruang memiliki fungsi-fungsi: luasPersegi(), luasPersegiPanjang(), luasSegitiga(), luasBalok(), luasKubus() dsb.
Fungsi-fungsi tersebut dapat saling membantu, contoh fungsi luasKubus() membutuhkan fungsi luasPersegi().

Rumus:

```
Luas Kubus = 6 *  luasPersegi;
Luas Persegi = sisi * sisi;
```

Maka programnya bisa dibuat seperti ini:

```
public class BangunRuang {

    public static void main(String[] args) {
        int s = 12;
        int luas = luasKubus(s);

        System.out.println(luas);
    }

    // membuat fungsi luasPersegi()
    static int luasPersegi(int sisi){
        return sisi * sisi;
    }

    // membuat fungsi luasKubus()
    static int luasKubus(int sisi){

        // memanggil fungsi luasPersegi
        return 6 * luasPersegi(sisi);
    }

}
```

Hasil output

```
864
```

Fungsi Static dan Non-Static

Pada contoh-contoh diatas, kita menggunakan kata kunci static sebelum membuat fungsi.
Kata kunci static akan membuat fungsi dapat dieksekusi langsung, tanpa harus membuat instansiasi objek dari class.

Contoh:

```
public class FungsiStatic {

    // fungsi non-static
    void makan(String makanan){
        System.out.println("Hi!");
        System.out.println("Saya sedang makan " + makanan);
    }

    // fungsi static
    static void minum(String minuman){
        System.out.println("Saya sedang minum " + minuman);
    }

    // fungsi main
    public static void main(String[] args) {

        // pemanggilan fungsi static
        minum("Kopi");


        // membuat instansiasi objek saya dari class FungsiStatic
        FungsiStatic saya = new FungsiStatic();
        // pemanggilan fungsi non-static
        saya.makan("Nasi Goreng");

    }
}
```

Pada contoh tersebut, fungsi `makan()` adalah fungsi *non-static*. Sedangkan fungsi `minum()` adalah fungsi *static*.

Hasil output dari program di atas:

```
Saya sedang minum Kopi
Hi!
Saya sedang makan Nasi Goreng
```

Fungsi Static dan Non-Static

Pada contoh-contoh diatas, kita menggunakan kata kunci static sebelum membuat fungsi.
Kata kunci static akan membuat fungsi dapat dieksekusi langsung, tanpa harus membuat instansiasi objek dari class.

Apabila kita tidak membuat objek untuk memanggil fungsi *non-static*, maka akan terjadi error.

Contoh:

```
public class FungsiStatic {

    // fungsi non-static
    void makan(String makanan){
        System.out.println("Hi!");
        System.out.println("Saya sedang makan " + makanan);
    }

    // fungsi static
    static void minum(String minuman){
        System.out.println("Saya sedang minum " + minuman);
    }

    // fungsi main
    public static void main(String[] args) {

        // pemanggilan fungsi static
        minum("Kopi");


        // membuat instansiasi objek saya dari class FungsiStatic
        FungsiStatic saya = new FungsiStatic();
        // pemanggilan fungsi non-static
        saya.makan("Nasi Goreng");

    }
}
```

Pada contoh tersebut, fungsi `makan()` adalah fungsi *non-static*. Sedangkan fungsi `minum()` adalah fungsi *static*.

Hasil output dari program di atas:

```
Saya sedang minum Kopi
Hi!
Saya sedang makan Nasi Goreng
```

Variabel Global dan Variabel Lokal pada Java

Variabel global adalah variabel yang bisa diakses dari semua fungsi. Sedangkan variabel lokal adalah variabel yang hanya bisa diakses dari dalam fungsi tempat variabel itu berada.

Saat pemanggilan fungsi `help()` kita membuat ulang variabel `nama`. Sehingga variabel nama menjadi variabel lokal pada fungsi `help()` dan nilainya berubah menjadi `"Petani Kode"`. Sedangkan, saat kita akases lagi variabel nama melalui fungsi `main()` nilainya tetap sama seperti yang didefinisikan.

Mari kita lihat contohnya:

```
class ProgramKu{

    // ini variabel global
    static String nama = "Programku";
    static String version = "1.0.0";

    static void help(){

        // ini variabel lokal
        String nama = "Petani Kode";

        // mengakses variabel global di dalam fungso help()
        System.out.println("Nama: " + nama);
        System.out.println("Versi: " + version);
    }

    public static void main(String args[]){

        // panggil fungsi help()
        help();

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

    }
}
```

Hasil outputnya:

```
Nama: Petani Kode
Versi: 1.0.0
Nama: Programku
Versi: 1.0.0
```

Contoh Program dengan Fungsi dan Prosedur

Program ini adalah program sederhana dengan fitur sebagai berikut:

1. Baca data dari ArrayList
2. Simpan data ke ArrayList
3. Ubah data
4. Hapus Data
5. Keluar

Penjelasan:

- Variabel `listBuah` adalah variabel global untuk menyimpan nama-nama buah.
- Variabel `isRunning` adalah variabel global untuk membuat loop.
- Kemudian `inputStreamReader` dan `input` adalah objek yang kita butuhkan untuk mengambil input dari keyboard.

Setelah itu, buat masing-masing fungsi.

Baiklah, silahkan buat class baru bernama `FungsiProsedur`. Lalu impor class-class yang dibutuhkan.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
```

Setelah itu buat variabel global di dalam class `FungsiProsedur`:

```
static ArrayList listBuah = new ArrayList();
static boolean isRunning = true;
static InputStreamReader inputStreamReader = new InputStreamReader(System.in
static BufferedReader input = new BufferedReader(inputStreamReader);
```

Fungsi untuk menampilkan menu:

```
static void showMenu() throws IOException {

    System.out.println("========= MENU ========");
    System.out.println("[1] Show All Buah");
    System.out.println("[2] Insert Buah");
    System.out.println("[3] Edit Buah");
    System.out.println("[4] Delete Buah");
    System.out.println("[5] Exit");
    System.out.print("PILIH MENU> ");

    int selectedMenu = Integer.valueOf(input.readLine());

    switch(selectedMenu){
        case 1:
            showAllBuah();
            break;
        case 2:
            insertBuah();
            break;
        case 3:
            editBuah();
            break;
        case 4:
            deleteBuah();
            break;
        case 5:
            System.exit(0);
            break;
        default:
            System.out.println("Pilihan salah!");

    }

}
```

Fungsi tersebut bertugas untuk menampilkan menu dan menentukan fungsi mana yang akan dipanggil berdasarkan nomer menu yang diinputkan.

Apa itu throws IOException?
Nanti saya akan bahas di kesempatan berikutnya. Untuk saat ini diabaikan saja dulu. Ini karena kita menggunakan Buffereader, jadi throws IOException wajib ditulis.

Fungsi untuk menampilkan data:

```
static void showAllBuah(){
    if(listBuah.isEmpty()){
        System.out.println("Belum ada data");
    } else {
        // tampilkan semua buah
        for(int i = 0; i < listBuah.size(); i++){
            System.out.println(String.format("[%d] %s",i, listBuah.get(i)));
        }
    }
}
```

Fungsi tersebut bertugas menampilkan isi dari listBuah. Kalau listBuah kosong, maka akan ditampilkan pesan "Belum ada data".

Fungsi untuk menambah data buah:

```
static void insertBuah() throws IOException{
    System.out.print("Nama buah: ");
    String namaBuah = input.readLine();
    listBuah.add(namaBuah);
}
```

Pada fungsi tersebut, kita menggunakan method listBuah.add(namaBuah); untuk menambah data ke dalam listBuah berdasarkan namaBuah yang diberikan.

Fungsi untuk mengubah data buah:

```
static void editBuah() throws IOException{
    showAllBuah();
    System.out.print("Pilih nomer buah: ");
    int indexBuah = Integer.valueOf(input.readLine());

    System.out.print("Nama Baru: ");
    String namaBaru = input.readLine();

    // ubah nama buah
    listBuah.set(indexBuah, namaBaru);
}
```

Pertama kita perlu tampilkan dulu daftar buahnya, lalu kita minta user untuk memilih buah mana yang akan diedit.
Setelah itu, kita update buahnya dengan method listBuah.set(indexBuah, namaBaru);.

Fungsi untuk menghapus buah:

```
static void deleteBuah() throws IOException{
    showAllBuah();
    System.out.print("Pilih nomer buah: ");
    int indexBuah = Integer.valueOf(input.readLine());
    // hapus buah
    listBuah.remove(indexBuah);
}
```

Hampir sama seperti edit buah, untuk menghapus buah kita juga butuh nomer indeks buah yang akan dihapus.
Lalu mengapusnya dengan method listBuah.remove(indexBuah);.

Fungsi main:

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

    do {
        showMenu();
    } while (isRunning);

}
```

Lengkap sudah, berikut ini bentuk kode lengkapnya.

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;

public class FungsiProsedur {

    static ArrayList listBuah = new ArrayList();
    static boolean isRunning = true;
    static InputStreamReader inputStreamReader = new InputStreamReader(Syste
    static BufferedReader input = new BufferedReader(inputStreamReader);

    static void showMenu() throws IOException{

        System.out.println("========= MENU ========");
        System.out.println("[1] Show All Buah");
        System.out.println("[2] Insert Buah");
        System.out.println("[3] Edit Buah");
        System.out.println("[4] Delete Buah");
        System.out.println("[5] Exit");
        System.out.print("PILIH MENU> ");
```

```
        int selectedMenu = Integer.valueOf(input.readLine());

        switch(selectedMenu){
            case 1:
                showAllBuah();
                break;
            case 2:
                insertBuah();
                break;
            case 3:
                editBuah();
                break;
            case 4:
                deleteBuah();
                break;
            case 5:
                System.exit(0);
                break;
            default:
                System.out.println("Pilihan salah!");

        }


    }

    static void showAllBuah(){
        if(listBuah.isEmpty()){
           System.out.println("Belum ada data");
        } else {
             // tampilkan semua buah
            for(int i = 0; i < listBuah.size(); i++){
                System.out.println(String.format("[%d] %s",i, listBuah.get(i
            }
        }
    }

    static void insertBuah() throws IOException{
        System.out.print("Nama buah: ");
        String namaBuah = input.readLine();
        listBuah.add(namaBuah);
    }
```

```
        }
    }

    static void insertBuah() throws IOException{
        System.out.print("Nama buah: ");
        String namaBuah = input.readLine();
        listBuah.add(namaBuah);
    }

    static void editBuah() throws IOException{
        showAllBuah();
        System.out.print("Pilih nomer buah: ");
        int indexBuah = Integer.valueOf(input.readLine());

        System.out.print("Nama Baru: ");
        String namaBaru = input.readLine();

        // ubah nama buah
        listBuah.set(indexBuah, namaBaru);
    }

    static void deleteBuah() throws IOException{
        showAllBuah();
        System.out.print("Pilih nomer buah: ");
        int indexBuah = Integer.valueOf(input.readLine());
        // hapus buah
        listBuah.remove(indexBuah);
    }

    public static void main(String[] args) throws IOException {

        do {
            showMenu();
        } while (isRunning);

    }

}
```

Setelah itu, silahkan dijalankan dan perhatikanlah hasilnya.

```
========= MENU ========
[1] Show All Buah
[2] Insert Buah
[3] Edit Buah
[4] Delete Buah
[5] Exit
PILIH MENU> 1
Belum ada data
========= MENU ========
[1] Show All Buah
[2] Insert Buah
[3] Edit Buah
[4] Delete Buah
[5] Exit
PILIH MENU> 2
Nama buah: Apel
========= MENU ========
[1] Show All Buah
[2] Insert Buah
[3] Edit Buah
[4] Delete Buah
[5] Exit
PILIH MENU> 1
[0] Apel
========= MENU ========
[1] Show All Buah
[2] Insert Buah
[3] Edit Buah
[4] Delete Buah
[5] Exit
PILIH MENU>
```

Silahkan coba untuk melakukan insert, edit, dan delete.

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Menghubungkan Java Dengan Database MySQL

CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

sofwere yang dibutuhkan :

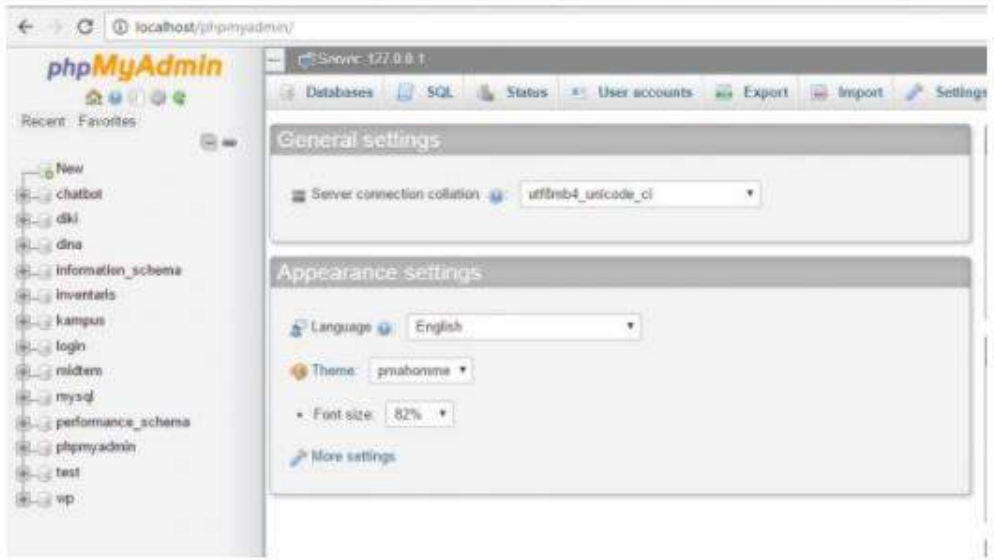
- java
- netbeans (saya menggunakan versi 8.2)
- Xampp

oke, langsung saja kita menuju pembahasan *Cara Menghubungkan Java Dengan Database MySQL*, berikut langka-langkah :

1. buka xampp dan aktifkan apache dan mysql

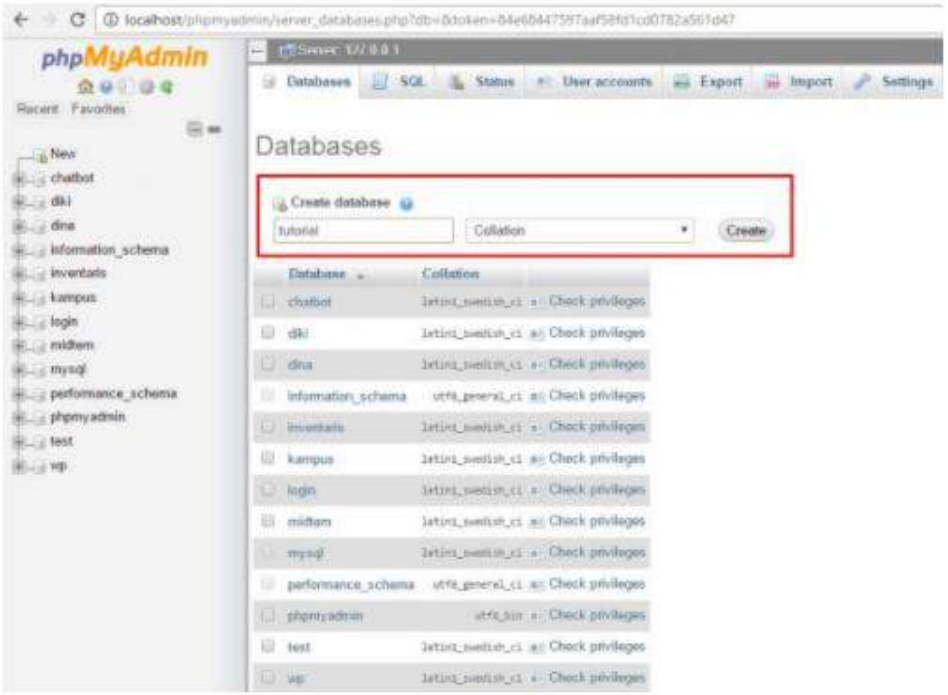
Cara Menghubungkan Java Dengan Database MySQL

2. buka browser dan akses *http://localhost/phpmyadmin/* ini fungsinya untuk membuat database



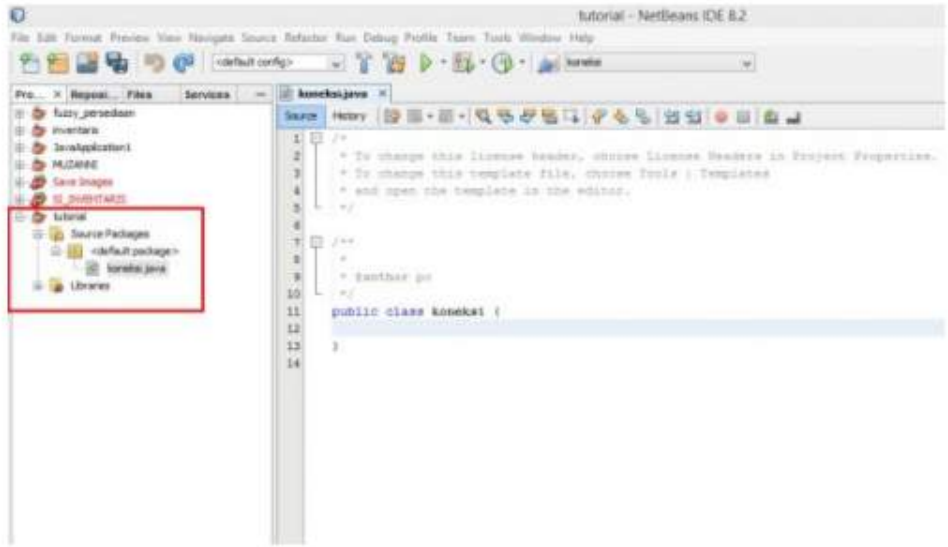
membuat database

3. buat database-nya, untuk penamaan database terserah (disni daya menggunakan nama tutorial)



membuat database

4. buat rojek baru java dengan netbeans dan buatkan sebuah class untuk file koneksi.



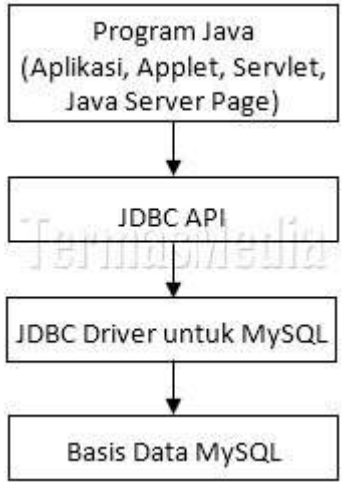
membuat projek baru menggunakan netbeans

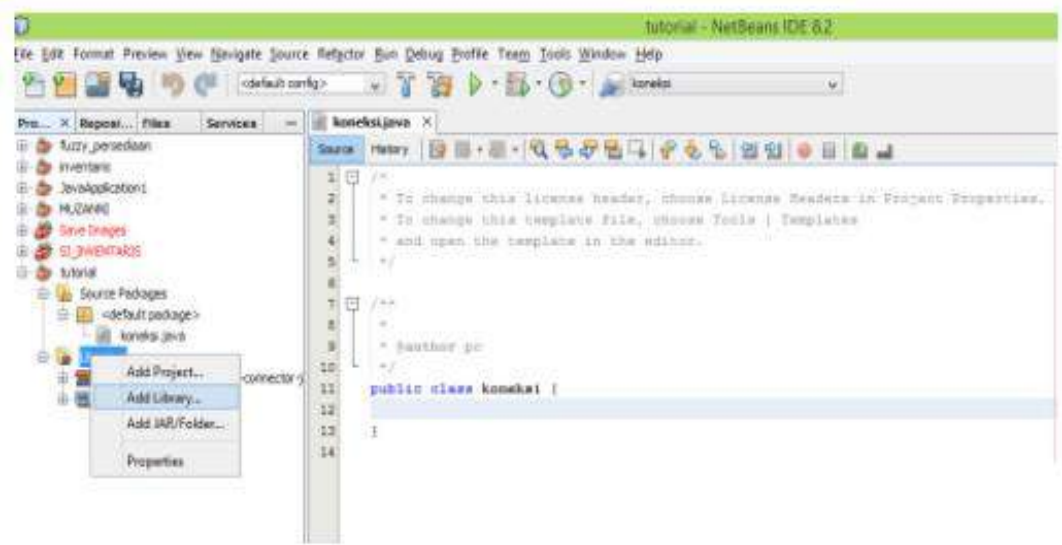
5. tambahkan library MySQL JDBC DRIVER untuk projek baru yang sudah kita buat tadi. untuk versi netbeans 8 ke atas itu sudah di sediakan, sedangkan untuk versi netbeansnya 7 kebawah harus di download.
cara menambahkan library ini yaitu : klik kanan pada folder library projek yang sudah kita buat tadinya, dan pilih add library, selanjutnya pilih library MySQL JDBC DRIVER.

Mengenal JDBC Dan JDBC Driver Untuk MySQL

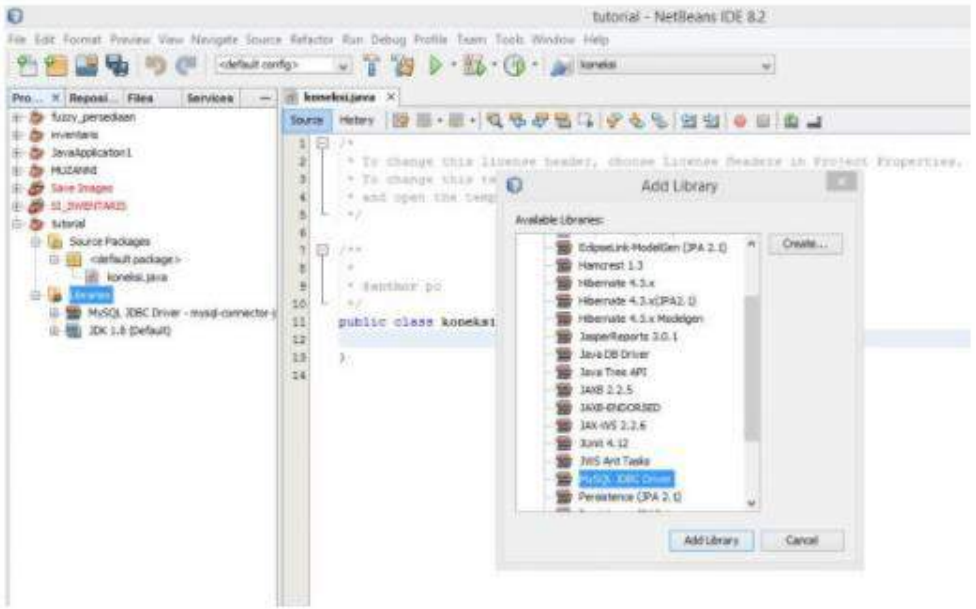
Java menyediakan standard API (application programming interface) untuk pengembangan program aplikasi basis data (database) yang disebut dengan JDBC API. JDBC adalah API Java untuk memanipulasi basis data. Dengan JDBC API, para pengembang aplikasi dan applet Java diberi kemudahan untuk mengakses berbagai tipe basis data dari berbagai penyedia basis data (database vendors) seperti MySQL Server, SQL Server, Oracle, Sybase dan sebagainya.

JDBC merupakan perantara antara Java dengan basis data. JDBC adalah sebuah spesifikasi yang menyediakan sekumpulan interface yang membolehkan akses portabel ke semua basis data. Dapat dikatakan pula bahwa JDBC hanya menyediakan interface standar, sedangkan masing-masing database vendors membuat driver yang diperlukan sebagai interface yang sebenarnya antara program Java (aplikasi, applet, servlet atau JSP) dengan basis data.





menambahkan library mysql jdbc driver

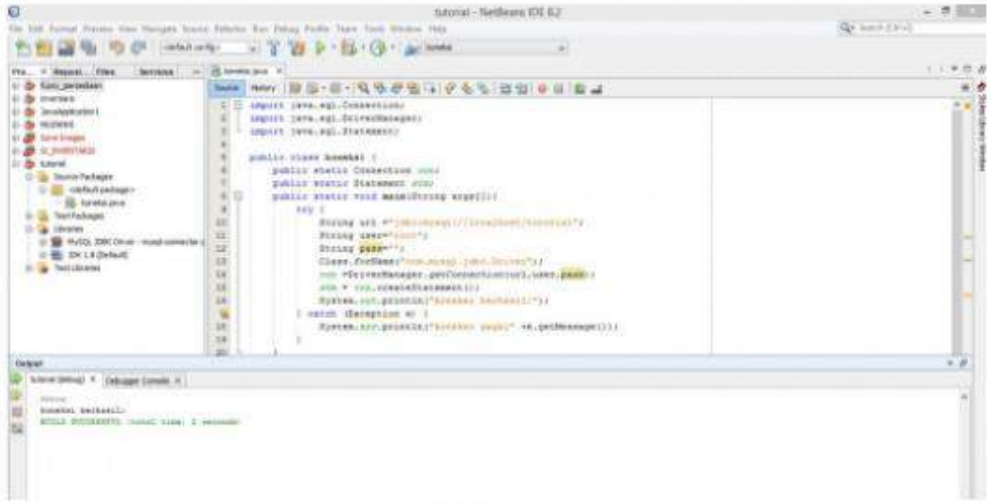


menambahkan library mysql jdbc driver

6. tambahkan sytax berikut kedalam file koneksi.java

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.Statement;

public class koneksi {
    public static Connection con;
    public static Statement stm;
    public static void main(String args[]){
        try {
            String url ="jdbc:mysql://localhost/tutorial";
            String user="root";
            String pass="";
            Class.forName("com.mysql.jdbc.Driver");
            con =DriverManager.getConnection(url,user,pass);
            stm = con.createStatement();
            System.out.println("koneksi berhasil;");
        } catch (Exception e) {
            System.err.println("koneksi gagal" +e.getMessage());
        }
    }

}
```

2. jalankan aplikasi yang yang tadi dan begini hasilnya :



outut berhasil

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Java untuk Pemrograman GUI

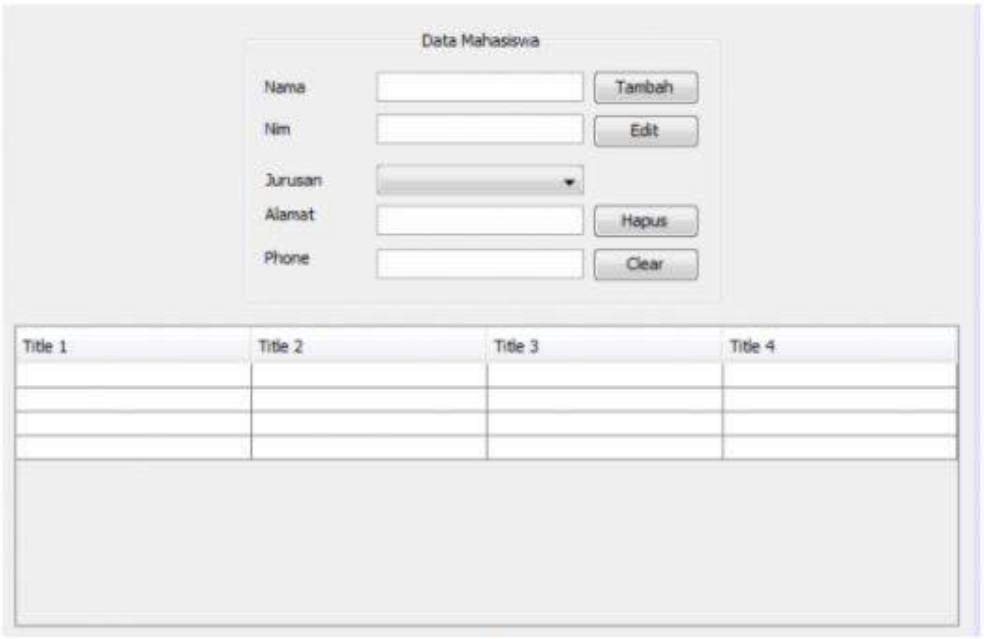
CYNTHIA HAYAT S.KOM., M.MSI

KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

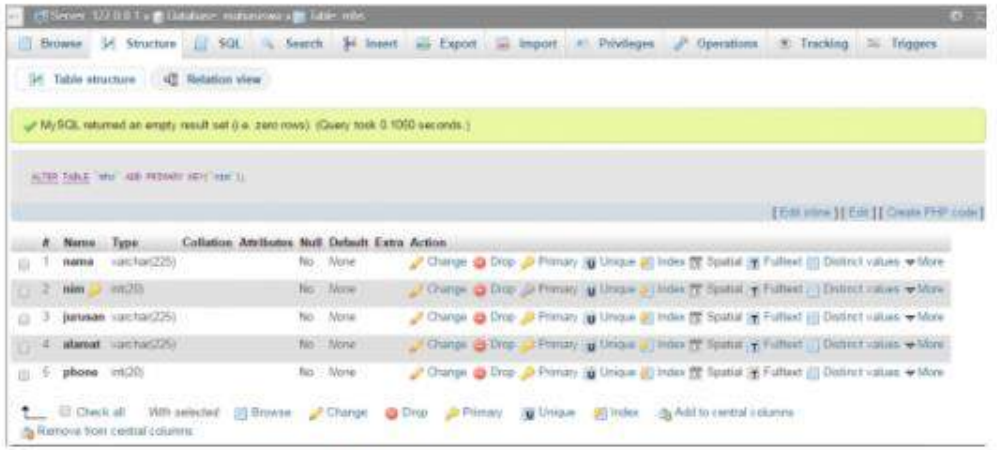
Cara Membuat CRUD Dengan Java MySQL

bagaimana cara membuat CRUD (Create, Read, Update, dan Delete) Dengan Java dan MySQL. dalam membuat aplikasi CRUD ini akan mengkoneksikan database dengan bantuan JDBC Driver.



Cara Membuat CRUD Dengan Java MySQL

Dalam membuat CRUD Java MySQL ini, sebelumnya harus membuat database MySQL terlebih dahulu. kira-kira seperti ini databasenya :



Cara Membuat CRUD Dengan Java MySQL

Membuat Project Baru Java Netbeans

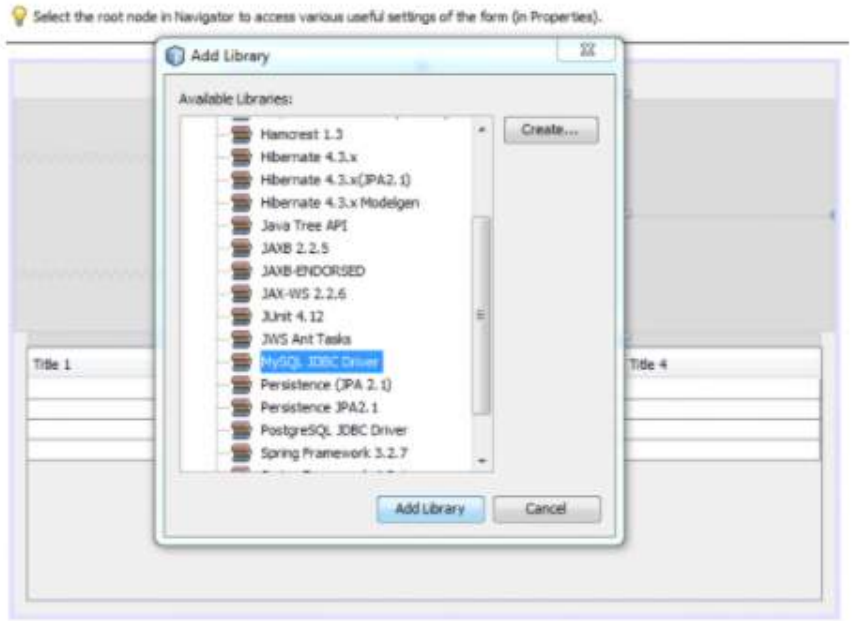
Buka Netbeans yang sudah di persiapkan, lalu tambahkan project baru. caranya : *File – New Project.. – Java – Java Application – klik Next* dan buatlah *nama project – hilangkan centang pada create main project* – lalu tekan *finish*. langkah selanjutnya kita akan membuat desain untuk form utuk aplikasi kita.

Membuat Jframe (Form untuk aplikasi CRUD Java MySQL)

dalam langkah ini kita akan membuat desain form untuk aplikasi CRUD Java MySQL. Untuk membuat form ini caranya : klik kana pada projet yang sudah kita buat tadi pilih *new – Jframe From – isikan nama* – klik *finish*. untuk desain form kira-kira seperti ini :

Membuat Koneksi Database MySQl (Config.java)

sebelum membuat class Config.java, pastikan terlebih dahulu kita sudah menambahkan Library MySQL JDBC Driver kedalam project kita. untuk menambahkan library caranya : klik kanan pada *library* yang ada dalam project – pilih *add library* – pilih *MySQL JDBC Driver* – klik *add library*.



Cara Membuat CRUD Dengan Java MySQL

ok, berikut syntax java untuk class Config.java

```java
// muzanni
// malasngoding.com
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class config {
    private static Connection mysqlconfig;
    public static Connection configDB()throws SQLException{
        try {
            String url="jdbc:mysql://localhost:3306/mahasiswa"; //url database
            String user="root"; //user database
            String pass=""; //password database
            DriverManager.registerDriver(new com.mysql.jdbc.Driver());
            mysqlconfig=DriverManager.getConnection(url, user, pass);
        } catch (Exception e) {
            System.err.println("koneksi gagal "+e.getMessage()); //perintah menampilk
        }
        return mysqlconfig;
    }
}
```

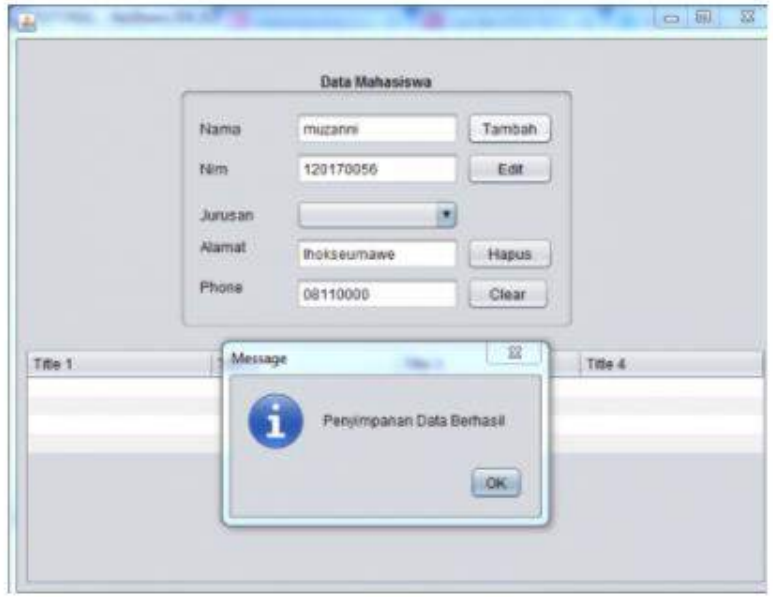
Cara Membuat CRUD Dengan Java MySQL

Create Data (tambah Data)

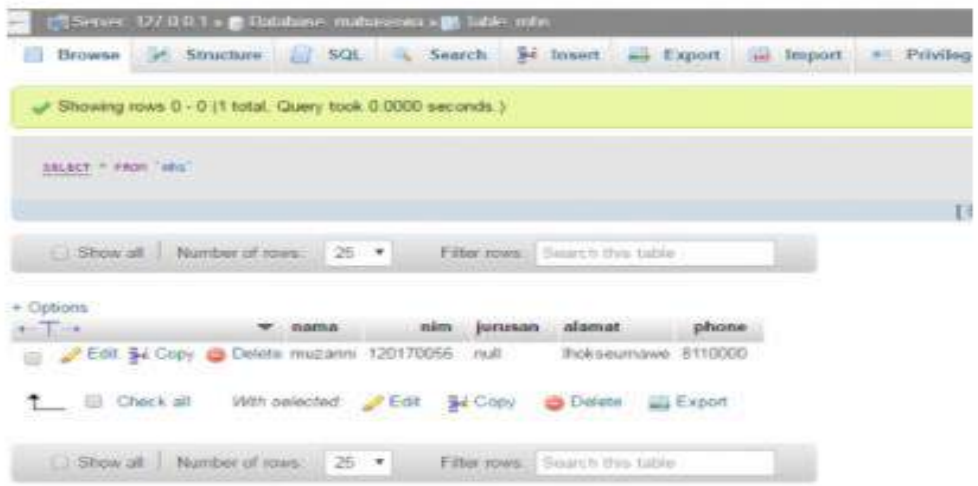
berikut code untuk membuat fungsi create (tambah) data dalam database. *klik kana pada tombol tambah – pilih event – Action – ActionPerformed.* lalu isikan syntax berikut :

```
try {
            String sql = "INSERT INTO mhs VALUES ('"+txt_nama.getText()+"','"+txt_nim.
            java.sql.Connection conn=(Connection)config.configDB();
            java.sql.PreparedStatement pst=conn.prepareStatement(sql);
            pst.execute();
            JOptionPane.showMessageDialog(null, "Penyimpanan Data Berhasil");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
```

untuk tes fungsi tambah data sudah bisa berjalan, Run project tadi, berikut hasilnya :



Tambah Data



Tambah Data

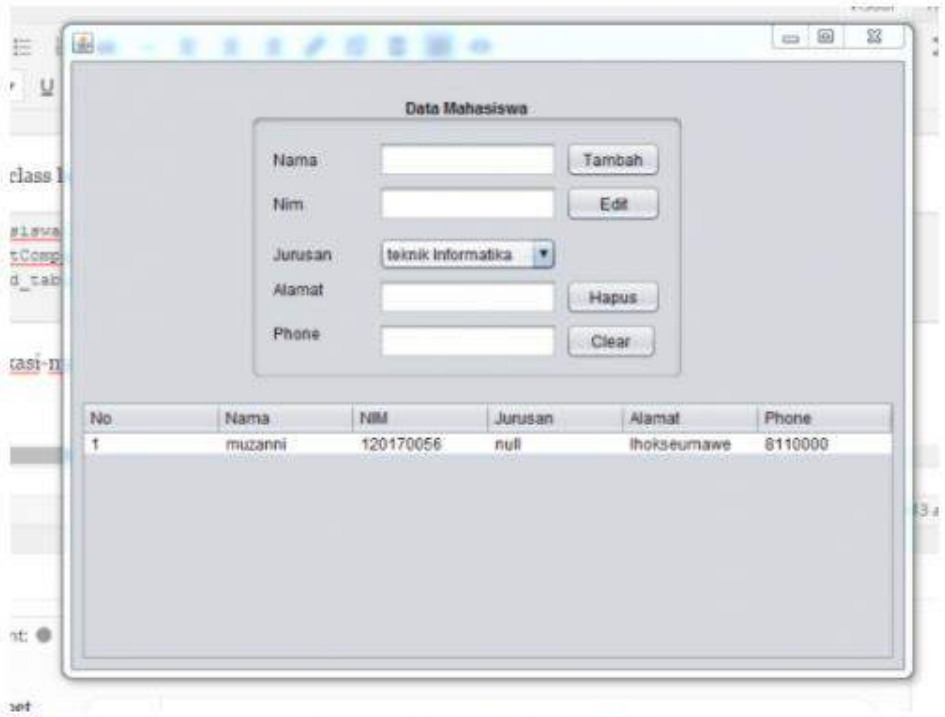
Menampilkan Data kedalam Tabel

```
2         // membuat tampilan model tabel
3         DefaultTableModel model = new DefaultTableModel();
4         model.addColumn("No");
5         model.addColumn("Nama");
6         model.addColumn("NIM");
7         model.addColumn("Jurusan");
8         model.addColumn("Alamat");
9         model.addColumn("Phone");
10
11        //menampilkan data database kedalam tabel
12        try {
13            int no=1;
14            String sql = "select * from mhs";
15            java.sql.Connection conn=(Connection)config.configDB();
16            java.sql.Statement stm=conn.createStatement();
17            java.sql.ResultSet res=stm.executeQuery(sql);
18            while(res.next()){
19                model.addRow(new Object[]{no++,res.getString(1),res.getString(2),res.
20            }
21            jTable1.setModel(model);
22        } catch (Exception e) {
23        }
```

deklarasikan class load_tabel ke dalam class mahasiswa, contohnya seperti ini :

```
1 public Mahasiswa() {
2         initComponents();
3         load_table();
4     }
```

alankan aplikasi-nya, maka tampilan akan seperti berikut :



Menampilkan data dalam tabel

Menghapus Isian Form setelah CRUD Data

setelah kita melakukan operasi misalkan menambahkan data, maka data yang kita tambahkan tadi masi ada pada form data mahasiswa, maka kita akan memberikan suatu fungsi () untuk menghapus otomatis setelah kita melakukan operasi tersebut. cara nya buat sebuah fungsi seperti syntak berikut :

```
private void kosong(){
        txt_alamat.setText(null);
        txt_nama.setText(null);
        txt_nim.setText(null);
        txt_pnone.setText(null);
        jComboBox1.setSelectedItem(this);
    }
```

deklarasikan dia kedalam class mahasiswa :

```
  public Mahasiswa() {
        initComponents();
        load_table();
        kosong();
    }
```

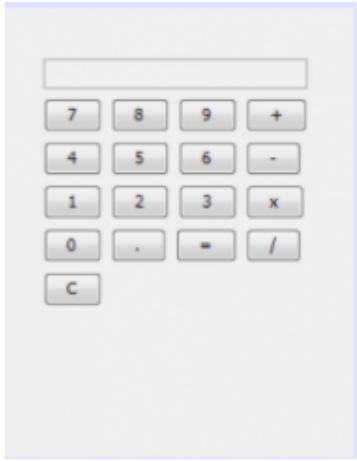
untuk menghapus otomatis tambahkan visibilitas kosong(), kedalam semua proses operasi, misalkan yang udah ada kedalam fungsi tambah data.

dan untuk menampilkan data secara otomoatis setelah melakukan operasi, misalkan menambahkan data tambahkan visibilitas load_tabel kedalam fungsi tambah data.

maka code untuk tambah data yang lengkapnya seperti berikut :

```
private void btn_tambahActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        try {
            String sql = "INSERT INTO mhs VALUES ('"+txt_nama.getText()+"','"+txt_nim
            java.sql.Connection conn=(Connection)config.configDB();
            java.sql.PreparedStatement pst=conn.prepareStatement(sql);
            pst.execute();
            JOptionPane.showMessageDialog(null, "Penyimpanan Data Berhasil");
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        load_table();
        kosong();
    }
```

Membuat Kalkulator Menggunakan Java



Membuat Kalkulator Menggunakan Java

4. deklarasikan dulu semua variabel yang dibutuhkan, contohnya seperti berikut :

```
String bil;
double jumlah,bil1,bil2;
int pilih;
```

5. langkah ini akan kita bahas cara memfungsikan semua tombol yang ada dalam desain aplikasi kita. caranya klik kanan pada tombol yang diinginkan pilih *event – action -action performed* atau bisa juga dengan mengklik 2 kali pada tombol yang di inginkan. berikut adalah syntak yang digunakan untuk memfungsikan semua tombol pada aplikasi kita.

```
private void bt_0ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        bil +="nilai"; // nilai ganti dengan angka 0, 1, 1
        txt_hasil.setText(bil); // menampilkan angka kedalam jTextField
}
```

gamba diatas merupakan desain sederhana kalkulator yang akan kita buat, sikahkan anda desain tamilan yang lebih menarik lagi untuk mengembangkan kalkulator yang dibahas dalam tutorial ini. 🙂

langsung kita menuju ke pembuatan aplikasi kalkulator menggunakan java.

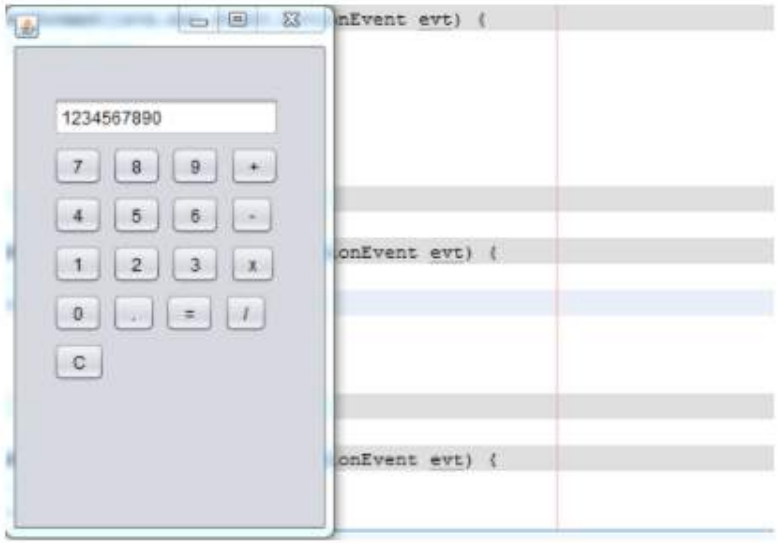
1. buat project baru pada texteditor netbeans, caranya buka netbeans, file – new project – java – java application – tuliskan nama – hilangkan centangan craeting main class – dan fininsh
2. buat sebuah class jFrame, class ini dimana tempat dimana kita akan mendesain dan membuat aplikasi kalkulator dengan java. caranya : klik kanan pada project aplikasi – new – jFrame From- tuliskan nama dan finish
3. desain tampilan kalkulator sesuai dengan keinginan, penulis hanya mendesain tamilan yang sederhana seperti berikut :

ok berikut syntax untuk masing- masing tombol :

```
private void bt_0ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 0
        bilangan +="0";
        txt_hasil.setText(bilangan);
}

private void bt_1ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 1
        bilangan +="1";
        txt_hasil.setText(bilangan);
}

private void bt_2ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 2
        bilangan +="2";
        txt_hasil.setText(bilangan);
    }

private void bt_3ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 3
        bilangan +="3";
        txt_hasil.setText(bilangan);
    }

    private void bt_4ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 4
        bilangan +="4";
        txt_hasil.setText(bilangan);
    }
```

```

    private void bt_5ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 5
        bilangan +="5";
        txt_hasil.setText(bilangan);
    }

    private void bt_6ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 6
        bilangan +="6";
        txt_hasil.setText(bilangan);
    }

    private void bt_7ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 7
        bilangan +="7";
        txt_hasil.setText(bilangan);
    }

    private void bt_8ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 8
        bilangan +="8";
        txt_hasil.setText(bilangan);
    }

    private void bt_9ActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol 9
        bilangan +="9";
        txt_hasil.setText(bilangan);
    }

private void bt_clearActionPerformed(java.awt.event.ActionEvent evt) {
        // fungsi untuk tombol c
        txt_hasil.setText(null);
        bil1=0.0;
        bil2=0.0;
        jumlah=0.0;
        bilangan="";
     }
```

kalo aplikasinya di run maka seperti berikut lah kira-kira akan muncul :

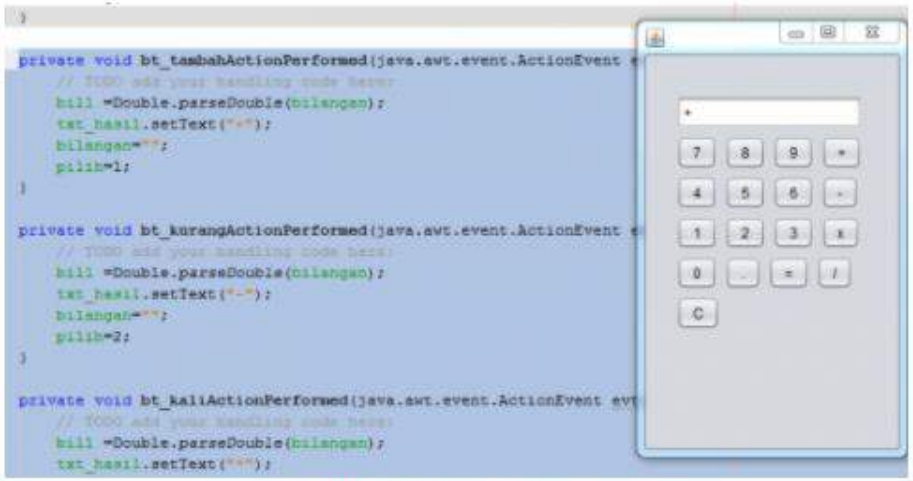


Membuat Kalkulator Menggunakan Java

setiap tombol yang di klik makan akan memunculkan fungsi yang sudah di masukkan dalam syntax di atas. kecuali pada tombol penjumlahan, pengurangan, kali, bagi, dan sama dengan.

6. membuat fungsi aritmatika pada kalkulator java
   sama halnya dengan membuat fungsi tombol untuk angka di atas, namun syntax untuk fungsinya berbeda. berikut syntak yang digunakan untuk fungsi aritmatika :

```
1  private void bt_tambahActionPerformed(java.awt.event.ActionEvent evt) {
2          // TODO add your handling code here:
3          bil1 =Double.parseDouble(bilangan);
4          txt_hasil.setText("+");
5          bilangan="";
6          pilih=1;
7      }
8
9      private void bt_kurangActionPerformed(java.awt.event.ActionEvent evt) {
10         // TODO add your handling code here:
11         bil1 =Double.parseDouble(bilangan);
12         txt_hasil.setText("-");
13         bilangan="";
14         pilih=2;
15     }
16
17     private void bt_kaliActionPerformed(java.awt.event.ActionEvent evt) {
18         // TODO add your handling code here:
19         bil1 =Double.parseDouble(bilangan);
20         txt_hasil.setText("*");
21         bilangan="";
22         pilih=3;
23     }
24
25     private void bt_bagiActionPerformed(java.awt.event.ActionEvent evt) {
26         // TODO add your handling code here:
27         bil1 =Double.parseDouble(bilangan);
28         txt_hasil.setText("/");
29         bilangan="";
30         pilih=4;
31     }
```



Membuat Kalkulator Menggunakan Java

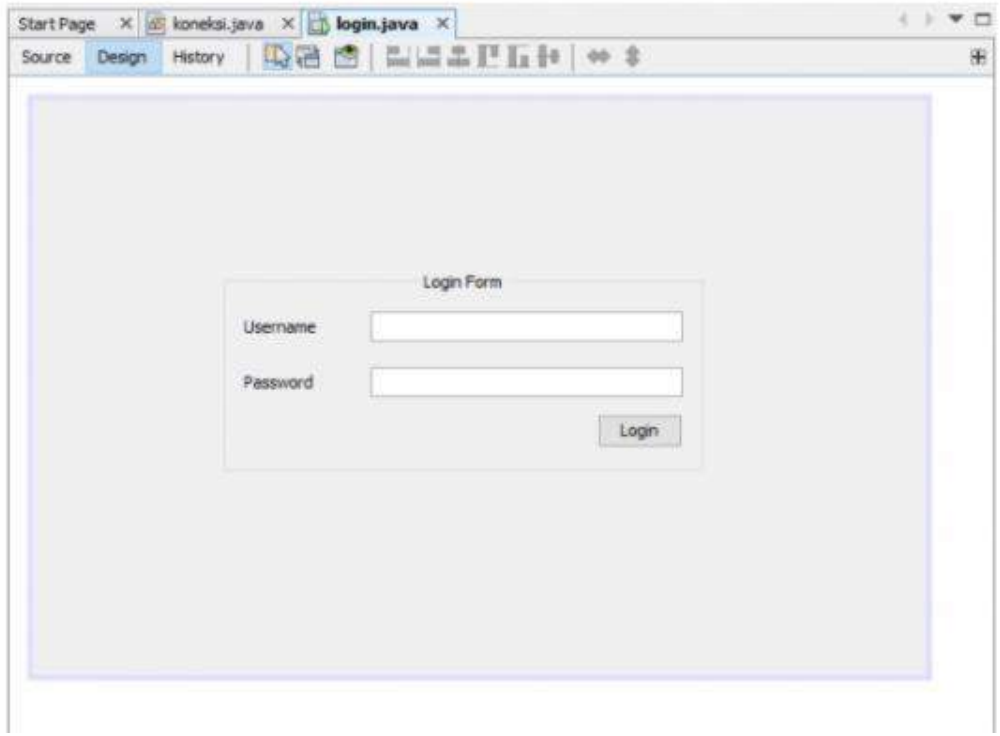
ketika tombol operasi nya di klik maka akan dimunculkan operasi mana yang akan dilakukan, tambah misalkan.

7. ini merupakan langkah terakhir, membuat fungsi untuk melakukan perhitungan. fungsi ini di letakkan dalam tombol (=) sama dengan. berikut syntax yang saya gunakan untuk melakukan operasi aritmatika :

```
1  private void bt_hasilActionPerformed(java.awt.event.ActionEvent evt) {
2          // TODO add your handling code here:
3          switch(pilih){
4          case 1:
5              bil2 = Double.parseDouble(String.valueOf(txt_hasil.getText()));
6              jumlah = bil1+bil2;
7              bilangan = Double.toString(jumlah);
8              break;
9          case 2:
10             bil2 =Double.parseDouble(String.valueOf(txt_hasil.getText()));
11             jumlah = bil1 - bil2;
12             bilangan = Double.toString(jumlah);
13             break;
14         case 3:
15             bil2 = Double.parseDouble(String.valueOf(txt_hasil.getText()));
16             jumlah = bil1 * bil2;
17             bilangan = Double.toString(jumlah);
18             break;
19         case 4:
20             bil2 = Double.parseDouble(String.valueOf(txt_hasil.getText()));
21             jumlah = bil1 / bil2;
22             bilangan = Double.toString(jumlah);
23             break;
24         }
25         txt_hasil.setText(bilangan);
26     }
```

Cara Membuat Login Pada Java Mysql

Sering kita lihat di kebanyakan aplikasi kita harus login terlebih dahulu baru bisa mengelola data yang ada dalam aplikasi tersebut. login ini adalah pembatas antara sistem dengan user. untuk membuat login ini kita membutuhkan database MySQL dan file library MySQL JDBC DRIVER serta satu buah file java untuk mengkoneksikan antara java dan database.



langkah pertama buat dulu database dan table (mohon maaf disini saya tidak mengajarkan cara membuat database dengan mysql). kira-kira databasenya seperti berikut :

```sql
-- phpMyAdmin SQL Dump
-- version 3.5.2.2
-- http://www.phpmyadmin.net

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Basis data: `malasngoding`
--

-- --------------------------------------------------------

--
-- Struktur dari tabel `admin`
--

CREATE TABLE IF NOT EXISTS `admin` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `username` varchar(20) NOT NULL,
  `password` varchar(20) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;

--
-- Dumping data untuk tabel `admin`
--

INSERT INTO `admin` (`id`, `username`, `password`) VALUES
(1, 'admin', '123');

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

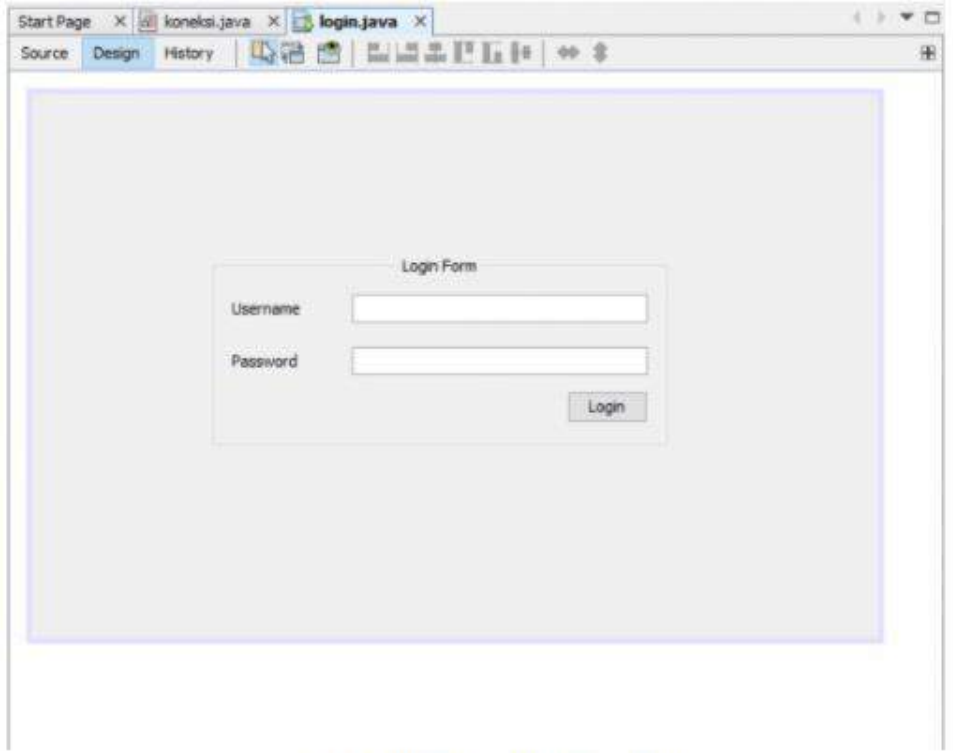
ok, saya anggap masalah database dan membuat project java baru pada netbeans sudah selesai. langkah selanjutnya membuat class *koneksi.java* dan mengimport library mysql.jadbc.driver.

mysql.jdbc.driver sendiri merupakan suatu library yang berfungsi sebagai penghubung antara java dan database mysql. cara menambahkan library ini kedalam project kita yaitu klik kanan pada *folder library* yang ada dalam project kita pilih *add library – mysql jdbc driver dan add library.* selesai kita menambahkan library mysql jdbc driver kedalam projek kita. selanjutnya buat sebuah class ***koneksi.java*** . class ini berfungsi untuk mengkoneksikan database dengan java. untuk lebih jelas tentang membuat koneksi java

berikut syntax untuk membuat class koneksi.java

```java
import java.sql.DriverManager;
import java.sql.Statement;
import javax.swing.JOptionPane;

public class koneksi {
    Connection con;
    Statement stm;

    public void config(){
        try {
            Class.forName("com.mysql.jdbc.Driver");
            con=DriverManager.getConnection("jdbc:mysql://localhost/malasngoding", "r
            stm = con.createStatement();
        } catch (Exception e) {
            JOptionPane.showMessageDialog(null, "koneksi gagal "+e.getMessage());
        }
    }
}
```

ok, class koneksi.java ini sudah selesai, langkah selanjutnya kita akan membuat form login beserta aksi login. klik kanan pada project yang sudah kita buat tadi, pilin new jFrame form, tuliskan nama dan selesai. untuk desain kira-kira sederhana-nya seperti ini :



Cara Membuat Login Java Mysql

untuk username menggunakan *JTextField*
dan password menggunakan *JPasswordField*

selanjutnya buka di bagian source dan isikan syntax berikut :

```
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;
```

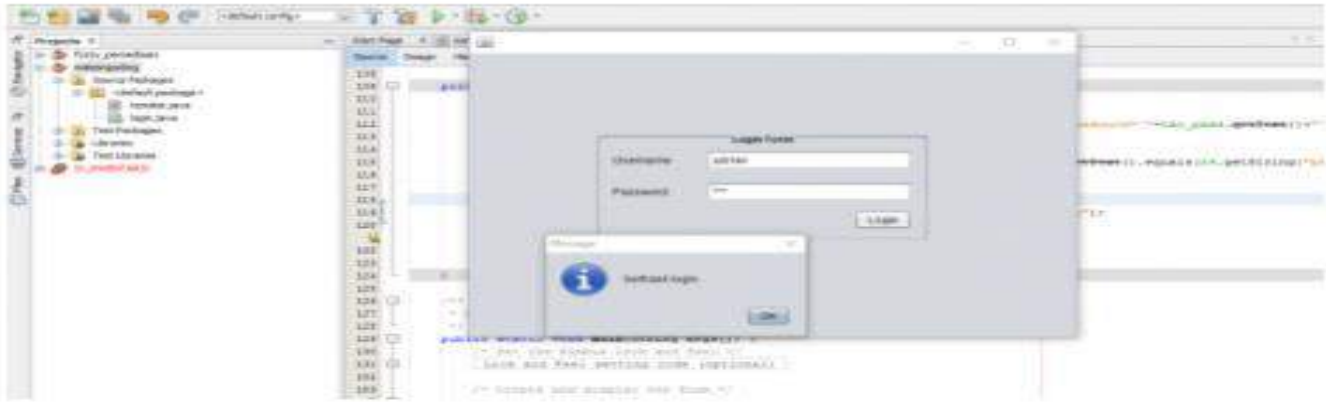
syntax ini fungsinya untuk memanggil beberapa library java yang kita gunakan untuk *membuat login java*. selanjutnya kita akan mendeklarasikan beberapa fungsi yang dibutuhkan untuk membuat login ini :

```
public class login extends javax.swing.JFrame {
    // deklarasi
    Connection con;
    Statement stat;
    ResultSet rs;
    String sql;

    public login() {
        initComponents();
        //pemanggilan fungsi koneksi database yang sudah kita buat pada class koneksi
        koneksi DB = new koneksi();
        DB.config();
        con = DB.con;
        stat = DB.stm;
    }
```

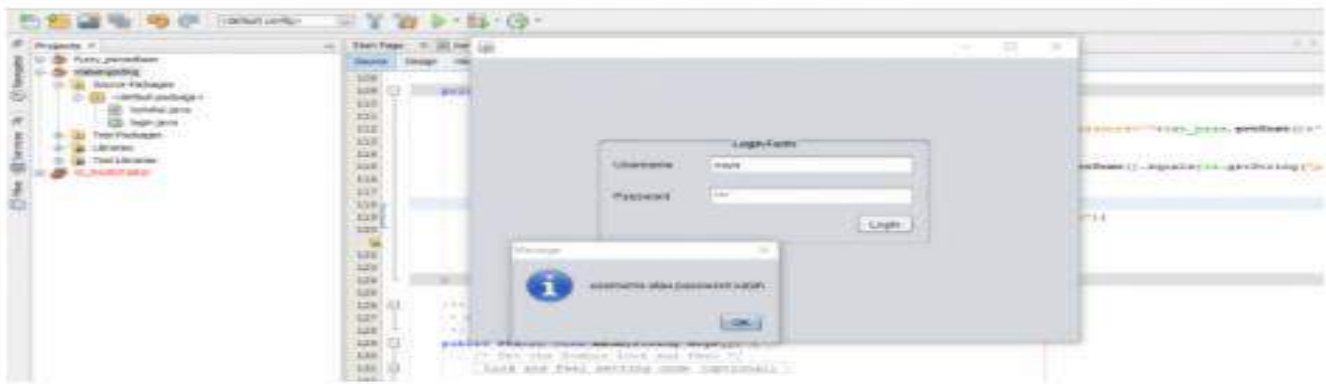
langkah terakhir membuat fungsi untuk login. klik kanan ada tombol login pilih event – action – actionPerformed dan isikan syntax berikut :

```
private void bt_loginActionPerformed(java.awt.event.ActionEvent evt) {

        try {
            sql = "SELECT * FROM admin WHERE username='"+txt_name.getText()+"' AND pa
            rs = stat.executeQuery(sql);
            if(rs.next()){
                if(txt_name.getText().equals(rs.getString("username")) && txt_pass.ge
                    JOptionPane.showMessageDialog(null, "berhasil login");
                }
            }else{
                    JOptionPane.showMessageDialog(null, "username atau password salah
                }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }
```

ok selesai kita membuat Membuat Login Java Mysql. berikut hasil demo aplikasinya :



Login Berhasil



Username dan Password Tidak Sesuai

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana

Membuat Aplikasi Sederhana pada Java

CYNTHIA HAYAT S.KOM., M.MSI
KRIDA WACANA CHRISTIAN UNIVERSITY
Faculty of Engineering and Computer Science
Departement of Information System

UKRIDA
Universitas Kristen Krida Wacana

Cara Membaca dan Menulis File di Java

Mengapa Kita Butuh File?

File adalah media penyimpanan eksternal bagi program Java.

Lalu mengapa kita membutuhkannya?

Ya, soalnya kalau kita hanya simpan data di dalam variabel saja.. Data itu akan hilang setelah program ditutup.

Karena itu, kita membutuhkan file untuk menyimpan data.

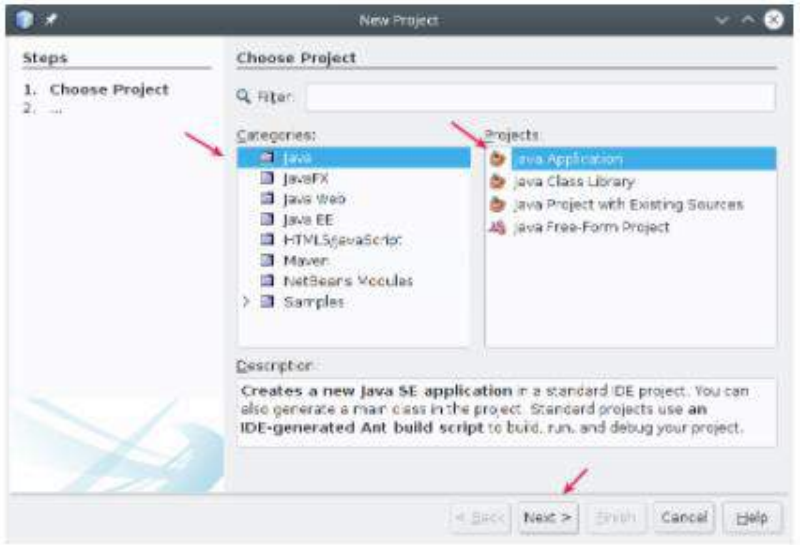
Walaupun program ditutup, data akan tetap ada di dalam file dan juga bisa kita baca lagi di program.

Bagaimana cara bacanya?

Silahkan simak:

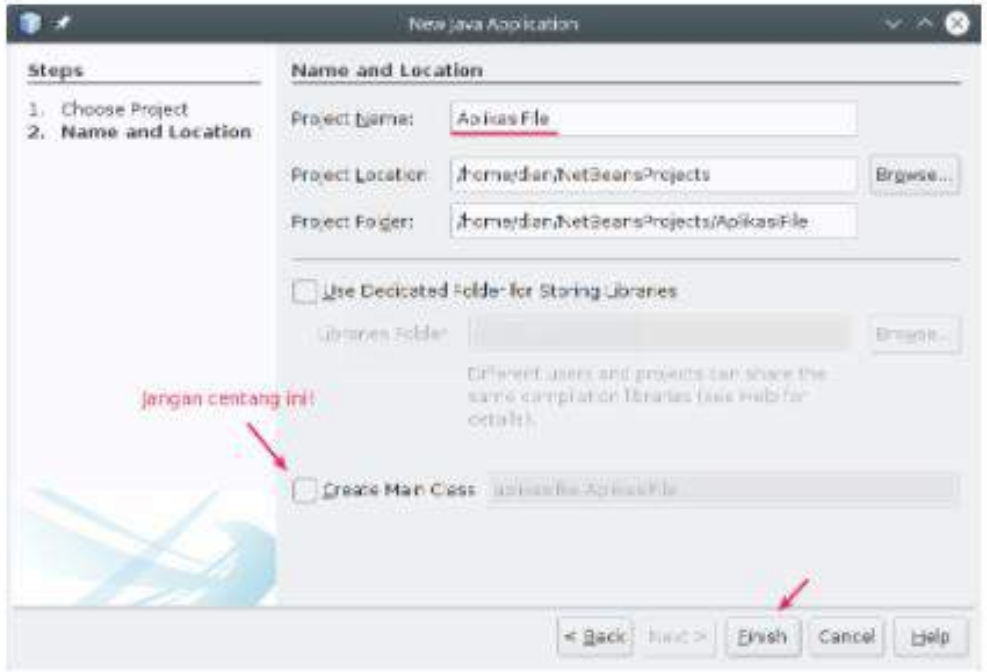
Cara Membaca File di Java

Silahkan buat project baru di Netbeans..

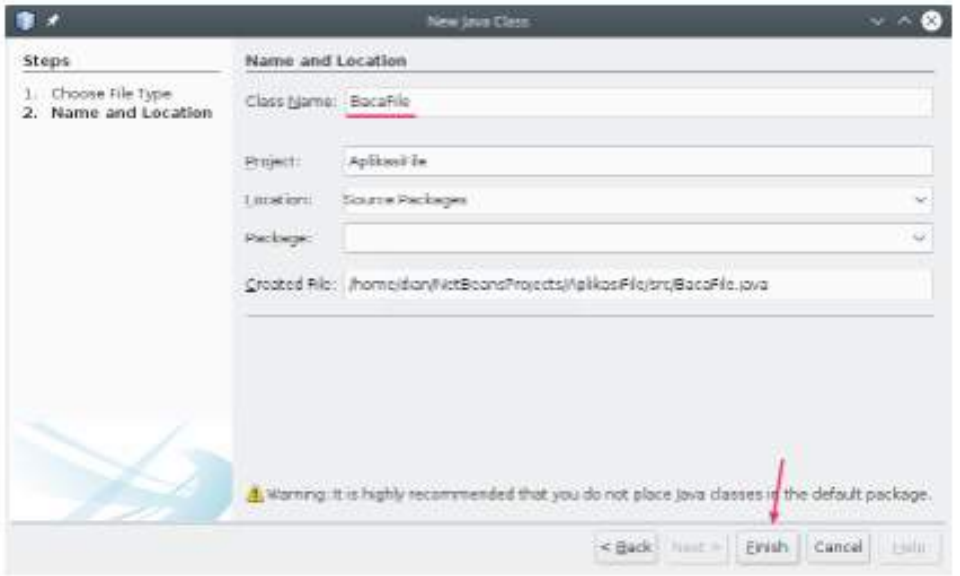


..berikan nama `AplikasiFile`.

..berikan nama `AplikasiFile`.



Setelah itu buat class baru di dalam `<default pacakage>` dengan nama `BacaFile`.



Isilah class tersebut dengan kode berikut:

```java
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class BacaFile {

    public static void main(String[] args) {

        String fileName = "src/puisi.txt" ;

        try {
            // membaca file
            File myFile = new File(fileName);
            Scanner fileReader = new Scanner(myFile);

            // cetak isi file
            while(fileReader.hasNextLine()){
                String data = fileReader.nextLine();
                System.out.println(data);
            }

        } catch (FileNotFoundException e) {
            System.out.println("Terjadi Kesalahan: " + e.getMessage());
            e.printStackTrace();
        }

    }
}
```

Pada program tersebut, kita membutuhkan tiga class untuk membantu kita membaca file.

Class `File` untuk membaca file, lalu class `FileNotFoundException` untuk mengatasi saat file tidak ditemukan.

Terakhir ada class `Scanner` untuk membaca isi File.

```java
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
```

Berikutnya kita membuat variabel dengan `fileName` untuk menyimpan nama file yang akan dibaca.

Pada blok `try..catch`, kita membaca file dan mencetak isinya.

Apabila file tidak ditemukan, maka blok `catch` akan dieksekusi.

Pada program di atas, kita membaca isi file dengan bantuan class `Scanner`. Selain menggunakan class ini, kita juga bisa menggunakan `BufferReader`.

Tapi menurut saya:

Menggunakan class `Scanner` lebih mudah dibandingkan dengan `BufferReader`.

Baiklah, berikutnya kita akan coba jalankan programnya:

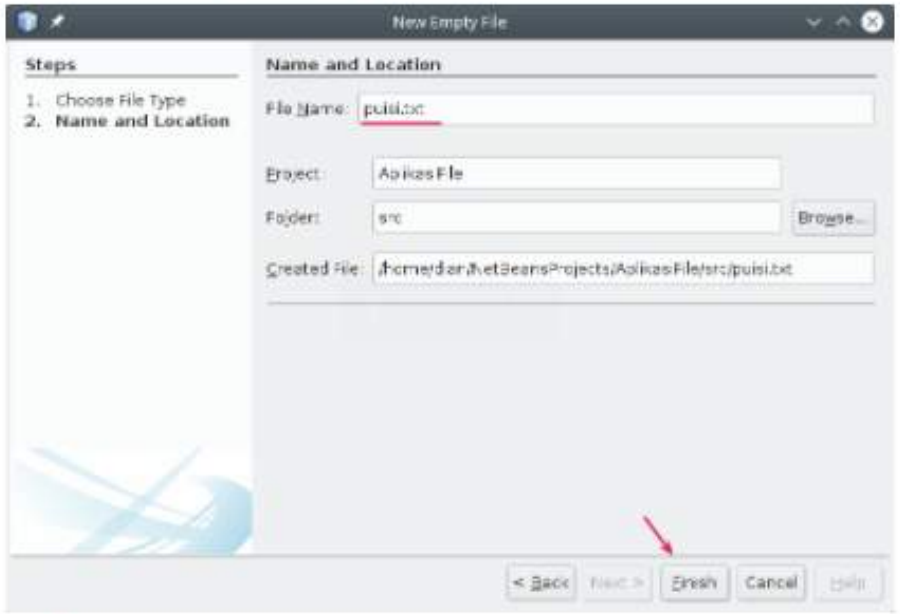
Klik kanan pada kode program lalu pilih **Run File** atau bisa juga tekan `Shift` + `F6`.

Hasilnya error, karena file `puisi.txt` belum ada.

Oke..

Kalau begitu, mari kita buat file `puisi.txt`. Klik kanan pada `<default package>` kemudian pilih **New->Empty File**.

Isi namanya dengan `puisi.txt`.

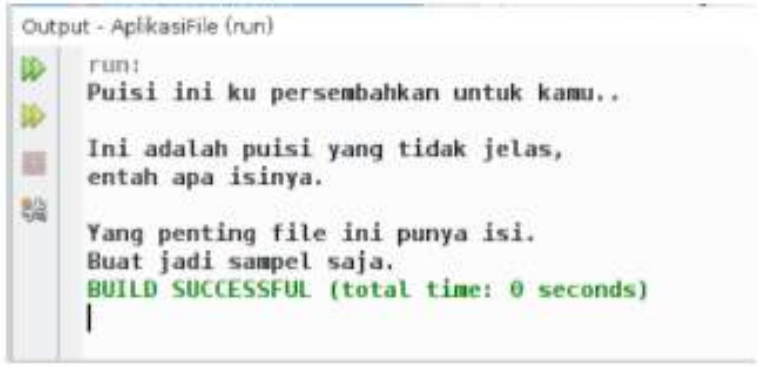


Setelah itu, isi file dengan teks berikut:

```
Puisi ini ku persembahkan untuk kamu..

Ini adalah puisi yang tidak jelas,
entah apa isinya.

Yang penting file ini punya isi.
Buat jadi sampel saja.
```

Sebenarnya kamu bebas mengisi dengan teks apapun.

Sekarang coba jalankan lagi programnya:



Berhasil!

Kita sudah bisa membaca file dengan Java.

Lalu gimana cara menulisnya?

Cara Menulis File di Java

Kita membutuhkan class `FileWriter` dan `IOException` untuk menulis file dengan Java.

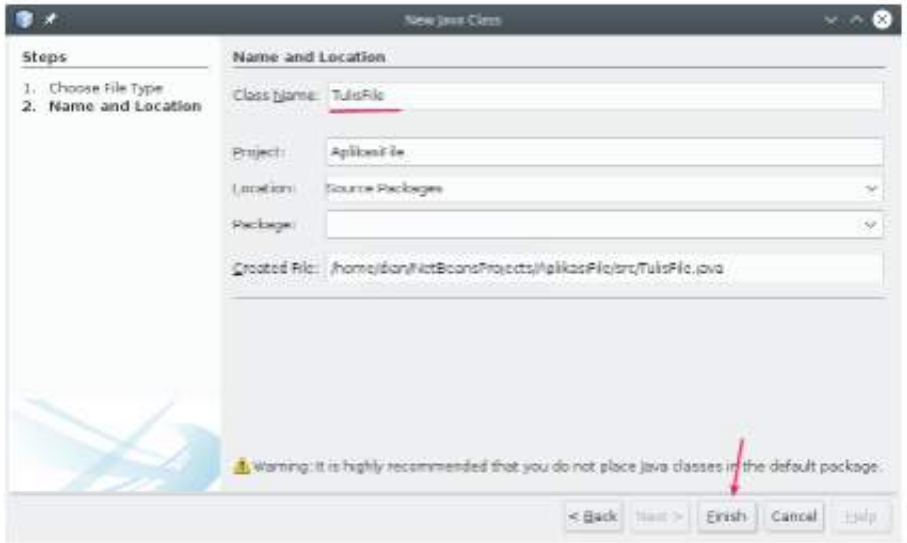
Class `FileWriter` berfungsi untuk menulis file baru. Apa bila filenya sudah ada ia akan menindih dengan file yang baru.

Kemudian class `IOException` kita butuhkan untuk mengatasi kalau terjadi error dalam penulisan file.

Misalnya, penulisannya gagal karena aksesnya dilarang.

Untuk lebih jelasnya, mari kita coba dalam program.

Buatlah class di dalam `<default package>` dengan nama `TulisFile`.



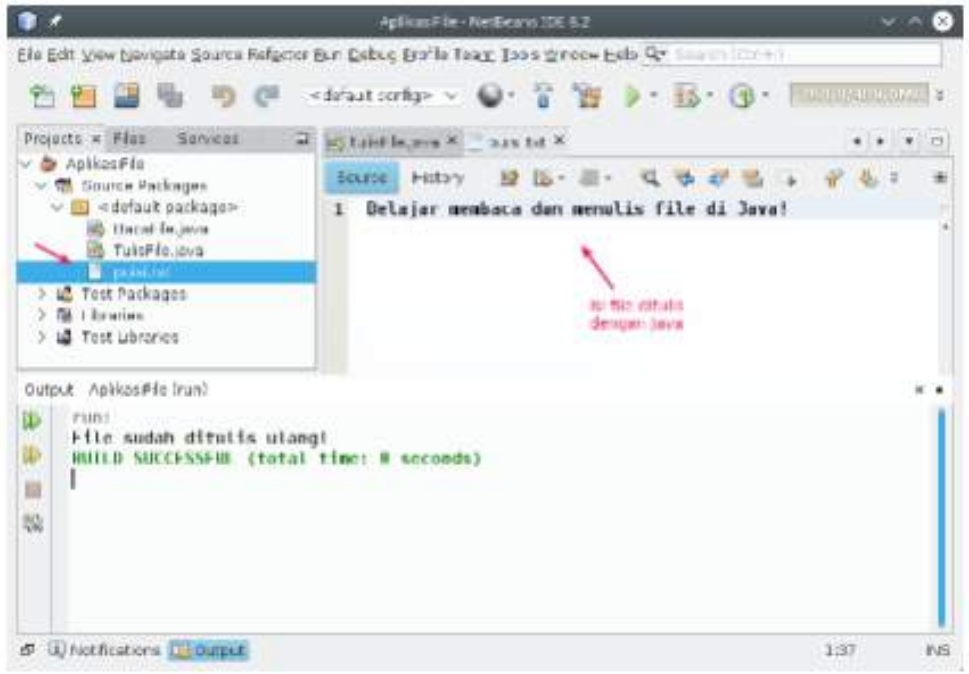
Setelah itu, isi dengan koder berikut:

```java
import java.io.FileWriter;
import java.io.IOException;

public class TulisFile {

    public static void main(String[] args) {

        String fileName = "src/puisi.txt";
        String fileContent = "Belajar membaca dan menulis file di Java!";

        try {
            FileWriter fileWriter = new FileWriter(fileName);
            fileWriter.write(fileContent);
            fileWriter.close();

            System.out.println("File sudah ditulis ulang!");
        } catch (IOException e) {
            System.out.println("Terjadi kesalahan karena: " + e.getMessage()
        }
    }
}
```

Setelah itu, coba jalankan programnya.

Setelah itu, coba jalankan programnya.

Maka hasilnya:



Membuat Aplikasi TodoList (CRUD) dengan Java dan File

Ini adalah lanjutan dari tutorial sebelumnya:

- Cara Membaca dan Menulis File di Java.

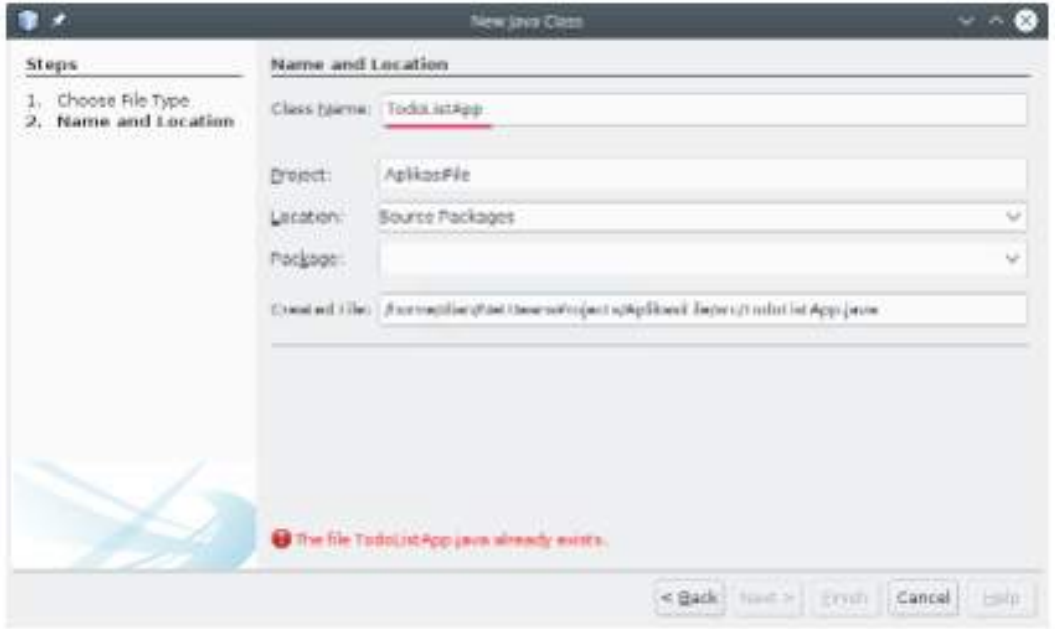
Pada tutorial ini, kita akan belajar membuat alikasi TodoList berbasis teks dengan memanfaatkan file sebagai media penyimpanan.

Langsung saja.

Mari kita mulai...

Membuat Class Baru

Buatlah class baru di dalam project `AplikasiFile` dengan nama `TodoListApp`.



Kemudian isi dengan kode berikut:

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class TodoListApp {

    // deklarasi variabel global
    static String fileName;
    static ArrayList<String> todoLists;
    static boolean isEditing = false;
    static Scanner input;

    public static void main(String[] args) {

    }

    static void clearScreen(){
    }

    static void showMenu() {
    }

    static void backToMenu() {
    }

    static void readTodoList() {
    }

    static void showTodoList() {
    }

    static void addTodoList() {
    }

    static void editTodoList() {
    }

    static void deleteTodoList() {
    }

}
```

Membuat Fungsi Clear Screen di Java

Fungsi ini kita butuhkan untuk membersihkan layar. Silahkan isi kode untuk fungsi `clearScreen()` seperti berikut ini:

```
static void clearScreen() {
    try {
        final String os = System.getProperty("os.name");
        if (os.contains("Windows")) {
            // clear screen untuk windows
            new ProcessBuilder("cmd", "/c", "cls")
                    .inheritIO()
                    .start()
                    .waitFor();
        } else {
            // clear screen untuk Linux, Unix, Mac
            Runtime.getRuntime().exec("clear");
            System.out.print("\033[H\033[2J");
            System.out.flush();
        }
    } catch (final Exception e) {
        System.out.println("Error karena: " + e.getMessage());
    }
}
```

Kalau kita perhatikan, fungsi ini akan menjalankan perintah `cls` dan `clear`.

Jika kamu membuka program ini dari Windows, maka perintah `cls` yang akan digunakan. Dan `clear` untuk Linux, Unix, dan Mac.

Tapi fungsi ini tidak bisa bekerja pada jendela output di Netbeans.

Membuat Fungsi `showMenu()`

Berikutnya kita membutuhkan fungsi `showMenu()` untuk menampilkan menu apa saja yang ada di aplikasi.

Silahkan isi kode untuk fungsi `showMenu()` seperti berikut ini:

```
static void showMenu() {
    System.out.println("=== TODO LIST APP ===");
    System.out.println("[1] Lihat Todo List");
    System.out.println("[2] Tambah Todo List");
    System.out.println("[3] Edit Todo List");
    System.out.println("[4] Hapus Todo List");
    System.out.println("[0] Keluar");
    System.out.println("----------------------");
    System.out.print("Pilih menu> ");

    String selectedMenu = input.nextLine();

    if (selectedMenu.equals("1")) {
        showTodoList();
    } else if (selectedMenu.equals("2")) {
        addTodoList();
    } else if (selectedMenu.equals("3")) {
        editTodoList();
    } else if (selectedMenu.equals("4")) {
        deleteTodoList();
    } else if (selectedMenu.equals("0")) {
        System.exit(0);
    } else {
        System.out.println("Kamu salah pilih menu!");
        backToMenu();
    }
}
```

Selain menampilkan menu, fungsi ini juga akan menentukan fungsi mana yang akan dieksekusi saat menu dipilih.

Misal, user memilih menu `1`, maka fungsi `showTodoList()` yang akan dieksekusi.

Membuat Fungsi `backToMenu()`

Fungsi ini kita butuhkan untuk kembali ke Menu Utama. Silahkan isi kode di dalam fungsi `backToMenu()` seperti ini:

```
static void backToMenu() {
    System.out.println("");
    System.out.print("Tekan [Enter] untuk kembali..");
    input.nextLine();
    clearScreen();
}
```

Fungsi ini akan meminta user untuk menekan `Enter`. Setelah itu akan membersihkan layar dan membuka menu Utama.

Walaupun kita tidak memanggil fungsi `showMenu()` di sana, tapi tetap akan kembali ke menu.

Ini karena kita akan kembali ke *main loop*.

Apa itu *main loop*?

Nanti kita akan buat dan bahas..

Membuat Fungsi `readTodoList()`

Fungsi ini kita butuhkan untuk membaca isi File. Silahkan isi kode untuk fungsi `readTodoList()` seperti berikut ini:

```
static void readTodoList() {
    try {
        File file = new File(fileName);
        Scanner fileReader = new Scanner(file);

        // load isi file ke dalam array todoLists
        todoLists.clear();
        while (fileReader.hasNextLine()) {
            String data = fileReader.nextLine();
            todoLists.add(data);
        }

    } catch (FileNotFoundException e) {
        System.out.println("Error karena: " + e.getMessage());
    }
}
```

Fungsi ini akan membaca file, lalu isinya akan disimpan dalam variabel array `todoLists`.

Fungsi ini akan kita panggil saat kita ingin membaca ulang isi file.

Membuat Fungsi `showTodoList()`

Fungsi ini kita butuhkan untuk menampilkan isi dari file `todolist.txt`.

Silahkan isi kode fungsi `showTodoList()` seperti berikut ini:

```
static void showTodoList() {
    clearScreen();
    readTodoList();
    if (todoLists.size() > 0) {
        System.out.println("TODO LIST:");
        int index = 0;
        for (String data : todoLists) {
            System.out.println(String.format("[%d] %s", index, data));
            index++;
        }
    } else {
        System.out.println("Tidak ada data!");
    }

    if (!isEditing) {
        backToMenu();
    }
}
```

Fungsi ini hanya melakukan print atau cetak isi file. Kita memanggil fungsi `readTodoList()` di sini, untuk membaca isi file.

Membuat Fungsi `addTodoList()`

Berikutnya, silahkan isi kode untuk fungsi `addTodoList()` seperti ini:

```
static void addTodoList() {
    clearScreen();

    System.out.println("Apa yang ingin kamu kerjakan?");
    System.out.print("Jawab: ");
    String newTodoList = input.nextLine();

    try {
        // tulis file
        FileWriter fileWriter = new FileWriter(fileName, true);
        fileWriter.append(String.format("%s%n", newTodoList));
        fileWriter.close();
        System.out.println("Berhasil ditambahkan!");
    } catch (IOException e) {
        System.out.println("Terjadi kesalahan karena: " + e.getMessage());
    }

    backToMenu();
}
```

Fungsi ini akan meminta user untuk menginputkan data todo list, kemudian menuliskannya ke dalam file.

Cara kerja fungsi ini ada beberapa langkah:

1. Meminta user untuk memilih nomer indeks dari data (indeks dalam array `todolist`)
2. Memeriksa indeks
3. Meminta user untuk menginputkan data baru
4. Menyimpan data baru ke dalam array `todolist`
5. Menulis ulang semua isi array `todolist` ke dalam file

Membuat Fungsi `editTodoList()`

Fungsi ini kita butuhkan untuk mengubah isi data. Silahkan isi kode untuk fungsi `editTodoList()` seperti berikut ini:

```
static void editTodoList() {
    isEditing = true;
    showTodoList();

    try {
        System.out.println("----------------");
        System.out.print("Pilih Indeks> ");
        int index = Integer.parseInt(input.nextLine());

        if (index > todoLists.size()) {
            throw new IndexOutOfBoundsException("Kamu memasukan data yang sa
        } else {

            System.out.print("Data baru: ");
            String newData = input.nextLine();

            // update data
            todoLists.set(index, newData);

            System.out.println(todoLists.toString());

            try {
                FileWriter fileWriter = new FileWriter(fileName, false);

                // write new data
                for (String data : todoLists) {
                    fileWriter.append(String.format("%s%n", data));
                }
                fileWriter.close();

                System.out.println("Berhasil diubah!");
            } catch (IOException e) {
                System.out.println("Terjadi kesalahan karena: " + e.getMessa
            }
        }
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }

    isEditing = false;
    backToMenu();
}
```

Membuat Fungsi `deleteTodoList()`

Fungsi ini kita butuhkan untuk menghapus data todolist. Silahkan tulis kode untuk fungsi `deleteTodoList()` seperti berikut ini.

```
static void deleteTodoList() {
    isEditing = true;
    showTodoList();

    System.out.println("------------------");
    System.out.print("Pilih Indeks> ");
    int index = Integer.parseInt(input.nextLine());

    try {
        if (index > todoLists.size()) {
            throw new IndexOutOfBoundsException("Kamu memasukan data yang sa
        } else {

            System.out.println("Kamu akan menghapus:");
            System.out.println(String.format("[%d] %s", index, todoLists.get
            System.out.println("Apa kamu yakin?");
            System.out.print("Jawab (y/t): ");
            String jawab = input.nextLine();

            if (jawab.equalsIgnoreCase("y")) {
                // hapus data
                todoLists.remove(index);

                // tulis ulang file
                try {
                    FileWriter fileWriter = new FileWriter(fileName, false);

                    // write new data
                    for (String data : todoLists) {
                        fileWriter.append(String.format("%s%n", data));
                    }
                    fileWriter.close();

                    System.out.println("Berhasil dihapus!");
                } catch (IOException e) {
                    System.out.println("Terjadi kesalahan karena: " + e.getM
                }
            }
        }
    } catch (Exception e) {
        System.out.println(e.getMessage());
    }

    isEditing = false;
    backToMenu();
}
```

Membuat Inisialisasi dan Main Loop

Terakhir, kita harus membuat inisialisasi dan *main loop* pada fungsi `main()`.

Silahkan tulis kode untuk fungsi `main()` seperti berikut ini.

```
public static void main(String[] args) {
    // initialize
    todoLists = new ArrayList<>();
    input = new Scanner(System.in);

    String filePath = System.console() == null ? "/src/todolist.txt" : "/tod
    fileName = System.getProperty("user.dir") + filePath;

    System.out.println("FILE: " + fileName);

    // run the program (main loop)
    while (true) {
        showMenu();
    }
}
```

Pertama kita harus melakukan inisalisasi atau mengisi nilai pertama untuk tiap variabel global yang belum memiliki isi agar tidak terjadi `NullPointerException`.

Untuk menentukan lokasi file yang akan dibaca, kita menulisnya seperti ini:

```
String filePath = System.console() == null ? "/src/todolist.txt" : "/todolis
fileName = System.getProperty("user.dir") + filePath;
```

Kaerna nanti kita akan coba membuka palikasi dari luar dan di dalam Netbeans.

Soalnya, alamat file (path) yang dibaca di Netbeans berbeda dengan yang di lauar.

Terakhir, kita membuat *main loop* menjaga agar program tidak berhenti selama kita tidak menutupnya.

```
while (true) {
    showMenu();
}
```

Fungsi `showMenu()` akan dipanggil berulang-ulang di dalam *main loop*.

Sekarang kode programnya sudah lengkap..

Uji Coba Program

Percobaan ini akan dilakukan di dua tempat, yakni Netbeans dan Terminal.

Percobaan dari Netbeans

1. Melihat isi Todo List

```
Output - AplikasiFile (run)
run:
FILE: /home/dian/NetBeansProjects/AplikasiFile/src/todolist.txt
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu> 1
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang

Tekan [Enter] untuk kembali..
```

2. Tambah Todo List

```
Output - AplikasiFile (run)
run:
FILE: /home/dian/NetBeansProjects/AplikasiFile/src/todolist.txt
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu> 2
Apa yang ingin kamu kerjakan?
Jawab: Siram tanaman
Berhasil ditambahkan!

Tekan [Enter] untuk kembali..
```

3. Edit Todo List

```
Output - AplikasiFile (run)
run:
FILE: /home/dian/NetBeansProjects/AplikasiFile/src/todolist.txt
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu> 3
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang
[2] Siram tanaman
-----------------
Pilih Indeks> 2
Data baru: Belajar
```

4. Hapus Todo List

```
Output - AplikasiFile (run)
run:
FILE: /home/dian/NetBeansProjects/AplikasiFile/src/todolist.txt
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu> 4
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang
[2] Belajar Java
-----------------
Pilih Indeks> 2
Kamu akan menghapus:
[2] Belajar Java
Apa kamu yakin?
Jawab (y/t): y
```

5. Salah Pilih Menu dan Keluar

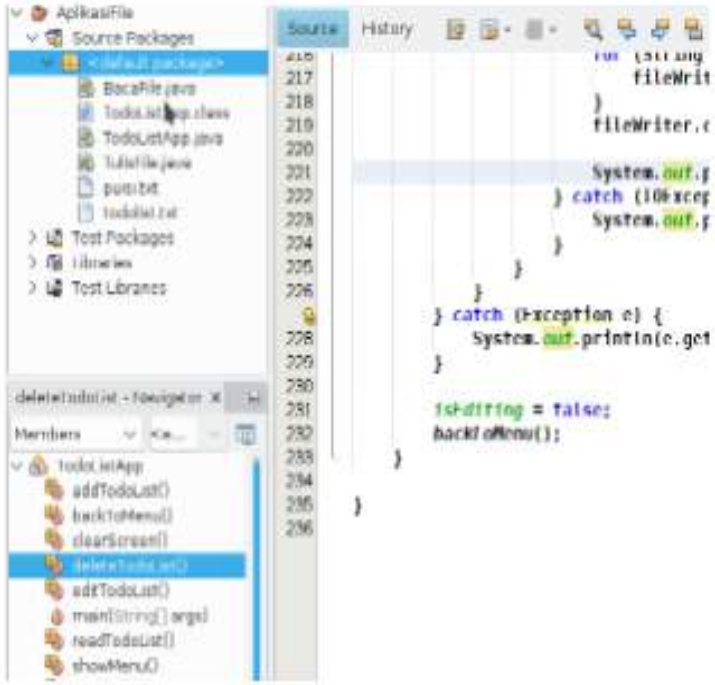
```
Output - AplikasiFile (run)
run:
FILE: /home/dian/NetBeansProjects/AplikasiFile/src/todolist.txt
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu> 99
Kamu salah pilih menu!

Tekan [Enter] untuk kembali..
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
```

Seperti yang kita lihat, fungsi Clear Screen tidak akan bisa bekerja di Netbeans.

Sekarang mari kita coba jalankan melalui Terminal.

Percobaan dari Terminal

Klik kanan pada <default package> kemudian, pilih **Tools->Open in Terminal**.



Setelah itu, compile programnya dengan perintah:

```
javac TodoListApp.java
```

Terminal - ...n/NetBeansProjects/AplikasiFile/src

```
Welcome to fish, the friendly interactive shell
dianpetanikode~/N/A/src>
```

Setelah itu, kita bisa buka programnya dengan perintah:

```
java TodoListApp
```

Mari kita coba semua fitur yang ada:

1. Melihat isi Todo List

Terminal - ...n/NetBeansProjects/AplikasiFile/src

```
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu>
```

2. Tambah Todo List

Terminal - ...n/NetBeansProjects/AplikasiFile/src

```
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang
[2] Beli Bibit Java

Tekan [Enter] untuk kembali..
```

3. Edit Todo List

Terminal - ...n/NetBeansProjects/AplikasiFile/src

```
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang
[2] Beli Bibit Java
-----------------
Pilih Indeks> 2
Data baru:
```

4. Hapus Todo List

Terminal - ...n/NetBeansProjects/AplikasiFile/src

```
TODO LIST:
[0] Belajar Masak
[1] Belajar Renang
[2] Beli Bibit Kotlin
-----------------
Pilih Indeks> 2
```

5. Salah Pilih Menu dan Keluar

Terminal - ...n/NetBeansProjects/AplikasiFile/src

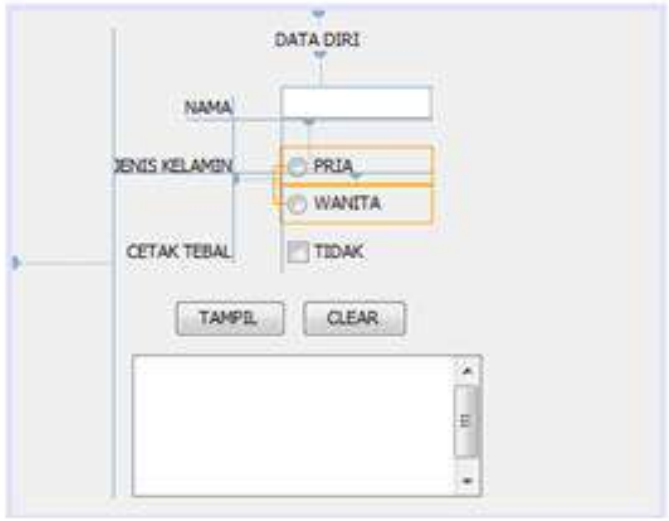
```
=== TODO LIST APP ===
[1] Lihat Todo List
[2] Tambah Todo List
[3] Edit Todo List
[4] Hapus Todo List
[0] Keluar
---------------------
Pilih menu>
```

Percobaan pada Terminal, berhasil!

Membuat Aplikasi java Netbeans (studi kasus menampilkan data dengan komponen swing)

a. Buat project Praktek2 – klik kanan project – pilih New – Klik jFrame Form

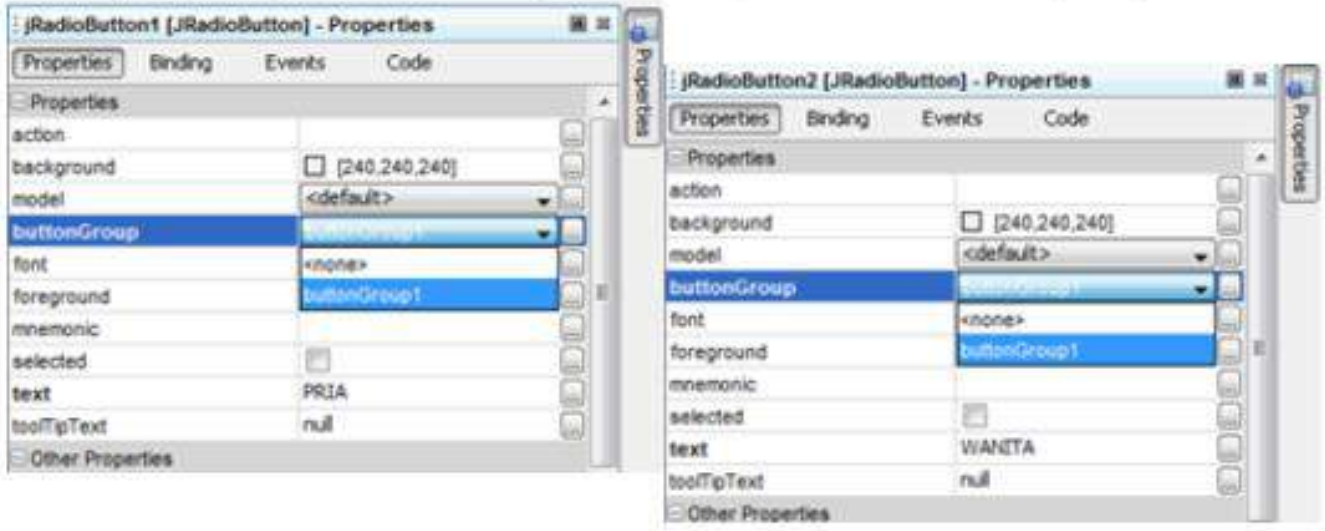
b. Masukkan jLabel (4), jTextField, jCheckBox, jRadioButton (2), jButton dan jTextArea. Atur posisi tiap-tiap komponen. Kemudian ubah propertie masing-masing komponen seperti gambar berikut :



objek	properties	nilai
JLabel1	Text	Biodata
	Font	Tahoma 12 Bold
JLabel2	Text	NAMA
JLabel3	Text	JENIS KELAMIN
JLabel4	Text	CETAK TEBAL
JTextField	Text	*dikosongkan*
JRadioButton1	Text	LAKI-LAKI
JRadioButton2	Text	PEREMPUAN
JCheckBox1	Text	TIDAK
JButton1	Text	TAMPIL
JTextArea1	Text	

c. Pilih dan drag ButtonGroup dari komponen pallete ke dalam jFrame seperti gambar diatas.

d. Ubah properties jRadioButton1 dan jRadioButton2 pada baris buttonGroup menjadi buttonGroup1, seperti gambar berikut :



e. Klik kanan pada jCheckBox1, pilih Events > Change > stateChanged

ketikan kode seperti gambar berikut :

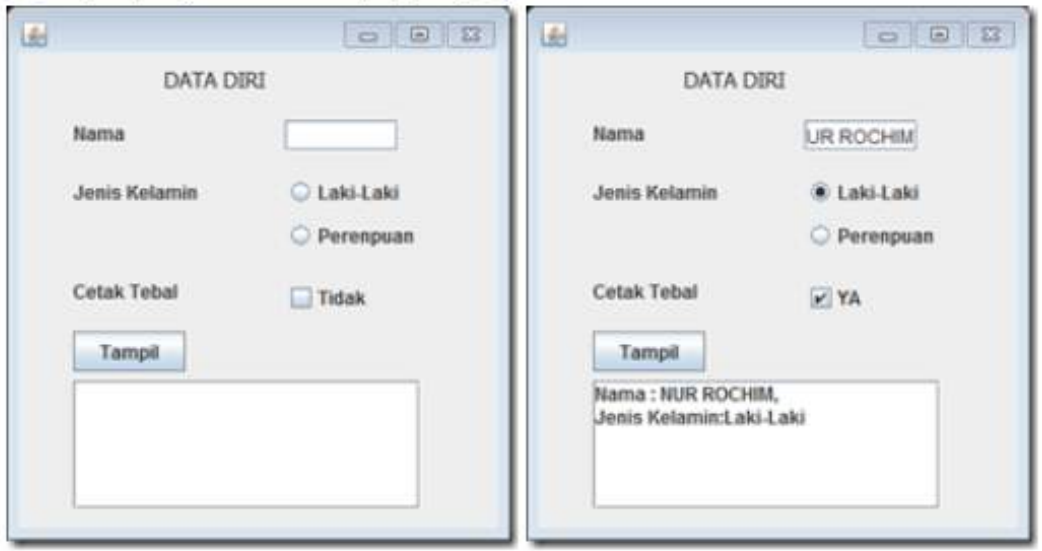
```
if (jCheckBox1.isSelected()){
    jCheckBox1.setText("YA");
}
else {
    jCheckBox1.setText("TIDAK"); }
```

f. Klik kanan jButton1, pilih Events > Mouse > mouseClicked.

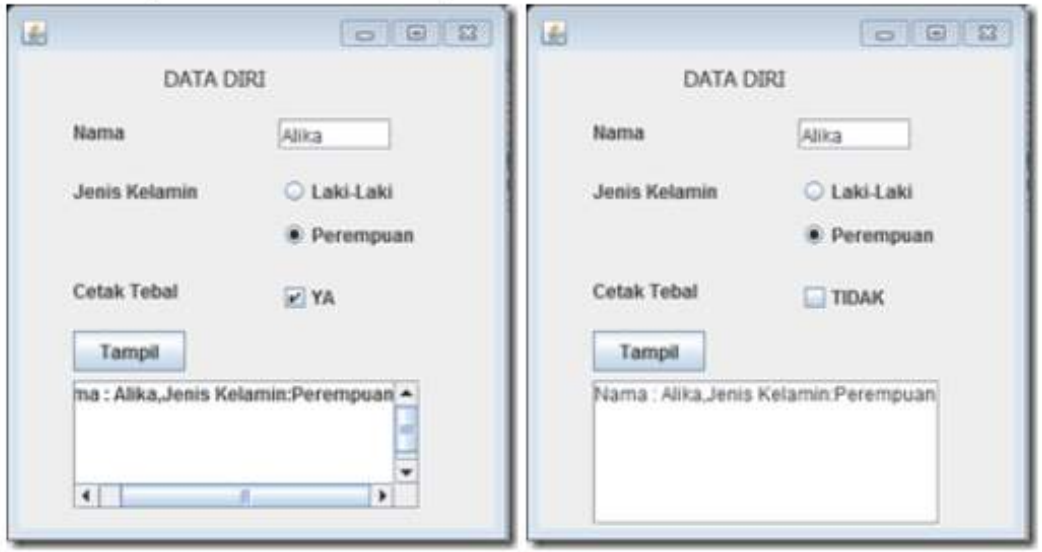
Ketikan kode seperti gambar berikut :

```
      Font tebal= new Font("Arial",Font.BOLD,12);
   Font biasa= new Font("Arial",Font.PLAIN,12);
    if (jRadioButton1.isSelected())
{ if (jCheckBox1.isSelected())
{ jTextArea1.setText("Nama : "+jTextField1.getText()+"\nJenis Kelamin "
       + ":"+jRadioButton1.getText());
           jTextArea1.setFont(tebal); }
       else{
           jTextArea1.setText("Nama : "+jTextField1.getText()+"\nJenis Kelamin "
                   + ":"+jRadioButton1.getText());
           jTextArea1.setFont(biasa);}
       }
       else  if(jRadioButton2.isSelected()){
           if (jCheckBox1.isSelected()){
               jTextArea1.setText("Nama : "+jTextField1.getText()+",Jenis Kelamin "
                       + ":"+jRadioButton2.getText());
           jTextArea1.setFont(tebal);
           }
           else{
             jTextArea1.setText("Nama : "+jTextField1.getText()+",Jenis Kelamin :"+jRadioButton2.getText());
           jTextArea1.setFont(biasa);
       }
   }
```

g. Compile (F11) dan Jalankan program (F6).



Analisa : Apabila jTextField dan jRadioButton kita inputkan maka akan menghasilkan keluaran yang akan ditampung di jTextArea. Contoh disini adalah jTextField : NUR ROCHIM | Jenis Kelamin : Laki-Laki.



Apabila jCheckBox di centang maka akan menghasilkan jenis huruf tebal pada jTextArea, apabila tidak dicentang maka jenis huruf tetap normal seperti gambar disebelah kanan.

Tambahkan juga Button Clear untuk mereset ulang apabila ingin menginputkan lagi.

Pembahasan

* Kode komponen jCheckBox1

```
if (jCheckBox1.isSelected()){
    jCheckBox1.setText("YA");
}
else {
    jCheckBox1.setText("TIDAK"); }
```

Analisa : Koding ini memiliki event stateChange yang bermaksut apabila kotak CheckBox dicentang maka akan tampil berubah.

Script diatas maksutnya adalah apabila jCheckBox1 di centang maka teksnya akan berubah menjadi YA, apabila tidak dicentang teksnya tetap TIDAK.

* Kode komponen jButton1

```
      Font tebal= new Font("Arial",Font.BOLD,12);
   Font biasa= new Font("Arial",Font.PLAIN,12);
    if (jRadioButton1.isSelected())
{ if (jCheckBox1.isSelected())
{ jTextArea1.setText("Nama : "+jTextField1.getText()+"\nJenis Kelamin "
       + ":"+jRadioButton1.getText());
          jTextArea1.setFont(tebal); }
       else{
          jTextArea1.setText("Nama : "+jTextField1.getText()+"\nJenis Kelamin "
                  + ":"+jRadioButton1.getText());
          jTextArea1.setFont(biasa);}
       }
       else  if(jRadioButton2.isSelected()){
          if (jCheckBox1.isSelected()){
              jTextArea1.setText("Nama : "+jTextField1.getText()+"Jenis Kelamin "
                      + ":"+jRadioButton2.getText());
          jTextArea1.setFont(tebal);
          }
          else{
            jTextArea1.setText("Nama : "+jTextField1.getText()+"Jenis Kelamin :"+jRadioButton2.getText());
          jTextArea1.setFont(biasa);
       }
   }
```

Analisa : Koding ini memiliki Event mouseClicked, apabila jButton di klik maka akan melaksanakan kode program selanjutnya.

Font tebal= new Font("Arial".Font.BOLD.J2);

Font biasa= new Font("Arial".Font.PLAIN.J2);

Script diatas untuk mengatur jenis dan ukuran font. Jangan lupa di baris paling awal kode ditambahkan import java.awt.Font agar font dapat berkerja.

```
if (jRadioButton1.isSelected())
{ if (jCheckBox1.isSelected())
{ jTextArea1.setText("Nama : "+jTextField1.getText()+",\nJenis Kelamin:"+jRadioButton1.getText());
jTextArea1.setFont(tebal); }
else{
jTextArea1.setText("Nama : "+jTextField1.getText()+",\nJenis Kelamin:"+jRadioButton1.getText());
jTextArea1.setFont(biasa);}
else if(jRadioButton2.isSelected()){
if (jCheckBox1.isSelected()){
jTextArea1.setText("Nama : "+jTextField1.getText()+",Jenis Kelamin:"+jRadioButton2.getText());
jTextArea1.setFont(tebal);}
else{
jTextArea1.setText("Nama : "+jTextField1.getText()+",Jenis Kelamin:"+jRadioButton2.getText());
jTextArea1.setFont(biasa);}}
```

Thank You

ukrida.ac.id

UKRIDA
Universitas Kristen Krida Wacana