

Pemrograman Java

Oleh: Rosihan Ari Yuana, S.Si, M.Kom
[\(http://blog.rosihanari.net\)](http://blog.rosihanari.net)

Tentang Java

Java adalah bahasa pemrograman dan platform komputasi pertama kali dirilis oleh Sun Microsystems pada tahun 1995. Java merupakan teknologi yang mendasari kekuatan program untuk utilitas, permainan, dan aplikasi bisnis. Java berjalan pada lebih dari 850 juta komputer pribadi di seluruh dunia, dan pada miliaran perangkat di seluruh dunia, termasuk ponsel dan perangkat TV.

Salah satu karakteristik Java adalah portabilitas, yang berarti bahwa program komputer yang ditulis dalam bahasa Java harus dijalankan secara sama, pada setiap hardware / platform sistem operasi. Hal ini dicapai dengan menyusun kode bahasa Java ke sebuah Java bytecode. Pengguna aplikasi biasanya menggunakan Java Runtime Environment (JRE) diinstal pada mesin mereka sendiri untuk menjalankan aplikasi Java, atau dalam browser web untuk applet Java.

Untuk pembuatan dan pengembangan aplikasi berbasis Java diperlukan Java Development Kit (JDK), dimana saat ini pemilik lisensi dari JDK adalah Oracle Corporation yang telah secara resmi mengakuisisi Sun Microsystem pada awal tahun 2010. Ada beberapa Java platform untuk keperluan development, yaitu:

- Java SE (Standard Edition), yang khusus digunakan untuk pengembangan aplikasi-aplikasi pada PC atau workstation.
- Java ME (Micro Edition), yaitu khusus digunakan untuk pengembangan aplikasi-aplikasi yang ada di perangkat mobile spt HP, smartphone, PDA, tablet dsb.
- Java EE (Enterprise Edition), yaitu khusus digunakan untuk pengembangan aplikasi skala besar (enterprise), dan aplikasi web berbasis java.

Instalasi Java Development Kit

Dalam tutorial ini hanya akan dibahas pengembangan aplikasi Java yang nantinya digunakan khusus di PC/workstation. Sehingga untuk keperluan ini, kita cukup menggunakan Java SE sebagai JDK nya.

Perangkat yang dibutuhkan untuk pembuatan aplikasi Java:

- **Java (SE – Standard Edition) Development Kit.**

JDK adalah suatu paket perangkat yang digunakan untuk membangun aplikasi, applet, dan komponen menggunakan bahasa Java. Berikut ini perangkat yang ada di dalam sebuah JDK: development tools , Java runtime environment (JRE), library, Java DB (Java relational database), demo aplikasi dan applet, serta contoh-contoh program.

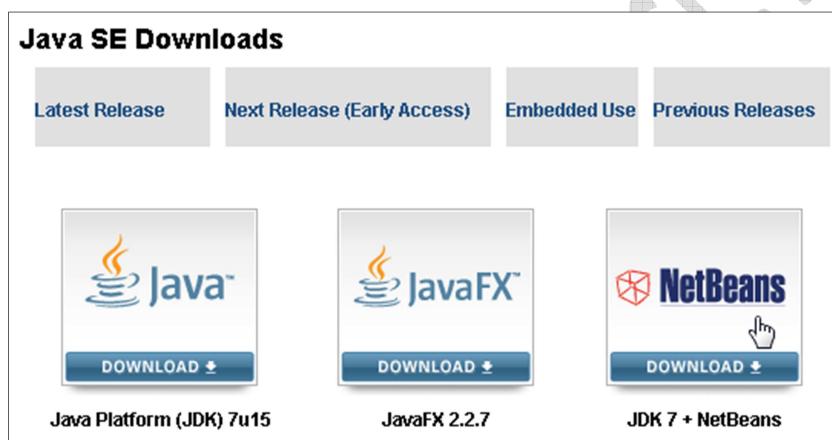
- **Java Development IDE (Integrated Development Environment)**, misal: NetBeans atau eclipse.

Berikut ini langkah instalasinya:

1. Unduh keduanya (JDK dan NetBeans) sekaligus di:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

2. Pilih JDK + Netbeans



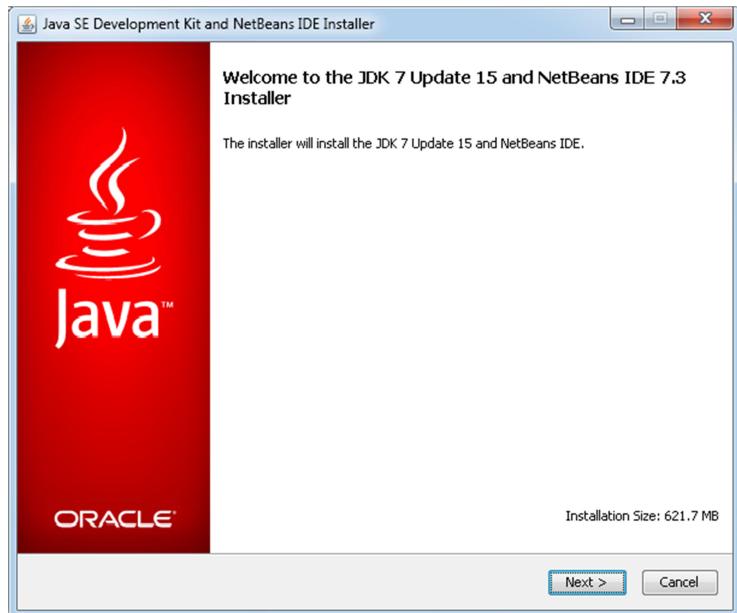
3. Pilih platform sistem operasi yang sesuai

Java SE and NetBeans Cobundle (JDK 7u15 and NB 7.3)		
Product / File Description	File Size	Download
Linux x86	178.92 MB	jdk-7u15-nb-7_3-linux-i586.sh
Linux x64	175.11 MB	jdk-7u15-nb-7_3-linux-x64.sh
Mac OS X x64	217.56 MB	jdk-7u15-nb-7_3-macosx-x64.dmg
Windows x86	184.15 MB	jdk-7u15-nb-7_3-windows-i586.exe
Windows x64	186.91 MB	jdk-7u15-nb-7_3-windows-x64.exe

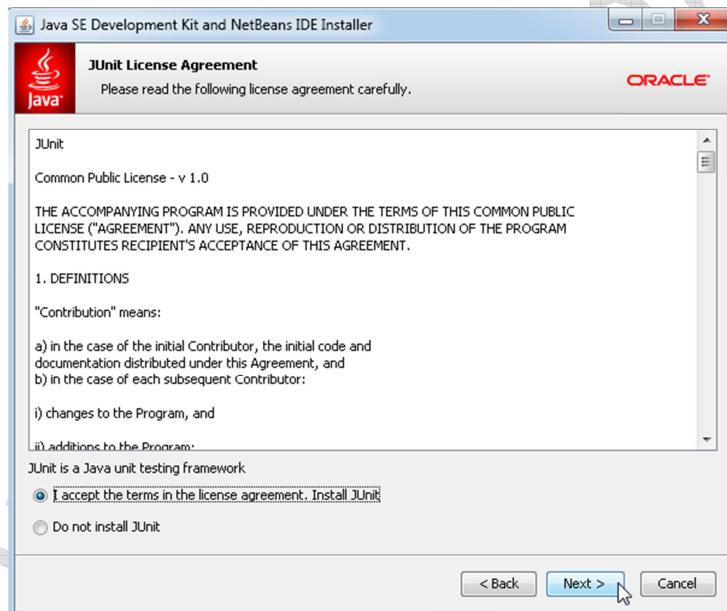
Keterangan:

Windows x64 adalah untuk sistem operasi Windows 64 bit.

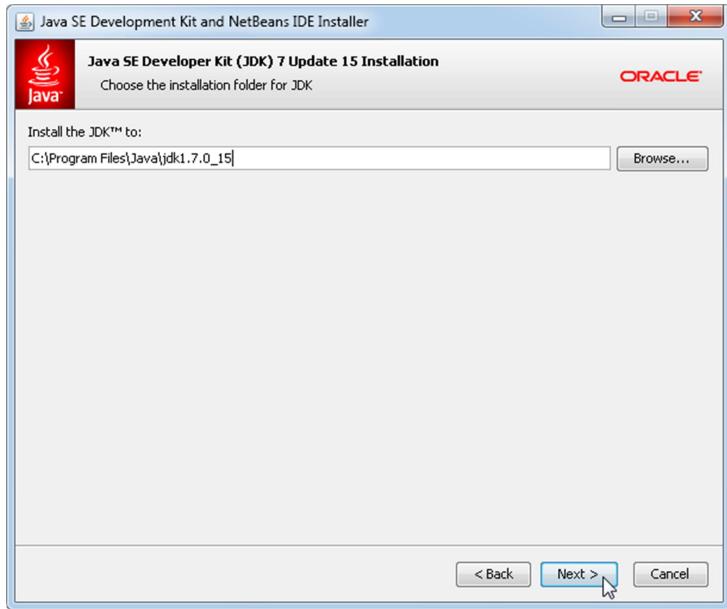
4. Jalankan instalasi JDK dan Netbeans hasil download



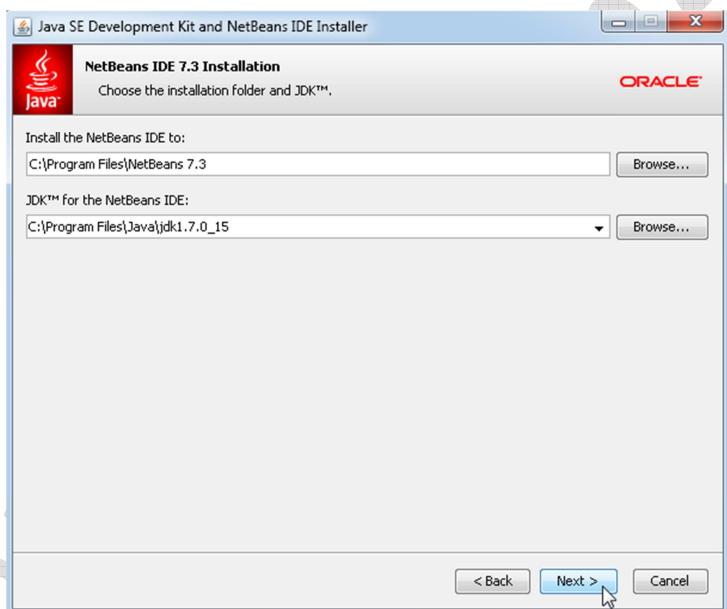
Klik NEXT



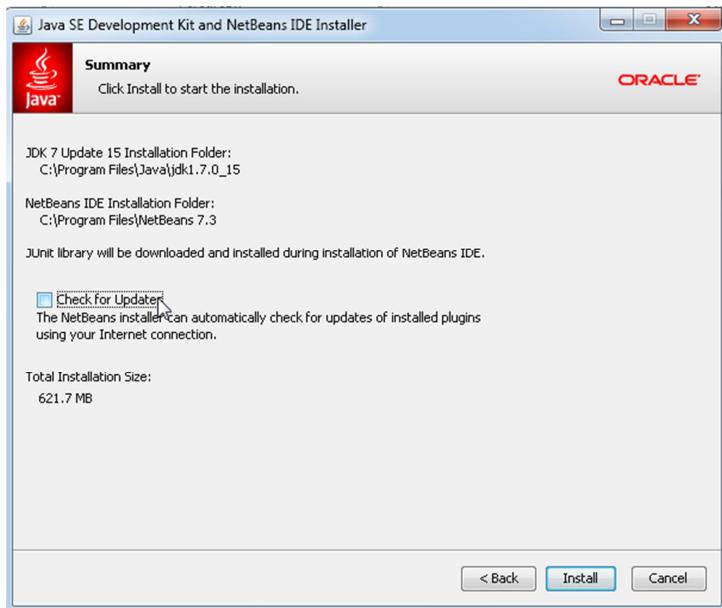
Pilih 'I accept the terms in the license agreement, Install JUnit', Klik NEXT



Tentukan PATH untuk lokasi instalasi JDK.



Tentukan PATH untuk NetBeans IDE



Keterangan: Hilangkan tanda cek pada 'Check for updates'

5. Klik INSTALL dan tunggu sampai instalasi selesai

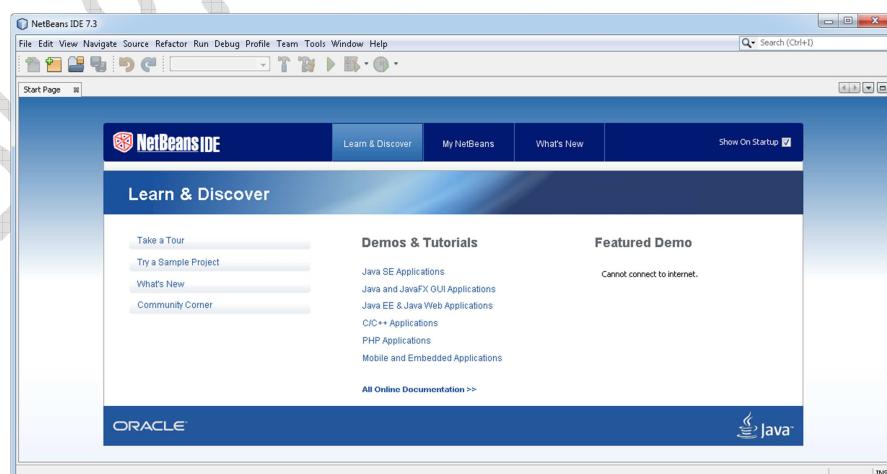
Membuat Program Java Pertama

Pada bab ini kita akan mencoba membuat program Java untuk yang pertama kalinya.

Example 1:

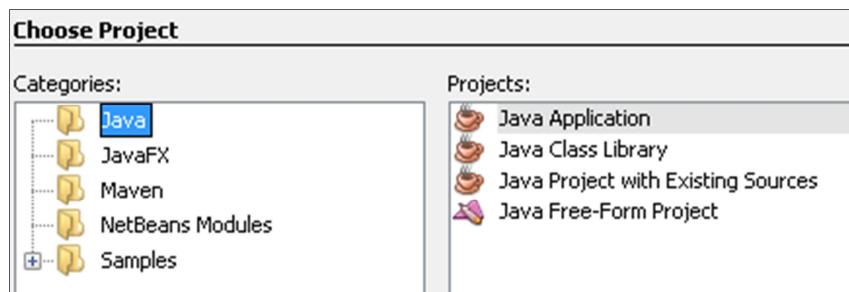
Program ini nanti hanya sekedar menampilkan sebuah pesan 'Hello World'. Berikut ini langkah pembuatannya:

1. Jalankan NetBeans



Tampilan awal NetBeans

2. Klik menu FILE – NEW PROJECT, setelah itu akan muncul PROJECT WIZARD yang memudahkan Anda untuk membuat Java Project
3. Pada bagian CATEGORIES, pilih JAVA, pada bagian PROJECT pilih JAVA APPLICATION

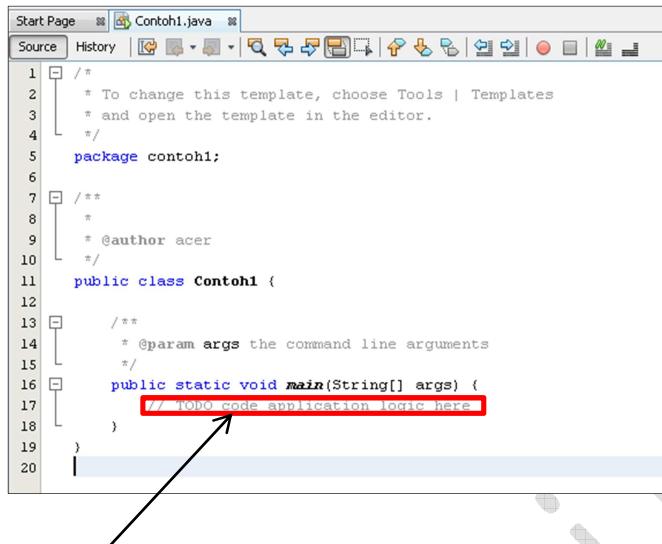


4. Klik NEXT
5. Beri nama Projectnya, misalnya: Contoh1, serta tentukan path untuk menyimpan projectnya. Dalam contoh ini, misalnya file project disimpan di D:\JavaApp. Pastikan folder 'JavaApp' tersebut sudah dibuat sebelumnya.

Project Name:	Contoh1
Project Location:	D:\JavaApp
Project Folder:	D:\JavaApp\Contoh1

6. Klik FINISH
7. Setelah itu NetBeans secara otomatis menyiapkan sebuah source program yang sudah lengkap strukturnya, dan tinggal Anda tambahkan beberapa perintah sesuai keinginan pada bagian

```
// TODO code application logic here
```



```

1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5   package contoh1;
6
7  /**
8   *
9   * @author acer
10  */
11 public class Contoh1 {
12
13 /**
14  * @param args the command line arguments
15 */
16 public static void main(String[] args) {
17     // TODO code application logic here
18 }
19
20

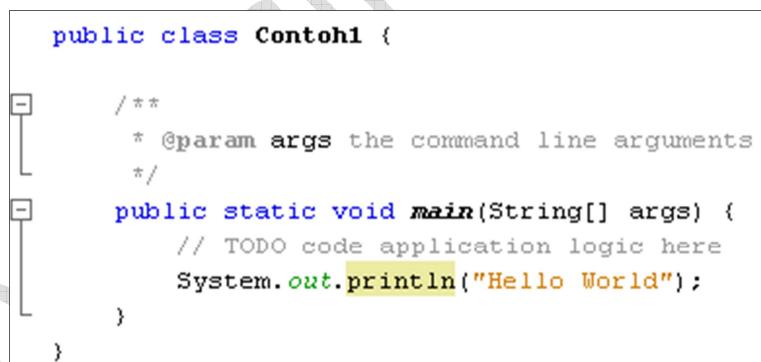
```

Tempat menyisipkan perintah yang diinginkan

8. Selanjutnya silakan tambahkan perintah berikut ini, untuk menampilkan pesan 'Hello World'.

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

Sehingga tampilan source Contoh1.java menjadi sbb:



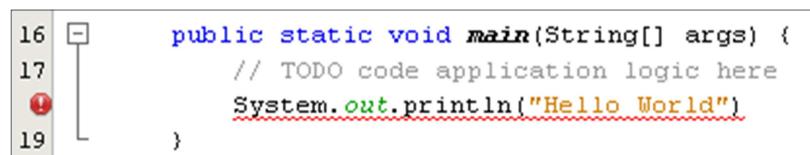
```

public class Contoh1 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        System.out.println("Hello World");
    }
}

```

9. Cek apakah ada tanda seru warna merah di sisi sebelah kiri source. Jika tanda ini muncul maka menandakan ada sintaks yang salah pada baris tersebut.



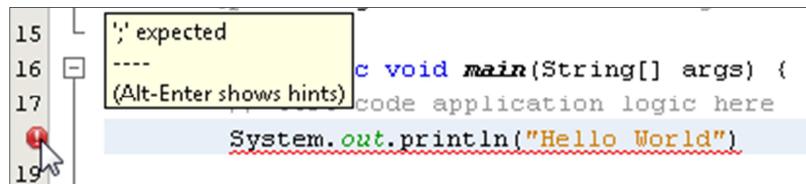
```

16
17     public static void main(String[] args) {
18         // TODO code application logic here
19         System.out.println("Hello World");

```

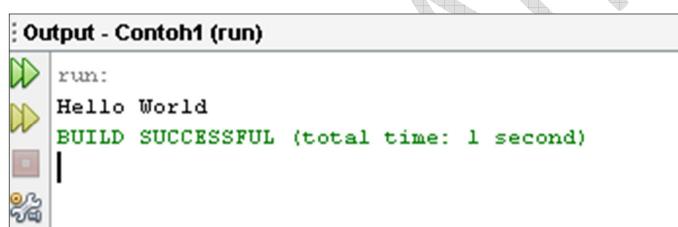
Contoh tampilan pesan kesalahan pada suatu baris program.

Untuk mengetahui apa kesalahannya, cukup dekatkan saja kursor pada tanda seru warna merah tersebut.



Untuk melihat ada tidaknya kesalahan sintaks, bisa juga dilakukan proses COMPILE, dengan cara mengklik menu RUN – COMPILE. Proses COMPILE hanya bisa dilakukan sekali saja.

10. Kita bisa melihat tampilan program Java yang sudah dibuat dengan mengklik menu RUN – RUN PROJECT.



11. Setelah kita melihat hasil running aplikasi Java kita, selanjutnya kita melakukan proses BUILD supaya dihasilkan file aplikasi Java yang executable. Hasil proses BUILD ini berupa file *.jar. Untuk melakukan proses BUILD suatu project, caranya klik RUN – BUILD PROJECT. Hasil dari proses BUILD project ini (*.jar), secara otomatis akan tersimpan di folder 'dist', dalam contoh ini di dalam direktori D:\JavaApp\Contoh1\dist.

Sedangkan source code nya sendiri tersimpan di direktori D:\JavaApp\Contoh1\src.

12. Selanjutnya, file Java Executable File (*.jar) bisa kita eksekusi via command prompt dengan perintah

```
java -jar "D:/JavaApp/Contoh1/dist/Contoh1.jar"
```



Keterangan:

Untuk pembuatan aplikasi Java dengan GUI (Graphics User Interface) akan dibahas pada bab yang lain.

Tipe Data Dalam Java

Sebagai mana bahasa pemrograman yang lain, di dalam Java juga dikenal istilah tipe data. Tipe data ini digunakan untuk pengalokasian memory guna menyimpan nilai/valuenya.

Di dalam Java, ada beberapa tipe data sebagai berikut:

Tipe Data	Range nilai	Keterangan
Byte	-128 ... 127	Bilangan bulat
Short	-32768 ... 32767	Bilangan bulat
Int	-2147483648 ... 2147483647	Bilangan bulat
Long	-9223372036854775808 ... 9223372036854775807	Bilangan bulat
Float		Bilangan riil
Double		Bilangan riil
Char		Karakter
String		String (beberapa karakter)
Boolean	true/false	-

Berikut ini contoh program Java untuk menyimpan nilai beberapa tipe data:

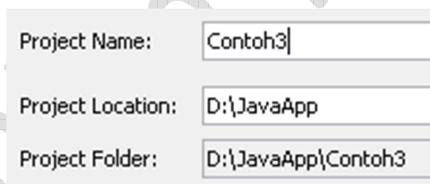
Example 2:

Misalkan kita buat project dengan nama 'Contoh3'. Langkahnya adalah:

1. Klik NEW PROJECT



2. Pilih JAVA pada Categories dan JAVA APPLICATION pada Projects.
3. Isikan nama projectnya dengan nama 'Contoh3'



4. Kemudian tulis kode berikut ini

```
package contoh3;

public class Contoh3 {
    public static void main(String[] args) {
        // deklarasi variabel dan tipe datanya
        int gajiPokok;
        float potonganGaji, gajiBersih;
```

```
String namaKaryawan, kodeKaryawan;
char golDarah;

// assignment nilai pada setiap variabel
kodeKaryawan = "K00001";
namaKaryawan = "ROSIHAN ARI YUANA";
golDarah = 'A';
gajiPokok = 3000000;
potonganGaji = (float) 0.2;
gajiBersih = gajiPokok - (potonganGaji * gajiPokok);

// tampilkan output
System.out.println("KODE KARYAWAN : "+kodeKaryawan);
System.out.println("NAMA KARYAWAN : "+namaKaryawan);
System.out.println("GOL DARAH      : "+golDarah);
System.out.println("GAJI POKOK     : Rp. "+gajiPokok);
System.out.println("GAJI BERSIH    : Rp. "+gajiBersih);
}
```

Output dari project ini adalah sbb:

```
run:
KODE KARYAWAN : K00001
NAMA KARYAWAN : ROSIHAN ARI YUANA
GOL DARAH      : A
GAJI POKOK     : Rp. 3000000
GAJI BERSIH    : Rp. 2400000.0
```

Operator di Java

Operator adalah suatu simbol atau tanda yang digunakan untuk mengoperasikan dua value atau lebih untuk mendapatkan hasil.

Jenis-jenis Operator di Java

Berikut ini beberapa jenis operator yang dikenal dalam bahasa Java

Operator Aritmatika

Simbol Operator	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus (sisa hasil bagi)

Operator Logika

Simbol Operator	Keterangan
&	Logika AND
	Logika OR
^	Logika XOR
!	Logika NOT

Operator Increment/Decrement

Simbol Operator	Keterangan
++	Value bertambah 1
--	Value berkurang 1

Operator Relasional

Simbol Operator	Keterangan
<	Lebih kecil dari
>	Lebih besar dari
>=	Lebih besar atau sama dengan dari
<=	Lebih kecil atau sama dengan dari
==	Sama dengan
!=	Tidak sama dengan

Operator Majemuk

Simbol Operator	Keterangan
+=	Contoh: $a += 2$, ekuivalen dg $a = a + 2$
-=	Contoh: $a -= 2$, ekuivalen dg $a = a - 2$
*=	Contoh: $a *= 2$, ekuivalen dg $a = a * 2$
/=	Contoh: $a /= 2$, ekuivalen dg $a = a / 2$
%=	Contoh: $a \%= 2$, ekuivalen dg $a = a \% 2$

Konsep Pemrograman Dalam Java (1)

Pemrograman Java menggunakan konsep Pemrograman Berorientasi Obyek (PBO) atau Object Oriented Programming (OOP). Semua program Java merupakan suatu obyek. Dasar-dasar OOP meliputi istilah berikut ini:

- Class
- Object
- Attribute
- Method

Secara umum, OOP adalah teknik yang memfokuskan desain program pada obyek dan class berdasarkan pada skenario di dunia nyata. Sebagai contoh, misalkan mobil. Sebuah mobil secara umum tentunya memiliki beberapa karakteristik, yaitu misalnya memiliki sejumlah roda, memiliki warna, memiliki beberapa pintu dsb. Selanjutnya mobil ini bisa terdapat berbagai macam merek, misalnya mobil Suzuki Ertiga, Toyota Avanza dsb. Sebuah mobil tentunya juga bisa dijalankan, baik maju maupun mundur atau dihentikan. Dalam OOP, mobil tersebut identik dengan Class, mobil Suzuki Ertiga, Avanza dll itu merupakan obyek. Jumlah roda, warna mobil, jumlah tempat duduk dll identik dengan atribut dari suatu obyek, serta proses untuk mengendalikan mobil (maju, mundur dan berhenti) itu dalam OOP identik dengan method dari suatu obyek.

Manfaat dari pemrograman yang menggunakan teknik OOP ini adalah kebebasan pengembangan, meningkatkan kualitas, mempermudah pemeliharaan, mempertinggi kemampuan dalam modifikasi dan meningkatkan penggunaan kembali software.

Class

Class adalah model dari suatu obyek yang menjelaskan karakteristik (sifat) serta fungsi yang dimiliki dari suatu obyek. Class merupakan wadah (tempat) yang digunakan untuk menciptakan suatu obyek. Dengan kata lain sebuah Class merupakan *blueprint* dari suatu obyek.

Berikut ini adalah aturan pembuatan class dalam Java:

```
public class namaclass
{
    .
    .
}
```

Aturan pemberian nama class:

- Dimulai dengan huruf, atau tanda _ atau tanda \$
- Tidak boleh menggunakan reserved word dalam Java
- Tidak boleh memuat operator aritmatika
- Bersifat case sensitive

Oleh karena itu, jika diperhatikan ketika membuat project baru, maka secara otomatis akan dibuat class sesuai nama projectnya. Misalkan Anda membuat project baru dengan nama 'project1', maka secara otomatis akan dibuat class dengan nama 'Project1'.

```
public class Project1 {
    .
    .
}
```

Dalam sebuah project, kita dapat membuat lebih dari satu class sebanyak kebutuhan.

Atribut

Atribut adalah elemen data dari suatu class. Atribut menyimpan informasi tentang class. Atribut dapat diartikan sebagai data, variabel, properti atau sebuah field.

Method

Method adalah sebuah function atau fungsi yang ada dalam suatu class. Setiap method memiliki tugas sendiri. Di dalam Java ada 2 jenis method yaitu void dan non void method. Method Void adalah method yang tidak mengembalikan nilai, sedang non void method adalah method yang mengembalikan suatu nilai.

Jika Anda perhatikan pula ketika membuat project baru misalnya 'project1', maka akan di dalam class 'project1' ini akan dibuat pula method dengan nama main().

```
public class Project1 {  
    public static void main(String[] args) {  
        .  
        .  
    }  
}
```

Method main() dalam suatu class menunjukkan method tersebut adalah method utama yang akan dijalankan pertama kali ketika program Java dijalankan. Khusus method main(), perlu diberikan 'static' setelah modifiernya.

Pada suatu class, kita bisa membuat method berapapun semau kita.

Perlu diingat juga bahwa di dalam Java, beberapa class itu bisa digabung atau disimpan menjadi satu dalam sebuah paket atau package jika diperlukan. Hal ini dimaksudkan untuk memudahkan pengelolaan class saja.

Example 3:

Sebagai contoh dari penerapan konsep OOP dalam pemrograman Java, misalkan kita akan membuat sebuah program untuk menjumlahkan dua buah bilangan.

Untuk langkah awal, kita desain terlebih dahulu bentuk class untuk penjumlahan bilangan tersebut. Misalkan kita buat class dengan nama 'operasiBilangan'. Di dalam class tersebut, misalkan kita buat atribut yaitu 'bilangan1' dan 'bilangan2', merupakan kedua bilangan yang akan dioperasikan, serta 'hasil' yang merupakan hasil dari operasi kedua bilangan. Selanjutnya di dalam class 'operasiBilangan' tersebut kita buat sebuah method 'jumlah' untuk menjumlahkan kedua bilangan, serta method untuk menampilkan hasil operasi bilangan.

Sesuai desain class tersebut, sekarang kita implementasikan di Java. Berikut ini langkah-langkahnya:

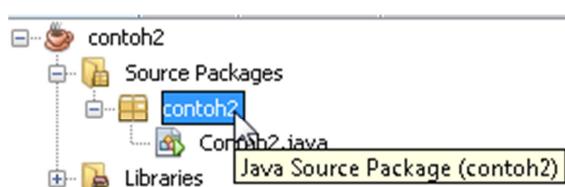
1. Klik NEW PROJECT



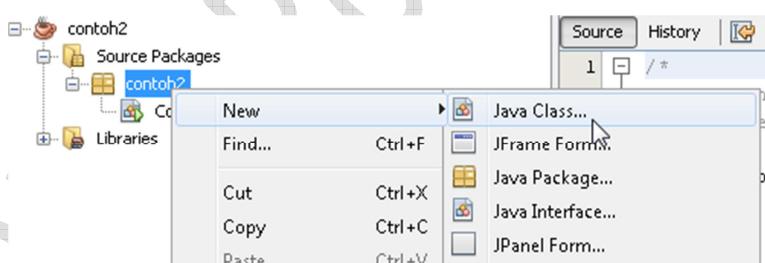
2. Pilih 'JAVA' pada Categories, dan 'Java Application' pada 'Project'
3. Misalkan untuk Nama Project, kita beri nama 'contoh2'



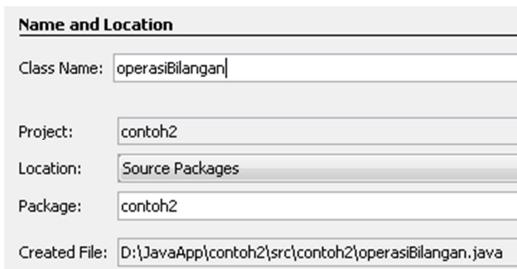
4. Setelah pembuatan project 'contoh2' ini berhasil, maka secara otomatis akan muncul package dengan nama 'contoh2'



5. Selanjutnya, kita buat Class dengan nama 'operasiBilangan' dalam package 'contoh2' tersebut yaitu dengan mengklik kanan pada nama package nya, lalu pilih NEW, dan pilih JAVA CLASS



6. Isikan 'operasiBilangan' pada isian nama class yang akan dibuat



7. Setelah membuat class 'operasiBilangan', maka secara otomatis Java akan membuat file dengan nama 'operasiBilangan.java' pada direktori project. Class 'operasiBilangan' ini terletak dalam package 'contoh2'. Dan tugas kita kemudian akan menulis kode dalam class operasiBilangan tersebut.

```
/*
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */
package contoh2;

/**
 *
 * @author acer
 */
public class operasiBilangan {
```

8. Tulislah kode berikut ini di dalam class operasiBilangan

```
public class operasiBilangan {

    // deklarasi atribut atau properties
    public int bilangan1;
    public int bilangan2;
    private int hasil;

    // method jumlah()
    public void jumlah()
    {
        this.hasil = this.bilangan1 + this.bilangan2;
    }

    // method tampilHasil()
    public void tampilHasil()
    {
        System.out.println("Hasil operasi bilangan : " + this.hasil);
    }
}
```

Keterangan:

- Bilangan1, bilangan2 dan hasil merupakan atribut atau properties dari class operasiBilangan. Sedangkan jumlah() dan tampilHasil() adalah methodnya.
- Perhatikan, di depan atribut atau method ada ‘public’ atau ‘private’. Jika diberikan ‘public’ maka atribut atau method tersebut bisa diakses dari class manapun (jika terdapat lebih dari satu class). Namun jika ‘private’, maka atribut atau method hanya bisa diakses di dalam class itu saja. Selain ‘public’ dan ‘private’ sebuah atribut atau method bisa juga diset dengan sifat ‘protected’ yang artinya hanya bisa diakses dalam class itu saja ata class lain yang masih dalam satu package yang sama. Keterangan ‘public’, ‘private’ dan ‘protected’ dalam OOP disebut modifier yang digunakan untuk menentukan aksesibilitas method atau atribut.
- Perintah ‘this.’ digunakan untuk mengakses atribut atau method yang ada dalam class tersebut.

9. Kemudian, di class ‘Contoh2’ nya (di file ‘Contoh2.java’) kita tulis kode program sebagai berikut

```
public class Contoh2 {  
  
    public static void main(String[] args) {  
  
        operasiBilangan op1 = new operasiBilangan();  
  
        op1.bilangan1 = 10;  
        op1.bilangan2 = 20;  
        op1.jumlah();  
        op1.tampilHasil();  
  
    }  
}
```

Keterangan:

Perintah

```
operasiBilangan op1 = new operasiBilangan();
```

digunakan untuk instantisasi, yaitu proses membuat obyek baru dengan nama ‘op1’. Obyek ini termasuk dalam class ‘operasiBilangan’.

Perintah

```
op1.bilangan1 = 10;
```

adalah mengeset atribut ‘bilangan1’ pada obyek ‘op1’ dengan suatu nilai. Demikian juga dengan perintah

```
op1.bilangan2 = 20;
```

Perintah

```
op1.jumlah();
```

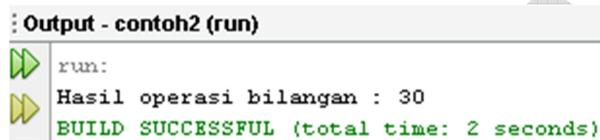
dimaksudkan untuk menjalankan method jumlah() yaitu menjumlahkan kedua nilai atribut ‘bilangan1’ dan ‘bilangan2’ pada obyek ‘op1’.

Sedangkan perintah

```
op1.tampilHasil();
```

digunakan untuk menjalankan method tampilHasil() yaitu menampilkan hasil penjumlahan.

- Untuk melihat hasil output program, Anda bisa mengcompilenya dahulu kemudian menjalankan RUN PROJECT. Adapun outputnya adalah sbb:



```
: Output - contoh2 (run)
run:
Hasil operasi bilangan : 30
BUILD SUCCESSFUL (total time: 2 seconds)
```

Dalam sebuah program, kita bisa membuat instantisasi beberapa obyek dari class yang sama.

Example 4 :

Berikut ini contoh yang merupakan pengembangan dari project ‘contoh2’.

```
public class Contoh2 {
    public static void main(String[] args) {

        // instantisasi obyek 'op1'
        operasiBilangan op1 = new operasiBilangan();

        op1.bilangan1 = 10;
        op1.bilangan2 = 20;
        op1.jumlah();
        op1.tampilHasil();

        // instantisasi obyek 'op2'
        operasiBilangan op2 = new operasiBilangan();

        op2.bilangan1 = 30;
```

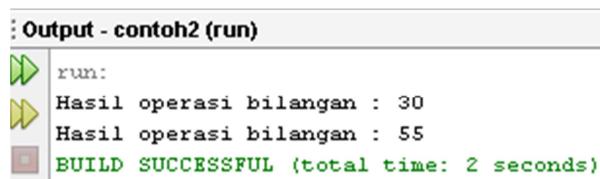
```

        op2.bilangan2 = 25;
        op2.jumlah();
        op2.tampilHasil();

    }
}

```

Dalam contoh di atas, dibuat 2 obyek dari class yang sama yaitu 'op1' dan 'op2'. Adapun hasil output dari program ini adalah



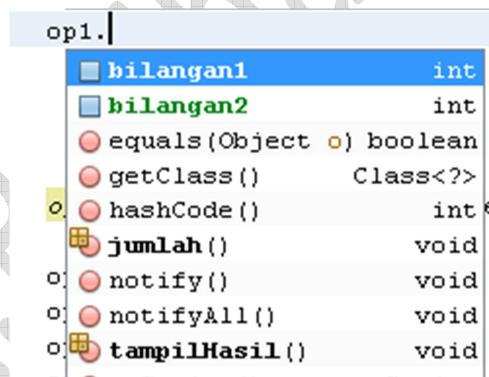
```

: Output - contoh2 (run)
run:
Hasil operasi bilangan : 30
Hasil operasi bilangan : 55
BUILD SUCCESSFUL (total time: 2 seconds)

```

Untuk mengecek apakah suatu class yang sudah dibuat itu betul atau tidak, atau bisa tidak diakses dari class lain, kita bisa melihat dari tool tips yang muncul ketika menulis kode program.

Sebagai contoh, misalkan di file 'Contoh2.java' ini kita tuliskan 'op1.' maka jika muncul tool tips seperti gambar di bawah ini



Yang menandakan bahwa atribut dan method yang ada dalam class 'operasiBilangan' bisa diakses. Atribut dan method yang muncul dalam tool tips hanyalah yang diset sebagai PUBLIC saja, sedangkan yang PRIVATE tidak muncul. Perhatikan, bahwa atribut 'hasil' yang sebelumnya kita set PRIVATE dalam class 'operasiBilangan' tidak muncul dalam tool tips.

Jika struktur penulisan class itu benar, maka secara otomatis class-class tersebut dapat diakses dari class lainnya dalam package yang sama. Namun, jika kita ingin melakukan instantisasi obyek dari suatu class yang class tersebut berasal dari package yang berbeda, maka perlu ditambahkan perintah:

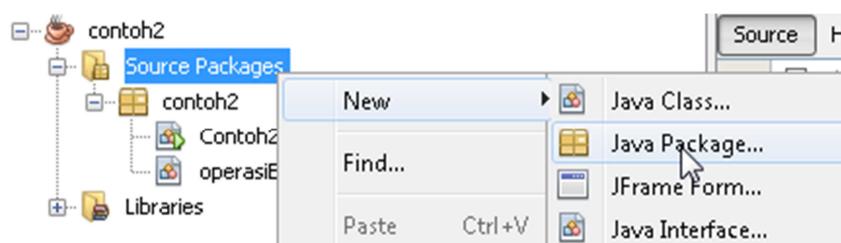
```
import namapackage.namaclass;
```

sebelum kode classnya.

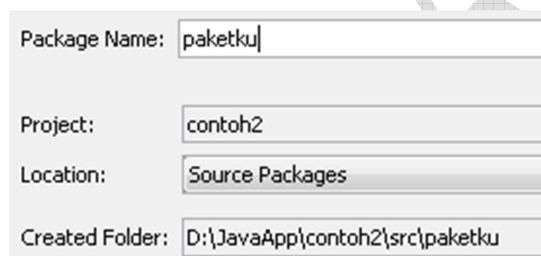
Example 5:

Sebagai contoh, misalkan kita akan membuat package baru dalam project ‘Contoh2’ dengan nama ‘paketku’. Di dalam package ‘paketku’ ini nanti kita buat class dengan nama ‘classku’. Selanjutnya di dalam class ‘classku’ dibuat sebuah method dengan nama ‘cetakHelloWorld’ untuk menampilkan pesan ‘Hello World’.

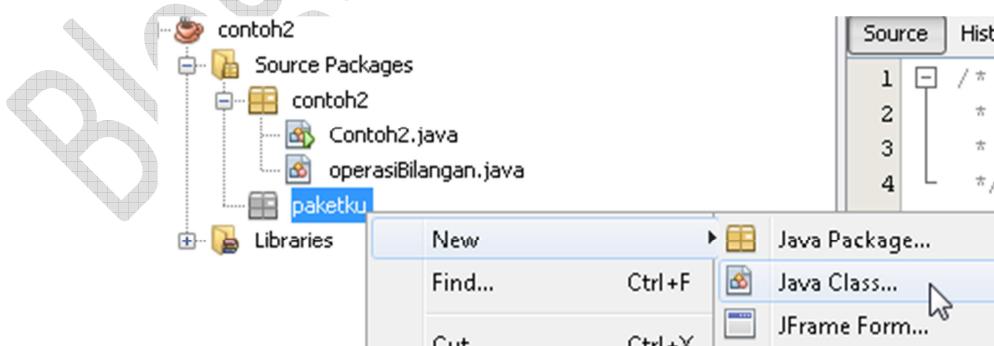
Adapun Cara membuat package baru dalam project ‘Contoh2’ ini adalah dengan mengklik kanan pada ‘Source Package’, lalu pilih NEW – JAVA PACKAGE



Kemudian isikan nama package yang akan dibuat, yaitu ‘paketku’



Setelah package ‘paketku’ dibuat, kita buat class dengan nama ‘classku’. Caranya adalah dengan mengklik kanan pada package ‘paketku’, pilih NEW – JAVA CLASS.



Lalu isikan nama class yang akan dibuat yaitu ‘classku’

Class Name:	classku
Project:	contoh2
Location:	Source Packages
Package:	paketku
Created File:	D:\JavaApp\contoh2\src\paketku\classku.java

Selanjutnya buat method cetakHelloWorld() di dalam class 'classku' yang sudah terbentuk sbb:

```
public class classku {  
  
    public void cetakHelloWorld()  
    {  
        System.out.println("Hello World");  
    }  
  
}
```

Kemudian, misalkan di method main() dalam class 'Contoh2' kita akan lakukan instantisasi suatu obyek dari class 'classku' ini, maka sebelum proses instantisasi ini dilakukan terlebih dahulu tambahkan perintah

```
import paketku.classku;
```

sebelum class 'Contoh2' nya (dalam file 'Contoh2.java'). Perhatikan gambar berikut.

```
package contoh2;  
import paketku.classku;  
  
public class Contoh2 {  
  
    public static void main(String[] args) {  
  
        // instantisasi obyek 'op1'  
        operasiBilangan op1 = new operasiBilangan();  
  
        op1.bilangan1 = 10;  
    }  
}
```

Setelah kita tambahkan perintah import, barulah kita bisa lakukan proses instantisasi suatu obyek dari class 'classku'. Misalkan:

```
package contoh2;  
import paketku.classku;
```

```
public class Contoh2 {  
  
    public static void main(String[] args) {  
  
        // instantisasi obyek 'op1'  
        operasiBilangan op1 = new operasiBilangan();  
  
        op1.bilangan1 = 10;  
        op1.bilangan2 = 20;  
        op1.jumlah();  
        op1.tampilHasil();  
  
        // instantisasi obyek 'op2'  
        operasiBilangan op2 = new operasiBilangan();  
  
        op2.bilangan1 = 30;  
        op2.bilangan2 = 25;  
        op2.jumlah();  
        op2.tampilHasil();  
  
        // instantitasasi obyek 'kelas' dari class 'classku'  
        classku kelas = new classku();  
  
        kelas.cetakHelloWorld();  
  
    }  
}
```

Mengapa sebelum instantisasi obyek 'kelas' yang termasuk class 'classku' ini perlu dilakukan import dari package 'paketku'? Ya... karena class 'Contoh2' ini beda package dengan class 'classku' di mana class 'Contoh2' ini ada di dalam package 'contoh2' sedangkan class 'classku' ada dalam package 'paketku'.

Adapun output dari program Java di atas adalah



```
Output - contoh2 (run)  
run:  
Hasil operasi bilangan : 30  
Hasil operasi bilangan : 55  
Hello World
```

Dalam kedua contoh sebelumnya, method yang dibuat dalam sebuah class merupakan void method (method yang tidak mengembalikan nilai). Dalam contoh ini akan ditunjukkan method yang non void.

Example 6:

Perhatikan kembali project 'Contoh2' sebelumnya. Sekarang kita akan modifikasi class 'operasiBilangan' nya sebagai berikut:

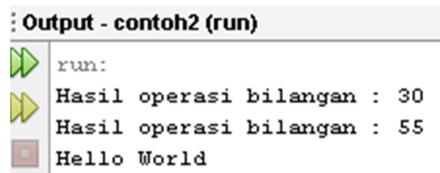
```
public class operasiBilangan {  
    public int bilangan1;  
    public int bilangan2;  
  
    private int jumlah()  
    {  
        int hasil;  
        hasil = this.bilangan1 + this.bilangan2;  
        return hasil;  
    }  
  
    void tampilHasil()  
    {  
        System.out.println("Hasil operasi bilangan : "+ this.jumlah());  
    }  
}
```

Dalam class ‘operasiBilangan’ tersebut, method jumlah() tidak lagi merupakan void method. Dalam contoh tersebut, method jumlah() misalkan kita buat PRIVATE. Kemudian di method tampilHasil() terdapat perintah untuk memanggil method jumlah() tersebut.

Selanjutnya, di class ‘Contoh2’ (contoh2.java) kita juga sedikit modifikasi programnya karena method jumlah() tidak bisa lagi diakses karena bersifat PRIVATE.

```
public class Contoh2 {  
  
    public static void main(String[] args) {  
  
        // instantisasi obyek 'op1'  
        operasiBilangan op1 = new operasiBilangan();  
  
        op1.bilangan1 = 10;  
        op1.bilangan2 = 20;  
        op1.tampilHasil();  
  
        // instantisasi obyek 'op2'  
        operasiBilangan op2 = new operasiBilangan();  
  
        op2.bilangan1 = 30;  
        op2.bilangan2 = 25;  
        op2.tampilHasil();  
  
        classku kelas = new classku();  
        kelas.cetakHelloWorld();  
    }  
}
```

Hasil outputnya pun akan diperoleh sama seperti sebelumnya yaitu



```
Output - contoh2 (run)
run:
Hasil operasi bilangan : 30
Hasil operasi bilangan : 55
Hello World
```

■ Sebuah method dapat pula memuat satu atau lebih parameter. Perhatikan contoh berikut ini.

Example 7:

Dalam contoh ini kita sedikit memodifikasi class 'classku' pada project 'Contoh2' dengan menambahkan method cetakString(). Tambahkan method cetakString() berikut ini pada class 'classku'

```
public class classku {

    public void cetakHelloWorld()
    {
        System.out.println("Hello World");
    }

    // method cetakKata() dengan parameter
    public void cetakKata(String kata)
    {
        System.out.println(kata);
    }
}
```

Dalam method cetakKata() tersebut sebuah parameter 'kata' bertipe data String. Nilai parameter 'kata' tersebut selanjutnya akan ditampilkan ke layar.

Kemudian, di method main() di class 'Contoh2' coba kita tambahkan perintah sbb:

```
kelas.cetakKata("Hallo apa kabar??");
```

Sehingga menjadi

```
public class Contoh2 {

    public static void main(String[] args) {

        // instantisasi obyek 'op1'
        operasiBilangan op1 = new operasiBilangan();

        op1.bilangan1 = 10;
        op1.bilangan2 = 20;
    }
}
```

```

        op1.tampilHasil();

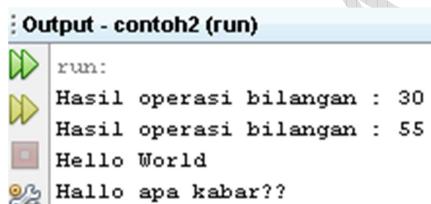
        // instantisasi obyek 'op2'
        operasiBilangan op2 = new operasiBilangan();

        op2.bilangan1 = 30;
        op2.bilangan2 = 25;
        op2.tampilHasil();

        classku kelas = new classku();
        kelas.cetakHelloWorld();
        // memanggil method cetakKata dg parameter
        kelas.cetakKata("Hallo apa kabar??");
    }
}

```

Hasil outputnya adalah:



```

: Output - contoh2 (run)
run:
Hasil operasi bilangan : 30
Hasil operasi bilangan : 55
Hello World
Hallo apa kabar??

```

Jika sebuah method memiliki lebih dari 1 parameter, maka setiap parameternya dipisahkan dengan koma (,) contohnya:

```

public void cetakKata(String kata1, String kata2)
{
    .
    .
}

```

Soal Latihan

- Buat project Java dengan nama 'suhu'. Aplikasi Java yang akan dibuat ini akan mengkonversi suhu dari celcius ke satuan suhu yang lain. Untuk keperluan ini, buatlah class 'konversiSuhu'. Lalu buatlah atribut/properties dan method pada class tsb sbb:

Nama Atribut	Sifat	Tipe Data	Keterangan
Celcius	Public	Float	Nilai suhu dalam celsius
Hasil	Private	Float	Hasil konversi suhu

Nama method	Sifat Method	Tipe Data Return Value	Keterangan
cToR()	Public (non void)	Float	Mengkonversi Celcius ke Reamur
cToK()	Public (non void)	Float	Mengkonversi Celcius ke Kelvin
cToF()	Public (non void)	Float	Mengkonversi Celcius ke Fahrenheit
tampilkanHasil()	Public (void)	-	Menampilkan hasil konversi

Kemudian hitunglah hasil konversi suhu dari celcius berikut ini ke Reamur, Kelvin dan Fahrenheit:

- 30 celcius
- 45 celcius
- 50.5 celcius
- 120 celcius

Keterangan:

- Untuk setiap nilai suhu celcius, buatlah obyek tersendiri.
- Untuk formula konversi suhu, silakan buka http://id.wikipedia.org/wiki/Rumus_konversi_suhu

2. Buat project Java dengan nama 'luasVolume'. Setelah package 'luasvolume' otomatis terbuat, selanjutnya buat 3 buah class dengan nama 'operasi' (operasi.java), 'luas' (luas.java), dan 'volume' (volume.java). Kemudian buat atribut dan method untuk masing-masing class dengan ketentuan sbb:

Class: operasi

Nama Atribut	Sifat	Tipe Data	Keterangan
bil1	Public	Float	Bilangan pertama yg akan dioperasikan
bil2	Public	Float	Bilangan kedua yg akan dioperasikan
hasil	Public	Float	Hasil operasi kedua bilangan

Nama method	Sifat Method	Tipe Data Return Value	Keterangan
dijumlahkan()	Public (non void)	Float	Menjumlahkan 2 bilangan
dikurangi()	Public (non void)	Float	Mengurangi 2 bilangan
dikalikan()	Public (non void)	Float	Mengalikan 2 bilangan
dibagi()	Public (non void)	Float	Membagi 2 bilangan
dipangkatkan2()	Public (non void)	Float	Memangkatkan 2 suatu bilangan

Class: luas

Nama Atribut	Sifat	Tipe Data	Keterangan
Luas	Private	Float	Luas suatu bangun

Nama method	Sifat Method	Tipe Data Return Value	Keterangan
luasPersegi(sisi)	Public (void)	-	Hitung luas persegi
luasLingkaran(r)	Public (void)	-	Hitung luas lingkaran
luasSegitiga(a, t)	Public (void)	-	Hitung luas segitiga
luasPersegiPanjang(p, l)	Public (void)	-	Hitung luas ps. Panjang
tampilLuas()	Public (void)	-	Menampilkan luas bangun

Keterangan: Gunakan class ‘operasi’ untuk mengoperasikan bilangan dalam perhitungan luasnya.

Class: volume

Nama Atribut	Sifat	Tipe Data	Keterangan
Volume	Private	Float	Volume suatu bangun ruang

Nama method	Sifat Method	Tipe Data Return Value	Keterangan
volKubus(sisi)	Public (void)	-	Hitung vol kubus
volBola(r)	Public (void)	-	Hitung vol bola
volBalok(p, l, t)	Public (void)	-	Hitung vol balok
volKerucut(r, t)	Public (void)	-	Hitung vol kerucut
volTabung(r, t)	Public (void)	-	Hitung vol tabung
tampilVolume()	Private (void)	-	Menampilkan vol bgn ruang

Keterangan: Gunakan class ‘operasi’ untuk mengoperasikan bilangan dalam perhitungan luasnya.

Selanjutnya, jawablah pertanyaan ini dengan menggunakan program Java yang Anda buat:

- Hitung luas bangun persegi, jika diketahui panjang sisi 5.8 satuan panjang
- Hitung luas segitiga, jika diketahui panjang alas 10.3 satuan panjang dan tingginya 7.86 satuan panjang
- Sebuah drum dengan diameter 65.8 cm dan tinggi 124.76 cm diisi penuh air. Selanjutnya sebuah drum yang lebih kecil berukuran diameter 30.72 cm dan tinggi 87.32 cm dimasukkan ke dalam drum yang lebih besar tadi, hingga ada air yang tumpah ke luar. Hitunglah berapa liter air yang tumpah. (Ket: untuk semua operasi bilangan, gunakan class ‘operasi’)

3. Sebuah arsitek ingin memperkirakan berapa biaya yang diperlukan untuk membangun sebuah rumah. Berikut ini adalah bahan material utama yang dibutuhkan untuk membangun rumah tersebut:

- Pasir	: 500 m ³ (harga Rp 120.000/m ³)
- Semen	: 70 sak (harga Rp 90.000/sak)
- Batu-Bata	: 10.000 buah (harga Rp 500/buah)
- Kayu kalimantan	: 300 m ³ (harga Rp 200.000/m ³)
- Batu kali untuk pondasi	: 200 buah (harga Rp 1000/buah)
- Genting	: 500 buah (harga Rp 2000/buah)

Buatlah program Java untuk menghitung berapa biaya yang diperlukan untuk membangun rumah tersebut, dengan terlebih dahulu merancang class-classnya, serta atribut dan method nya.

4. Sebuah mobil akan melakukan perjalanan dari kota A ke kota F. Untuk menuju ke kota F, mobil harus melewati beberapa kota yaitu B, C, D dan E dengan jarak sebagai berikut:

Kota A ke B	: 45 km
Kota B ke C	: 51 km
Kota C ke D	: 38 km
Kota D ke E	: 104 km
Kota E ke F	: 93 km

Jika konsumsi bensin untuk mobil tersebut adalah 1 liter untuk tiap 9 km, maka hitunglah berapa liter yang dibutuhkan mobil untuk:

- Berjalan dari kota A ke F
- Berjalan dari kota B ke E
- Berjalan dari kota A ke F, kemudian kembali lagi ke kota B
- Berjalan dari kota A ke F – E – D – C – D – E – D – E – F

Buatlah program Java untuk menjawab persoalan di atas, dengan terlebih dahulu merancang class, atribut dan method nya.

Input dan Output dalam Java

Dalam bagian ini akan dibahas cara membaca data input melalui console serta menampilkan outputnya juga melalui console.

Input Data via Console

Untuk keperluan input data via console, perlu kita buat class khusus.

Example 8:

Sebagai contoh, berikut ini adalah sebuah pembuatan program Java untuk perhitungan gaji karyawan yang beberapa datanya diinput lewat console:

1. Buat Project dengan nama '**gajikaryawan**', untuk package nya juga diberi nama '**gajikaryawan**'
2. Buat class '**inputConsole**' yang disimpan dalam file **inputConsole.java**



dengan isi kode sbb:

inputConsole.java

```
import java.io.*;
public class inputConsole {

    // membaca data string
    public String bacaString()
    {
        BufferedReader bfr = new BufferedReader(new
            InputStreamReader(System.in), 1);
        String string = "";
        try
        {
            string = bfr.readLine();
        }
        catch (IOException ex)
        {
            System.out.println(ex);
        }
        return string;
    }
    // membaca data integer
    public int bacaInt()
    {
        return Integer.parseInt(bacaString());
    }
    // membaca data float
    public float bacaFloat()
    {
        return Float.parseFloat(bacaString());
    }
    // membaca data long integer
    public long bacaLong()
```

```
{  
    return Long.parseLong(bacaString());  
}  
}
```

Secara umum, di dalam class ‘inputConsole’ tersebut, mekanisme method-method untuk membaca input dalam berbagai tipe data itu adalah membaca setiap input dalam bentuk string kemudian input string tersebut diubah ke tipe data yang bersesuaian.

3. Selanjutnya buat beberapa kode di bawah ini pada class Gajikaryawan

```
public class Gajikaryawan {  
  
    public static void main(String[] args) {  
  
        String nama, kodekar;  
        int gapok, jmlanak;  
        float gaber, tunjanak;  
  
        inputConsole input1 = new inputConsole();  
        // input kode karyawan  
        System.out.print("KODE KARYAWAN : ");  
        kodekar = input1.bacaString();  
        // input nama karyawan  
        System.out.print("NAMA KARYAWAN : ");  
        nama = input1.bacaString();  
        // input gaji pokok karyawan  
        System.out.print("GAJI POKOK : ");  
        gapok = input1.bacaInt();  
        // input jumlah anak  
        System.out.print("JML ANAK : ");  
        jmlanak = input1.bacaInt();  
  
        // hitung tunjangan anak -> setiap anak 10% dari gaji pokok  
        tunjanak = (float) ((float) gapok * 0.1 * jmlanak);  
        // hitung gaji bersih = gaji pokok + tunj anak  
        gaber = gapok + tunjanak;  
  
        // output  
        System.out.println("NAMA KARYAWAN : "+nama+"("+kodekar+"));  
        System.out.println("GAJI BERSIH : Rp. "+gaber);  
    }  
}
```

Output Via Console

Secara umum perintah untuk menampilkan output ke layar console adalah

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

atau

```
System.out.print(string);
```

Perbedaan keduanya adalah jika dengan `println()` setelah menampilkan suatu string ke output console, maka terjadi perpindahan baris pada pointernya. Sedangkan untuk `System.out.print()` tidak terjadi perpindahan baris pointernya.

Mengatur Digit Presisi Bilangan Riil (Float)

Secara default, Java akan menampilkan bilangan riil atau float dalam bentuk 15 digit di belakang koma, misalnya:

```
System.out.print(22./7);
```

akan muncul hasil di layar, bilangan 3.142857142857143

Selanjutnya bagaimana jika kita ingin membatasi digit presisi di belakang komanya, misalnya hanya 3 digit saja? Caranya adalah dengan memanfaatkan built in class ‘DecimalFormat’ yang sudah tersedia dalam Java. Berikut ini contohnya:

Example 9:

Contoh program Java untuk menampilkan 3 digit di belakang koma untuk bilangan Pi (22/7)

```
import java.text.DecimalFormat;
public class Contoh2 {

    public static void main(String[] args) {
        // membuat obyek dari class DecimalFormat untuk 3 digit presisi
        DecimalFormat jmldigit = new DecimalFormat("0.000");
        System.out.println("Bilangan Pi: " + jmldigit.format(22./7));
    }
}
```

Class `DecimalFormat` ada dalam suatu package Java dengan nama ‘`java.text.DecimalFormat`’ sehingga di bagian atas program perlu ditambahkan perintah:

```
import java.text.DecimalFormat;
```

Input Data Via GUI (Graphics User Interface)

Selain via console, input data juga bisa dilakukan via GUI. Di dalam Java, untuk membuat aplikasi berbasis GUI bisa menggunakan SWING sebagai package nya, sehingga di dalam program perlu melakukan import dengan perintah sebagai berikut:

```
import javax.swing.*;
```

Berikut ini contoh kode Java untuk menerima input melalui form GUI kemudian outputnya melalui console.

Example 10:

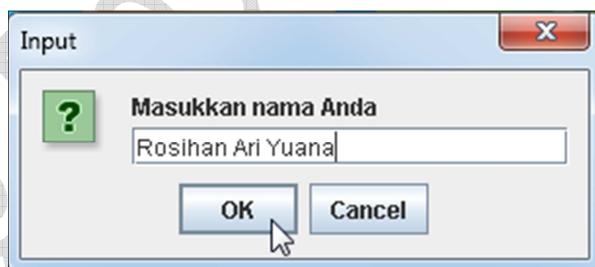
Contoh program Java yang menerima input berupa nama (string) kemudian menampilkan nama yang tadi diinputkan via console.

```
import javax.swing.*;
public class Contoh2 {

    public static void main(String[] args) {

        String nama;
        nama = JOptionPane.showInputDialog("Masukkan nama Anda");
        System.out.println("Hallo selamat datang, " + nama);
    }
}
```

Tampilan dari kode di atas setelah dirunning adalah sbb:



dan outputnya:

```
Output - contoh2 (run)
run:
Hallo selamat datang, Rosihan Ari Yuana
```

Example 11:

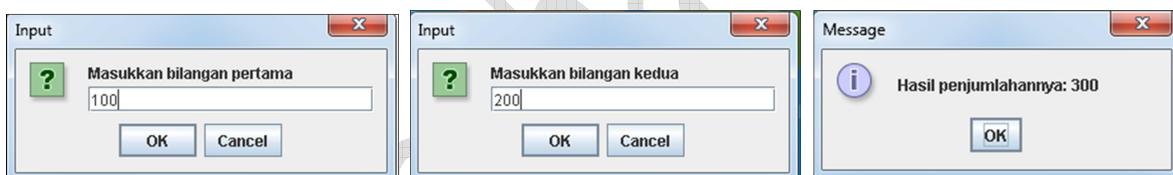
Contoh program Java untuk menjumlahkan dua buah bilangan integer yang diinput melalui GUI kemudian menampilkan hasilnya via Message Dialog.

```
import javax.swing.*;
public class Contoh2 {

    public static void main(String[] args) {

        int bill1;
        int bil2;
        int hasil;
        // baca nilai bilangan ke-1
        bill1 = Integer.parseInt(JOptionPane.showInputDialog("Masukkan
bilangan pertama"));
        // baca nilai bilangan ke-2
        bil2 = Integer.parseInt(JOptionPane.showInputDialog("Masukkan
bilangan kedua"));
        // jumlahkan kedua bilangan
        hasil = bill1 + bil2;
        // tampilkan hasil penjumlahannya via message dialog
        JOptionPane.showMessageDialog(null, "Hasil penjumlahannya:
"+hasil);
    }
}
```

Adapun tampilan program Java tersebut adalah sebagai berikut:



Latihan

1. Tambahkan beberapa method di class 'input Console' untuk membaca input dalam bentuk tipe data short dan double
2. Buatlah project Java untuk menghitung banyaknya masing-masing pecahan uang Rp 100.000, Rp 50.000, Rp 20.000, Rp 10.000 dan Rp 5.000 dari sebuah nilai uang. Sebagai contoh misalkan jumlah uangnya Rp. 1.280.000 maka jumlah pecahannya Rp 100.000 adalah 12, pecahan Rp 50.000 adalah 1, pecahan Rp 20.000 adalah 1, pecahan Rp 10.000 adalah 1 dan pecahan Rp 5.000 nya adalah 0. Input program adalah jumlah nilai uang yang akan dicari pecahannya. Rancanglah terlebih dahulu class, atribut dan method-methodnya.
3. Seseorang dengan tinggi badan 180 cm ingin mengukur tinggi sebuah pohon yang berjarak 20 meter di depan posisi orang tersebut berdiri. Jika sudut pandang orang tersebut terhadap titik tertinggi pohon adalah 30°, maka hitunglah tinggi pohon dengan aplikasi Java yang Anda buat. Input program adalah jarak pohon dengan orang, tinggi orang, dan sudut pandang orang terhadap titik tertinggi pohon.

Petunjuk:

Gunakan method-method built in dari Java berikut ini untuk perhitungan trigonometri

- Math.sin(x) : menghitung nilai sinus x dlm radian (return value: double)
- Math.cos(x) : menghitung nilai cosinus x dlm radian (return value: double)
- Math.tan(x) : menghitung nilai tangen x dlm radian (return value: double)

Keterangan:

Input dan output program menggunakan GUI

Struktur Kontrol Proses

Struktur kontrol proses bertujuan untuk dapat menentukan urutan statement/perintah yang akan dikerjakan atau diproses. Struktur kontrol proses ini antara lain:

Struktur Kontrol Kondisional

Struktur kontrol ini untuk menyatakan proses yang berbentuk persyaratan/kondisional.

Statement IF

Tata cara penulisan statement IF:

```
if (syarat)
{
    statement;
    statement;
    .
    .
}
```

bisa juga berbentuk sebagai berikut

```
if (syarat)
{
    statement;
    statement;
    .
    .
}
else
{
    statement;
    statement;
    .
    .
}
```

atau bisa juga berbentuk

```
if (syarat1)
{
    statement;
    statement;
    .
    .
}
else if (syarat2)
{
    statement;
```

```
        statement;  
        .  
        .  
    }  
else if (syarat3)  
{  
    statement;  
    statement;  
    .  
    .  
}  
. .  
else  
{  
    statement;  
    statement;  
    .  
    .  
}
```

Statement SWITCH

Struktur penulisan statement SWITCH adalah sebagai berikut:

```
switch(ekspresi)  
{  
    case variabel1 : statement;  
    statement;  
    .  
    .  
    break;  
    case variabel2 : statement;  
    statement;  
    .  
    .  
    break;  
    .  
    default : statement;  
    statement;  
    .  
    .  
}
```

Struktur Kontrol Perulangan (Looping)

Struktur kontrol perulangan digunakan untuk mengatur proses yang dijalankan secara berulang-ulang. Berikut ini beberapa statement yang dapat digunakan untuk mengatur proses perulangan:

Statement FOR

Aturan penulisan (syntax) nya adalah:

```
for(ekspresiawal; syarat; ekspresiakhir)
{
    statement;
    statement;
    .
    .
}
```

Statement WHILE

Aturan penulisannya:

```
while(syarat)
{
    statement;
    statement;
    .
    .
}
```

Statement DO WHILE

Aturan penulisannya:

```
do
{
    statement;
    statement;
    .
    .
}
while (syarat);
```

Example 12.

Dalam contoh ini, kita akan membuat sebuah project untuk menentukan gaji bersih karyawan dengan ketentuan:

Gaji bersih = gaji pokok + tunjangan istri + tunjangan anak – potongan

Di mana tunjangan istri diberikan sebesar 10% dari gaji pokok, dan tunjangan anak adalah 5% dari tiap anak. Sedangkan potongannya adalah 5% dari total gaji pokok dan tunjangan-tunjangan.

Pertama kita buat dahulu project dengan nama misalnya: 'projectGaji'

Project Name:	projectGaji
Project Location:	D:\JavaApp
Project Folder:	D:\JavaApp\projectGaji

Selanjutnya kita desain class, method dan atributnya sbb:

Nama Class: 'gaji'

Nama Atribut	Sifat	Tipe Data	Keterangan
kodeKaryawan	Public	String	Kode karyawan
namaKaryawan	Public	String	Nama karyawan
gajipokok	Public	Float	Gaji pokok karyawan
statusMenikah	Public	Char	Status menikah 'Y' atau 'N'
jmlAnak	Public	Integer	Jumlah anak

Nama method	Sifat Method	Tipe Data Return Value	Keterangan
hitungTunjIstri(char s) s: status menikah (y/n)	Public (non void)	Float	Menghitung tunjangan istri
hitungTunjAnak(int n) n: jumlah anak	Public (non void)	Float	Menghitungan tunjangan anak
hitungGajiBersih()	Public (non void)	Float	Menghitung gaji bersih
hitungPotongan()	Public (non void)	Float	Menghitung potongan

Kemudian kita implementasikan desain di atas ke dalam bentuk coding di dalam class 'gaji'

gaji.java

```
package projectgaji;

public class gaji {
    // deklarasi untuk atribut class 'gaji'
    public String kodekaryawan;
    public String namakaryawan;
    public float gajipokok;
    public char statusMenikah;
    public int jmlAnak;

    // deklarasi untuk method dari class 'gaji'
    public float hitungTunjIstri(char s)
    {
        float tunjIstri = 0;
        if (s == 'Y')
```

```
{  
    tunjIstri = (float) (0.1 * this.gajiPokok);  
}  
  
return tunjIstri;  
}  
  
public float hitungTunjAnak(int n)  
{  
    float tunjAnak;  
    tunjAnak = (float) (n * 0.05 * this.gajiPokok);  
    return tunjAnak;  
}  
  
public float hitungPotongan()  
{  
    float jmlPotongan;  
    jmlPotongan = (float) (0.05 * (this.gajiPokok +  
this.hitungTunjAnak(this.jmlAnak) +  
this.hitungTunjIstri(this.statusMenikah)));  
    return jmlPotongan;  
}  
  
public float hitungGajiBersih()  
{  
    float gaber;  
    gaber = (float) (this.gajiPokok +  
this.hitungTunjAnak(this.jmlAnak) +  
this.hitungTunjIstri(this.statusMenikah) - this.hitungPotongan());  
    return gaber;  
}  
}
```

Sedangkan berikut ini isi dari method main() dalam class projectGaji nya.

ProjectGaji.java

```
package projectgaji;  
  
public class ProjectGaji {  
  
    public static void main(String[] args) {  
  
        gaji g1 = new gaji();  
  
        g1.kodekaryawan = "K001";  
        g1.namakaryawan = "ROSIHAN ARI";  
        g1.statusMenikah = 'y';  
    }  
}
```

```
g1.jmlAnak = 3;
g1.gajiPokok = 2500000;

System.out.println("=====");
System.out.println("KODE KARYAWAN : " + g1.kodekaryawan);
System.out.println("NAMA KARYAWAN : " + g1.namakaryawan);
System.out.println("STATUS MENIKAH : " + g1.statusMenikah);
System.out.format("GAJI POKOK : Rp %10.1f \n", g1.gajiPokok);

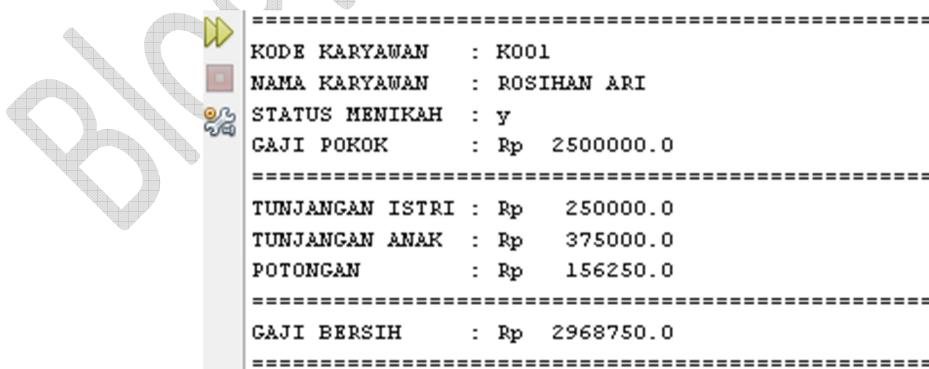
System.out.println("=====");
System.out.format("TUNJANGAN ISTRI : Rp %10.1f \n",
    g1.hitungTunjIstri(g1.statusMenikah));
System.out.format("TUNJANGAN ANAK : Rp %10.1f \n",
    g1.hitungTunjAnak(g1.jmlAnak));
System.out.format("POTONGAN : Rp %10.1f \n",
    g1.hitungPotongan());

System.out.println("=====");
System.out.format("GAJI BERSIH : Rp %10.1f \n",
    g1.hitungGajiBersih());
System.out.println("=====")
}
```

Keterangan:

Perintah `System.out.format()` digunakan untuk memformat tampilan, khususnya untuk pengaturan bilangan yang dalam contoh di atas digunakan untuk memformat tampilan bilangan riil dengan 1 digit angka di belakang koma (%10.1f). Perintah ini sebagai alternatif cara untuk memformat tampilan output selain yang pernah di bahas di bab sebelumnya.

Sedangkan berikut ini tampilan output dari program Javanya



```
=====
KODE KARYAWAN : K001
NAMA KARYAWAN : ROSIHAN ARI
STATUS MENIKAH : Y
GAJI POKOK : Rp 2500000.0
=====

TUNJANGAN ISTRI : Rp 250000.0
TUNJANGAN ANAK : Rp 375000.0
POTONGAN : Rp 156250.0
=====

GAJI BERSIH : Rp 2968750.0
=====
```

Example 13

Dalam contoh ini, akan sedikit dimodifikasi project dari **Example 12** sebelumnya yaitu dengan menambahkan modul untuk simulasi jumlah angsuran pinjaman karyawan per bulan sampai pinjamannya lunas. Adapun besaran angsuran pinjaman perbulannya tergantung golongan karyawannya dengan ketentuan sbb:

- Jika golongannya IIIa, maka angsuran perbulannya Rp 50.000,-
- Jika golongannya IIIb, maka angsuran perbulannya Rp 100.000,-
- Jika golongannya IIIc, maka angsuran perbulannya Rp 150.000,-

Dari simulasi jumlah ansuran per bulan tersebut akan diketahui sampai berapa bulan pinjaman seorang karyawan lunas.

Untuk mengimplementasikan hal ini, kita coba tambahkan 2 atribut dalam class 'gaji' yaitu

```
public float totPinjaman;  
public String gol;
```

dimana 'totPinjaman' menunjukkan jumlah total pinjaman yang dipinjam seorang karyawan, dan 'gol' adalah golongan seorang karyawan.

Selanjutnya, kita buat method **simulasiAngsuranPinjaman()** dalam class 'gaji' sebagai berikut

```
public void simulasiAngsuranPinjaman()  
{  
    float angsuran = 0, totalAngsuran = 0;  
    int bulan;  
    if ("IIIa".equals(this.gol))  
    {  
        angsuran = (float) 50000;  
    }  
    else if ("IIIb".equals(this.gol))  
    {  
        angsuran = (float) 100000;  
    }  
    else if ("IIIc".equals(this.gol))  
    {  
        angsuran = (float) 150000;  
    }  
  
    // menampilkan total pinjaman  
    System.out.format("TOTAL PINJAMAN : Rp %10.1f \n",  
this.totPinjaman);  
  
    // menampilkan angsuran perbulannya  
    bulan = 0;  
    while (totalAngsuran < this.totPinjaman)  
    {
```

```
        totalAngsuran += angsuran;
        bulan++;
        System.out.format("ANGSURAN S/D BLN KE-%2d : Rp %10.1f \n",
bulan, totalAngsuran);
    }

}
```

Keterangan:

Maksud dari statement

```
if ("IIIa".equals(this.gol))
{
    ...
}
```

dari method di atas adalah membandingkan nilai dari string **this.gol** apakah sama dengan 'IIIa' atau tidak, jika sama maka akan menjalankan statement di dalam blok IF nya.

Setelah itu kita bisa panggil method **simulasiAngsuranPinjaman()** tersebut di method **main()** nya dengan terlebih dahulu menset golongan dan total pinjaman si karyawan. Misalnya:

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

    gaji g1 = new gaji();

    g1.kodekaryawan = "K001";
    g1.namakaryawan = "ROSIHAN ARI";
    g1.statusMenikah = 'Y';
    g1.jmlAnak = 3;
    g1.gajiPokok = 2500000;

    // setting golongan karyawan
    g1.gol = "IIIc";
    // setting total pinjaman
    g1.totPinjaman = 1750000;

    System.out.println("=====");
    System.out.println("KODE KARYAWAN : "+g1.kodekaryawan);
    System.out.println("NAMA KARYAWAN : "+g1.namakaryawan);
    System.out.println("STATUS MENIKAH : "+g1.statusMenikah);
    System.out.format("GAJI POKOK : Rp %10.1f \n", g1.gajiPokok);

    System.out.println("=====");
    System.out.format("TUNJANGAN ISTRI : Rp %10.1f \n",
g1.hitungTunjIstri(g1.statusMenikah));
}
```

```

System.out.format("TUNJANGAN ANAK : Rp %10.1f \n",
g1.hitungTunjAnak(g1.jmlAnak));
System.out.format("POTONGAN           : Rp %10.1f \n",
g1.hitungPotongan());

System.out.println("=====");
System.out.format("GAJI BERSIH       : Rp %10.1f \n",
g1.hitungGajiBersih());

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

// memanggil method simulasiAngsuranPinjaman()
g1.simulasiAngsuranPinjaman();
}

```

Adapun hasil atau output dari program ini adalah

```

=====
KODE KARYAWAN   : K001
NAMA KARYAWAN   : ROSIHAN ARI
STATUS MENIKAH  : Y
GAJI POKOK      : Rp  2500000.0
GOLONGAN        : IIIc
=====

TUNJANGAN ISTRI : Rp  250000.0
TUNJANGAN ANAK  : Rp  375000.0
POTONGAN        : Rp  156250.0
=====

GAJI BERSIH     : Rp  2968750.0
=====

TOTAL PINJAMAN  : Rp  1750000.0
ANGSURAN BLN KE- 1 : Rp  150000.0
ANGSURAN BLN KE- 2 : Rp  300000.0
ANGSURAN BLN KE- 3 : Rp  450000.0
ANGSURAN BLN KE- 4 : Rp  600000.0
ANGSURAN BLN KE- 5 : Rp  750000.0
ANGSURAN BLN KE- 6 : Rp  900000.0
ANGSURAN BLN KE- 7 : Rp  1050000.0
ANGSURAN BLN KE- 8 : Rp  1200000.0
ANGSURAN BLN KE- 9 : Rp  1350000.0
ANGSURAN BLN KE-10 : Rp  1500000.0
ANGSURAN BLN KE-11 : Rp  1650000.0
ANGSURAN BLN KE-12 : Rp  1800000.0

```

Latihan

- Dari hasil terakhir project yang ada di Example 13, ubahlah sistem penggajian karyawannya dengan ketentuan bahwa besarnya gaji pokoknya tergantung golongannya:

GOLONGAN	GAJI POKOK
IIIa	Rp. 1.500.000,-
IIIb	Rp 1.850.000,-
IIIc	Rp 2.100.000,-
IIId	Rp 2.350.000,-

Dengan demikian, golongan karyawan akan menentukan besarnya gaji bersih setiap karyawan.

2. Dari hasil terakhir **Latihan No 1** sebelumnya, tambahkan method baru dengan nama **simulasiAngsuranRumah()**. Method ini nantinya digunakan untuk mensimulasikan jumlah angsuran sampai dengan bulan tertentu, hingga lunas. Adapun ketentuan jumlah angsuran tiap bulannya tergantung golongan karyawan yaitu:

GOLONGAN	ANGSURAN PER BULAN
IIIa	5% dari gaji pokok
IIIb	7% dari gaji pokok
IIIc	10% dari gaji pokok
IIId	15% dari gaji pokok

3. Dengan menggunakan hasil Latihan No 2 di atas, tentukan berapa lama waktu (dalam bulan) yang diperlukan karyawan-karyawan berikut ini supaya angsuran rumahnya lunas?

KODE KARY	NAMA KARY	GOL	HARGA RUMAH
K001	JOKO SETIAWAN	IIIb	Rp 120jt
K002	RUDI HARTONO	IIIc	Rp 145jt
K003	BAYU HARJONO	IIIa	Rp 165jt

Constructor

Di dalam OOP, ada istilah ‘constructor’. ‘Constructor’ ini melekat pada suatu class, yang dengannya kita bisa menset beberapa nilai atribut sekaligus dari suatu obyek ketika proses instansiasi. Jika sebelumnya setiap kita ingin menset nilai atribut dari sebuah obyek, maka prosesnya adalah instansiasi baru set nilai atribut, namun dengan ‘constructor’ ini kedua langkah tersebut bisa dijadikan dalam satu langkah saja.

Example 14

Berikut ini contoh constructor yang ada dalam sebuah class. Perhatikan contoh class ‘operasi’ berikut ini

```
public class operasi {  
  
    public int bil1;  
    public int bil2;  
    public int hasil;  
  
    private void jumlahkan()  
    {  
        this.hasil = this.bil1 + this.bil2;  
    }  
  
    public void tampilhasil()  
    {  
        System.out.println("Hasil penjumlahannya : " + this.hasil);  
    }  
}
```

Apabila kita ingin menjumlahkan 2 bilangan misalnya 10 dan 20, maka kita lakukan perintah sbb:

```
operasi opl = new operasi();  
opl.bil1 = 10;  
opl.bil2 = 20;  
opl.jumlahkan();  
opl.tampilhasil();
```

Selanjutnya, misalkan kita buat constructor sbb:

```
public class operasi {  
  
    public int bil1;  
    public int bil2;  
    public int hasil;  
  
    // constructor class operasi  
    public operasi(int x, int y)  
    {  
        this.bil1 = x;  
    }  
}
```

```
        this.bil2 = y;
    }

    public void jumlahkan()
    {
        this.hasil = this.bil1 + this.bil2;
    }

    public void tampilhasil()
    {
        System.out.println("Hasil penjumlahannya : " + this.hasil);
    }
}
```

Setelah dibuat constructor, kita dapat memberikan perintah berikut ini

```
operasi opl = new operasi(10, 20);
```

untuk proses instansiasi sekaligus menset atribut bil1 dan bil2 nya. Sehingga secara umum perintah untuk menjumlahkan dua bilangannya adalah sbb:

```
operasi opl = new operasi(10, 20);
opl.jumlahkan();
opl.tampilhasil();
```

Latihan

1. Buatlah project dengan nama ‘bangunDatar’ untuk mencari luas dan keliling beberapa buah bangun datar. Di dalam project tersebut, buatlah sebuah class dengan nama ‘persegiPanjang’ dan ‘segiTiga’. Tentukan constructor yang tepat untuk class-class tersebut. Dalam setiap class, buat method dengan nama ‘hitungLuas’ dan ‘hitungKeliling’.
2. Buatlah project dengan nama ‘deret’ untuk menampilkan deret aritmatika. Selanjutnya buat class dengan nama ‘aritmatika’ dengan constructor a (suku awal), b (selisih), dan n (jumlah suku). Kemudian buat method tampilDeret() untuk menampilkan deret aritmatikanya.

Larik (Array)

Seperti halnya bahasa pemrograman yang lain, di dalam Java juga ada penggunaan Array. Di dalam java nomor indeks suatu array dimulai dari 0.

Berikut ini cara mendeklarasikan sebuah array dengan n buah elemen

```
tipedata[] namaarray = new tipedata[n];
```

Sebagai contoh, perhatikan perintah berikut ini untuk membuat array dengan nama arrayku bertipe data integer dengan jumlah elemennya 10.

```
int[] arrayku = new int[10];
```

Example 15

Berikut ini contoh penggunaan array dalam bentuk string. Misalkan diberikan 5 buah data string berupa nama karyawan, selanjutnya program Java akan menampilkan panjang karakter untuk setiap nama karyawan tersebut.

Pertama, buatlah project dengan nama 'arrayString'

Selanjutnya buat class dengan nama 'operasistring', lalu buat method sbb:

```
package arraystring;
public class operasistring {
    // method untuk mencari panjang suatu string x
    public int panjangstring(String x)
    {
        return x.length();
    }
}
```

Kemudian, buat class dengan nama 'stringku'

```
package arraystring;
public class stringku {

    // atribut untuk menyimpan 5 nama karyawan berupa array
    public String[] nama = new String[5];

    // method untuk menampilkan nama setiap karyawan dan pjg
    // karakternya
    public void tampildata()
    {
        int i;

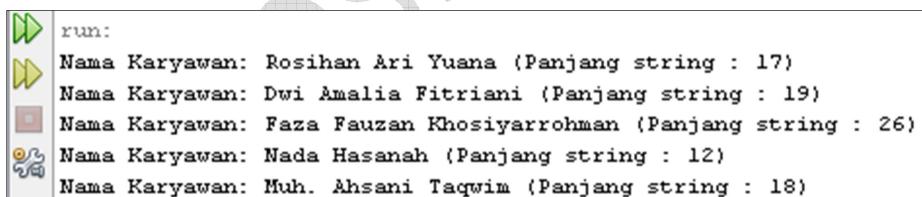
        for(i=0; i<this.nama.length; i++)
    }
```

```
{  
    operasistring os1 = new operasistring();  
    System.out.println("Nama Karyawan: " + this.nama[i] + "  
(Panjang string : " + os1.panjangstring(this.nama[i]) + ")");  
}  
}  
}
```

Terakhir kita buat program berikut ini di main class nya

```
package arraystring;  
public class ArrayString {  
    public static void main(String[] args) {  
        // instansiasi obyek  
        stringku str = new stringku();  
        // set nama setiap karyawan ke dalam array  
        str.nama[0] = "Rosihan Ari Yuana";  
        str.nama[1] = "Dwi Amalia Fitriani";  
        str.nama[2] = "Faza Fauzan Khosiyarrohman";  
        str.nama[3] = "Nada Hasanah";  
        str.nama[4] = "Muh. Ahsani Taqwim";  
        // panggil method untuk menampilkan data  
        str.tampildata();  
    }  
}
```

Adapun output dari project di atas adalah



```
run:  
Nama Karyawan: Rosihan Ari Yuana (Panjang string : 17)  
Nama Karyawan: Dwi Amalia Fitriani (Panjang string : 19)  
Nama Karyawan: Faza Fauzan Khosiyarrohman (Panjang string : 26)  
Nama Karyawan: Nada Hasanah (Panjang string : 12)  
Nama Karyawan: Muh. Ahsani Taqwim (Panjang string : 18)
```

Example 17

Berikut ini contoh penggunaan array untuk proses pengolahan data karyawan. Dalam project ini nantinya, user diminta memasukkan jumlah karyawan yang mau dientri. Kemudian aplikasi meminta memasukkan data NIK (nomor induk karyawan), nama karyawan, dan gaji pokok untuk setiap karyawan. Selanjutnya, seluruh data karyawan ini akan ditampilkan dalam bentuk tabel dan ditambahkan informasi berupa rata-rata besar gaji pokoknya.

Untuk kasus ini, silakan membuat project dengan nama 'arrayKaryawan'

Kemudian, buat class untuk kepentingan input data via console seperti yang pernah di bahas di bab sebelumnya.

Class 'myInput'

```
package arraykaryawan;
import java.io.*;

public class myInput {

    // membaca data string
    public String bacaString()
    {
        BufferedReader bfr = new BufferedReader(new
                                                InputStreamReader(System.in), 1);
        String string = "";
        try
        {
            string = bfr.readLine();
        }
        catch (IOException ex)
        {
            System.out.println(ex);
        }
        return string;
    }
    // membaca data integer
    public int bacaInt()
    {
        return Integer.parseInt(bacaString());
    }
}
```

Selanjutnya, kita buat class 'dataKaryawan'

```
package arraykaryawan;

public class dataKaryawan {

    // atribut untuk NIK (nomor induk karyawan) berupa array string
    // max 100 data
    private String[] nik = new String[100];
    // atribut untuk nama karyawan berupa array string
    // max 100 data
    private String[] nama = new String[100];
    // atribut untuk gaji pokok karyawan berupa array int
    // max 100 data
    private int[] gapok = new int[100];
    // atribut untuk banyak karyawan
    public int n;
```

```
// method untuk menyimpan NIK x pada data array karyawan ke-i
public void setNIK(int i, String x)
{
    this.nik[i] = x;
}

// method unt menyimpan nama karyawan x pd data array karyawan ke-i
public void setNama(int i, String x)
{
    this.nama[i] = x;
}

// method unt menyimpan gapok x pd data array karyawan ke-i
public void setGapok(int i, int x)
{
    this.gapok[i] = x;
}

// method untuk menghitung rata2 gapok dari n karyawan
public float rerataGapok()
{
    int sum = 0, i;
    for(i=0; i<this.n; i++)
    {
        sum += this.gapok[i];
    }
    return sum/this.n;
}

// method untuk menset atribut jumlah karyawan sejumlah x
public void setJmlKaryawan(int x)
{
    this.n = x;
}

// method untuk menampilkan data karyawan
public void tampilData()
{
    int i;

    System.out.println("=====");
    System.out.println("NIK      NAMA KARYAWAN                      GAJI");
    System.out.println("POKOK");
    System.out.println("=====");
    for(i=0; i<this.n; i++)
    {
        System.out.println(String.format("%-5s %-35s Rp %10d",
            this.nik[i], this.nama[i], this.gapok[i]));
    }
}
```

```
System.out.println("=====");
    System.out.println("Rata-rata gaji Pokok dari " + this.n +
karyawan adalah Rp " + this.rerataGapok());
}
}
```

Keterangan:

Format specifier `%-35s` digunakan untuk mengatur posisi string supaya rata kiri, dengan space 35 karakter.

Terakhir, di bagian main class kita buat sebagai berikut:

```
package arraykaryawan;

public class ArrayKaryawan {

    public static void main(String[] args) {

        int i;
        // instansiasi obyek dataKar dari class dataKaryawan
        dataKaryawan dataKar = new dataKaryawan();
        // instansiasi obyek input1 dari class myInput
        myInput input1 = new myInput();
        // input jumlah karyawan
        System.out.print("Berapa jumlah karyawan : ");
        // set jumlah karyawan ke atribut n dari obyek dataKar
        dataKar.setJmlKaryawan(input1.bacaInt());

        // looping untuk proses input data karyawan
        for(i=0; i<dataKar.n; i++)
        {
            System.out.println("Karyawan ke-"+(i+1));
            System.out.print("Masukkan NIK : ");
            // set nilai atribut nik dalam array
            dataKar.setNIK(i, input1.bacaString());
            System.out.print("Masukkan Nama Karyawan : ");
            // set nilai atribut nama dalam array
            dataKar.setNama(i, input1.bacaString());
            System.out.print("Masukkan Gaji Pokok : ");
            // set nilai atribut gapok dalam array
            dataKar.setGapok(i, input1.bacaInt());
        }

        // tampil data
        dataKar.tampilData();
    }
}
```

Berikut ini tampilan project ketika dijalankan

```

Berapa jumlah karyawan : 2
Karyawan ke-1
Masukkan NIK : K001
Masukkan Nama Karyawan : Rosihan Ari Yuana
Masukkan Gaji Pokok : 2500000
Karyawan ke-2
Masukkan NIK : K002
Masukkan Nama Karyawan : Dwi Amalia Fitriani
Masukkan Gaji Pokok : 1750000
=====
NIK      NAMA KARYAWAN                      GAJI POKOK
=====
K001    Rosihan Ari Yuana                  Rp    2500000
K002    Dwi Amalia Fitriani                Rp    1750000
=====
Rata-rata gaji Pokok dari 2 karyawan adalah Rp 2125000.0
  
```

Selanjutnya bagaimana jika array nya merupakan array 2 dimensi? Berikut ini adalah cara pendeklarasiannya

```
tipedata[][] namaarray = new tipedata[n][m];
```

di mana n adalah jumlah elemen untuk baris, dan m adalah untuk jumlah elemen kolom.

Di bawah ini adalah contoh pendeklarasian suatu variabel bertipe data array ‘contoharray’ bertipe data float dengan jumlah elemen 10 x 20;

```
float[][] contoharray = new float[10][20];
```

Example 18

Berikut ini contoh program Java untuk menjumlahkan 2 buah matriks A dan B dengan ukuran masing-masing 2 x 3.

Buat project baru dengan nama ‘matriks’

Kemudian buat class dengan nama ‘operasi’

```

package matriks;

public class operasi {

    // deklarasi matriks A, B, C dengan ukuran 2x3 bertipe integer
    private int[][] matriksA = new int[2][3];
  
```

```
private int[][] matriksB = new int[2][3];
private int[][] matriksC = new int[2][3];

// set elemen matriks A
public void isimatriksA()
{
    this.matriksA[0][0] = 2;
    this.matriksA[0][1] = 3;
    this.matriksA[0][2] = 1;
    this.matriksA[1][0] = -4;
    this.matriksA[1][1] = 0;
    this.matriksA[1][2] = -1;
}

// set elemen matriks B
public void isimatriksB()
{
    this.matriksB[0][0] = 1;
    this.matriksB[0][1] = 2;
    this.matriksB[0][2] = 1;
    this.matriksB[1][0] = -3;
    this.matriksB[1][1] = 0;
    this.matriksB[1][2] = -5;
}

// jumlahkan setiap elemen matriks A dan B, simpan di matriks C
public void jumlahkan()
{
    this.matriksC[0][0] = this.matriksA[0][0] +
this.matriksB[0][0];
    this.matriksC[0][1] = this.matriksA[0][1] +
this.matriksB[0][1];
    this.matriksC[0][2] = this.matriksA[0][2] +
this.matriksB[0][2];
    this.matriksC[1][0] = this.matriksA[1][0] +
this.matriksB[1][0];
    this.matriksC[1][1] = this.matriksA[1][1] +
this.matriksB[1][1];
    this.matriksC[1][2] = this.matriksA[1][2] +
this.matriksB[1][2];
}

// method untuk menampilkan matriks A, B atau C
public void tampilmatriks(char namamatriks)
{
    int i, j;

    if (namamatriks == 'A')
    {
```

```
        for(i=0; i<matriksA.length; i++)
    {
        for(j=0; j<matriksA[i].length; j++)
        {
            System.out.print(String.format("%4d",
this.matriksA[i][j]));
        }
        System.out.println();
    }
}
else if (namamatriks == 'B')
{
    for(i=0; i<matriksB.length; i++)
    {
        for(j=0; j<matriksB[i].length; j++)
        {
            System.out.print(String.format("%4d",
this.matriksB[i][j]));
        }
        System.out.println();
    }
}
else if (namamatriks == 'C')
{
    for(i=0; i<matriksC.length; i++)
    {
        for(j=0; j<matriksC[i].length; j++)
        {
            System.out.print(String.format("%4d",
this.matriksC[i][j]));
        }
        System.out.println();
    }
}
}
```

Selanjutnya di class utama 'Matriks' buat coding berikut ini

```
package matriks;

public class Matriks {

    public static void main(String[] args) {

        operasi op1 = new operasi();
        // panggil method untuk menset nilai matriks A
        op1.isimatriksA();
        System.out.println("MATRIKS A");
```

```
// panggil method untuk tampilan matriks A  
opl.tampilmatriks('A');  
// panggil method untuk menset nilai matriks B  
opl.isimatriksB();  
System.out.println("MATRIKS B");  
// panggil method untuk tampilan matriks B  
opl.tampilmatriks('B');  
// panggil method untuk menjumlahkan matriks A dan B  
opl.jumlahkan();  
System.out.println("MATRIKS C");  
// panggil method untuk tampilan matriks C  
opl.tampilmatriks('C');  
}  
}
```

Latihan

1. Dengan menggunakan array, buat project Java untuk menampilkan bilangan maksimum, minimum, dan rata-rata dari beberapa bilangan yang diinput melalui console.
2. Dalam Example 17, silakan dimodifikasi untuk menampilkan data karyawan yang memiliki gaji pokok paling rendah dan paling tinggi
3. Tambahkan method dalam Example 18 untuk mencari pengurangan matriksnya
4. Analog dengan Exampe 18, kembangkan program Java nya sehingga bisa melakukan penjumlahan dan pengurangan untuk sembarang matriks $m \times n$

Pewarisan (Inheritance)

Di dalam Java, sifat suatu class dapat diturunkan atau diwariskan pada sebuah class lain. Istilah sifat yang diwariskan ini adalah atribut atau method. Class yang sifatnya diwariskan ini dinamakan superclass, dan class yang sifatnya mewarisi class lain dinamakan subclass.

Pewarisan ini merupakan keuntungan dalam PBO karena suatu sifat atau method yang didefinisikan dalam suatu superclass dapat diwariskan pada semua subclassnya. Sehingga di dalam subclass tersebut tidak perlu menulis kode program lagi untuk method tersebut.

Example 19

Misalkan diberikan sebuah class dengan nama ‘kendaraan’ sbb:

```
public class kendaraan {  
  
    public int jmlRoda;  
    public int jmlSeat;  
    public String nama;  
  
    public void tampilJmlRoda()  
    {  
        System.out.println(this.nama + " jumlah rodanya: " +  
this.jmlRoda);  
    }  
  
    public void tampilJmlSeat()  
    {  
        System.out.println(this.nama + " jumlah seat: " +  
this.jmlSeat);  
    }  
  
}
```

Selanjutnya kita buat class di file yang lain yang merupakan turunan atau warisan dari class ‘kendaraan’ tersebut dengan nama ‘keretaApi’. Dalam hal ini class ‘kendaraan’ disebut superclass, dan ‘keretaApi’ adalah subclass.

```
public class keretaApi extends kendaraan {  
  
    public int jmlGerbong;  
  
    public void tampilJmlGerbong()  
    {  
        System.out.println(this.nama + " jumlah gerbongnya " +  
this.jmlGerbong);  
    }  
}
```

{}

Perhatikan, di dalam class ‘keretaApi’ terdapat atribut tambahan yaitu ‘jmlgerbong’ dan method ‘tampilJmlGerbong’.

Sekarang, bagaimana cara menggunakan kedua class tersebut? Perhatikan contohnya berikut ini

```
public static void main(String[] args) {  
  
    kendaraan ob1 = new kendaraan();  
    ob1.nama = "Kijang Innova";  
    ob1.jmlRoda = 4;  
    ob1.jmlSeat = 6;  
    ob1.tampilJmlRoda();  
    ob1.tampilJmlSeat();  
  
    keretaApi ob2 = new keretaApi();  
    ob2.nama = "KA. Argo Lawu";  
    ob2.jmlRoda = 100;  
    ob2.jmlSeat = 500;  
    ob2.jmlGerbong = 20;  
    ob2.tampilJmlGerbong();  
    ob2.tampilJmlRoda();  
    ob2.tampilJmlSeat();  
  
}
```

Perhatikan contoh di atas, bahwa obyek ‘ob2’ dari instansiasi class ‘keretaApi’ bisa diset atribut-atributnya seperti halnya ‘ob1’, demikian juga method-methodnya. Khusus atribut ‘jmlGerbong’ dan method ‘tampilJmlGerbong()’ hanya dimiliki oleh class ‘keretaApi’ saja.

Keterangan:

Semua atribut dan method dari superclass yang memiliki modifier ‘public’ akan bisa langsung diwariskan pada subclassnya, demikian juga ‘protected’ namun dengan syarat subclass dan superclass nya berada dalam package yang sama.

Example 20

Dalam contoh ini, kita sedikit modifikasi program di Example 19 yaitu dengan menambahkan constructor dengan sebuah parameter untuk setting atribut nama kendaraan pada class ‘kendaraan’ nya.

```
public class kendaraan {  
  
    public int jmlRoda;  
    public int jmlSeat;  
    public String nama;  
  
    // constructor kendaraan
```

```
public kendaraan(String x)
{
    this.nama = x;
}

public void tampilJmlRoda()
{
    System.out.println(this.nama + " jumlah rodanya: " +
this.jmlRoda);
}

public void tampilJmlSeat()
{
    System.out.println(this.nama + " jumlah seat: " +
this.jmlSeat);
}
```

Jika di dalam class ‘kendaraan’ terdapat constructor, maka begitu juga di class warisannya yaitu ‘keretaApi’. Dalam contoh ini, misalkan kita buat constructor dengan tambahan 1 parameter, yaitu untuk setting atribut tipe KA. Sedangkan 1 parameternya lagi untuk nama kendaraan.

```
public class keretaApi extends kendaraan {

    public int jmlGerbong;
    public String tipeKA;

    // constructor keretaApi
    public keretaApi(String x, String y)
    {
        super(x);
        this.tipeKA = y;
    }

    public void tampilJmlGerbong()
    {
        System.out.println(this.nama + " jumlah gerbongnya " +
this.jmlGerbong);
    }

    public void tampilTipeKA()
    {
        System.out.println(this.nama + " adalah jenis kereta " +
this.tipeKA);
    }
}
```

Perhatikan constructor dalam class 'keretaApi' tersebut.

```
public keretaApi(String x, String y)
{
    super(x);
    this.tipeKA = y;
}
```

Perintah super(x) menunjukkan bahwa parameter x ini terkait dengan parameter 'nama kendaraan' dari constructor class kendaraan. Sedangkan parameter y nya untuk setting atribut tipe KA nya. Berikut ini contoh pemanggilan class-class nya dalam implementasi.

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

    kendaraan ob1 = new kendaraan("Kijang Innova");
    ob1.jmlRoda = 4;
    ob1.jmlSeat = 6;
    ob1.tampilJmlRoda();
    ob1.tampilJmlSeat();

    keretaApi ob2 = new keretaApi("KA. Argo Lawu", "Executive");
    ob2.jmlRoda = 100;
    ob2.jmlSeat = 500;
    ob2.jmlGerbong = 20;
    ob2.tampilJmlGerbong();
    ob2.tampilJmlRoda();
    ob2.tampilJmlSeat();
    ob2.tampilTipeKA();
}
```

Adapun outputnya adalah sbb:

```
Kijang Innova jumlah rodanya: 4
Kijang Innova jumlah seat: 6
KA. Argo Lawu jumlah gerbongnya 20
KA. Argo Lawu jumlah rodanya: 100
KA. Argo Lawu jumlah seat: 500
KA. Argo Lawu adalah jenis kereta Executive
```

Selanjutnya bagaimana jika kita ingin supaya sebuah class tidak bisa dibuat subclass atau class turunannya lagi? Caranya hanya dengan menambahkan kata kunci 'final', contoh:

```
public final class kendaraan {
    .
    .
}
```

Overriding Method

Misalkan kita memiliki sebuah superclass sebagai berikut

```
public class superclass1 {  
  
    public int atribut1;  
    public int atribut2;  
  
    public void method1()  
    {  
        System.out.println("Ini adalah hasil method1() superclass");  
    }  
  
}
```

Kemudian kita buat beberapa subclass dari superclass di atas, yaitu

```
public class subclass1 extends superclass1 {  
  
    public void method1()  
    {  
        System.out.println("Ini adalah hasil method1() subclass1");  
    }  
  
}
```

Dan

```
public class subclass2 extends superclass1 {  
  
    public void method1()  
    {  
        System.out.println("Ini adalah hasil method1() subclass2");  
    }  
  
}
```

Jika diperhatikan, maka baik di superclass maupun di subclass1 dan subclass2 terdapat method dengan nama yang sama yaitu method1(). Dalam hal ini, method1() yang ada di subclass1 dikatakan overriding method dari method1() yang ada di superclass1. Demikian juga untuk method1() yang ada di subclass2. Jika suatu obyek diinstansiasikan dari class subclass1, dan kemudian obyek tersebut diberikan method method1() maka akan memanggil method1() yang ada dalam subclass1 tersebut karena sudah mengoverride method1() yang ada di superclass1.

Berikut ini contoh implementasinya untuk beberapa obyek dari class yang berbeda.

```
public static void main(String[] args) {  
  
    superclass1 ob1 = new superclass1();  
    subclass1 ob2 = new subclass1();  
    subclass2 ob3 = new subclass2();  
  
    // memanggil method1 dari superclass1  
    ob1.method1();  
    // memanggil method1 dari subclass1  
    ob2.method1();  
    // memanggil method1 dari subclass2  
    ob3.method1();  
  
}
```

Akan menghasilkan output:

```
Ini adalah hasil method1() superclass  
Ini adalah hasil method1() subclass1  
Ini adalah hasil method1() subclass2
```

Polimorfisme

Istilah polimorfisme secara umum adalah suatu materi yang bisa memiliki banyak bentuk. Dalam PBO, istilah polimorfisme adalah kemampuan untuk mendefinisikan karakteristik subclass-subclass secara unik.

Untuk contoh gambaran dari kemampuan polimorfisme dalam Java ini, adalah tinjau kembali superclass1, subclass1 dan subclass2 dari bab **Overriding Method**.

Selanjutnya di main method nya, kita buat seperti ini:

```
public static void main(String[] args) {  
  
    superclass1 ob1, ob2, ob3;  
  
    ob1 = new superclass1();  
    ob2 = new subclass1();  
    ob3 = new subclass2();  
  
    ob1.method1();  
    ob2.method1();  
    ob3.method1();  
  
}
```

Perhatikan pada perintah:

```
superclass1 ob1, ob2, ob3;
```

Meskipun kita definisikan ob1, ob2, dan ob3 sebagai tipe data superclass1 yang sama, namun method method1() yang dipanggil dari tiap-tiap obyek melalui perintah

```
ob1.method1();  
ob2.method1();  
ob3.method1();
```

menghasilkan output yang berbeda –beda:

```
Ini adalah hasil method1() superclass  
Ini adalah hasil method1() subclass1  
Ini adalah hasil method1() subclass2
```

Hal ini dikarenakan sebelum pemanggilan method method1() untuk setiap obyeknya, terlebih dahulu kita instansiasi masing-masing obyeknya untuk class yang berbeda:

```
ob1 = new superclass1();  
ob2 = new subclass1();  
ob3 = new subclass2();
```

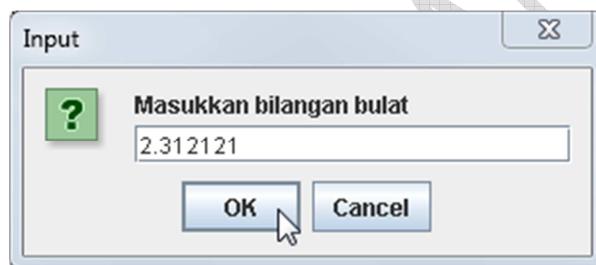
Penanganan Exception

Exception adalah peristiwa yang terjadi ketika proses running program yang mengakibatkan program berhenti, ditandai dengan munculnya pesan error.

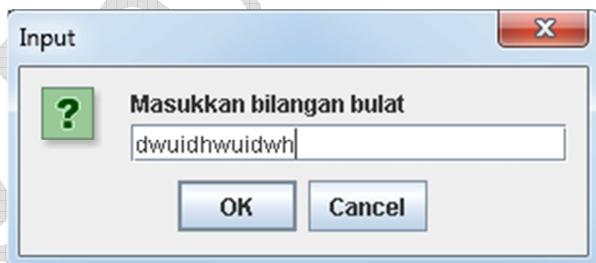
Sebagai contoh, misalkan kita punya program Java sbb:

```
public static void main(String[] args) {  
  
    int bilBulat;  
    bilBulat = Integer.parseInt(JOptionPane.showInputDialog("Masukkan  
    bilangan bulat"));  
}  
}
```

Ketika program sederhana di atas dijalankan, maka akan meminta masukan sebuah bilangan bulat (integer). Namun, apa yang akan terjadi jika yang dimasukkan bukan bilangan bulat?



Atau



Maka selanjutnya akan muncul pesan error Exception sbb:

```
run:  
Exception in thread "main" java.lang.NumberFormatException: For input string: "dwuidhwuidwh"  
at java.lang.NumberFormatException.forInputString(NumberFormatException.java:65)  
at java.lang.Integer.parseInt(Integer.java:492)  
at java.lang.Integer.parseInt(Integer.java:527)  
at javaapplication39.JavaApplication39.main(JavaApplication39.java:22)  
Java Result: 1
```

Untuk mengantisipasi munculnya Exception tersebut, kita bisa melakukan penanganan dengan statement

```
try
{
    ...
}
catch (namaexception var)
{
    ...
}
```

Keterangan:

- ‘namaexception’ nantinya diisikan dengan nama exception yang muncul, dalam contoh kasus sebelumnya yang merupakan nama exception adalah ‘**NumberFormatException**’
- Secara umum, kita bisa menggunakan keyword ‘**Exception**’ pada ‘namaexception’ untuk menangkap semua exception yang terjadi. Keyword ‘Exception’ adalah top level dari semua exception.

```
run:
Exception in thread "main" java.lang.NumberFormatException: For input
    at java.lang.NumberFormatException.forInputString(NumberFormat
    at java.lang.Integer.parseInt(Integer.java:492)
```

Sedangkan ‘var’ diisikan dengan sembarang nama variabel.

Dengan statement try-catch di atas, maka pesan error exception tidak akan muncul namun akan digantikan dengan pesan atau prosedur lain yang kita tuliskan dalam bagian **catch**.

Berikut ini contoh penanganan Exception dari kasus sebelumnya.

```
try
{
    bilBulat = Integer.parseInt(JOptionPane.showInputDialog("Masukkan
bilangan bulat"));
}
catch (NumberFormatException e)
{
    JOptionPane.showMessageDialog(null, "Input salah");
}
```

Dengan penanganan di atas, maka ketika input yang dimasukkan bukan bilangan bulat maka akan muncul pesan ‘Input Salah’.

Blok try-catch juga dapat berbentuk sbb:

```
try
{
    ...
}
catch (exception1 var1)
{
    ...
}
catch (exception2 var2)
{
    ...
}
.
.
.
catch (exceptionn varn)
{
    ...
}
```

Blok try-catch juga dapat diletakkan di sembarang struktur control, misalnya dalam looping.

```
public static void main(String[] args)
{
    int bil1, bil2;

    // selama input untuk bil1 bukan bilangan bulat
    // maka akan terus mengulang input
    while (true)
    {
        try
        {
            bil1 =
Integer.parseInt(JOptionPane.showInputDialog("Masukkan Bilangan 1
(Integer)"));
            break;
        }
        catch (Exception e)
        {
            JOptionPane.showMessageDialog(null, "Bukan bilangan
integer");
        }
    }

    // selama input untuk bil2 bukan bilangan bulat
    // maka akan terus mengulang input

    while (true)
    {
        try
```

```
{  
    bil2 =  
    Integer.parseInt(JOptionPane.showInputDialog("Masukkan Bilangan 2  
(Integer)"));  
    break;  
}  
catch (Exception e)  
{  
    JOptionPane.showMessageDialog(null, "Bukan bilangan  
integer");  
}  
}  
  
JOptionPane.showMessageDialog(null, "Hasil penjumlahannya: " +  
(bil1+bil2));  
}
```

Bekerja Dengan GUI (Graphics User Interface)

Dengan GUI, tampilan interface aplikasi menjadi lebih *user friendly* dan menarik. Java menyediakan dua class untuk GUI antara lain: AWT (Abstract Windowing Toolkit) yang ada dalam package `java.awt` dan SWING yang ada dalam package `javax.swing`.

Dalam modul ini hanya akan dibahas mengenai class SWING saja mengingat SWING merupakan pengembangan dari AWT. Untuk membuat project Java dengan menggunakan interface GUI cukup mudah sekali dengan memanfaatkan NETBEANS. Kita cukup meletakkan komponen-komponen GUI seperti `TextField`, `Label`, `ComboBox`, `RadioButton` dll yang diinginkan ke dalam sebuah container (bisa berupa `Frame` atau `Panel`), kemudian kita bisa berikan event pada komponen tersebut. Event di dalam GUI adalah suatu action yang akan terjadi ketika sebuah komponen diberikan sebuah trigger (pemicu). Sebagai contoh misalkan kita ingin supaya suatu proses terjadi ketika sebuah tombol diklik melalui mouse, maka kita bisa memberikan event `MouseClicked` pada tombol tersebut.

Example 21:

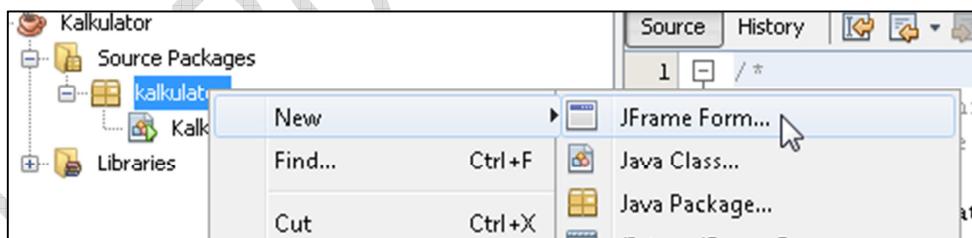
Misalkan akan dibuat sebuah project untuk membuat kalkulator sederhana dengan menggunakan GUI.

Langkah ke-1 (CREATE PROJECT):

Buat project baru dengan nama '**Kalkulator**'

Langkah ke-2 (BUAT FRAME):

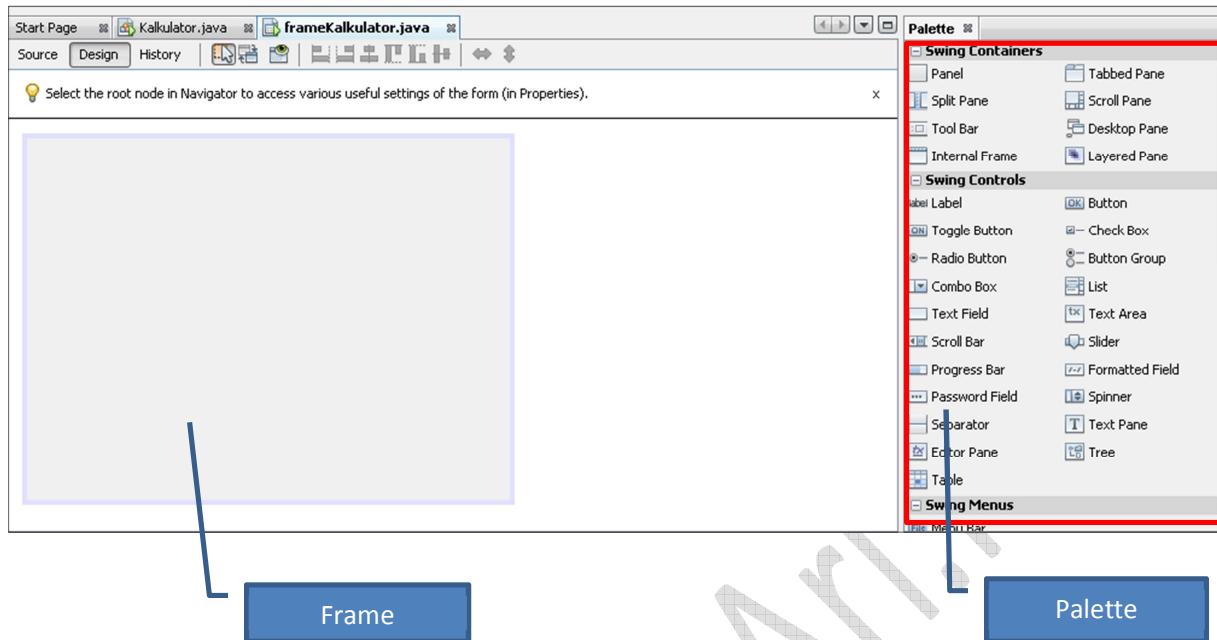
Buat Frame untuk meletakkan komponen-komponen GUI. Dengan cara klik kanan pada package, pilih NEW – JFrame Form



Selanjutnya beri nama class untuk Framenya, misalnya: **frameKalkulator**

Langkah ke-3 (LETAKKAN KOMPONEN):

Selanjutnya proses pengaturan tata letak komponen pada frame yang sudah dibuat. Komponen-komponen yang bisa diletakkan di frame atau container secara umum ada dalam palette



Untuk meletakkan komponen ke dalam frame cukup klik pada komponennya, lalu drop pada framenya. atau bisa juga Anda lakukan klik komponen, lalu drag ke arah frame.

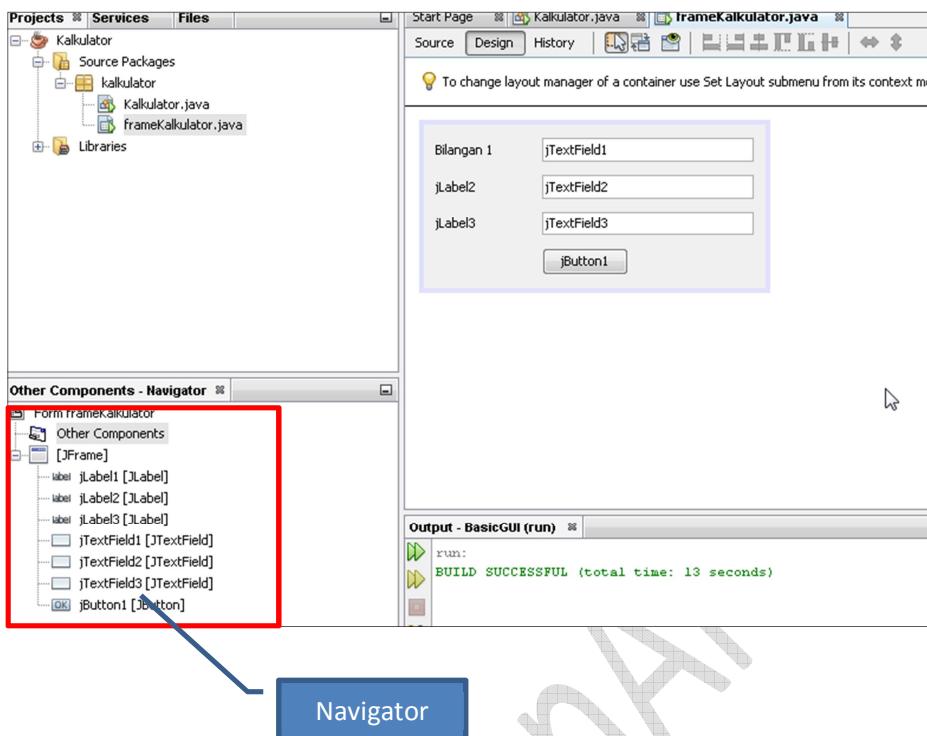
Silakan Anda atur tampilan komponen pada frame seperti berikut



Keterangan:

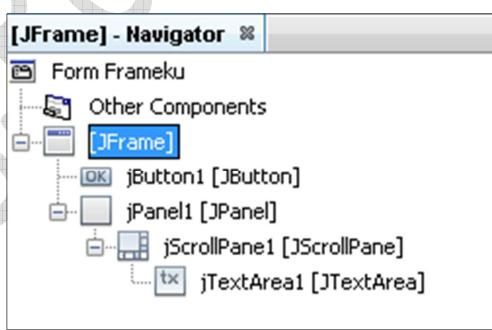
Dalam frame di atas, terdapat beberapa komponen yang diletakkan yaitu 3 buah label, 3 buah textfield dan 1 buah button.

Semua komponen yang kita letakkan dalam frame memiliki nama sendiri-sendiri yang unik, secara default NetBeans akan memberi nama masing-masing komponen yang bisa dilihat dibagian NAVIGATOR



Sebagai contoh, misalkan untuk tombol yang ada dalam Frame di atas, nama komponennya adalah JButton1. Sedangkan kotak isian (TextField) untuk memasukkan bilangan pertama, nama komponennya adalah jTextField1, dst.

Fungsi dari Navigator juga dapat digunakan untuk melihat letak suatu komponen berada di dalam container yang mana. Sebagai contoh perhatikan tampilan Navigator sbb:



Pada struktur hirarki yang tampak pada navigator di atas, JFrame merupakan top level container, di dalamnya ada 1 tombol (Jbutton1) dan sebuah container dengan nama JPanel1. Di dalam JPanel1 terdapat komponen JscrollPane1 yang juga merupakan sebuah container., yang mana di dalamnya terdapat komponen dengan JtextArea1.

Langkah ke-4 (UBAH PROPERTIES KOMPONEN)

Ubah properties masing-masing komponen sesuai yang diinginkan. Properties di sini adalah segala sifat yang melekat dari sebuah obyek komponen, misalnya warna background, jenis font, ukuran font dsb. Untuk mengubah properties dari suatu komponen, caranya dengan klik kanan pada komponen yang akan diubah propertiesnya, lalu pilih 'Properties'. Selanjutnya akan muncul beberapa properties dari komponen yang bisa diubah. Sebagai contoh akan diubah properties TEXT dari komponen jLabel1 menjadi 'Bilangan 1'

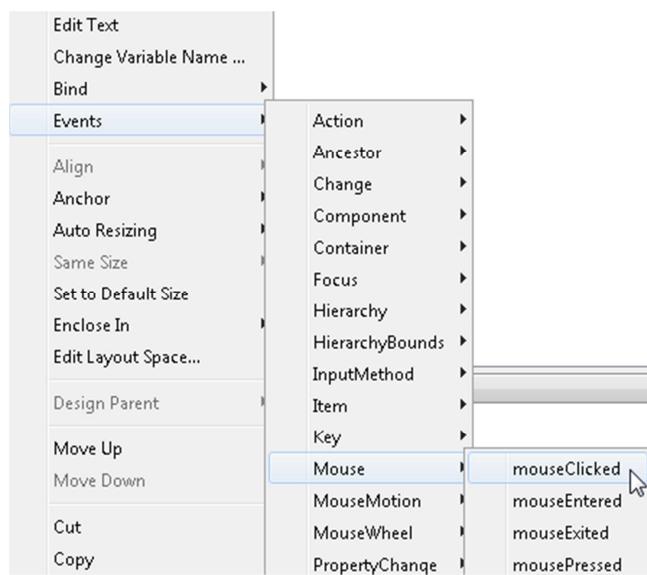
horizontalAlignment	LEADING
icon	
labelFor	<none>
text	Bilangan 1
toolTipText	

Sekarang ubah properties TEXT pada tiap-tiap komponen sehingga diperoleh tampilan sbb:

**Langkah ke-5 (BUAT EVENT):**

Setelah tata letak komponen dan pengaturan propertiesnya sudah dilakukan berikutnya adalah memberikan event pada komponennya. Sebagai contoh, misalnya kita ingin setelah user mengisikan angka dan kotak isian bilangan 1 dan 2 kemudian akan muncul hasil penjumlahannya ketika si user mengklik tombol +. Sesuai skenario ini, kita akan tambahkan sebuah event pada tombol + ini yaitu menampilkan hasil penjumlahan bilangan yang dipicu ketika tombol tersebut diklik.

Untuk membuat event mouse klik pada suatu komponen caranya klik kanan komponennya, lalu pilih EVENTS – MOUSE – MOUSECLICKED



Selanjutnya, ketikkan kode pada tempat yang di sediakan

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}
```

Ketikkan kode berikut ini

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {

    // deklarasi variabel
    float bil1, bil2, hasil;

    // baca nilai float dari komponen jTextField1
    bil1 = Float.parseFloat(jTextField1.getText());
    // baca nilai float dari komponen jTextField2
    bil2 = Float.parseFloat(jTextField2.getText());
    // jumlahkan
    hasil = bil1 + bil2;

    // tampilkan hasil ke komponen jTextField3
    jTextField3.setText(String.valueOf(hasil));
}
```

Langkah ke-6 (TAMPILKAN FRAME VIA MAIN CLASS)

Setelah event diberikan pada komponen, terakhir kita lakukan instansiasi untuk frame yang tadi telah kita buat dalam main class nya.

Sisipkan kode berikut ini pada class 'Kalkulator' dalam file 'Kalkulator.java'

```
public class Kalkulator {  
  
    public static void main(String[] args) {  
  
        // instansiasi dari class frameKalkulator  
        frameKalkulator kalkulator1 = new frameKalkulator();  
        // tampilkan frame  
        kalkulator1.setVisible(true);  
  
    }  
}
```

Untuk melihat hasilnya, silakan jalankan programnya.



Sebuah method dari event yang telah kita berikan pada suatu komponen bisa dihapus apabila tidak lagi digunakan dengan cara, klik kanan komponen yang akan dihapus event nya, pilih properties kemudian pilih Events



Kemudian hapus nama event yang telah dibuat

itemStateChanged	<none>
keyPressed	<none>
keyReleased	<none>
keyTyped	<none>
mouseClicked	Button1MouseClicked
mouseDragged	<none>
mouseEntered	<none>
mouseExited	<none>

Setelah dihapus menjadi

inputMethodTextChanged	<none>
itemStateChanged	<none>
keyPressed	<none>
keyReleased	<none>
keyTyped	<none>
mouseClicked	
mouseDragged	<none>
mouseEntered	<none>
mouseExited	<none>

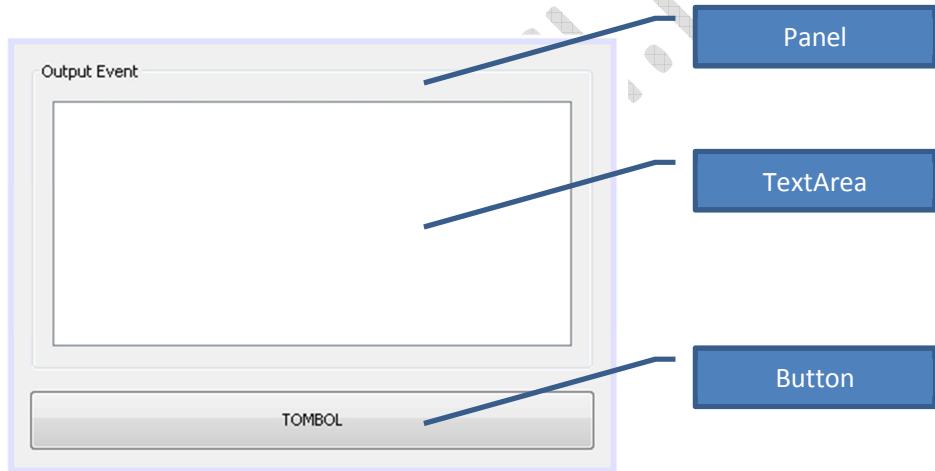
Dalam contoh sebelumnya, kita sudah mengenal apa itu event **MouseClicked**, yaitu event yang terjadi ketika suatu klik mouse dilakukan pada komponen tersebut. Selain **MouseClicked**, masih ada beberapa event lain yang terkait dengan mouse yaitu

- **MouseEntered** : ketika posisi kursor mouse mengenai komponen
- **MouseExited** : ketika posisi kursor mouse keluar dari komponen
- **MousePressed** : ketika tombol mouse ditekan pada komponen
- **MouseReleased** : ketika tombol mouse dilepaskan (setelah ditekan) pada komponen

Untuk melihat efek beberapa event terkait dengan mouse di atas, perhatikan contoh project berikut ini:

Example 22:

Buatlah project baru, kemudian buat Frame dalam class '**Frameku**' , sisipkan beberapa komponen serta aturlah tata letaknya sbb:



Selanjutnya ubah propertiesnya sehingga tampilannya seperti di atas.

Kemudian tambahkan event **MouseEntered** pada tombolnya, dan tulis kodennya berikut ini:

```
private void jButton1MouseEntered(java.awt.event.MouseEvent evt) {  
    // menampilkan pesan ke textarea  
    jTextArea1.setText("ini efek dari event MouseEntered");  
}
```

Kemudian tambahkan event **MouseExited** pada tombolnya, dengan kode berikut ini:

```
private void jButton1MouseExited(java.awt.event.MouseEvent evt) {  
    jTextArea1.setText("ini efek dari event MouseExited\\");  
}
```

Tambahkan event **MousePressed**, pada tombol, dan tulis kode ini:

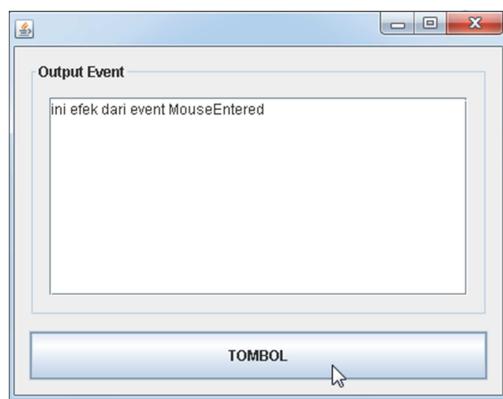
```
private void jButton1MousePressed(java.awt.event.MouseEvent evt) {
    jTextArea1.setText("ini efek dari event MousePressed");
}
```

Tambahkan event **MouseReleased**, pada tombol, dan tulis kode ini:

```
private void jButton1MouseReleased(java.awt.event.MouseEvent evt) {
    jTextArea1.setText("ini efek dari event MouseReleased");
}
```

Terakhir, jangan lupa lakukan instansiasi pada main class supaya frame bisa muncul ketika project di run.

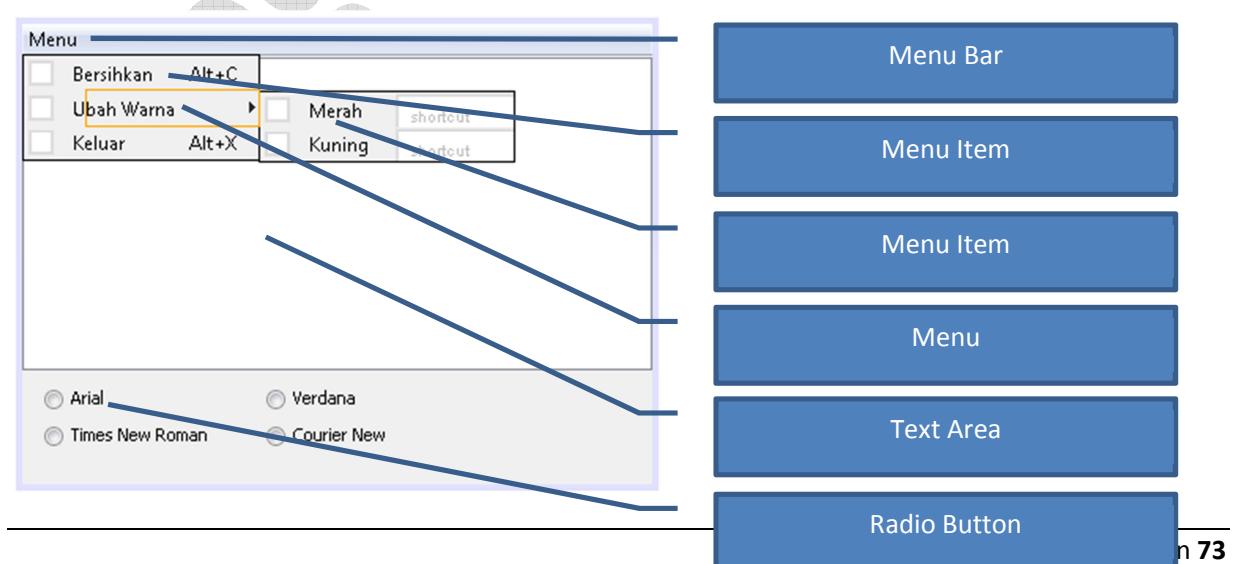
Selanjutnya jalankan program, dan perhatikan beberapa efek dari event tersebut.



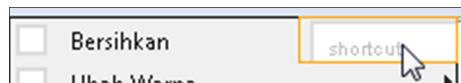
Example 23:

Dalam contoh ini, kita akan membuat project yang di dalamnya terdapat beberapa komponen lainnya yaitu TextArea, Menu, dan RadioButton.

Desainlah sebuah frame dengan tampilan sebagai berikut



Untuk membuat tombol short cut menu, caranya cukup mendouble click 'shortcut' pada sub menu yang ingin ditambahkan short cutnya.



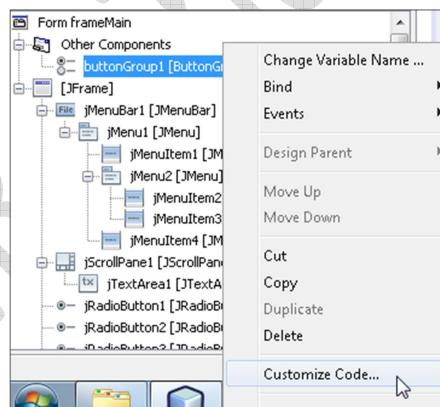
Kemudian tentukan short cut nya dengan memilih tombol yang dikehendaki



Selanjutnya, supaya di antara beberapa komponen radio button tersebut hanya bisa dipilih salah satu maka radio button harus dimasukkan ke dalam Button Group. Oleh karena itu, tambahkan komponen Button Group ke dalam frame.



Kemudian untuk memasukkan radio button-radio button ke dalam button group tersebut caranya, klik kanan pada komponen Button Group melalui Navigator, dan pilih Customize Code



Selanjutnya tambahkan kode berikut ini

```
buttonGroup1 = new javax.swing.ButtonGroup();
buttonGroup1.add(jRadioButton1);
buttonGroup1.add(jRadioButton2);
buttonGroup1.add(jRadioButton3);
buttonGroup1.add(jRadioButton4);
```

Berikutnya tambahkan event-event pada komponen berikut ini:

Komponen : jMenuItem1 (menu 'Bersihkan')
Event : actionPerformed

```
private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt)
{
    // membersihkan text area
    jTextAreal.setText("");
}
```

Komponen : jMenuItem2 (menu 'Ubah – Warna Merah')
Event : actionPerformed

```
private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah warna text area menjadi merah
    jTextAreal.setBackground(Color.red);
}
```

Komponen : jMenuItem3 (menu 'Ubah – Warna Kuning')
Event : actionPerformed

```
private void jMenuItem3ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah warna text area menjadi kuning
    jTextAreal.setBackground(Color.yellow);
}
```

Komponen : jMenuItem4 (menu 'Keluar')
Event : actionPerformed

```
private void jMenuItem4ActionPerformed(java.awt.event.ActionEvent evt)
{
    // menampilkan konfirmasi
    int hasil = JOptionPane.showConfirmDialog(this, "Apakah Anda
yakin mau keluar?", "Konfirmasi", JOptionPane.OK_CANCEL_OPTION);

    // jika yang dipilih OK, maka program EXIT
    if (hasil == JOptionPane.OK_OPTION)
    {
        System.exit(0);
    }
}
```

Komponen : jRadioButton1 (Radio button - Arial)
Event : actionPerformed

```
private void jRadioButton1ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah font Text Area menjadi Arial, Plain, size 12 pt
    Font huruf = new Font("Arial", Font.PLAIN, 12);
    jTextAreal.setFont(huruf);
}
```

Komponen : jRadioButton2 (Radio button –Times New Roman)
Event : actionPerformed

```
private void jRadioButton2ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah font Text Area menjadi Times New Roman, Plain, size 12 pt
    Font huruf = new Font("Times New Roman", Font.PLAIN, 12);
    jTextAreal.setFont(huruf);
}
```

Komponen : jRadioButton3 (Radio button - Verdana)
Event : actionPerformed

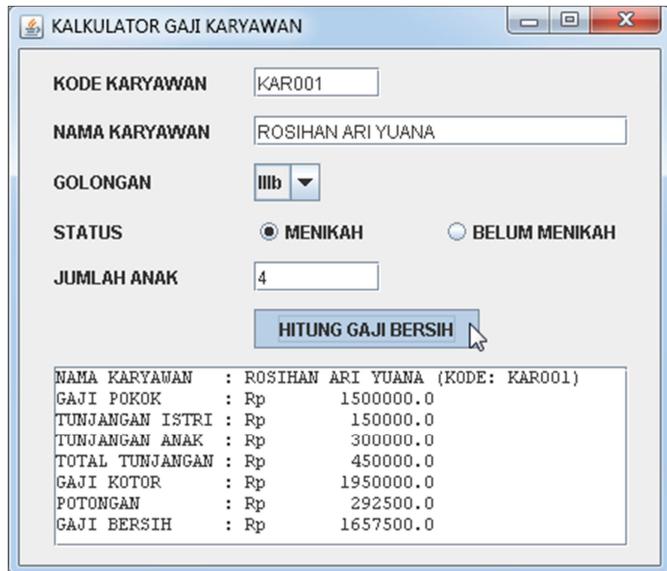
```
private void jRadioButton3ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah font Text Area menjadi Verdana, Plain, size 12 pt
    Font huruf = new Font("Verdana", Font.PLAIN, 12);
    jTextAreal.setFont(huruf);
}
```

Komponen : jRadioButton4 (Radio button – Courier New)
Event : actionPerformed

```
private void jRadioButton4ActionPerformed(java.awt.event.ActionEvent evt)
{
    // mengubah font Text Area menjadi Courier New, Plain, size 12 pt
    Font huruf = new Font("Courier New", Font.PLAIN, 12);
    jTextAreal.setFont(huruf);
}
```

Example 24:

Pada contoh ini, project yang kita buat adalah membuat aplikasi perhitungan gaji karyawan dengan tampilan sbb:



Keterangan:

Komponen yang diberikan event pada project ini adalah Radio Button (Menikah dan Belum Menikah), yaitu event MouseClicked. Jika yang dipilih adalah Menikah, maka TextField untuk mengisi Jumlah Anak aktif sehingga user bisa mengisikan jumlah anaknya. Sedangkan jika yang dipilih Belum Menikah, maka TextField Jumlah Anak akan diisi 0 dan dinonaktifkan sehingga user tidak bisa mengisi jumlah anaknya.

Event juga diberikan pada button Hitung Gaji Bersih, yaitu MouseClicked yang nantinya akan melakukan proses perhitungan gaji.

Untuk lebih detilnya, silakan mengunduh file project nya di

<http://rosihanari.net/tutorial/project-gaji.zip>

dan silakan dipelajari sendiri :-)

Latihan

1. Tinjau kembali Example 21, tambahkan button untuk melakukan operasi pengurangan, perkalian dan pembagian. Kemudian buat event pada button-button tersebut. Usahakan tidak terjadi kemungkinan Error yang disebabkan oleh Exception.
2. Pada Example 24, modifikasi projectnya supaya user tidak bisa memasukkan jumlah anak berupa bilangan real atau negatif. Jika jumlah anak berupa bilangan real, maka akan muncul Exception.