

**20**  
JAM

BELAJAR CODING



# Belajar **JAVA** dari NOL

**Isa Hamdan, S.Kom**  
*Founder KomputerKit.com*

**EDISI KEDUA**



**KomputerKit.com**  
Tempat Terbaik Belajar Coding !

## **Dari Penulis**

Belajar *coding* atau program sekarang ini tentu berbeda dari beberapa tahun sebelumnya. Dengan makin banyaknya sumber daya yang bisa digunakan untuk belajar seperti youtube, sosial media, maupun situs berbagi *coding* seperti github dan seterusnya.

Yang paling dibutuhkan oleh calon programmer saat ini adalah kemampuan membaca *coding* yang dibagikan dari situs berbagi *coding* atau dari forum – forum di sosial media, sehingga programmer bisa langsung berkarya menggunakan *coding* yang sudah di gabung dan dimodifikasi.

Buku ini dibuat untuk belajar memahami *coding* dari dasar. Pada tahap selanjutnya jika sudah difahami dengan baik anda bisa langsung belajar dari *coding* orang lain baik melalui youtube, google, forum di sosial media atau situs berbagi coding seperti github. Dengan memahami *coding* orang lain maka anda dapat mempercepat belajar anda.

Buku ini adalah buku PRAKTEK dan BUKAN BUKU BACAAN. Lakukan praktek di depan komputer atau laptop anda ketika membaca buku ini. Perhatikan yang tampil di monitor anda dan fahami maksudnya. Kunci untuk menjadi programmer adalah LATIHAN dan LATIHAN.

Terimakasih sudah membaca dan menggunakan buku ini sebagai bahan untuk belajar. Jika ada masukan kritik atau saran demi perbaikan isi silahkan kirim ke email :  
[Komputerkit.dev@gmail.com](mailto:Komputerkit.dev@gmail.com)

## **Edisi Buku**

Edisi Pertama - November 2017

Edisi Kedua - Desember 2017

Untuk kedua putriku yang baik  
Ica & Hana

## **Daftar Isi**

1.	Konsep Program Komputer .....	5
2.	Kenapa harus Java? .....	5
3.	Program pertama .....	11
4.	Perbedaan Aplikasi Executable dan Aplikasi Bytecode .....	17
5.	Variabel dan Tipe data.....	17
6.	Input data menggunakan Scanner .....	20
7.	Operator.....	21
8.	Pengujian .....	24
9.	Pengujian Ganda .....	26
10.	Pengujian Bersarang (IF di dalam IF).....	27
11.	Pemilihan (Selection) .....	28
12.	Pengulangan ( <i>Looping</i> ) .....	29
13.	Pengulangan Bersarang ( <i>Nested Loop</i> ) .....	30
14.	Array (Larik / Rantang).....	32
15.	Method (Blok Program) .....	35
16.	Properti, Method, Class .....	40
17.	Fungsi Static Pada Method .....	44
18.	Konstruktor (method yang langsung dijalankan ketika objek dibuat).....	45
19.	Inheritance (Pewarisan) .....	47
20.	Abstract Class.....	49
21.	Encapsulasi (Public, Private, protected).....	51
22.	Polimorfisme (Nama Method yang sama dengan tugas yang berbeda ).....	55
23.	Overriding .....	57
24.	Overloading.....	59
25.	Interface.....	60
26.	Enumerasi (tipe data bentukan) .....	62
27.	Java SWING (Form java) .....	63
28.	Collection .....	75
29.	Keyword this, final, super .....	77
30.	Try catch.....	83

31.	Try Catch Finally .....	84
32.	Thread .....	85
	Self Assessment (Penilaian Diri Sendiri).....	89
	DAFTAR PUSTAKA .....	91

## **1. Konsep Program Komputer**

Komputer adalah sebuah mesin yang bekerja berdasarkan instruksi atau perintah yang dibuat oleh manusia. Instruksi yang dibuat disesuaikan dengan kebutuhan pengguna atau user dari komputer tersebut. Daftar instruksi yang ditulis dan jalankan oleh komputer disebut dengan *program*. Orang yang membuat program disebut dengan *programmer*.

Secara umum program berisi 3 proses, yaitu:

### **1. Input**

Input adalah perintah atau nilai yang dimasukan oleh pengguna saat menggunakan komputer.

Contoh :

saat anda mengetik menggunakan keyboard, saat anda mengklik mouse, saat anda men tap handphone anda, saat anda memasukan nilai angka pada kalkulator, dan sebagainya.

### **2. Proses**

Proses adalah pengolahan input untuk mendapatkan hasil yang di inginkan.

Contoh:

Saat anda memasukan 2 nilai pada kalkulator maka kalkulator akan menghitung terlebih dahulu penjumlahan dari 2 nilai tersebut sebelum ditampilkan. Proses perhitungan inilah yang disebut proses dalam program.

### **3. Output**

Output adalah hasil akhir yang diharapkan dari jalannya program.

Contoh:

Saat anda ingin melihat hasil akhir penjumlahan di kalkulator maka tampilan hasil akhir inilah yang disebut dengan output. Tampilan dilayar yang anda lihat ketika menjalankan komputer, atau laptop atau handphone anda adalah output dari program yang sudah ditanam di komputer anda.

## **2. Kenapa harus Java?**

Java adalah bahasa pemrograman yang banyak digunakan saat ini. Banyak aplikasi atau program yang dibuat menggunakan java. Kemampuan memahami java mutlak diperlukan bagi anda yang ingin belajar pemrograman.

### **Java Development KIT dan Java Virtual Machine**

Java adalah sebuah bahasa pemrograman yang dibuat oleh sun microsystem yang sekarang dibeli oleh oracle. Untuk bisa membuat aplikasi menggunakan java maka komputer harus di instal aplikasi JDK (*Java Development Kit*). Aplikasi java berjalan di atas java virtual mesin, yang dihasilkan oleh JRE (*Java Runtime Environment*). JRE wajib di instal untuk bisa menjalankan program yang dihasilkan oleh java. JRE bisa di instal di hampir semua sistem operasi (*OS-Operating System*). Karena JRE bisa di instal di hampir semua OS maka aplikasi yang dihasilkan oleh java bisa berjalan diatas semua OS.

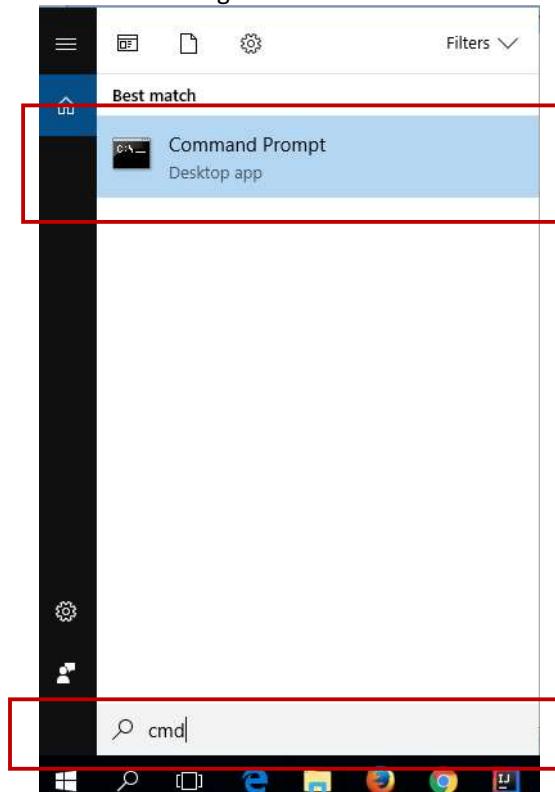
### Instalasi Java Development KIT (JDK)

Aplikasi JDK bisa di download dari situs:

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

Cara pengujian JDK apakah sudah ter instal dengan baik adalah:

- masuk ke cmd (**command Prompt**)
- ketik sesuai gambar dibawah dan tekan enter



Ketik sesuai lokasi dari JDK di komputer anda.

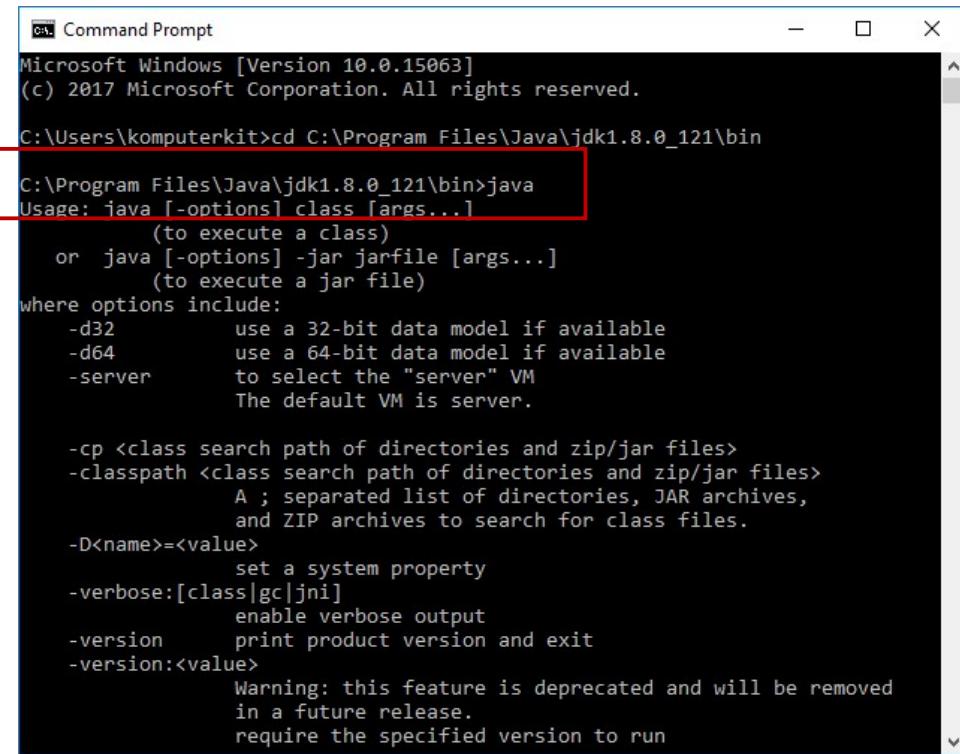
A screenshot of a Microsoft Command Prompt window. The title bar says "Command Prompt". The window shows the following text:

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\komputerkit>cd C:\Program Files\Java\jdk1.8.0_121\bin
```

The last line of text, which is the command being typed, is highlighted with a red rectangular box.

- Kemudian **ketik java**. Jika tampilan sudah sesuai gambar dibawah maka instalasi java berhasil



```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\komputerkit>cd C:\Program Files\Java\jdk1.8.0_121\bin

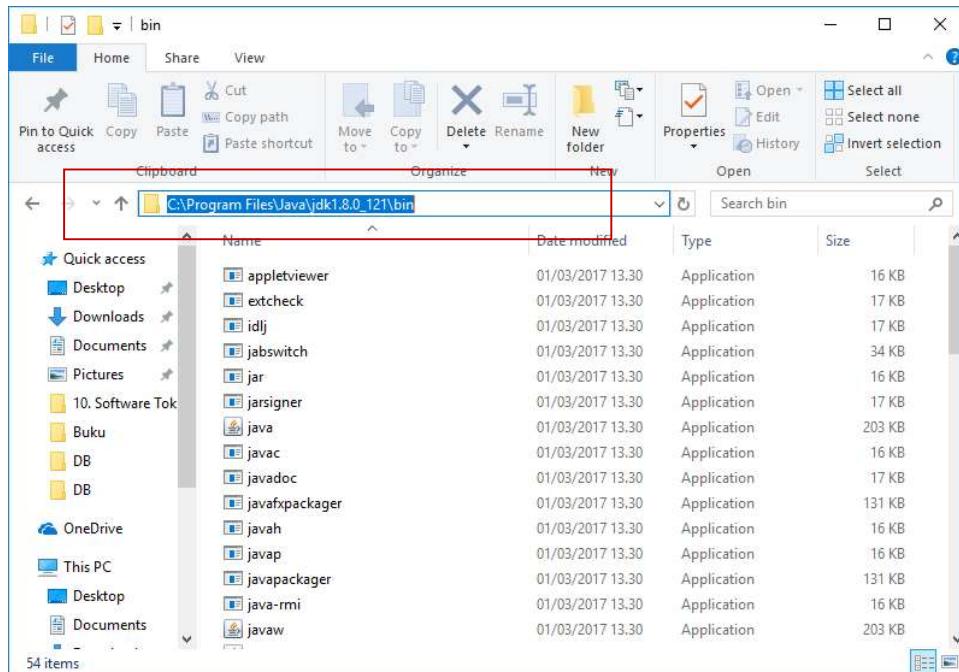
C:\Program Files\Java\jdk1.8.0_121\bin>java
Usage: java [-options] class [args...]
           (to execute a class)
       or  java [-options] -jar jarfile [args...]
           (to execute a jar file)
where options include:
  -d32      use a 32-bit data model if available
  -d64      use a 64-bit data model if available
  -server    to select the "server" VM
             The default VM is server.

  -cp <class search path of directories and zip/jar files>
  -classpath <class search path of directories and zip/jar files>
             A ; separated list of directories, JAR archives,
             and ZIP archives to search for class files.

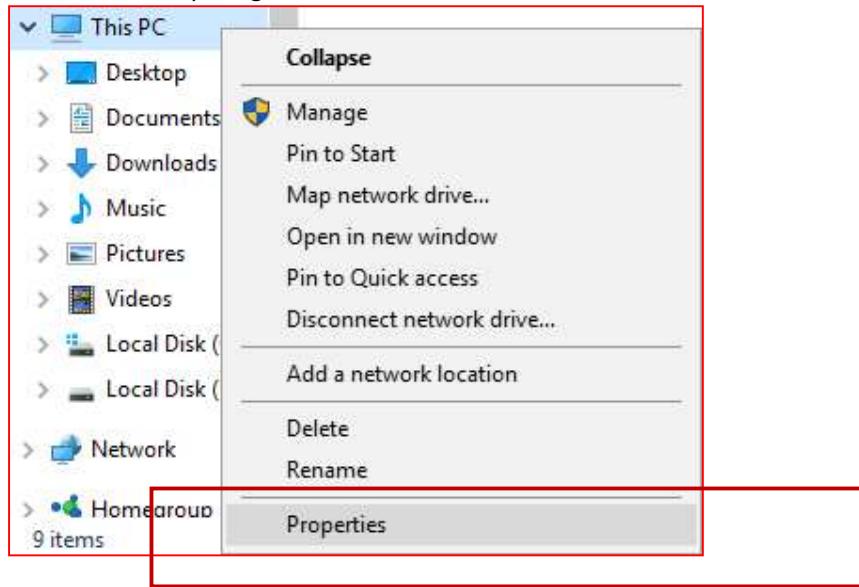
  -D<name>=<value>
             set a system property
  -verbose:[class|gc|jni]
             enable verbose output
  -version     print product version and exit
  -version:<value>
             Warning: this feature is deprecated and will be removed
             in a future release.
             require the specified version to run
```

### Setting agar java bisa dijalankan dari semua lokasi.

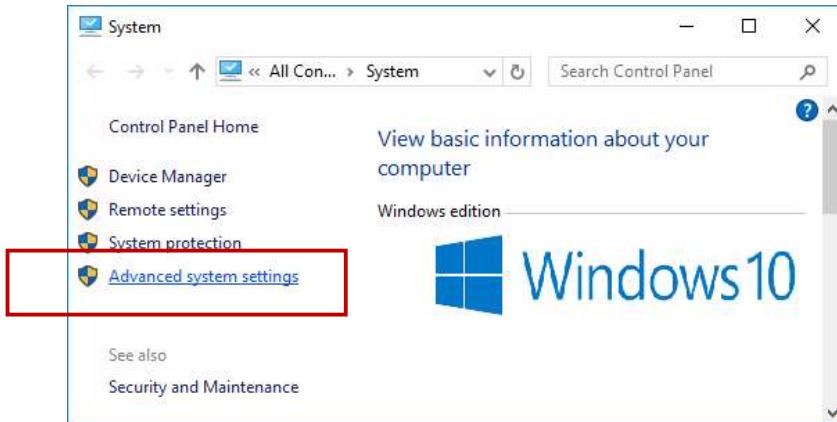
- Masuk ke windows explorer dan copy path seperti gambar dibawah



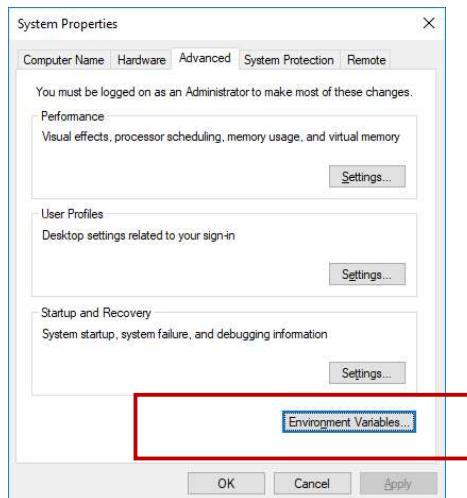
- Klik kanan seperti gambar dibawah



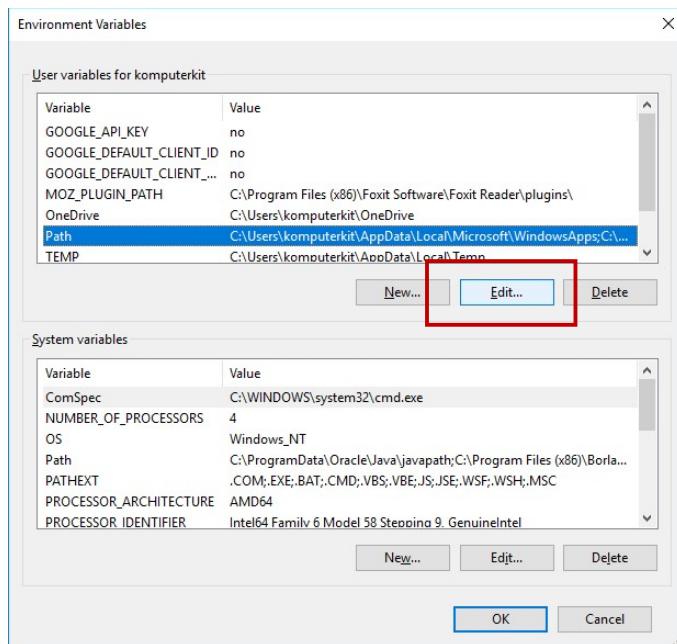
- Klik advance system settings



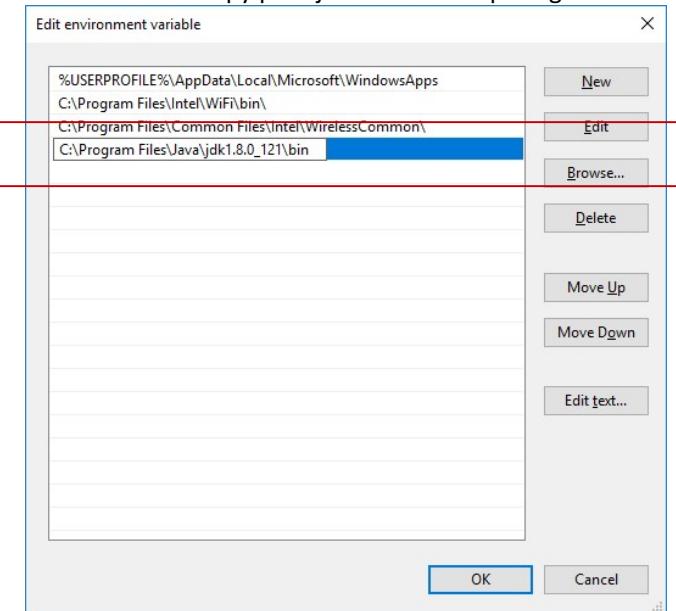
- Klik Environment variables



- Klik edit



- Paste hasil copy path java di lokasi seperti gambar dibawah kemudian klik OK



- buka cmd (**command Prompt**) dan ketik **javac** untuk mencoba **compiler** java.  
**Compiler** adalah program yang akan merubah program yang kita tulis menjadi bahasa mesin sehingga program yang kita buat bisa dijalankan oleh komputer.

```

D:\>javac
Usage: javac <options> <source files>
where possible options include:
  -g                           Generate all debugging info
  -g:none                      Generate no debugging info
  -g:{lines,vars,source}          Generate only some debugging info
  -nowarn                       Generate no warnings
  -verbose                      Output messages about what the compiler is doing
  -deprecation                  Output source locations where deprecated APIs are used
  -classpath <path>              Specify where to find user class files and annotation pro-
                                cessors
  -cp <path>                    Specify where to find user class files and annotation pro-
                                cessors
  -sourcepath <path>             Specify where to find input source files
  -bootclasspath <path>          Override location of bootstrap class files
  -extdirs <dirs>                Override location of installed extensions
  
```

### 3. Program pertama

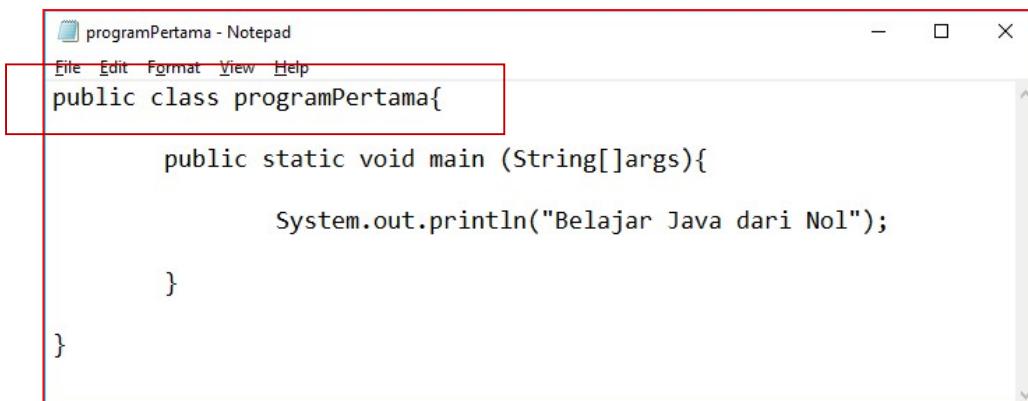
Buka aplikasi notepad atau notepad++ ketik kode program seperti gambar dibawah.

Simpan kode tersebut dengan nama file **programPertama.java**

Nama file yang disimpan harus sama dengan nama class

**Perhatikan huruf besar dan kecil di java itu berbeda.**

Contoh : *String* dan **string** itu tidak sama



```
programPertama - Notepad
File Edit Format View Help
public class programPertama{

    public static void main (String[]args){

        System.out.println("Belajar Java dari Nol");

    }

}
```

A screenshot of a Windows Notepad window titled "programPertama - Notepad". The window contains the following Java code:

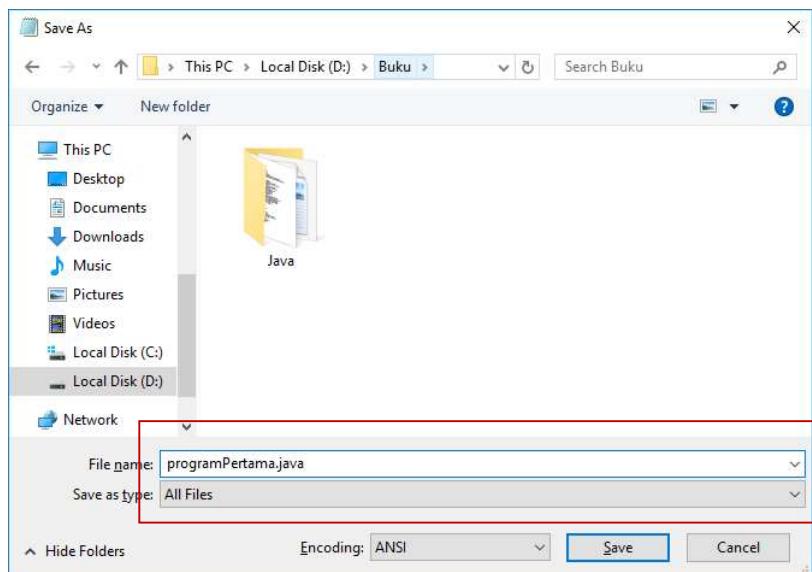
```
public class programPertama{

    public static void main (String[]args){

        System.out.println("Belajar Java dari Nol");

    }

}
```

The entire code block is highlighted with a red rectangular selection.

Setelah disimpan buka cmd (**command Prompt**). Arahkan ke folder dimana anda menyimpan file yang sudah dibuat. Kemudian lakukan compile pada program dengan perintah **programPertama.java**

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The window title bar includes standard icons for minimize, maximize, and close. The main area displays the following command-line session:

```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\komputerkit>d:
D:>cd buku
D:\Buku>dir
 Volume in drive D has no label.
 Volume Serial Number is CA1E-78B5

 Directory of D:\Buku

10/11/2017 15.42 <DIR> .
10/11/2017 15.42 <DIR> ..
10/11/2017 15.27 <DIR> Java
10/11/2017 15.29 135 programPertama.java
               1 File(s)    135 bytes
               3 Dir(s)  4.785.242.112 bytes free

D:\Buku>javac programPertama.java
D:\Buku>
```

A red rectangular box highlights the directory listing and the compilation command. Another red rectangular box highlights the output of the compilation command, which is the generated bytecode file.

Jalankan program Pertama yang sudah di compile dengan cara mengetikan  
**Java programPertama**

The screenshot shows a Windows Command Prompt window titled "Command Prompt". The main area displays the following command-line session:

```
D:\Buku>dir
 Volume in drive D has no label.
 Volume Serial Number is CA1E-78B5

 Directory of D:\Buku

10/11/2017 15.42 <DIR> .
10/11/2017 15.42 <DIR> ..
10/11/2017 15.27 <DIR> Java
10/11/2017 15.29 135 programPertama.java
               1 File(s)    135 bytes
               3 Dir(s)  4.785.242.112 bytes free

D:\Buku>javac programPertama.java
D:\Buku>java programPertama
Belajar Java dari Nol

D:\Buku>
```

A red rectangular box highlights the execution command and its output, which is the printed message "Belajar Java dari Nol".

### **Instalasi IntelliJ IDEA**

Untuk membuat aplikasi java diperlukan *code editor* (tempat programmer menuliskan program). Pilihan menggunakan IntelliJ IDEA karena setelah belajar java akan dilanjutkan belajar membuat aplikasi android menggunakan android studio. Tampilan dari android studio sama dengan tampilan dari IntelliJ IDEA, karena kedua aplikasi tersebut dihasilkan oleh developer yang sama.

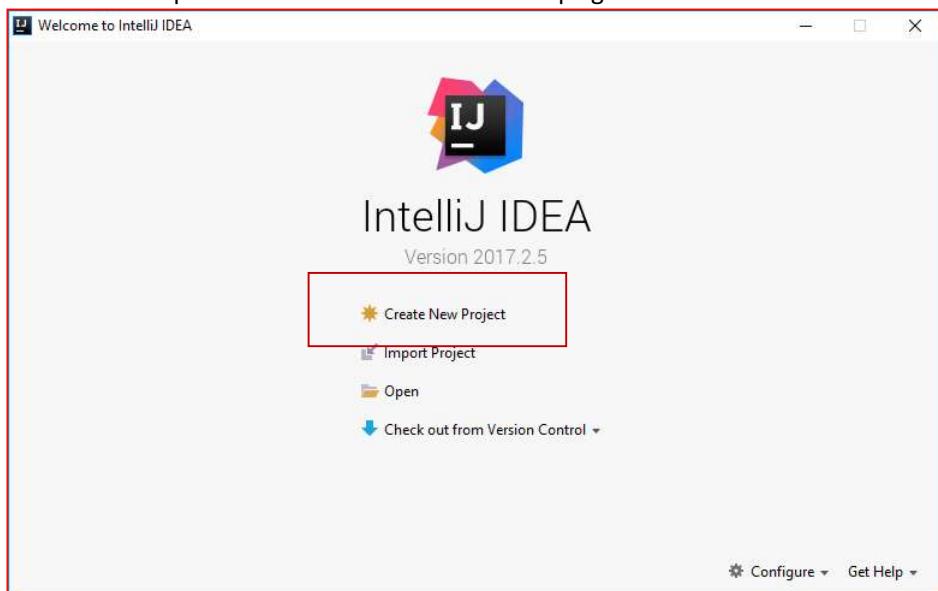
Aplikasi IntelliJ IDEA bisa di download dari situs:

<https://www.jetbrains.com/idea/download/#section=windows>

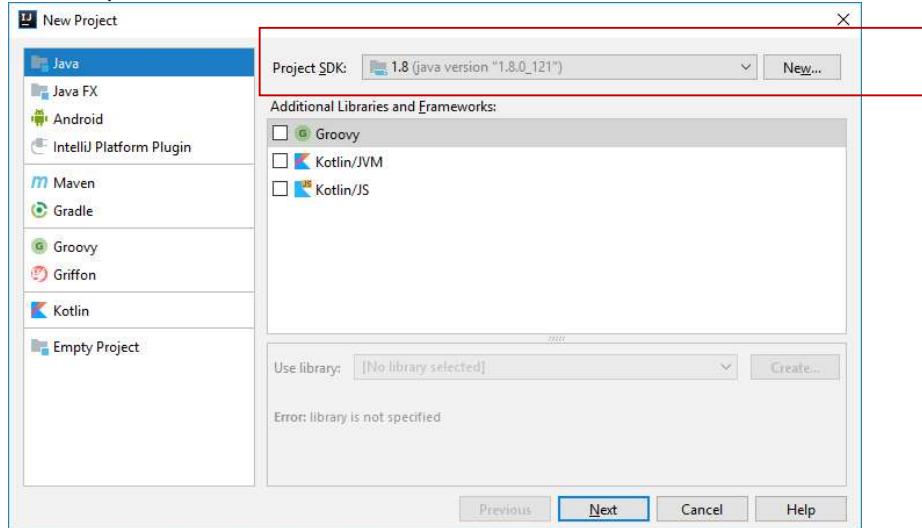


### **Cara menggunakan IntelliJ IDEA**

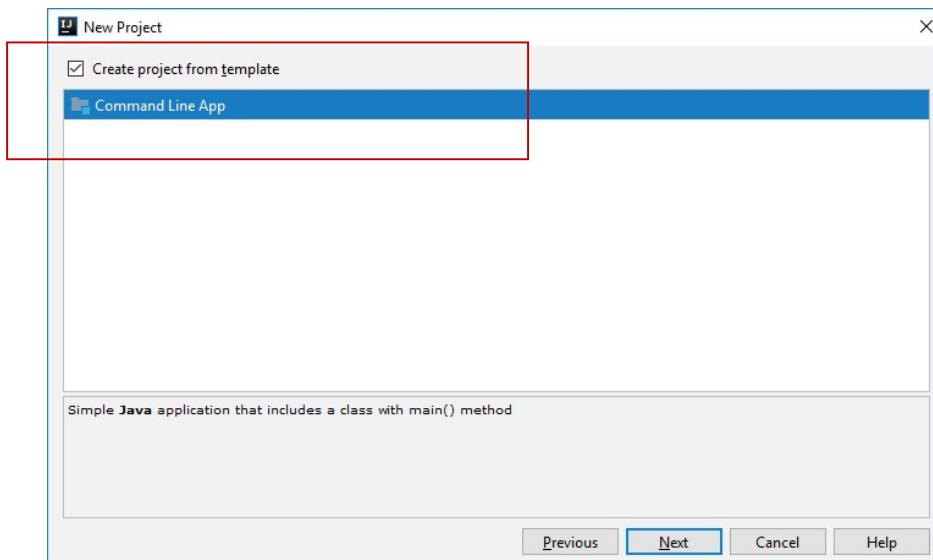
- Jalankan aplikasi intelliJ IDEA maka akan tampil gambar dibawah



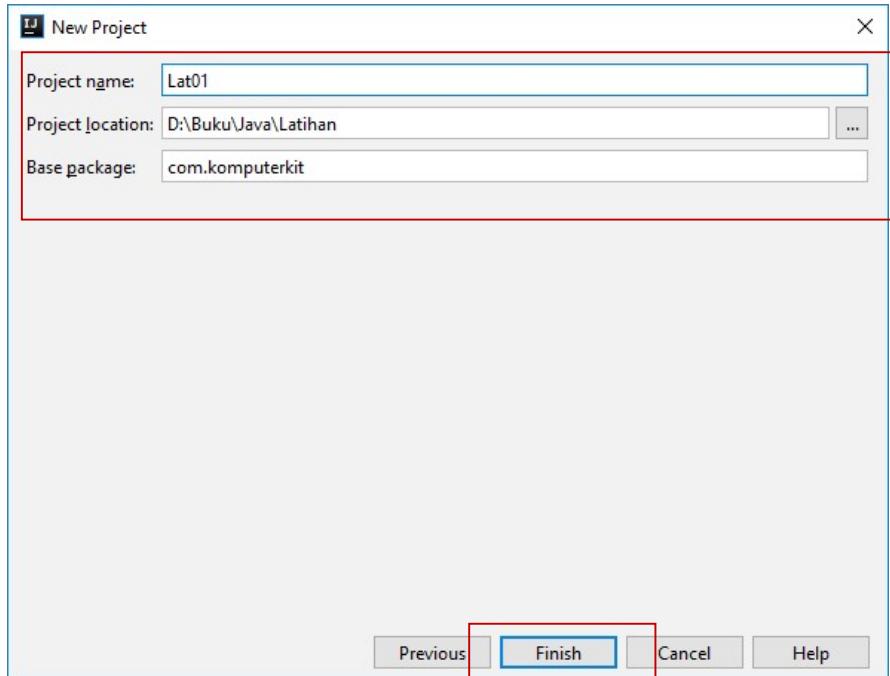
- Jika sudah tampil maka akan muncul tampilan seperti dibawah, pastikan project SDK mengarah ke JDK. Jika belum, arahkan project SDK mengarah ke JDK di komputer anda:



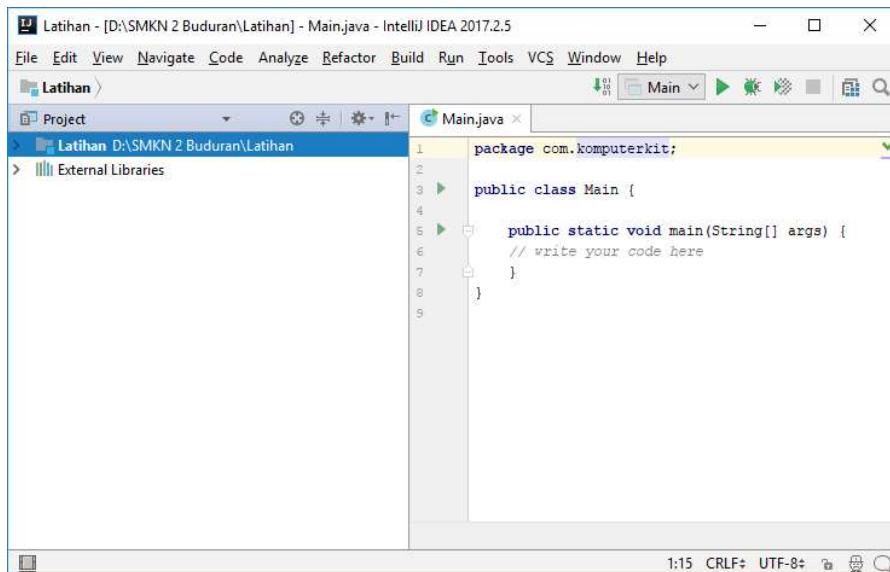
- Kemudian klik next maka akan tampil seperti gambar dibawah, kemudian centang 'Create project from template' dan klik Next



- Setelah klik Next maka akan muncul tampilan seperti gambar dibawah, arahkan project location ke folder yang sudah anda siapkan. Kemudian klik finish



- Akan muncul tampilan seperti pada gambar



- Ketik program seperti pada gambar

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        System.out.println("Saya belajar java dari NOL");
    }
}
```

Jalankan program yang sudah dibuat dengan meng klk PLAY



Output atau hasil dari program akan tampil seperti pada gambar dibawah ini.

The terminal window shows the command "java Main" being run, followed by the output "Saya belajar java dari NOL". It also indicates that the process finished with exit code 0.

Penjelasan program :

Baris program	Penjelasan
<code>package com.komputerkit;</code>	Package adalah nama projek yang sedang dikerjakan. Semua baris coding dengan nama package yang sama akan dianggap sebagai satu kesatuan atau bagian dari projek yang sama
<code>public class Main {</code> <code>    public static void main(String[] args) {</code> <code>        System.out.println("Saya belajar java dari NOL");</code> <code>    }</code>	Penulisan coding java harus ditulis di dalam <b>class</b> . Nama <b>class</b> yang ditandai dengan <b>{</b> kurung kurawal buka sebagai permulaan dari <b>class</b> dan kurung kurawal tutup <b>}</b> sebagai akhir dari <b>class</b>

## 4. Perbedaan Aplikasi Executable dan Aplikasi Bytecode

### Aplikasi Executable

Aplikasi yang langsung bisa dijalankan di atas OS tanpa membutuhkan aplikasi yang lain.

Format aplikasi ini adalah

*namaaplikasi.exe*

### Aplikasi Bytecode

Aplikasi yang membutuhkan aplikasi lain untuk bisa berjalan diatas OS. Aplikasi yang dibuat menggunakan java baru bisa berjalan jika di OS tersebut sudah terinstal JRE.

Kelebihan java adalah JRE bisa di instal di hampir semua OS maka aplikasi yang dihasilkan oleh java juga bisa berjalan di hampir semua OS. Format aplikasi ini adalah *namaclass.java*

## 5. Variabel dan Tipe data

### Konsep

Joni adalah seorang siswa yang akan makan siang. Ibu nya dirumah sudah menyediakan makanan di dapur. Joni pergi ke dapur untuk mengambil piring sebagai tempat nasi. Setelah piring diambil joni menuangkan nasi ke dalam piring yang diambil. Setelah mengambil nasi dan lauk kemudian joni mengambil gelas. Setelah gelas diambil joni menuangkan air minum ke dalam gelasnya.

Dari cerita diatas dapat diambil kesimpulan

Pikiran di dalam program disebut <b>Tipe data</b>	Wadah / Tempat Di dalam program disebut <b>Variabel atau Properti</b>	Isi Di dalam program disebut <b>Data</b>
<i>Mau ambil nasi</i>	piring	nasi
<i>Mau ambil minum</i>	Gelas	Air

Sebelum mengambil wadah atau tempat maka tentukan dulu yang akan diambil baru kemudian ambil tempat atau wadahnya.

Didalam pembuatan program juga sama. Tentukan dulu data yang akan dimasukan siapkan tempatnya baru masukan isinya.

- **Tipe Data Tulisan**

String, Char

Tipe Data	Contoh
String	String nama;
char	char a=66;

- **Tipe data angka**

Integer (utuh)

Tipe Data	Ukuran (bit)	Range
Byte	8	-128 s.d. 127
Short	16	-32768 s.d. 32767
Int	32	-2147483648 s.d. 2147483647
Long	64	-9223372036854775808 s.d. 9223372036854775807

Pecahan

Tipe	Ukuran		Range	Presisi (jumlah digit)
	bytes	bit		
float	4	32	+/- 3.4 x 1038	6-7
double	8	64	+/- 1.8 x 10308	15

- Tipe data boolean

Tipe Data	Contoh
boolean	<code>boolean kondisi = true;</code> <code>boolean kondisi = false;</code>

- Tipe data referensi

Tipe Data	Penjelasan
array	Data dalam larikan atau rantangan
class	Kumpulan dari variabel dan function (method)
interface	Kumpulan method yang dideklarasikan

- Aturan penamaan variabel, method, dan class

Aturan	Contoh
Huruf / abjad, Karakter mata uang, Underscore (_)	<code>String nama;</code> <code>String \$nama;</code> <code>String _nama;</code>
<b>TIDAK BOLEH</b> mengandung @, spasi atau diawali dengan angka. Selain itu, tidak boleh menggunakan keyword atau katakata yang memiliki arti atau digunakan dalam pemrograman Java.	<code>String @nama;</code> <code>String na ma</code> <code>String 7nama;</code> <code>String void;</code>

Buat program seperti gambar dibawah dan perhatikan hasilnya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        String nama="KomputerKit";
        int umur = 2;
        Double uang=30.51;
        char huruf=65;
        float phi;
        phi = (float) 3.14;
        boolean kondisi=true;

        System.out.println(nama);
        System.out.println(uang);
        System.out.println(huruf);
        System.out.println(phi);
        System.out.println(kondisi);

    }
}
```

## 6. Input data menggunakan Scanner

Scanner digunakan untuk memasukan data yang kita input dari keyboard.  
Tambahkan import dan buat deklarasi scanner seperti gambar dibawah ini

```
package com.komputerkit;
import java.util.*;
public class Main {
    public static void main(String[] args) {
        String nama;
        int umur;
        double uang;
        char huruf;
        float phi;
        boolean kondisi;
        Scanner input = new Scanner(System.in);

        System.out.print("Inputkan Nama :");
        nama = input.nextLine();
        System.out.println("Nama Saya : " + nama);

        System.out.print("Inputkan Umur :");
        umur = input.nextInt();
        System.out.println("Umur Saya : " + umur);

        System.out.print("Inputkan Uang :");
        uang = input.nextDouble();
        System.out.println("Uang Saya : " + uang);

        System.out.print("Inputkan Phi :");
        phi = input.nextFloat();
        System.out.println("Uang Saya : " + phi);

        System.out.print("Inputkan Kondisi :");
        kondisi = input.nextBoolean();
        System.out.println("Kondisi Saya : " + kondisi);
    }
}
```

Jalankan program hasilnya seperti gambar dibawah



```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Inputkan Nama :komputerkit
Nama Saya : komputerkit
Inputkan Umur : 3
Umur Saya : 3
Inputkan Uang : 55,8
Uang Saya : 55.8
Inputkan Phi : 3,14
Uang Saya : 3.14
Inputkan Kondisi : true
Kondisi Saya : true

Process finished with exit code 0
```

## 7. Operator

Operator adalah simbol yang digunakan untuk menjalankan aksi operand. Operand adalah sesuatu yang di operasikan oleh operator.

- Operator Matematika (untuk proses hitung matematika)

Operator	Keterangan
+	penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Sisa Pembagian

Buat program seperti gambar dibawah. Penulisan variabel huruf besar dan kecil dengan bacaan sama akan dianggap berbeda oleh java. Jalankan program jika sudah selesai.

```
package com.komputerkit;
public class Main {
    public static void main(String[] args) {
        int hasil;
        hasil = 10 + 5;
        System.out.println(hasil);

        double Hasil;
        Hasil = 7.5 - 2;
        System.out.println(Hasil);

        float HAsil;
        HAsil = (float) (3.14 -2);
        System.out.println(HAsil);

        double HASil;
        HASil= 8 / 3;
        System.out.println(HASil);

        int HASIL;
        HASIL = 7 % 2;
        System.out.println(HASIL);
    }
}
```

- Operator Penugasan (pengisian variabel)

Operator	Persamaan	Keterangan
=	=	Pemberian Nilai
+=	Angka = Angka + 2	Penambahan Bilangan
-=	Angka = Angka - 2	Pengurangan Bilangan
/=	Angka = Angka / 2	Pembagian Bilangan
%	%	Peroleh Sisa Pembagian

Buat program seperti gambar dibawah. Jalankan program jika sudah selesai.

```
package com.komputerkit;
public class Main {
    public static void main(String[] args) {
        int angka = 10;
        System.out.println(angka);

        angka +=5;
        System.out.println(angka);

        angka -=3;
        System.out.println(angka);

        int bilangan = 7;
        bilangan /= 3;
        System.out.println(bilangan); |
```

```
        int bagi;
        bagi = 7 % 3;
        System.out.println(bagi);
    }
}
```

- Operator Pembanding (membandingkan variabel atau data)

Operator	Keterangan
==	Sama Dengan
!=	Tidak sama Dengan
>	Lebih Besar dari
<	Kurang Dari
>=	Lebih besar sama dengan
<=	Kurang dari sama dengan

Buat program seperti gambar dibawah ini dan perhatikan hasilnya

```
package com.komputerkit;
public class Main {
    public static void main(String[] args) {
        boolean hasil;
        hasil = 5 == 6;
        System.out.println(hasil);

        hasil = 5 != 6;
        System.out.println(hasil);

        hasil = 5 < 6;
        System.out.println(hasil);

        hasil = 5 > 6;
        System.out.println(hasil);

        hasil = 5 <= 6;
        System.out.println(hasil); |

        hasil = 5 >= 6;
        System.out.println(hasil);
    }
}
```

- Operator Logika (operasi dengan hasil kondisi benar atau salah)

Operator	Keterangan
&&	AND (hasil BENAR jika semua yang di uji BENAR)
	AND (hasil SALAH jika semua yang di uji SALAH)

Buat program seperti gambar dibawah

```
package com.komputerkit;
public class Main {
    public static void main(String[] args) {
        boolean hasil;
        hasil = true && true && true;
        System.out.println(hasil);

        hasil = false && false && false;
        System.out.println(hasil);

        hasil = true && false && true;
        System.out.println(hasil);

        hasil = true || true || true;
        System.out.println(hasil);

        hasil = false || false || false;
        System.out.println(hasil);

        hasil = true || false || false;
        System.out.println(hasil);
    }
}
```

## 8. Pengujian

### Konsep

Joni mengerjakan soal ujian jika joni mendapatkan nilai lebih besar dari 55 maka joni "LULUS". Jika tidak maka joni "TIDAK LULUS". Dari permasalahan diatas anda diminta membuat sebuah logika pengujian terhadap nilai yang diperoleh oleh joni.

Dari permasalahan diatas kita bisa melakukan pengujian dengan tabel dibawah:

Pengujian (nilai > 55)	Hasil
Jika benar	"LULUS"
Jika salah	"TIDAK LULUS"

Pengujian hanya akan menguji kondisi benar atau salah.

Kondisi pengujian ada di dalam kurung (**pengujian**)

Buat program pengujian benar atau salah seperti dibawah, kemudian jalankan

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int nilai = 56;
        boolean hasil = nilai > 55;

        if (hasil){
            System.out.println("LULUS");
        } else {
            System.out.println("TIDAK LULUS");
        }

    }
}
```

Ubah nilai menjadi 50 seperti gambar dibawah

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int nilai = 50;
        boolean hasil = nilai > 55;

        if (hasil){
            System.out.println("LULUS");
        } else {
            System.out.println("TIDAK LULUS");
        }

    }
}
```

Untuk menyederhanakan program pengujian nilai bisa dilakukan di dalam IF seperti pada gambar dibawah ini:

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        int nilai = 50;

        if (nilai > 55){
            System.out.println("LULUS");
        } else {
            System.out.println("TIDAK LULUS");
        }
    }
}
```

## 9. Pengujian Ganda

### Konsep

Nilai yang diperoleh oleh joni adalah nilai antara 0 dan 100 jika nilai yang dimasukan lebih kecil dari nol maka nilai dianggap salah. Jika nilai yang dimasukan lebih besar dari 100 maka nilai juga dianggap salah. Nilai yang benar adalah nilai antara 0 sampai 100.

Dari permasalahan diatas kita bisa melakukan pengujian dengan tabel dibawah:

Pengujian	Hasil
Nilai $\geq 0$ , nilai $\leq 100$	"Nilai Benar"
Nilai $< 0$ , nilai $> 100$	"Nilai Salah"

Buat aplikasi seperti gambar dibawah ini

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        int nilai = 50;

        if ( (nilai >= 0) && (nilai <=100) ){
            System.out.println("Nilai Benar");
        } else {
            System.out.println("Nilai Salah");
        }
    }
}
```

## 10. Pengujian Bersarang (IF di dalam IF)

### Konsep

Nilai yang dimasukan ke dalam pengujian adalah nilai antara 0 dan 100. Jika nilai yang masuk sudah benar antara 0 dan 100, maka nilai tersebut akan di uji kembali apakah nilai tersebut lebih besar dari 55. Jika nilai nya lebih besar dari 55 maka nilai joni dianggap lulus.

Buat program seperti gambar dibawah ini. Perhatikan hasilnya

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int nilai = 50;

        if ( (nilai >= 0) && (nilai <=100) ){
            if(nilai > 55){
                System.out.println("LULUS");
            }else {
                System.out.println("TIDAK LULUS");
            }
        } else {
            System.out.println("Nilai Salah");
        }
    }
}
```

## 11. Pemilihan (Selection)

Pada program disediakan fasilitas pilihan yang digunakan untuk menguji suatu keadaan yang sudah pasti. Keadaan tersebut kemudian di cocokan dengan pilihan yang ada di program.

Pada contoh dibawah ini diberikan pilihan keadaan 1 dan 2. Dan pilihan yang tidak tersedia. Jika data yang dimasukan 1 maka pilihan yang tersedia adalah "minggu". Jika data yang dimasukan 2 maka pilihannya adalah "senin". Jika data yang dimasukan selain 1 dan 2 maka akan tampil "Tidak ada".

```
package com.komputerkit;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        int x;
        x=1;
        switch (x){
            case 1:
                System.out.println("Minggu");
                break;

            case 2:
                System.out.println("Senin");
                break;

            default:
                System.out.println("Tidak ada");
        }
    }
}
```

Buat program seperti gambar dibawah, jalankan dan lihat hasilnya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        String hari;
        hari ="minggu";

        switch (hari){
            case "minggu":
                System.out.println(1);
                break;
            case "senin":
                System.out.println(2);
                break;
            default:
                System.out.println(0);
        }
    }
}
```

## 12. Pengulangan (*Looping*)

### Konsep

Pada hari senin joni ada pelajaran olah raga di lapangan bersama guru olah raga dan teman satu kelasnya. Guru olah raga memerintahkan semua siswa untuk lari keliling lapangan sebanyak 5 kali. Guru olah raga berdiri di dekat gawang sebelah utara. Para siswa mulai berlari di awali dari tempat tersebut. Saat guru mengatakan SATU (1) murid mulai berlari keliling, ketika melewati gawang sebelah utara guru mengatakan DUA (2), murid kemdian berlari keliling lagi sampai melewati gawang dan guru mengatakan TIGA (3), murid keliling lagi sampai guru menyelesaikan hitungannya sebanyak LIMA (5)

Dari permasalahan diatas kita bisa membuat tabel sebagai berikut:

Pengulangan	Yang dikerjakan
Hitungan ke - 1	Keliling lapangan 1 kali
Hitungan ke - 2	Keliling lapangan 2 kali
Hitungan ke .. n	Keliling lapangan n kali

Dari konsep diatas bisa dibuat program sebagai berikut:

Penghitungan menggunakan for

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        for (int guru=1; guru<=5; guru++){
            System.out.println("Keliling lapangan ke : "+guru);
        }
    }
}
```

### Penjelasan program

`int guru=1;` adalah hitungan awal pengulangan, `guru<=5;` hitungan akhir  
`guru++` pengulangan, nilai penambahan pengulangan sebanyak 1 dari nilai awal.  
Contoh diatas nilai awalnya 1 jika sudah satu keliling nilai akan ditambah 1 dari nilai sebelumnya dari 1,2,3,4,5

## 13. Pengulangan Bersarang (*Nested Loop*)

### Konsep

Pada hari senin joni ada pelajaran olah raga di lapangan bersama guru olah raga dan teman satu kelasnya. Guru olah raga memerintahkan joni berlari sebanyak 3 kali keliling lapangan saat berlari keliling joni diminta mengambil 2 rumput untuk setiap lari keliling.

Dari permasalahan diatas maka bisa dibuat program seperti gambar dibawah

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        for (int guru=1; guru<=3; guru++){

            System.out.println("Keliling ke : "+guru);

            for (int rumput=1; rumput<=2; rumput++){
                System.out.println("Rumput ke : "+rumput);
            }
        }
    }
}
```

Hasil program akan menampilkan seperti gambar dimana perhitungan akan dimulai dari luar dulu kemudian akan menyelesaikan yang di dalam sampai selesai. Setelah yang di dalam selesai yang diluar naik lagi satu putaran.

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Keliling ke : 1
Rumput ke : 1
Rumput ke : 2
Keliling ke : 2
Rumput ke : 1
Rumput ke : 2
Keliling ke : 3
Rumput ke : 1
Rumput ke : 2

Process finished with exit code 0
```

### Pengulangan dengan while

Pengulangan menggunakan while diawali dengan deklarasi variabel [int x], penentuan nilai awal [x=0], nilai akhir [x<10] dan perhitungan naik [x++].

```
package com.komputerkit;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        int x;
        x=0;
        while (x<10){
            System.out.println("pengulangan ke : "+ x);
            x++;
        }
    }
}
```

### Pengulangan dengan do while

Pengulangan menggunakan while diawali dengan deklarasi variabel [int x], penentuan nilai awal [x=0], perhitungan naik [x++], dan nilai akhir [x<10].

```
package com.komputerkit;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        int x;
        x=0;
        do{
            System.out.println("pengulangan ke : "+ x);
            x++;
        }while(x<10);
    }
}
```

### Pengulangan menurun

Pada pengulangan menurun nilai awal adalah nilai yang besar [x=5], nilai akhir adalah nilai terkecil [x>0], dan untuk menurunkan nilai menggunakan [x--]

```
package com.komputerkit;

import java.util.*;

public class Main {
    public static void main(String[] args) {
        int x=5;
        while (x>0){
            System.out.println(x);
            x--;
        }
    }
}
```

## 14. Array (Larik / Rantang)

### Konsep

Joni diminta ibunya membawa makanan dalam rantang 4 susun dengan sekali bawa. Isi susunan rantang [0] adalah nasi, rantang [1] sayur, rantang [2] ikan, rantang [3] sambal. Rantang dibuat untuk memudahkan membawa makanan tersebut.

Dari cerita tersebut maka di dalam program data yang dimasukan ke dalam bentuk rantangan disebut dengan **array**.

```
package com.komputerkit;

public class Main {
    public static void main(String[] args) {
        String [] rantang = {"nasi", "sayur", "ikan", "sambal"};
        System.out.println(rantang[0]);
    }
}
```

Deklarasi array adalah

String [] rantang

Isi array

{"nasi", "sayur", "ikan", "sambal"};

Untuk menampilkan array diawali dari posisi terbawah atau indeks ke 0  
**Pengisian Array**

Pengisian array bisa dilakukan saat program dijalankan diawali dengan deklarasi jumlah array yang akan dimasukan

```
String [] rantang;  
rantang = new String[4];
```

Kemudian pengisian array menggunakan scanner dan looping

```
for (int x=0; x<4; x++){  
    System.out.print("isi rantang : ");  
    rantang[x]=input.nextLine();  
}
```

Untuk menampilkan array juga menggunakan looping

```
for (int x=0; x<4; x++){  
    System.out.println(rantang[x]);  
}
```

Berikut adalah program lengkapnya

```
package com.komputerkit;  
import java.util.*;  
public class Main {  
  
    public static void main(String[] args) {  
  
        String [] rantang;  
        rantang = new String[4];  
  
        Scanner input = new Scanner(System.in);  
  
        for (int x=0; x<4; x++){  
            System.out.print("isi rantang : ");  
            rantang[x]=input.nextLine();  
        }  
  
        for (int x=0; x<4; x++){  
            System.out.println(rantang[x]);  
        }  
    }  
}
```

### Array multidimensi

Array bisa dibuat lebih dari satu rantangan, misal kita akan membawa dua rantang satu berisi camilan dan satu berisi minuman maka kita bisa membuat array nya.

```
package com.komputerkit;
import java.util.*;
public class Main {

    public static void main(String[] args) {

        String [][] makanan ={ {"kue", "donat", "martabak"}, {"teh", "kopi"}};

        System.out.println(makanan[0][0]);
        System.out.println(makanan[1][0]);
    }
}
```

Untuk menampilkan isi array dengan cara memanggil indeks atau atau nomer rantang nya dan di ikuti dengan nomer susun nya

```
System.out.println(makanan[0][0]);
System.out.println(makanan[1][0]);
```

Hasil dari program diatas adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
kue
teh

Process finished with exit code 0
```

## 15. Method (Blok Program)

Method atau blok program digunakan sebagai wadah dari proses untuk memudahkan penggunaan dan pemanfaatan prosesnya. Contoh proses yang tidak dimasukan ke wadah

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int r = 5;
        double luasLingkaran = 3.14 * r * r;

        System.out.println(luasLingkaran);
    }
}
```

Ketika program dijalankan maka proses menghitung luas lingkaran akan langsung dijalankan. Perhatikan contoh yang kedua ketika proses hitung luas lingkaran dimasukan kedalam wadah

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        luasLingkaran(5);
    }

    public static void luasLingkaran(int r){
        double luas = 3.14 * r * r;
        System.out.println(luas);
    }
}
```

Pada contoh diatas wadah prosesnya adalah

```
public static void luasLingkaran(int r){
    double luas = 3.14 * r * r;
    System.out.println(luas);
}
```

Nama wadahnya adalah **luasLingkaran** ketika akan menjalankan prosesnya cukup dipanggil nama dari wadahnya

```
public static void main(String[] args) {
    luasLingkaran(5);
}
```

### Method Tanpa Inputan

Pembuatan program akan lebih mudah dilakukan dengan sistem BLOK dengan menggunakan sistem blok semua proses akan disimpan dalam blok. Jika terjadi kesalahan cukup melihat ke blok yang salah tersebut.  
Buat program seperti contoh, jalankan program dan perhatikan hasilnya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        luasLingkaran();
        luasPersegi();
        luasSegitiga();
    }

    public static void luasLingkaran() {
        System.out.println("Luas Lingkaran");
    }

    public static void luasSegitiga(){
        System.out.println("Luas Segitiga");
    }

    public static void luasPersegi(){
        System.out.println("Luas Persegi");
    }
}
```

### Method Dengan Inputan

Method atau BLOK program bisa diberikan inputan atau masukan agar input tersebut bisa di proses di dalam blok.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        luasLingkaran( r: 3.5);
        luasPersegi( p: 3);
        luasSegitiga( t: 4);
    }

    public static void luasLingkaran(double r) {
        System.out.println("Luas Lingkaran dengan jejari : "+r);
    }

    public static void luasSegitiga(int t){
        System.out.println("Luas Segitiga dengan tinggi : "+t);
    }

    public static void luasPersegi(int p){
        System.out.println("Luas Persegi dengan panjang : "+p);
    }
}
```

Tampilan ketika program ketika dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Luas Lingkaran dengan jejari : 3.5
Luas Persegi dengan panjang : 3
Luas Segitiga dengan tinggi : 4

Process finished with exit code 0
```

Bagian inputan harus di deklarasikan saat pembuatan method seperti berikut

```
luasLingkaran(double r)
Inputan juga harus langsung di isi ketika method dipanggil
luasLingkaran( r: 3.5);
```

### Method dengan Output

Pada proses tertentu method dibutuhkan untuk mengeluarkan output agar bisa diproses lagi di tempat lain.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int t = 10;
        double volumeTabung;
        volumeTabung = luasLingkaran( r: 3.5) * t;
        System.out.println(volumeTabung);

    }

    public static double luasLingkaran(double r) {
        double luas = 3.14 * r * r ;
        return luas;
    }

}
```

Output method ditandai dengan **return**

```
public static double luasLingkaran(double r) {
    double luas = 3.14 * r * r ;
    return luas;
}
```

Perbedaan penulisan method dengan output dan method tanpa output adaalah ada pada deklarasi nama method.

Jika method tanpa output deklarasinya adalah

```
public static void luasPersegi()
```

Sedangkan method dengan output kalimat void diganti dengan tipe data dari output yang ingin dihasilkan oleh method

```
public static double luasLingkaran(double r)
```

Contoh method dengan input integer dan output boolean.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        System.out.println(periksaTanggal( 35));
    }

    public static boolean periksaTanggal(int t){
        boolean hasil;

        if(t < 1 || t > 31){
            hasil = false;
        }else {
            hasil = true;
        }

        return hasil;
    }
}
```

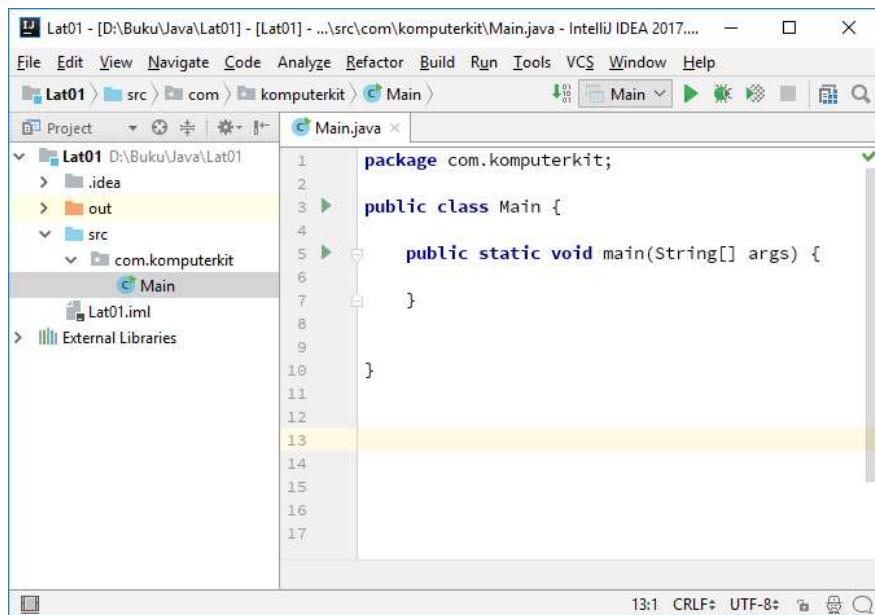
# PBO

Pemrograman Berorientasi Objek

## 16. Properti, Method, Class

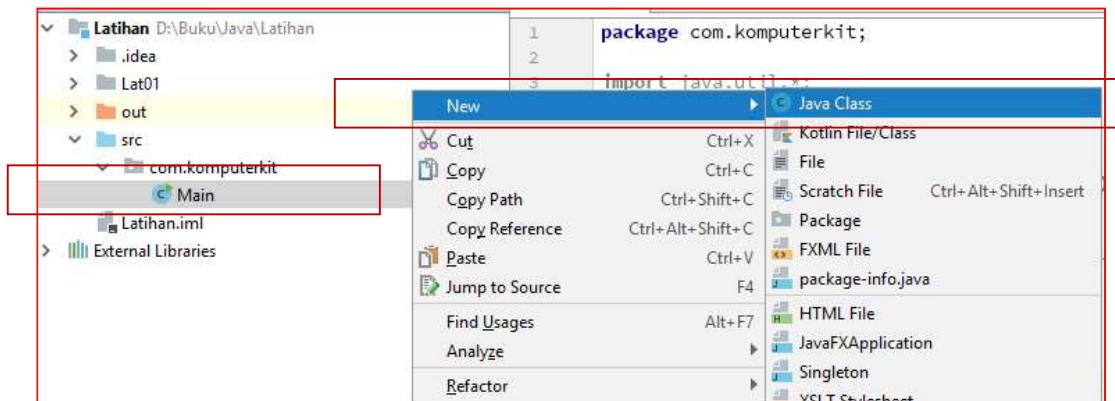
Bagian	Penjelasan
<b>Properti</b>	Variabel di dalam oop disebut juga sebagai properti. Properti berfungsi sebagai wadah atau tempat dari data yang akan di proses atau data yang di hasilkan oleh proses/ <b>Contoh :</b> Properti <b>panjang</b> pada program luasPersegi digunakan sebagai wadah dari input yang akan di proses Properti <b>luas</b> pada program digunakan sebagai wadah dari output yang dihasilkan oleh program luasPersegi.
<b>Method</b>	Method adalah BLOK program yang mempunyai tugas tertentu. Pembuatan blok program digunakan untuk mempermudah proses pembuatan program dan pemeliharaan program. Method pada bahasa pemrograman lain dikenal sebagai prosedur atau function.
<b>Class</b>	Class adalah SUPER BLOK atau rumah bagi method dan properti. Method dan properti yang mempunyai kesamaan tugas bisa dikumpulkan ke dalam satu class. Dengan menggunakan class program akan lebih mudah dikembangkan dan di kerjakan bersama dalam satu TIM. Dimana masing - masing anggota tim cukup mengerjakan class tertentu kemudian tinggal menggabungkan class nya saja.

Buat projek baru kemudian simpan di folder yang anda tentukan.

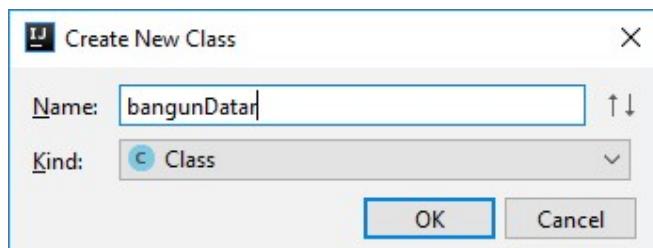


```
Lat01 - [D:\Buku\Java\Lat01] - [Lat01] - ...\\src\\com\\komputerkit\\Main.java - IntelliJ IDEA 2017.... - □ X
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Lat01 > src > com > komputerkit > Main > Main.java
Project Main.java
1 package com.komputerkit;
2
3 public class Main {
4
5     public static void main(String[] args) {
6
7         }
8
9     }
10
11
12
13
14
15
16
17
```

Klik kanan pada **Main**, pilih **New**, pilih **Java Class** sesuai dengan gambar



Beri nama class dengan **bangunDatar**



Class **bangunDatar** sudah terbentuk



Buat isi dari class **bangunDatar** sebagai berikut

```
package com.komputerkit;

public class bangunDatar {

    public int panjang;

    public void luasPersegi(){
        System.out.println("Luas Persegii");
    }
}
```

Pada class Main buat coding sebagai berikut:

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        bangunDatar bangun = new bangunDatar();

        bangun.panjang = 5;
        System.out.println(bangun.panjang);

        bangun.luasPersegi();

    }
}
```

Bagian	Penjelasan
<b>bangunDatar bangun = new bangunDatar();</b>	Pembuatan obyek atau disebut juga <b>INSTANCE</b> , caranya adalah: Nama class <b>bangunDatar</b> kemudian nama obyek <b>bangun</b> kemudian tanda = (sama dengan) = Kemudian new <b>new</b> kemudian nama class ikuti dengan tanda kurung buka kurung tutup (). <b>bangunDatar();</b>
<b>bangun.panjang = 5;</b>	Setelah objek terbentuk, maka semua isi dari class akan menjadi bagian dari objek. Cara menggunakan bagian class yang sudah menjadi bagian dari objek adalah tulis nama objeknya ikuti dengan nama properti atau method nya. Contoh:

	Pada properti panjang akan di isi nilai 5. Properti panjang merupakan isi dari class bangunDatar yang sudah dibuatkan objeknya.
System.out.println(bangun.panjang);	Menampilkan isi dari properti panjang
bangun.luasPersegi();	Menjalankan method yang ada di class bangunDatar. Caranya tulis nama objeknya ikuti dengan nama method yang ada di class yang sudah dijadikan objek. Contoh: <b>bangun</b> adalah nama objek dan <b>luasPersegi()</b> adalah nama method yang dijalankan.

Tampilan program

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
5
Luas Persegi

Process finished with exit code 0
```

**Semua method yang ada di dalam class bisa dipanggil kalau sudah dibuat menjadi objek.**

## 17. Fungsi Static Pada Method

Fungsi static pada method adalah agar method bisa langsung dipanggil tanpa harus membuat objek terlebih dahulu.

Buat method static di class bangunDatar

```
package com.komputerkit;

public class bangunDatar {

    public int panjang;

    public static void luasPersegi(){

        System.out.println("Luas Persegi");
    }
}
```

Panggil method static di class bangunDatar dari Main, cara panggilnya adalah nama class ikuti dengan nama method nya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        bangunDatar.luasPersegi();
    }
}
```

Tampilan outputnya adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Luas Persegi

Process finished with exit code 0
```

## 18. Konstruktor (method yang langsung dijalankan ketika objek dibuat)

Aturan pembuatan konstruktor

- Nama method konstruktor harus sama dengan nama class
- Dalam satu class hanya ada satu konstruktor
- Tidak boleh ada return (output method)
- Method bisa berisi parameter input

Buat coding di class bangunDatar sebagai berikut

```
package com.komputerkit;

public class bangunDatar {
    public int panjang;
    bangunDatar(){
        System.out.println("ini konstruktor");
    }
}
```

Buat coding di class Main sebagai berikut

```
package com.komputerkit;

public class Main {
    public static void main(String[] args) {
        bangunDatar bangun = new bangunDatar();
    }
}
```

Hasil ketika dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
ini konstruktor

Process finished with exit code 0
```

## Konstruktor Dengan Parameter Input

Buat class bangunDatar seperti gambar dengan parameter

```
package com.komputerkit;

public class bangunDatar {

    public int panjang;

    bangunDatar(int p, int l){
        System.out.println("Panjang : "+ p + " Lebar : "+l);
    }
}
```

Buat class Main seperti gambar

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        bangunDatar bangun = new bangunDatar( p: 3, l: 4);
    }
}
```

Hasil ketika dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Panjang : 3 Lebar : 4

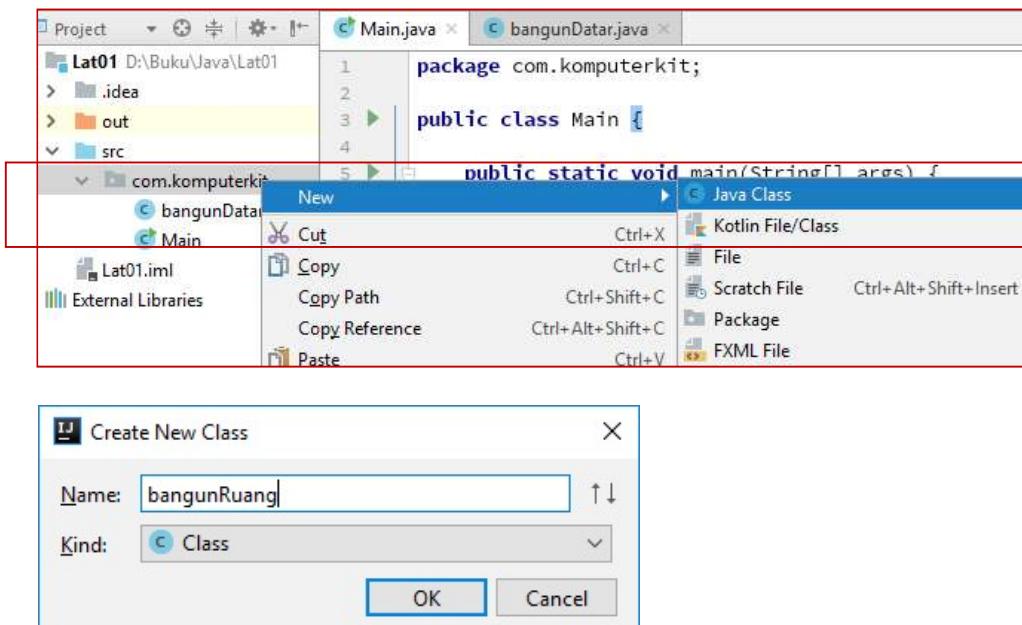
Process finished with exit code 0
```

## 19. Inheritance (Pewarisan)

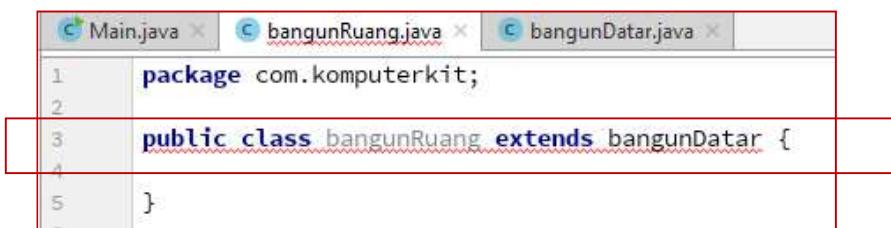
Pewarisan atau inheritance adalah pemberian semua isi properti dan method dari class induk ke class anak. Sebagai contoh kita akan membuat 2 class sebagai berikut

- Class bangunDatar sebagai class induk
- Class bangunRuang sebagai class anak

Isi dari class bangunDatar akan digunakan pada class bangunRuang



Kita sekarang mempunyai 2 class. Tambahkan extends bangunDatar di class bangunRuang



Tambahkan coding di class bangunRuang

```
package com.komputerkit;  
  
public class bangunRuang extends bangunDatar {  
  
    public void volumeKubus(){  
        System.out.println("Volume Kubus");  
    }  
}
```

Sekarang kita mempunyai 2 class yang bisa dijadikan objek, jika kita menjadikan class bangunRuang sebagai obyek maka secara otomatis isi dari class bangunDatar akan ikut, karena class bangunRuang mewarisi isi dari class bangunDatar

Isi dari class bangunDatar

```
package com.komputerkit;

public class bangunDatar {

    public int panjang;

    public static void luasPersegi (){
        System.out.println("Luas Persegi");
    }
}
```

Isi dari class bangunRuang

```
package com.komputerkit;

public class bangunRuang extends bangunDatar {

    public void volumeKubus(){
        System.out.println("Volume Kubus");
    }
}
```

Kita akan membuat objek dari class bangunRuang. Dengan membuat class dari bangunRuang maka kita bisa memanggil isi dari class bangunDatar

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        bangunRuang bangun = new bangunRuang();

        bangun.volumeKubus();
        bangun.luasPersegi();

    }
}
```

Kita juga bisa membuat objek menggunakan class bangunDatar  
Method di class anak tidak bisa digunakan oleh objek yang menggunakan class induk.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        bangunDatar bangun = new bangunDatar();
        bangun.luasPersegi();
    }
}
```

## 20. Abstract Class

*Abstract class adalah class yang tidak bisa dibuat menjadi objek*, tapi abstract class bisa dijadikan sebagai class induk.

Buat class induk dengan abstract class sebagai berikut

```
package com.komputerkit;

abstract class bangunDatar {

    public int panjang;

    public static void luasPersegi (){

        System.out.println("Luas Persegii");
    }
}
```

Buat class anak dengan extends sebagai berikut

```
package com.komputerkit;  
  
public class bangunRuang extends bangunDatar {  
    public void volumeKubus(){  
        System.out.println("Volume Kubus");  
    }  
}
```

Buat objek dengan menggunakan class bangunRuang

```
package com.komputerkit;  
  
public class Main {  
    public static void main(String[] args) {  
        bangunRuang bangun = new bangunRuang();  
        bangun.luasPersegi();  
    }  
}
```

## 21. Encapsulasi (Public, Private, protected)

Ada 3 cara untuk menggunakan properti, method, dan class yaitu:

- **Public**  
Bisa digunakan di semua lokasi baik dalam class nya sendiri, luar class nya sendiri, dalam package nya sendiri, luar package nya sendiri.
- **Private**  
Hanya bisa digunakan dalam class nya sendiri
- **Protected**  
Bisa digunakan dalam class nya sendiri, luar class nya sendiri, dalam package nya sendiri, TETAPI TIDAK bisa diluar packagennya sendiri

### Contoh Public

Buat program seperti dibawah ini

```
package com.komputerkit;

public class bangunDatar {

    public int panjang;

    public void luasPersegi (){

        System.out.println("Luas Persegi");
    }
}
```

Buat objek menggunakan class bangunDatar

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        bangunDatar bangun = new bangunDatar();

        bangun.panjang = 10;
        System.out.println(bangun.panjang);

        bangun.luasPersegi();
    }
}
```

### Contoh Private

Buat program seperti dibawah

```
package com.komputerkit;

public class bangunDatar {

    private void luasPersegi (){

        System.out.println("Luas Persegi");
    }

    public void panggil(){
        luasPersegi();
    }
}
```

Karena method luasPersegi di **private** maka harus dibuat method yang **public** agar method luasPersegi bisa digunakan.

Buat objek menggunakan class bangunDatar. Objek yang sudah dibuat hanya bisa menggunakan method **public**.

```
package com.komputerkit;

public class Main {

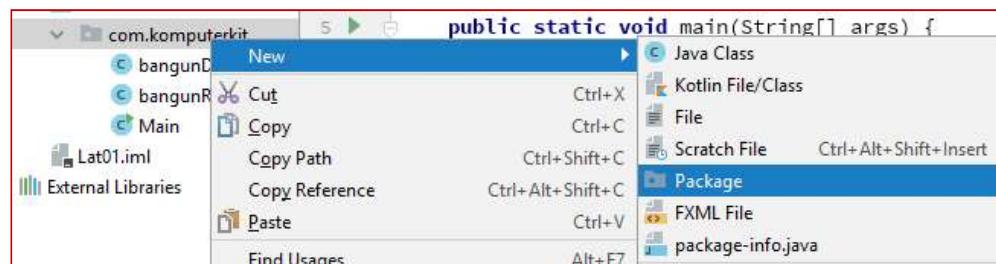
    public static void main(String[] args) {

        bangunDatar bangun = new bangunDatar();

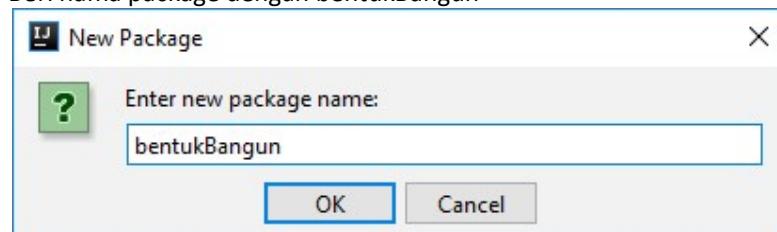
        bangun.panggil();
    }
}
```

### Contoh Protected

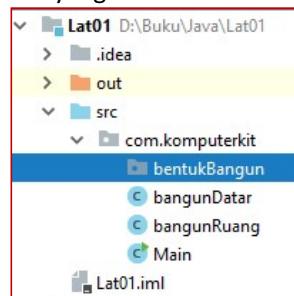
Buat package baru, klik kanan pilih [New] pilih [Package]



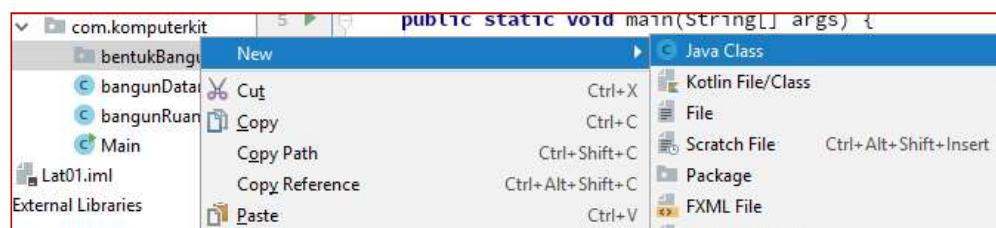
Beri nama package dengan bentukBangun



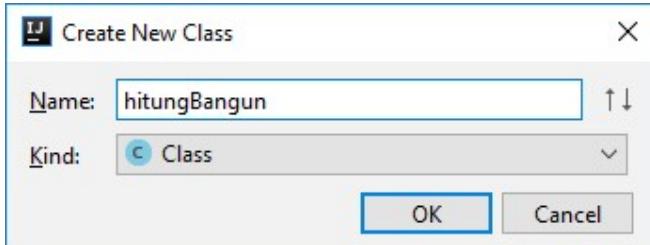
File yang sudah terbentuk



Buat class baru pada package bentukBangun, klik kanan pada package bentukBangun



Beri nama class dengan hitungBangun



Buat isi class hitungBangun sebagai berikut, perhatikan nama package nya

```
package com.komputerkit.bentukBangun;

public class hitungBangun {

    protected void caraHitung(){
        System.out.println("Cara Hitung Bangun");
    }

    public void rumusBangun(){
        System.out.println("Rumus Bangun");
    }
}
```

Tambahkan import pada Main class, kemudian buat objek dan panggil method yang ada di class hitungBangun

```
package com.komputerkit;

import com.komputerkit.bentukBangun.*;

public class Main {

    public static void main(String[] args) {

        hitungBangun bangun = new hitungBangun();

        bangun.rumusBangun();
    }
}
```

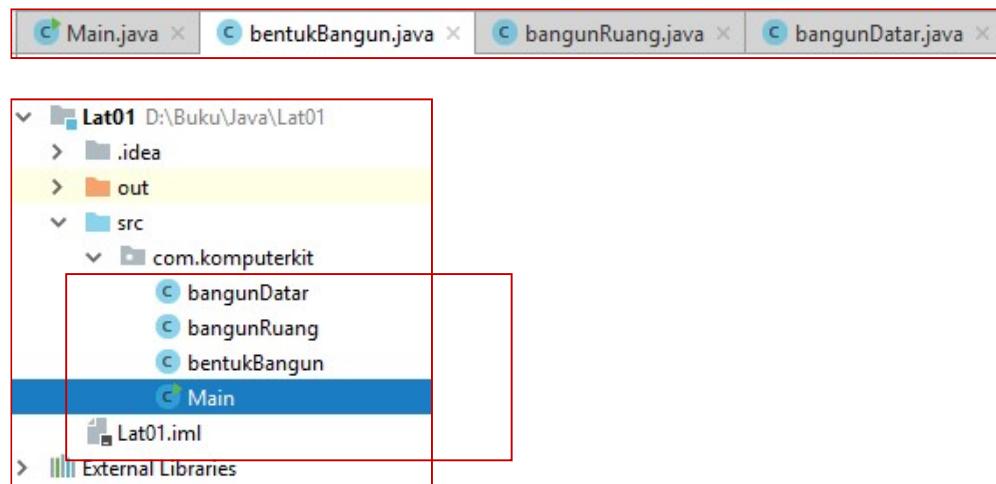
Cara menulis import adalah, nama package di ikuti tanda \* yang berarti semua method yang akan di import.

```
import com.komputerkit.bentukBangun.*;
```

## 22. Polimorfisme (Nama Method yang sama dengan tugas yang berbeda )

Polimorfisme berasal dari kata poli yang berarti banyak dan morfisme yang berarti banyak bentuk. Dalam java polimorfisme digunakan untuk menjelaskan method dengan nama yang sama namun dengan tugas yang berbeda.

Buat 3 kelas dengan nama sebagai berikut



Buat class induk dengan nama bentukBangun

```
package com.komputerkit;

public class bentukBangun {

    public void hitung(){
        System.out.println(" Hitung Bangun ");
    }
}
```

Buat class anak dengan nama bangunDatar

```
package com.komputerkit;

public class bangunDatar extends bentukBangun{

    public void hitung (){
        System.out.println(" Hitung Bangun Datar ");
    }
}
```

Buat class anak dengan nama bangunRuang

```
package com.komputerkit;

public class bangunRuang extends bentukBangun {

    public void hitung(){
        System.out.println(" Hitung Bangun Ruang ");
    }
}
```

Deklarasikan class induk dan nama objek nya. Buat objek dari masing – masing class dan jalankan method nya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        bentukBangun bangun;

        bangun = new bentukBangun();
        bangun.hitung();

        bangun = new bangunDatar();
        bangun.hitung();

        bangun = new bangunRuang();
        bangun.hitung();
    }
}
```

Output ketika program dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Hitung Bangun
Hitung Bangun Datar
Hitung Bangun Ruang

Process finished with exit code 0
```

## 23. Overriding

Overriding adalah penggunaan nama class yang sama dengan tugas yang berbeda pada class *induk* dan class *anak*.

Buat class induk dengan nama bentukBangun

```
package com.komputerkit;

public class bentukBangun {

    public void hitung(){

        System.out.println(" Hitung Bangun ");
    }
}
```

Buat class anak dengan nama bangunDatar

```
package com.komputerkit;

public class bangunDatar extends bentukBangun{

    public void hitung (){
        System.out.println(" Hitung Bangun Datar ");
    }
}
```

Buat 2 objek dengan menggunakan class induk dan class anak. Panggil nama method yang sama dari objek yang berbeda

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        bentukBangun bentuk = new bentukBangun();
        bangunDatar bangun = new bangunDatar();

        bentuk.hitung();

        bangun.hitung();
    }
}
```

Beberapa aturan overriding

- Nama method harus sama antara method pertama dan method berikutnya
- Jika method mempunyai output maka tipe data outputnya harus sama
- Method harus dalam bentuk public

## 24. Overloading

Overloading adalah pembuatan method yang sama dengan tugas yang berbeda dalam satu class.

Buat class bentukBangun dengan 3 method yang sama dengan tugas yang berbeda, sebagai berikut

```
package com.komputerkit;

public class bentukBangun {

    public void hitung(){
        System.out.println("Hitung Bangun ");
    }

    public void hitung(int r){
        System.out.println("Hitung Lingkaran dengan jari : "+r);
    }

    public double hitung (double r){
        double luas = 3.14 * r * r;
        return luas;
    }
}
```

Buat objek menggunakan class bentukBangun kemudian panggil nama method nya

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        bentukBangun bentuk = new bentukBangun();

        bentuk.hitung();

        bentuk.hitung( r: 3);

        System.out.println(bentuk.hitung( r: 10.0));
    }
}
```

Output program yang dihasilkan dari overloading adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
Hitung Bangun
Hitung Lingkaran dengan jari : 3
314.0
```

#### Perbedaan overriding dan overloading

- Overloading, nama method yang sama dengan tugas yang berbeda pada class induk dan class anak
- Overloading, nama method yang sama dengan tugas yang berbeda pada satu class

## 25. Interface

Interface digunakan untuk mencatat method yang akan digunakan dalam class. Proses pencatatan hanya pada nama method saja tanpa isinya. Seperti contoh dibawah terdapat nama interface rumus.

```
interface rumus{
    void luasLingkaran();
    void luasPersegi();
    void luasSegitiga();
}
```

Kegunaan interface adalah untuk mencatat method yang akan digunakan selama pembuatan program. Biasanya programmer akan mencatat dulu secara umum method – method yang akan dibuat dalam program. Setelah mencatat semua kebutuhan method baru kemudian programmer akan membuat isi dari method nya.

Buat class bentukBangun sebagai berikut

```
package com.komputerkit;

interface rumus{

    void luasLingkaran();

    void luasPersegi();

    void luasSegitiga();

}

public class bentukBangun implements rumus {

    public void luasLingkaran(){
        System.out.println("luas Lingkaran");
    }

    public void luasPersegi(){
        System.out.println("luas persegi");
    }

    public void luasSegitiga(){
        System.out.println("luas segitiga");
    }

}
```

Buat objek menggunakan class bentukBangun

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        bentukBangun bangun = new bentukBangun();

        bangun.luasLingkaran();
        bangun.luasPersegi();
        bangun.luasSegitiga();

    }
}
```

Output dari program adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
luas lingkaran
luas persegi
luas segitiga

Process finished with exit code 0
```

## 26. Enumerasi (tipe data bentukan)

enumerasi adalah konstanta yang sudah ditentukan. Bentuk konstanta seperti array. Perbedaan array dengan enumerasi adalah, kalau array bersifat dinamis atau isinya masih bisa dirubah, kalau enumerasi datanya tetap sesuai dengan yang sudah ditentukan.

Buat program seperti berikut

```
package com.komputerkit;

public class Main {

    public enum bangun {persegi, lingkaran, segitiga}

    public static void main(String[] args) {

        bangun[] b = bangun.values();

        System.out.println(b[1]);

        for (bangun bb:bangun.values()){
            System.out.println(bb);
        }
    }
}
```

Deklarasi array dari data enumerasi

```
bangun[] b = bangun.values();
```

Untuk menampilkan satu elemen data enum menggunakan perintah

```
System.out.println(b[1]);
```

Untuk menampilkan semua isi dari data enumerasi

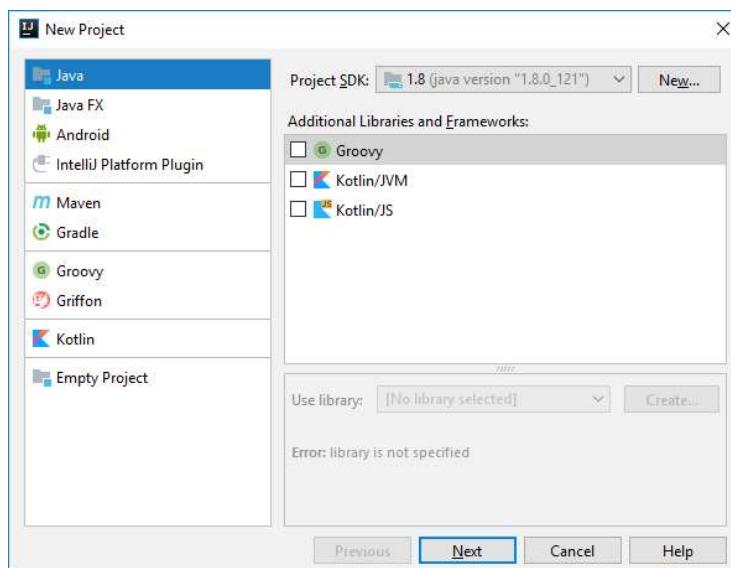
```
for (bangun bb:bangun.values()){
    System.out.println(bb);
}
```

## 27. Java SWING (Form java)

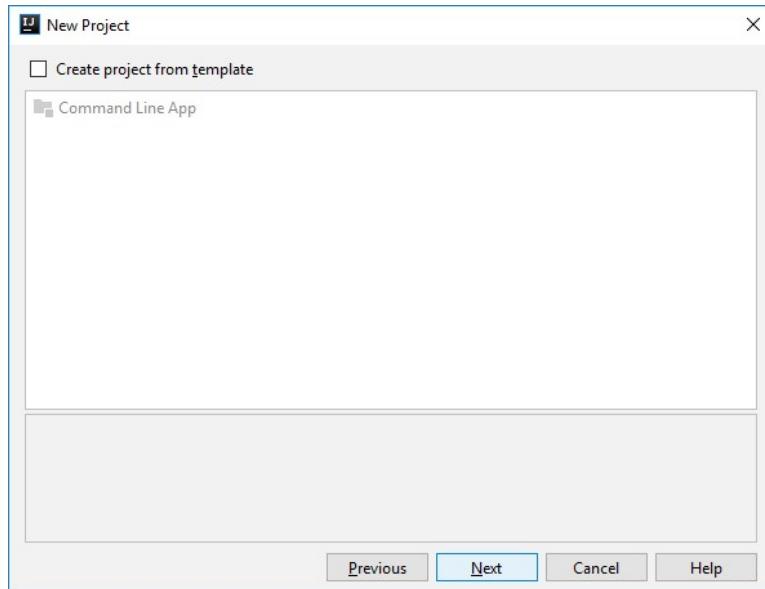
Java swing adalah aplikasi java yang dibuat menggunakan form.  
Buat projek baru dengan cara, klik [ **Create New Project** ]



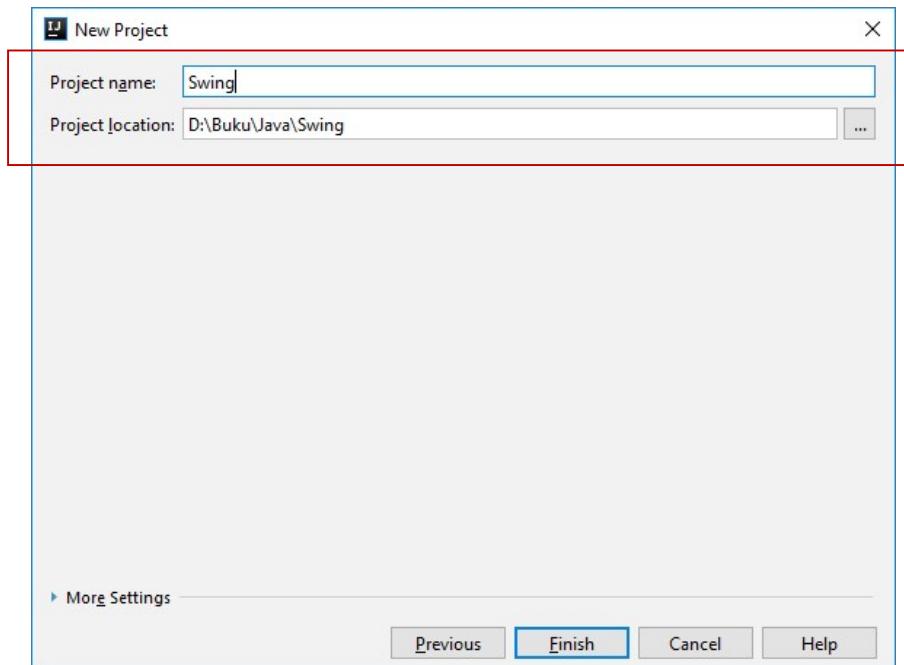
Klik [ **next** ]



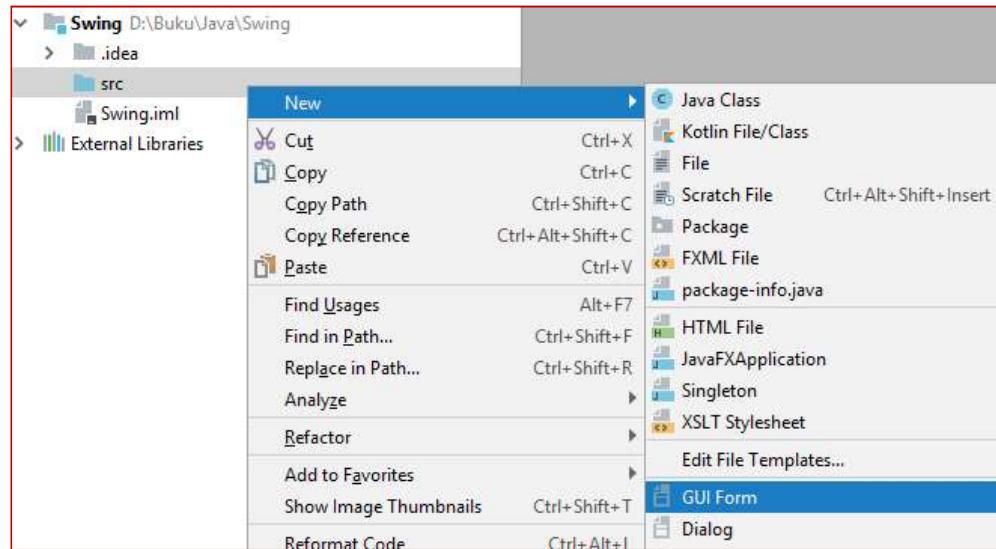
Klik [ next ]



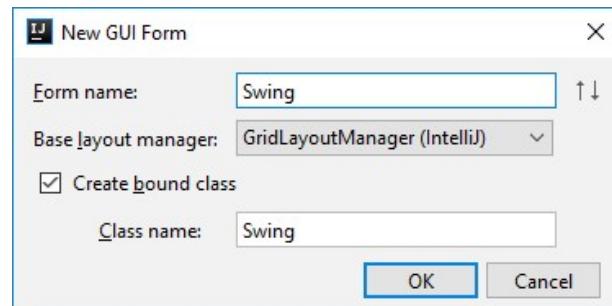
Siapkan folder, beri nama folder dengan swing. Arahkan project location ke folder tersebut. Kemudian klik [finish]



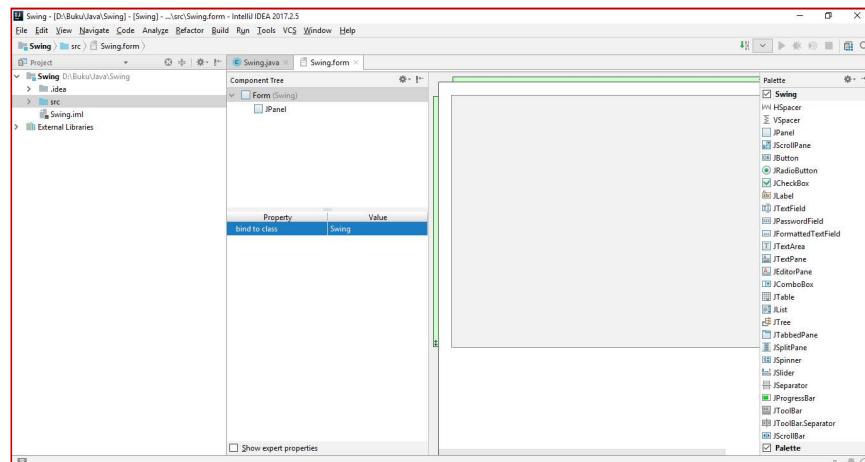
Klik kanan pada folder [ src ] pilih [ New ] pilih dan klik [ GUI Form ]



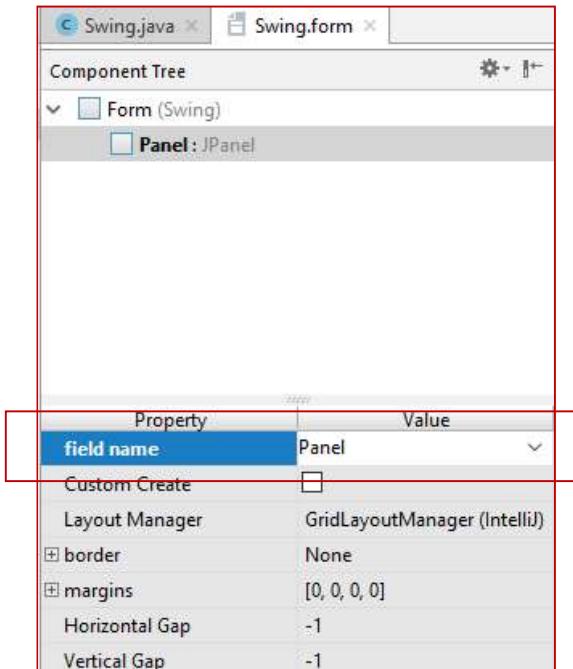
Beri nama [ Swing ]



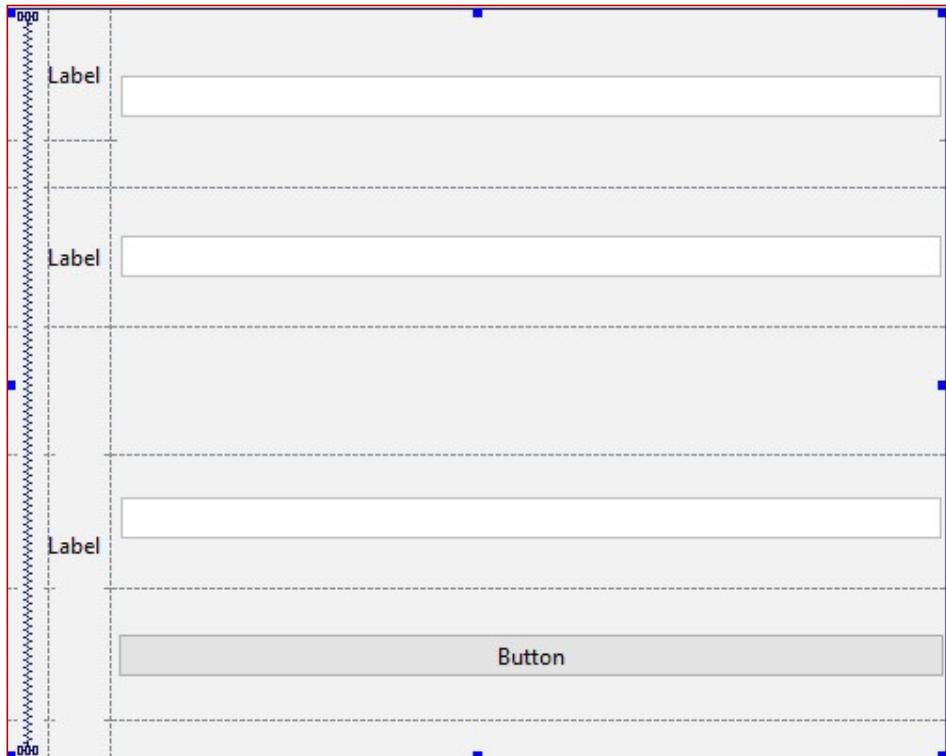
Tampilan swing



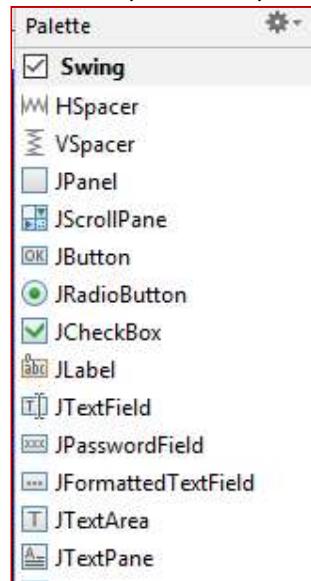
Beri nama panel



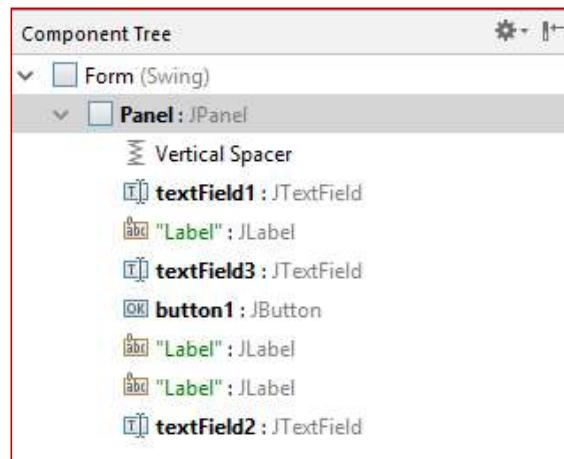
Buat tampilan sebagai berikut



Ambil komponen dari palet



Lihat hasilnya di component tree

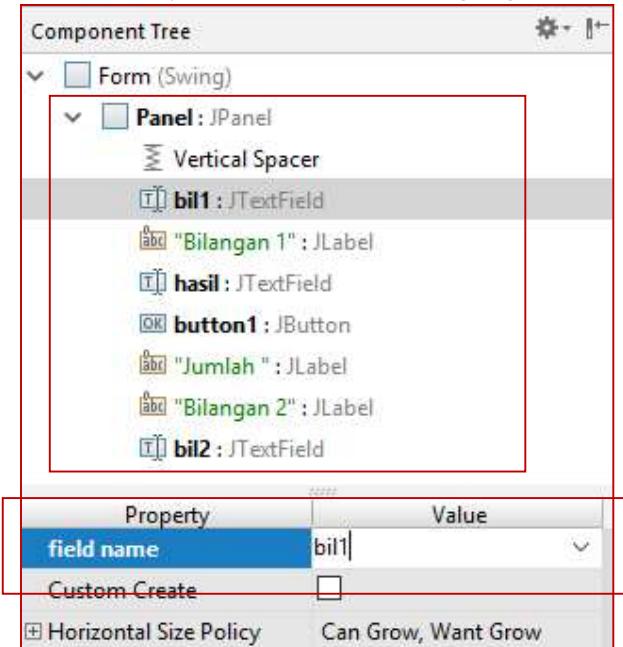


Pada [ text ] isi dengan

textField1 : JTextField
"Bilangan 1" : JLabel
textField3 : JTextField

icon	
labelFor	
text	Bilangan 1

Isi field name pada JTextField sesuai dengan gambar



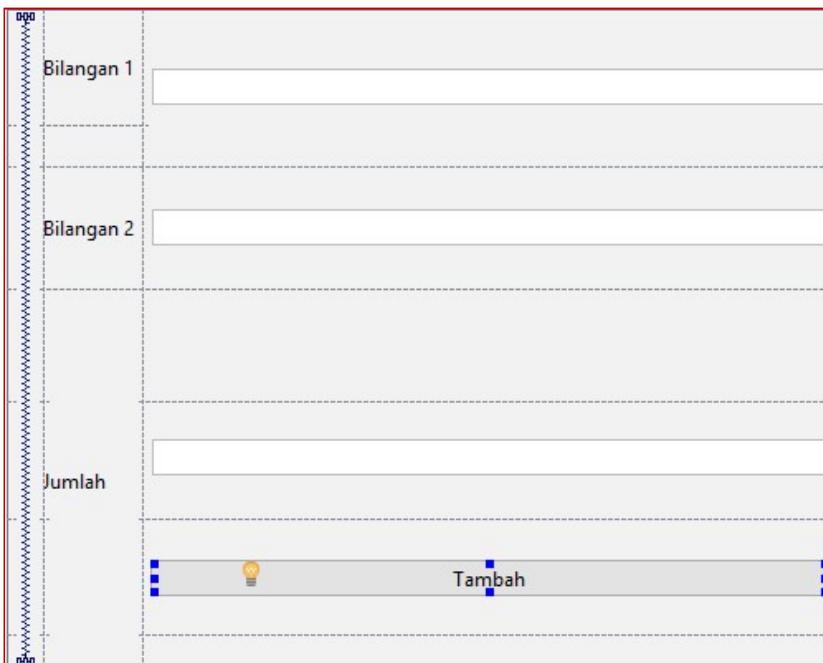
Isi field name JButton dengan btambah



Isi text JButton dengan tambah

horizontalAlignment	Center
horizontalTextPosition	Trailing
icon	
<b>text</b>	<b>Tambah</b>
toolTipText	
verticalAlignment	Center

Hasilnya seperti berikut



Berpindah ke tab swing.java

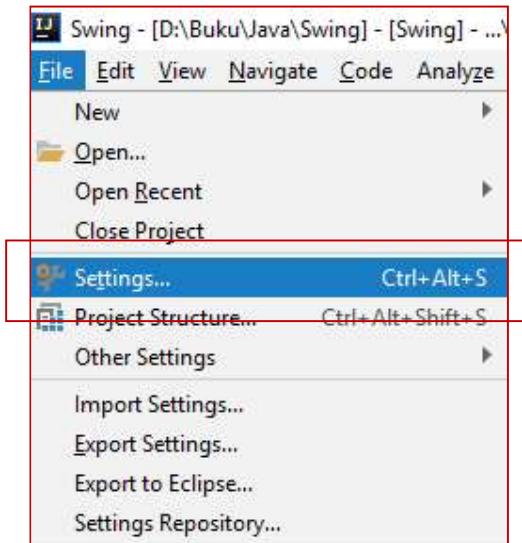
```
1 import javax.swing.*;
2
3 public class Swing {
4     private JPanel Panel;
5     private JTextField bil1;
6     private JTextField hasil;
7     private JButton btambah;
8     private JTextField bil2;
9 }
10
11
```

The screenshot shows a Java code editor with two tabs: "swing.java" and "swing.form". The "swing.java" tab is active, displaying the following code:

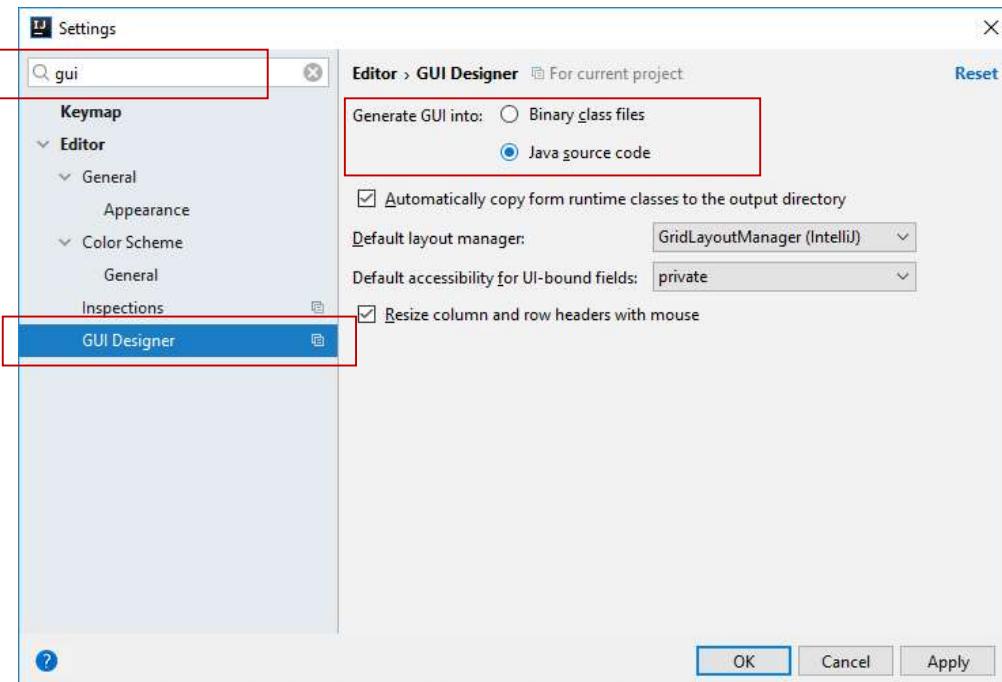
```
1 import javax.swing.*;
2
3 public class Swing {
4     private JPanel Panel;
5     private JTextField bil1;
6     private JTextField hasil;
7     private JButton btambah;
8     private JTextField bil2;
9 }
10
11
```

The code defines a class named "Swing" with several private instance variables: "Panel", "bil1", "hasil", "btambah", and "bil2". Lines 1 through 9 are numbered on the left, and line 11 is highlighted with a yellow background.

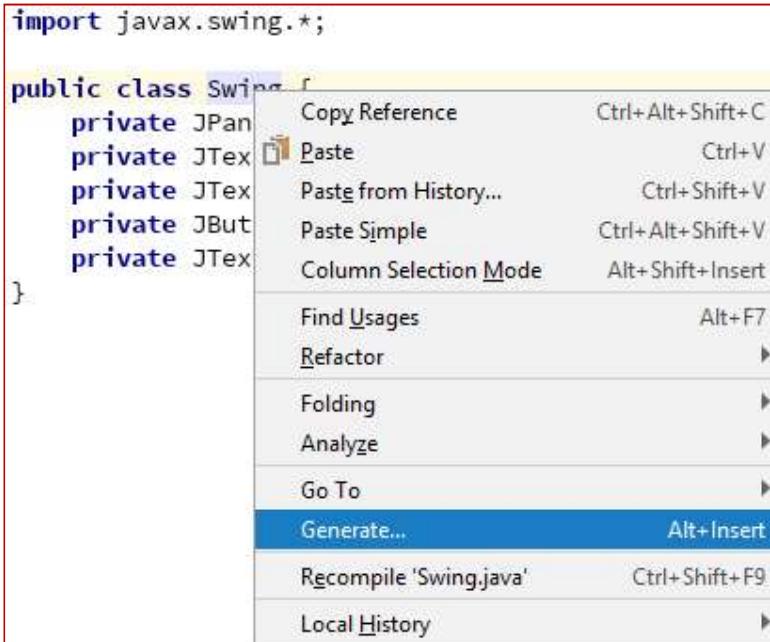
Klik file pilih setting



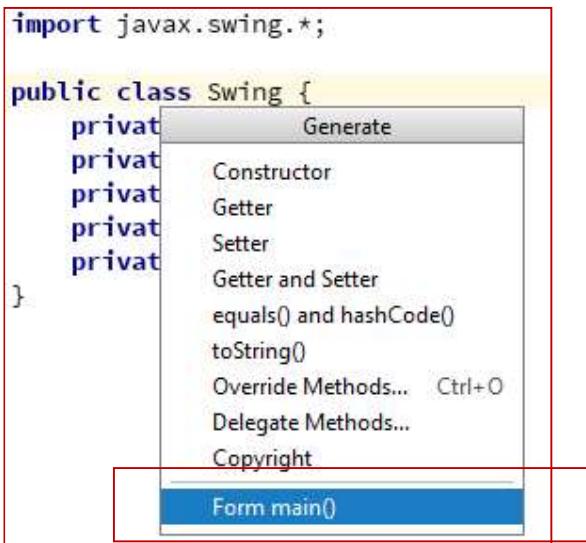
Ketik GUI pilih GUI Designer, klik Java source code dan klik ok



Klik kanan pada public class Swing pilih Generate



Pilih Form main()



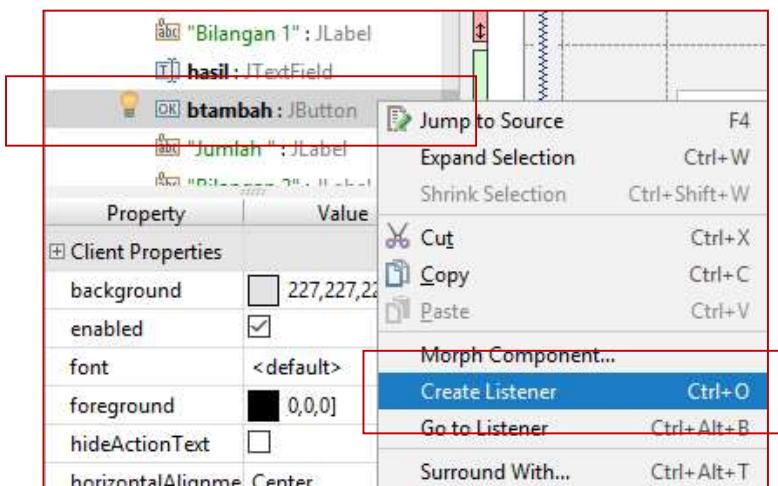
Setelah di klik hasilnya sebagai berikut

```
import javax.swing.*;  
  
public class Swing {  
    private JPanel Panel;  
    private JTextField bil1;  
    private JTextField hasil;  
    private JButton btambah;  
    private JTextField bil2;  
  
    public static void main(String[] args) {  
        JFrame frame = new JFrame("Swing");  
        frame.setContentPane(new Swing().Panel);  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
        frame.pack();  
        frame.setVisible(true);  
    }  
}
```

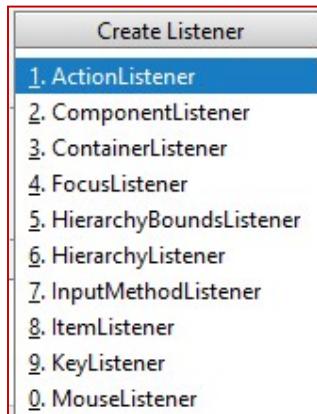
Jalankan programnya hasilnya sebagai berikut



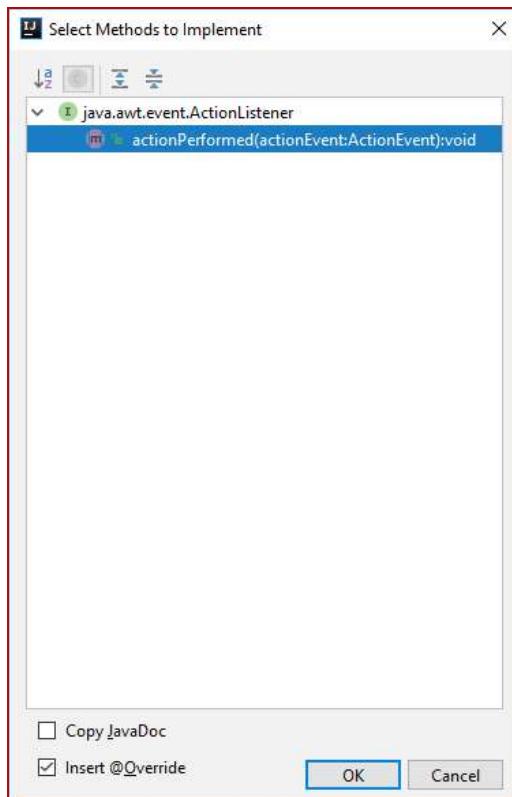
Klik kanan pada component tree pilih JButton, klik create listener



### Pilih ActionListener



Klik OK



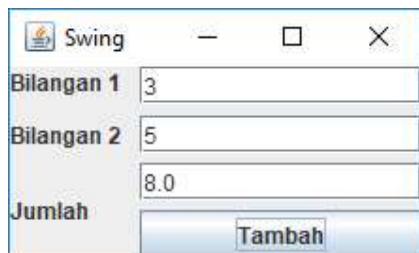
Tambahkan coding berikut kemudian jalankan

```
public Swing() {
    btambah.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent actionEvent) {
            double a = Double.parseDouble(bil1.getText());
            double b = Double.parseDouble(bil2.getText());

            double jumlah = a + b;

            hasil.setText(String.valueOf(jumlah));
        }
    });
}
```

Hasil program yang dibuat adalah



## 28. Collection

Collection adalah array dinamis dimana jumlah isi atau elemen array yang akan dibuat tidak perlu dideklarasikan.

Cara membuat collection adalah:

1. Buat Properti dengan deklarasi ArrayList
2. Isi properti tersebut
3. Untuk menampilkan panggil properti tersebut

```
package com.komputerkit;

import java.util.ArrayList;

public class Main {

    public static void main(String[] args) {
        ArrayList<String> bangun = new ArrayList<String>();
        bangun.add("lingkaran");
        bangun.add("persegi");
        bangun.add("segitiga");

        System.out.println(bangun);
    }
}
```

Tampilan isi dari collection

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
[lingkaran, persegi, segitiga]
```

```
Process finished with exit code 0
```

Jika ingin menampilkan 1 elemen dari collection, gunakan perintah:

```
System.out.println(bangun.get(0));
```

### Mengisi elemen collection saat program dijalankan.

Buat program seperti dibawah ini. Kita akan mengisi collection sebanyak jumlah looping yang dibuat.

```
package com.komputerkit;

import java.util.ArrayList;
import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        String isi;
        Scanner input = new Scanner(System.in);
        ArrayList<String> bangun = new ArrayList<String>();

        for (int a=0; a<3; a++){
            System.out.print("isi bangun : ");
            isi = input.nextLine();
            bangun.add(isi);
        }

        System.out.println(bangun);
    }
}
```

Tampilan ketika program dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
isi bangun : lingkaran
isi bangun : segitiga
isi bangun : persegi
[lingkaran, segitiga, persegi]
```

## 29. Keyword this, final, super

Keyword adalah kata kunci yang bisa digunakan pada program untuk kebutuhan tertentu

### 1. This

This sering digunakan pada deklarasi constructor dimana this **digunakan untuk menunjukkan kepemilikan properti.**

Contoh:

Buat program seperti dibawah, buat class baru dengan nama **Bangun**

```
java Bangun.java
package com.komputerkit;

public class Bangun {

    private String var;

    Bangun(String var){
        var = var;
    }

    public void tampil (){
        System.out.println(var);
    }

}
```

Buat object di Main

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        Bangun b=new Bangun(var: "belajar this") ;
        b.tampil();
    }
}
```

Jika dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
null
```

```
Process finished with exit code 0
```

### **Penjelasan**

Pada class **Bangun** terdapat 2 properti yang bernama **var**. Properti pertama dimiliki oleh **class Bangun**. Properti kedua dimiliki oleh **Constructor Bangun**. **Method tampil** akan menggunakan properti milik **class Bangun** untuk menampilkan isi nya. Karena terdapat 2 properti yang sama dimana **var** akan mengisi **var** maka properti yang dimiliki **class Bangun** tidak akan terisi. Karena **var** yang dianggap adalah **var** milik **Constructor Bangun**.

Untuk bisa mengisi properti **class Bangun** harus digunakan **this** agar **var** yang menggunakan **this** bisa menunjukan kalau **var** tersebut adalah milik **class Bangun**.

```
java X Bangun.java X
package com.komputerkit;

public class Bangun {

    private String var;

    Bangun(String var){
        this.var = var;
    }

    public void tampil (){
        System.out.println(var);
    }
}
```

Hasil ketika dijalankan

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
belajar this

Process finished with exit code 0
```

## 2. Final

- Final di properti

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        int bil;
        bil = 10;
        System.out.println(bil);

        bil = 5;
        System.out.println(bil);
    }
}
```

Hasilnya adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
10
5

Process finished with exit code 0
```

Jika ditambahkan final pada properti. Maka property tersebut hanya bisa digunakan satu kali. Lihat tanda merah pada penggunaan yang kedua.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        final int bil;

        bil = 10;
        System.out.println(bil);

        bil = 5;
        System.out.println(bil);
    }
}
```

- **Final di Method**

Buat class dengan nama Bangun dengan method luas

```
java x Bangun.java x BangunDatar.java x
package com.komputerkit;

public class Bangun {

    void luas(){
        System.out.println("luas Lingkaran");
    }

}
```

Buat class turunan dari Bangun dengan nama BangunDatar

```
java x Bangun.java x BangunDatar.java x
package com.komputerkit;

public class BangunDatar extends Bangun {

    void luas(){
        System.out.println("luas persegi");
    }
}
```

Buat objek di Main kemudian jalankan

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        BangunDatar b = new BangunDatar();
        b.luas();

    }

}
```

Hasilnya adalah

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
luas persegi

Process finished with exit code 0
```

Sekarang tambahkan final di method luas class Bangun

```
ava ✘ Bangun.java ✘ BangunDatar.java ✘  
package com.komputerkit;  
  
public class Bangun {  
  
    final void luas(){  
        System.out.println("luas lingkaran");  
    }  
  
}
```

Perhatikan tanda merah yang menunjukan error pada class BangunDatar

```
ava ✘ Bangun.java ✘ BangunDatar.java ✘  
package com.komputerkit;  
  
public class BangunDatar extends Bangun {  
  
    void luas(){  
        System.out.println("luas persegi");  
    }  
}
```

### Kesimpulan

Dengan menggunakan final maka method tidak dapat di overriding atau digunakan kembali pada class lain.

- **Final pada class**

Pada class Bangun tambahkan final, perhatikan error yang tampil pada class BangunDatar.

```
ava ✘ Bangun.java ✘ BangunDatar.java ✘  
package com.komputerkit;  
  
public final class Bangun {  
  
    void luas(){  
        System.out.println("luas lingkaran");  
    }  
  
}
```

### Kesimpulan

Class yang diberi final tidak bisa diturunkan atau di extends

### 3. Super

Buat class Bangun

```
ava ✘ Bangun.java ✘ BangunDatar.java ✘
package com.komputerkit;

public class Bangun {

    String kata="belajar super";

    void luas(){
        System.out.println("luas lingkaran");
    }

}
```

Buat class turunan dengan BangunDatar

```
ava ✘ Bangun.java ✘ BangunDatar.java ✘
package com.komputerkit;

public class BangunDatar extends Bangun {

    String kata = "KomputerKit";

    void luas(){

        super.luas();

        System.out.println("luas persegi");

        System.out.println(kata);

        System.out.println(super.kata);
    }
}
```

Buat objek dari class BangunDatar

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        BangunDatar b = new BangunDatar();

        b.luas();
    }
}
```

Jalankan programnya

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
luas lingkaran
luas persegi
KomputerKit
belajar super
```

```
Process finished with exit code 0
```

### Kesimpulan

Jika ada nama properti atau method yang sama di dalam class induk dan class anak, kita bisa menggunakan super untuk memanggil atau menggunakan properti atau method di class induk.

## 30. Try catch

Try catch digunakan agar jika terjadi error atau kesalahan saat program dijalankan, maka kesalahan atau error tersebut tidak ditampilkan. Yang ditampilkan adalah pesan yang di tulis dibawah catch. Contoh jika ada integer dibagi dengan NOL maka akan eror. Agar error tidak ditampilkan maka proses pembagian di letakan dibawah try

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int a, b;
        a = 0;
        try {
            b = 5 / a;
            System.out.println(b);

        } catch (Exception e){
            System.out.println("bilangan salah");
        }
    }
}
```

### 31. Try Catch Finally

Try catch finally digunakan jika terjadi error maka error akan diatangkap oleh catch kemudian akan diteruskan ke finally. Biasanya finally digunakan untuk menghentikan proses agar proses tersebut bisa dikeluarkan dari memori. Finally akan selalu dijalankan setelah catch.

Contoh: buat program seperti dibawah dan lihat hasilnya.

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {

        int a = 5;
        int b;

        try {
            b = 5 /0;
            System.out.println(b);

        }catch (Exception e){
            System.out.println(e);

        } finally {
            System.out.println("ada yang salah, aplikasi ditutup");
        }
    }
}
```

Jika dijalankan akan tampil

```
"C:\Program Files\Java\jdk1.8.0_121\bin\java" ...
java.lang.ArithmetricException: / by zero
ada yang salah, aplikasi ditutup

Process finished with exit code 0
```

## 32. Thread

Thread adalah pengaturan method atau constructor yang akan dijalankan ketika sebuah aplikasi berjalan. Thread berguna untuk menangani kebutuhan dari beberapa method yang akan dijalankan secara bersamaan.

Contoh:

Misalnya anda melakukan copy file dalam ukuran yang besar, tentu membutuhkan waktu yang lama. Sambil menunggu proses copy selesai anda menjalankan aplikasi pemutar musik agar tidak jemu menunggu. Proses tersebut disebut dengan **multitasking**. Proses **multitasking** di dalam aplikasi java disebut dengan thread.

Dengan menggunakan thread maka kita bisa menentukan apakah proses akan dijalankan, di hentikan sementara, atau dimatikan.

**Penggunaan thread harus menggunakan class induk thread yang sudah disediakan oleh java. Nama method yang digunakan juga harus mengikuti nama yang sudah ditentukan di class induk.**

Buat program seperti dibawah dengan class anak Bangun dan class induk Thread

```
package com.komputerkit;

public class Bangun extends Thread {

    public void run(){

        for (int a=0; a<5; a++){

            System.out.println(a);

            try {
                Thread.sleep( 2000 );
            } catch (InterruptedException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Pada contoh diatas. Proses thread akan menghentikan looping selama 2000 mili detik atau 2 detik. Setelah 2 detik looping akan dilanjutkan sampai selesai.

Buat class Main dan jalankan. Perhatikan outputnya

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        Bangun b = new Bangun();
        b.run();
    }
}
```

Penulisan thread menggunakan implements

```
package com.komputerkit;

public class Bangun implements Runnable {
    private Thread t;
    private String namaThread;

    Bangun(String nama){
        namaThread = nama;
        System.out.println("nama thread "+nama);
    }
    public void run() {
        System.out.println("thread jalan .. "+namaThread);
        for (int a=4; a>0; a--){
            System.out.println(namaThread +" ke :"+ a);
            try {
                Thread.sleep( 5000 );
            } catch (InterruptedException e) {
                System.out.println(namaThread + a + e);
            }
        }
        System.out.println("Thread "+namaThread);
    }

    public void start(){
        System.out.println("start "+namaThread);
        if (t == null){
            t = new Thread( runnable: this, namaThread);
            t.start();
        }
    }
}
```

```
package com.komputerkit;

public class Main {

    public static void main(String[] args) {
        Bangun b = new Bangun( nama: "lingkaran");
        b.start();

        Bangun c = new Bangun( nama: "persegi");
        c.start();
    }
}
```

Setelah dijalankan akan tampil

```
nama thread lingkaran
start lingkaran
nama thread persegi
start persegi
thread jalan .. lingkaran
lingkaran ke :4
thread jalan .. persegi
persegi ke :4
lingkaran ke :3
persegi ke :3
lingkaran ke :2
persegi ke :2
lingkaran ke :1
persegi ke :1
Thread lingkaran
Thread persegi

Process finished with exit code 0
```

Penggunaan implements dan extends akan menghasilkan output yang sama.

## **Self Assessment (Penilaian Diri Sendiri)**

Penilaian sendiri adalah penilaian yang anda lakukan sendiri untuk mengukur sejauh mana keberhasilan anda dalam belajar. Berikut adalah aturan penilaian yang kami siapkan.

Isikan poin pada setiap bagian kompetensi penilaian dengan aturan sebagai berikut:

<b>POIN</b>	<b>ATURAN</b>
0	<i>Anda belum pernah mengikuti atau belajar sama sekali</i>
1	<i>Pernah belajar atau mengikuti pembelajaran namun belum bisa</i>
2	<i>Pernah belajar dan bisa mengikuti namun harus di bimbing oleh instruktur dan melihat panduan, atau masih melihat video pembelajaran.</i>
3	<i>Bisa mengerjakan sendiri tanpa dibimbing instruktur, tanpa melihat panduan, dan tanpa melihat video pembelajaran</i>
4	<i>Materi baru yang bisa dikuasai sendiri tanpa bantuan instruktur, cukup dengan browsing di google atau youtube.</i>

### **KOMPETENSI PENILAIAN**

<b>NO</b>	<b>KOMPETENSI</b>	<b>POIN ANDA</b>
1	<i>Konsep Program Komputer</i>	
2	<i>Kenapa harus Java?</i>	
3	<i>Program pertama</i>	
4	<i>Perbedaan Aplikasi Executable dan Aplikasi Bytecode</i>	
5	<i>Variabel dan Tipe data</i>	
6	<i>Input data menggunakan Scanner</i>	
7	<i>Operator</i>	
8	<i>Pengujian</i>	
9	<i>Pengujian Ganda</i>	
10	<i>Pengujian Bersarang (IF di dalam IF)</i>	
11	<i>Pemilihan (Selection)</i>	
12	<i>Pengulangan (Looping)</i>	

13	<i>Pengulangan Bersarang (Nested Loop)</i>
14	<i>Array (Larik / Rantang)</i>
15	<i>Method (Blok Program)</i>
16	<i>Properti, Method, Class</i>
17	<i>Fungsi Static Pada Method</i>
18	<i>Konstruktor (method yang langsung dijalankan ketika objek dibuat)</i>
19	<i>Inheritance (Pewarisan)</i>
20	<i>Abstract Class</i>
21	<i>Encapsulasi (Public, Private, protected)</i>
22	<i>Polimorfisme (Nama Method yang sama dengan tugas yang berbeda )</i>
23	<i>Overriding</i>
24	<i>Overloading</i>
25	<i>Interface</i>
26	<i>Enumerasi (tipe data bentukan)</i>
27	<i>Java SWING (Form java)</i>
28	<i>Collection</i>
29	<i>Keyword this, final, super</i>
30	<i>Try catch</i>
31	<i>Try Catch Finally</i>
32	<i>Thread</i>

Setelah di isi semua poin penilaian nya, jumlahkan kemudian **dibagi** dengan jumlah kompetensi, jumlah kompetensi diatas adalah 32. Hasil pembagiannya sesuaikan dengan tabel berikut:

HASIL	PREDIKAT
<= 2,5	<i>Belum LULUS</i>
2,6 – 3,0	<i>LULUS</i>
3,1 – 3,5	<i>BAIK.</i>
3,6 – 4	<i>HEBAT</i>

## **DAFTAR PUSTAKA**

1. <https://docs.oracle.com/javase/tutorial/>
2. <https://www.tutorialspoint.com/java/>
3. <https://www.javatpoint.com/java-tutorial>
4. [https://www.youtube.com/watch?v=SSmB\\_HA0edc&list=PLonJJ3BVjZW6\\_q8gh7XoLUIhRlyBcYJLP](https://www.youtube.com/watch?v=SSmB_HA0edc&list=PLonJJ3BVjZW6_q8gh7XoLUIhRlyBcYJLP)
5. <https://www.youtube.com/watch?v=bwUWEioRlgE&list=PLlxmoA0rQ-LzrFmbcHy2vUYlr9ZRo83yv>
6. <https://dedykuncoro.com/2013/03/macam-macam-tipe-data-pada-java.html>
7. [http://jagocoding.com/tutorial/1140/Belajar\\_Membuat\\_Thread\\_di\\_Java](http://jagocoding.com/tutorial/1140/Belajar_Membuat_Thread_di_Java)
8. Java\_A Beginner's Guide\_Create, Compile, and Run Java Programs Today (6th ed.)  
[Schildt 2014] (badly formatted)