



Modul #2

# *Class, Object, Method.*

-CLASS, OBJECT, & METHOD-

## DAFTAR ISI

<b>A. PENDAHULUAN</b>	1
1. Identitas Modul	1
2. Tujuan Pembelajaran	1
3. Deskripsi Materi	1
<b>B. DASAR TEORI</b>	2
1. Konsep <i>Class</i> , <i>Object</i> , dan <i>Method</i> dalam Java	2
2. Deklarasi <i>Class</i> dan <i>Method</i> dalam Java	5
3. Pembuatan <i>Object</i> dari <i>Class</i> dan Pemanggilan <i>Method</i> dalam <i>Class</i>	6
4. <i>Input</i> dan <i>Output</i> dalam Pemrograman Java	6
<b>C. TUGAS PRAKTIKUM</b>	8
1. Menerapkan Penggunaan <i>Class</i> , <i>Object</i> , dan <i>Method</i>	8
2. Menerapkan Penggunaan Variabel Global dan Variabel Local	10
3. Input Karakter dan <i>String</i> dengan <i>Class</i> <i>BufferedReader</i>	11
4. Input Bilangan dengan <i>Class</i> <i>BufferedReader</i>	12
5. Input Data dengan <i>Class</i> <i>Scanner</i>	12
<b>D. EVALUASI</b>	14
1. Program Mencetak Kata	14
2. Program Konversi Nilai	14

## A. PENDAHULUAN

### Kompetensi Dasar:

Membuat *Class* dan *Object* dalam Java

#### 1. Identitas Modul

Mata Pelajaran : Pemrograman Berorientasi Objek

Kelas : XI

Judul Modul : Class, Object, dan Method dalam Java

#### 2. Tujuan Pembelajaran

Setelah mempelajari bab ini diharapkan siswa akan mampu :

- a Menerapkan konsep class, object, dan method dalam Java.
- b Mendeklarasikan class dan method dalam Java.
- c Membuat object dari class.
- d Memanggil method dalam class.

#### 3. Deskripsi Materi

Modul 2 yang berjudul *Class, Object, dan Method* dalam Java ini memaparkan tentang bagaimana cara mendeklarasikan *class*, membuat *object* dari suatu *class* serta membuat dan memanggil *method*. Di sini pembuatan program sudah mulai dirancang untuk dapat menerima *input* dan menampilkan *output* sesuai dengan data yang dimasukkan oleh *user*.

Agar saat dijalankan program dapat menerima *input* dari luar kode program, maka harus meng-*import class* Scanner dari *package* java.util atau *class* BufferedReader dari *package* java.io di bagian paling atas kode program sesuai dengan prosedur yang dipaparkan dalam modul.

## B. DASAR TEORI

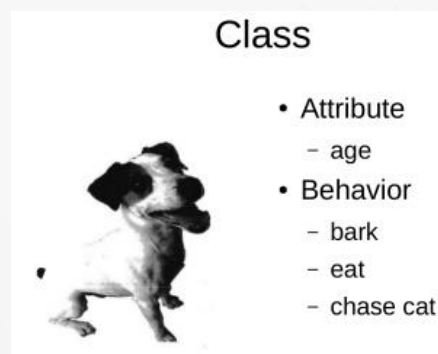
### 1. Konsep *Class*, *Object*, dan *Method* dalam Java

*Class* (kelas) merupakan wadah yang akan digunakan untuk menciptakan *object* sehingga sebelum membuat *object* harus membuat *class* terlebih dahulu. *Object* (objek) adalah sekumpulan data dalam program yang terdiri dari variabel dan *method* terkait. *Object* saling berinteraksi dengan cara saling memanggil *method* dari satu *object* ke *object* lainnya yang disebut message. Dengan kata lain, *object* merupakan *instance* sebenarnya dari sebuah *class*. *Instance* dibuat saat dilakukan inisialisasi *class* dengan menggunakan kata kunci *new*, sedangkan *method* adalah prosedur atau fungsi yang dimiliki oleh sebuah *object*. Karena pada dasarnya merupakan prosedur atau fungsi, maka *method* ini akan mengolah atau mengubah data atau variabel yang ada di dalam *object* sesuai dengan operasi yang telah ditentukan. *Method* disebut juga behaviour.

Di dalam *class* juga sering dideklarasikan berbagai variabel yang memiliki ruang lingkup yang berbeda. Ruang lingkup atau cakupan variabel (*variable scope*) ini menentukan seberapa luas variabel dapat diakses. Cakupan variabel ini terbagi menjadi dua, yaitu *global* dan *local*. Variabel *global* merupakan variabel yang ada di dalam *class* dan dapat diakses oleh semua *method* dalam *class*, sedangkan variabel *local* merupakan variabel yang ada di dalam *method* dari suatu *class* sehingga hanya dapat diakses oleh *method* itu sendiri.

#### a. **Class**

*Class* merupakan blueprint/rancangan dari suatu objek. *Class* adalah gambaran umum dari suatu objek. Dikatakan bahwa Anjing adalah *class*, maka *class* Anjing memiliki karakteristik/*atribute* dan perilaku/*behavioral* yang dimiliki oleh anjing pada umumnya. Untuk lebih jelasnya perhatikan ilustrasi berikut:



Ilustrasi diatas memperlihatkan bahwa *class* Dog memiliki *attribute* (*age*) dan *behavioral* (*bark*, *eat*, dan *chase cat*). *Attribute* dan *behavioral* tersebut umum dimiliki oleh anjing sehingga setiap objek yang memiliki *attribute* dan *behavioral* tersebut dikategorikan sebagai anjing. Di Java, untuk mendeklarasikan *class* menggunakan kata kunci "class" yang diikuti dengan nama *class*-nya.

```

1 package dog;
2 public class Dog {
3     int age;
4     void bark() {
5         System.out.println("anjing menggongong");
6     }
7     void eat() {
8         System.out.println("anjing makan");
9     }
10    void chaseCat() {
11        System.out.println("anjing mengejar kucing");
12    }
13 }

```

Pada baris 1 dideklarasikan "class Dog". Class Dog ini memiliki atribut "age" dan behavioral "bark", "eat", dan "chaseCat" yang dideklarasikan di dalam kurung kurawal. Atribut dan behavioral tersebut merupakan anggota/member dari class. Penjelasan detail tentang atribut dan behavioral dibahas pada bahasan berikutnya.

Penamaan class mengikuti aturan penamaan seperti variable namun ada perbedaan sedikit, yaitu setiap kata harus diawali dengan huruf kapital/huruf besar. Misal, class "mydog" terdiri dari kata "my" dan "dog" maka huruf "m" pada "my" huruf kapital dan huruf "d" pada "dog" harus kapital juga. Dari "mydog" menjadi "MyDog". Hal ini dimaksud supaya nama suatu class itu mudah untuk dibaca.

## b. Method

Method merupakan sebutan untuk *behavioral/function* di Java. Method selalu memiliki kurung lengkung atau "()", kurung lengkung tersebut bisa juga disemati suatu variable atau parameter. Parameter sendiri adalah sebutan dari variabel yang terletak dalam kurung lengkung suatu method. Aturan penamaan method sama dengan aturan penamaan variable. Pembahasan tersebut bisa dilihat di penjelasan tentang variabel. Method sendiri ada dua jenis: (1) void dan (2) return.

### 1. void Method

void method adalah method yang tidak mengembalikan suatu nilai. Dilihat sekilas, void method dapat diidentifikasi dengan adanya kata kunci "void" di depan nama method.

```

void eat() {
    System.out.println("anjing makan");
}

```

Ilustrasi di atas adalah method void karena secara kasat mata method tersebut di depan nama method-nya (eat) memiliki kata kunci "void". Void bisa disebut dengan method tidak mengembalikan nilai karena method ini ketika dipanggil tidak memiliki nilai yang bisa disimpan di suatu variabel.



## 2. return Method

return method adalah method yang mengembalikan nilai. Method ini bisa diidentifikasi dengan adanya data type di depan nama method-nya dan kata kunci return di dalam method-nya.

```
int getAge() {  
    return 3;  
}
```

Pada baris 1 di depan nama method, "getAge" terdapat data type "int" sehingga di dalam method ini harus mengembalikan "int" juga. Kata kunci "return" digunakan untuk mengembalikan nilai. Dilihat pada baris 2, method "getAge" ini mengembalikan nilai 3, nilai 3 tersebut adalah nilai yang ber-data type "int".

## c. Method

Object adalah representasi dari class. Katakan bahwa Anjing adalah class-nya maka doggy, pretty, dan sweety adalah objectnya. Class masih berupa blueprint/rancangan sedangkan object adalah wujud nyata. Cara mendeklarasikan object sebagai berikut:



Untuk membuat object terlebih dahulu harus tahu class yang akan dibuatkan object-nya. Pada ilustrasi diatas "Dog" adalah class yang akan dibuatkan object-nya. Object dari class "Dog" tersebut bernama "pretty". Berikutnya, untuk benar-benar "pretty" adalah object dari class "Dog" maka pada ilustrasi diatas ditambahkan assignment operator (=) yang digunakan untuk memberikan nilai object "pretty" ini dengan object baru dari class Dog, yaitu caranya dengan menambahkan kata kunci "new" yang diikuti dengan "constructor Dog" / "Dog()", constructor akan dibahas pada bahasan berikutnya. Sedangkan, penamaan object mengikuti penamaan pada variable. Sekali object tersebut dibuat, semua anggota (attribute dan method) bisa diakses oleh object tersebut.

Source Code Dog.java

```

1 package dog;
2 public class Dog {
3     int age;
4
5     int getAge(){
6         return 3;
7     }
8     void bark(){
9         System.out.println("anjing menggongong");
10    }
11    void eat(){
12        System.out.println("anjing makan");
13    }
14    void chaseCat(){
15        System.out.println("anjing mengejar kucing");
16    }
17 }

```

Source Code DogTest.java

```

1 package dog;
2 public class DogTest {
3     public static void main(String [] args){
4         Dog pretty = new Dog();
5         pretty.bark();
6         pretty.eat();
7         pretty.chaseCat();
8     }
9 }

```

Ada dua file java yang saling terkait, yaitu Dog.java dan DogTest.java. Dog.java adalah blueprint dari Dog sedangkan DogTest.java adalah penerapan blueprint tersebut dalam bentuk object. Pada file DogTest.java baris 5 menunjukkan sistem membuat object baru, pretty, dari class Dog. Sekali object tersebut dibuat, semua member (attribute / method) bisa diakses. Pada baris 6 – 8 menunjukkan cara mengakses member Dog, yaitu dengan cara memanggil nama object-nya diikuti dengan titik beserta member yang ingin diakses. Apabila program tersebut di jalankan, maka hasilnya menjadi:

Output - Dog (run) X	Action Items
run:	
anjing menggongong	
anjing makan	
anjing mengejar kucing	
BUILD SUCCESSFUL (total time: 0 seconds)	

## 2. Deklarasi Class dan Method dalam Java

Class dideklarasikan dengan cara sebagai berikut:

```

modifier class nama_class
{
    /*body dari class
    deklarasi atribut

```

```
    deklarasi konstruktor
    deklarasi method*/
}
```

*Method dideklarasikan dengan cara sebagai berikut:*

```
modifier type nama_method(parameter_input)
{
    //body dari method
}
```

### 3. Pembuatan *Object* dari *Class* dan Pemanggilan *Method* dalam *Class*

*Object* dibuat dengan cara sebagai berikut:

```
nama_class nama_objek = new nama_class();
```

*Pemanggilan method* dilakukan dengan cara sebagai berikut:

```
nama_objek.nama_method();
```

### 4. *Input* dan *Output* dalam Pemrograman Java

*Input* data oleh user dan menampilkan *output* pada layar dapat dilakukan dengan dua cara, yaitu dengan menggunakan *class* *BufferedReader* dari *package* *java.io* dan *class* *Scanner* dari *package* *java.util*.

Hal-hal yang perlu diperhatikan saat menggunakan *class* *BufferedReader*:

- Meng-import *package* *java.io* dengan cara menuliskan kode berikut ini di bagian paling atas kode program:

```
import java.io.*;
```

- Membuat objek dari *class* *BufferedReader* yang dapat terhubung dengan keyboard sehingga dapat membaca input dari user dengan cara menambahkan statement berikut ini:

```
BufferedReader bufReader = new BufferedReader(new
    InputStreamReader(System.in));
```

atau

```
InputStreamReader insReader = new InputStreamReader(System.in);
BufferedReader bufReader = new BufferedReader(insReader);
```

- Jika data yang dimasukkan berupa bilangan, maka harus mendeklarasikan variabel string temporary dan menggunakan fungsi *readLine()* di dalam blok *try-catch* untuk membaca input:

```
try {
```



```
        temporary = bufReader.readLine();  
        //kode program yang mungkin mengalami kesalahan  
    }  
    catch(IOException exc)  
    {  
        //menangkap kesalahan yang terjadi  
        //memberikan pesan kesalahan  
    }  
}
```

Hal-hal yang perlu diperhatikan saat menggunakan class Scanner:

- a. Meng-import package java.util dengan cara menuliskan kode berikut ini di bagian paling atas kode program:

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

- b. Membuat object dari class Scanner dengan cara seperti berikut ini:

```
Scanner inScanner = new Scanner(System.in);
```

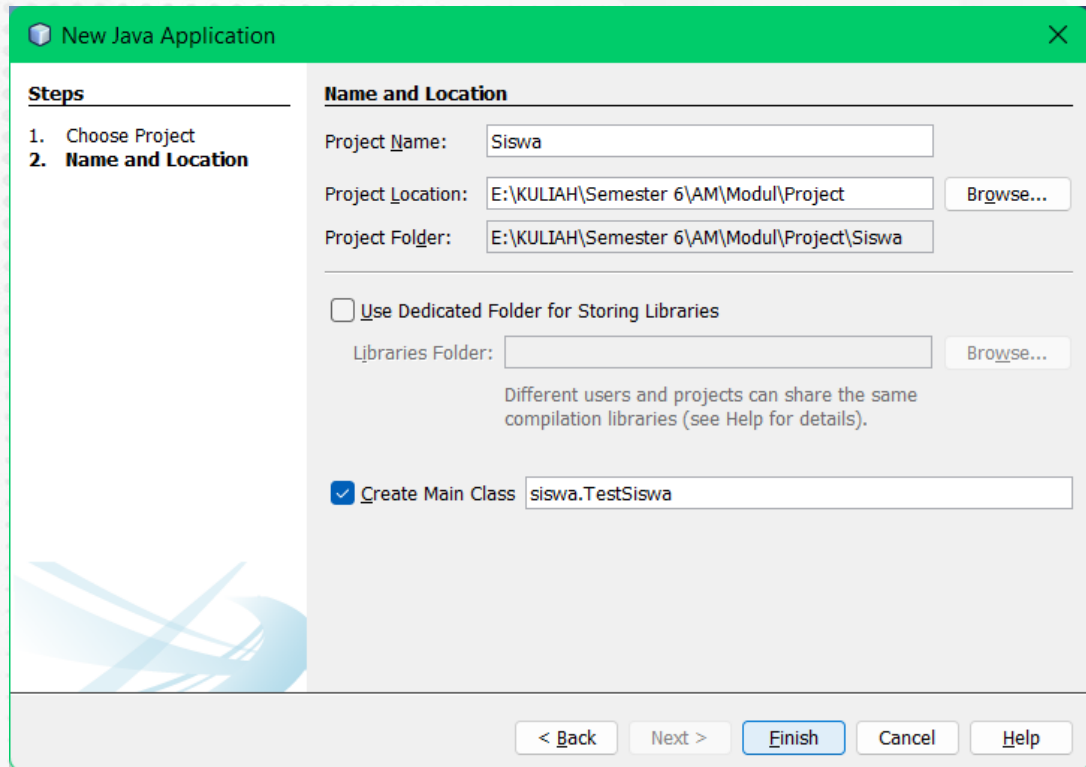
- c. Membaca dan menyimpan input dari keyboard ke dalam variabel dengan cara seperti berikut ini:

```
kata = inScanner.nextLine(); //jika input bertipe data string  
bilBulat = inScanner.nextInt(); //jika input bertipe data  
integer  
bilReal = inScanner.nextDouble(); //jika input bertipe data  
double
```

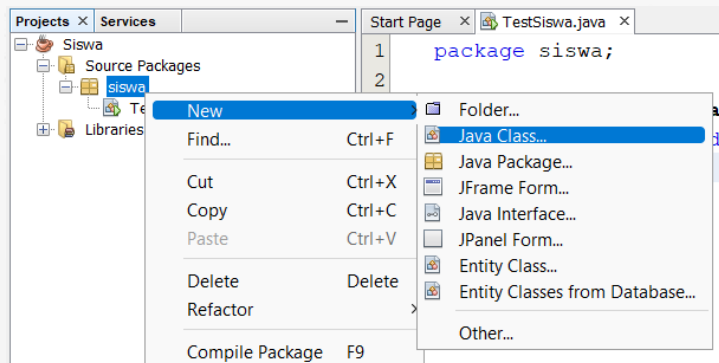
## C. TUGAS PRAKTIKUM

### 1. Menerapkan Penggunaan *Class*, *Object*, dan *Method*

Buatlah Project Baru di Netbeans dengan nama siswa, kemudian untuk Main Class bernama siswa.TestSiswa

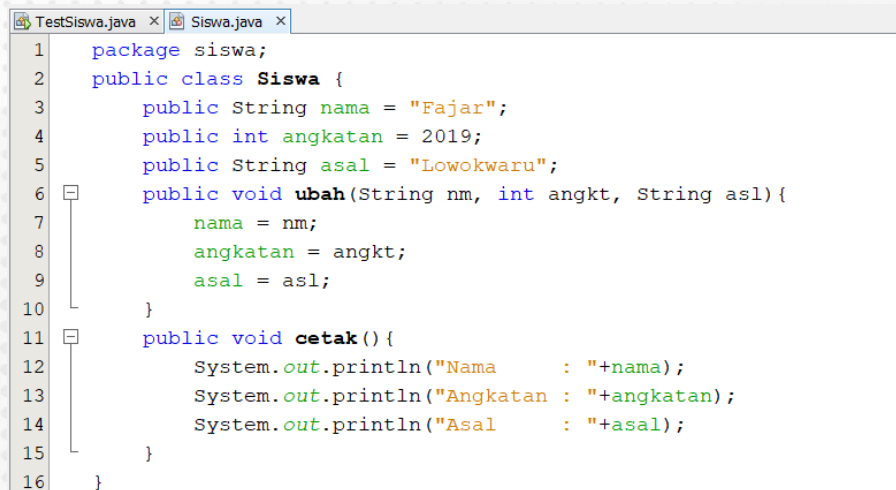


Kemudian buatlah sebuah class baru dengan nama siswa



Tuliskan dan simpan kode program di bawah ini:

Kode program class Siswa.java

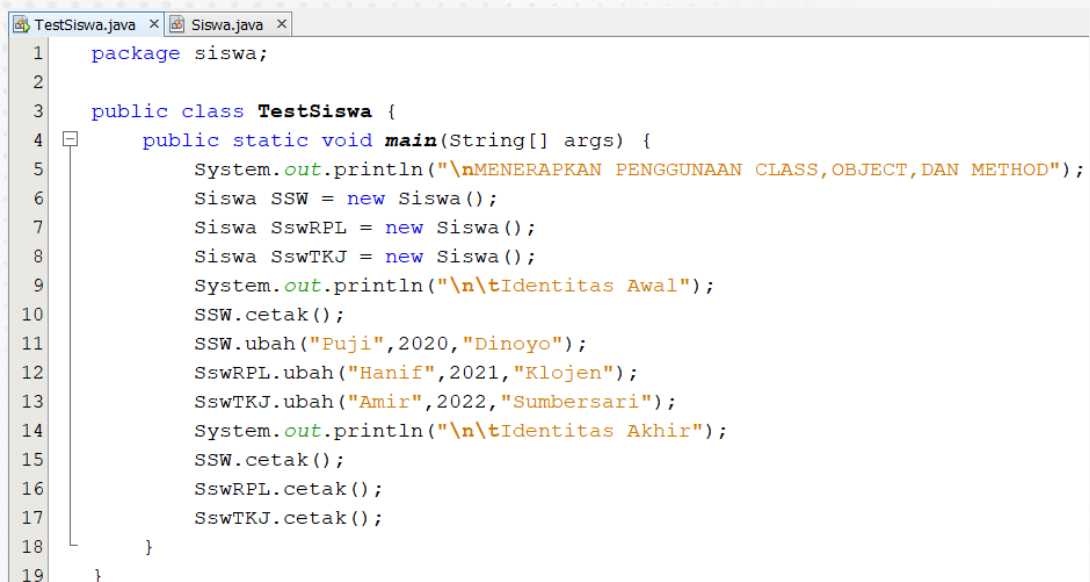


```

1 package siswa;
2 public class Siswa {
3     public String nama = "Fajar";
4     public int angkatan = 2019;
5     public String asal = "Lowokwaru";
6     public void ubah(String nm, int angkt, String asl){
7         nama = nm;
8         angkatan = angkt;
9         asal = asl;
10    }
11    public void cetak(){
12        System.out.println("Nama      : "+nama);
13        System.out.println("Angkatan : "+angkatan);
14        System.out.println("Asal      : "+asal);
15    }
16 }

```

Kode program class TestSiswa.java



```

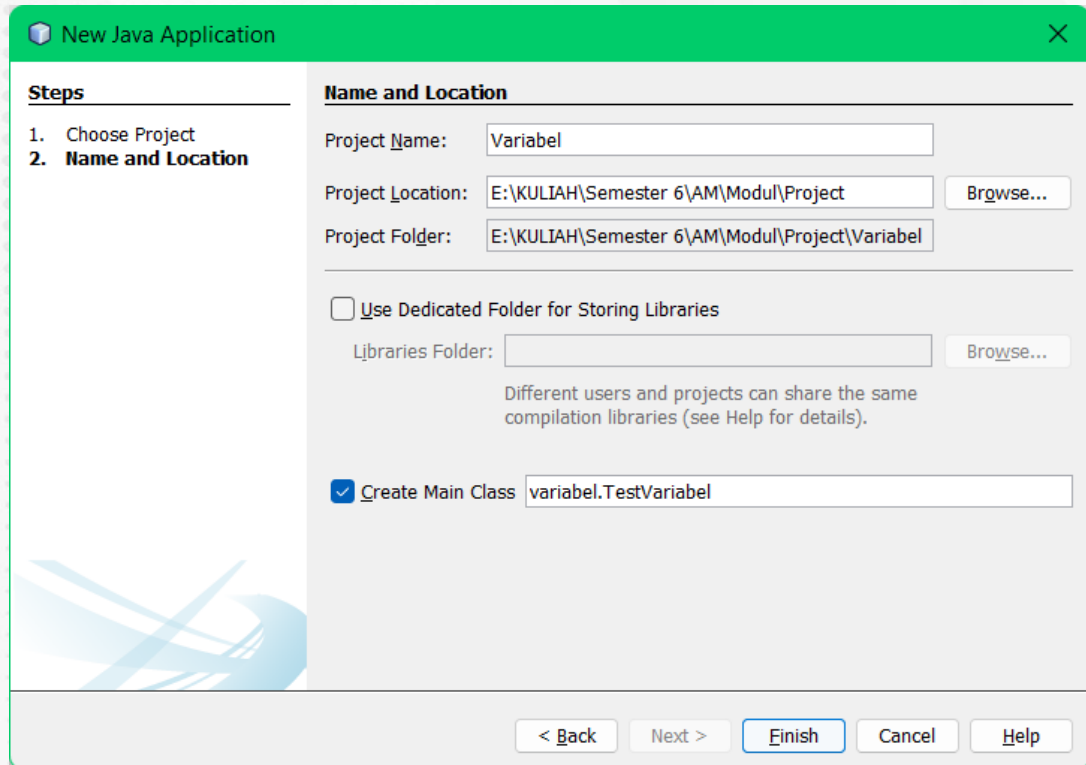
1 package siswa;
2
3 public class TestSiswa {
4     public static void main(String[] args) {
5         System.out.println("\nMENERAPKAN PENGGUNAAN CLASS, OBJECT, DAN METHOD");
6         Siswa SSW = new Siswa();
7         Siswa SswRPL = new Siswa();
8         Siswa SswTKJ = new Siswa();
9         System.out.println("\n\tIdentitas Awal");
10        SSW.cetak();
11        SSW.ubah("Puji", 2020, "Dinoyo");
12        SswRPL.ubah("Hanif", 2021, "Klojen");
13        SswTKJ.ubah("Amir", 2022, "Sumbersari");
14        System.out.println("\n\tIdentitas Akhir");
15        SSW.cetak();
16        SswRPL.cetak();
17        SswTKJ.cetak();
18    }
19 }

```

- Lakukan kompilasi dan eksekusi program terhadap class Siswa kemudian tunjukkan hasilnya!
- Lakukan kompilasi dan eksekusi program terhadap class TestSiswa kemudian tunjukkan hasilnya!
- Mengapa saat kompilasi program terhadap class Siswa tidak terjadi *error* tetapi eksekusi programnya tidak dapat menampilkan data dari class tersebut? Mengapa data dalam class Siswa baru tampil saat eksekusi program dilakukan pada class TestSiswa?
- Lakukan modifikasi kode program untuk membuat dua *object* lagi dengan nama SswRPL dan SswTKJ! Lakukan perubahan data, panggil *method* ubah dan *method* cetak! Tunjukkan hasil modifikasi kode program yang telah Anda lakukan!
- Lakukan kompilasi dan eksekusi program terhadap class yang telah Anda modifikasi kemudian tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program ini!

## 2. Menerapkan Penggunaan Variabel Global dan Variabel Local

Buatlah Project Baru di Netbeans dengan nama `Variabel`, kemudian untuk Main Class bernama `siswa.TestVariabel`



Kemudian buatlah sebuah class baru dengan nama `Variabel`

Tuliskan dan simpan kode program di bawah ini:

Kode program class `Variabel.java`

```
TestVariabel.java x Variabel.java x
1  package variabel;
2  public class Variabel {
3      String sifat = "Pintar";
4      void infoVariabel() {
5          String sifat = "Rajin";
6          System.out.println("\nMenampilkan Variabel Global");
7          System.out.println("Sifat milik class : "+this.sifat);
8          System.out.println("\nMenampilkan Variabel Local");
9          System.out.println("Sifat pada method : "+sifat);
10     }
11 }
```

Kode program class `TestVariabel.java`

```

TestVariabel.java x Variabel.java x
1 package variabel;
2 public class TestVariabel {
3     public static void main(String[] args) {
4         System.out.println
5             ("\nMENERAPKAN. PENGGUNAAN. VARIABEL. GLOBAL. DAN. LOCAL");
6         Variabel info = new Variabel();
7         info.infoVariabel();
8     }
9 }

```

- Amati *folder* penyimpanan file.java! Lakukan kompilasi terhadap *class* TestLingkup! File.class apa saja yang terbentuk? Jelaskan mengapa bisa terbentuk file.class tersebut!
- Lakukan eksekusi program dan tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program ini!
- Apa fungsi dari *keyword* **"this"**?

### 3. Input Karakter dan *String* dengan *Class* *BufferedReader*

Buatlah Project Baru di Netbeans dengan nama InputKarakter, kemudian untuk Main Class bernama inputkarakter.InputKarakter  
Tulislah dan simpan kode program di bawah ini:

```

InputKarakter.java x
1 package inputkarakter;
2 import java.io.*;
3
4 public class InputKarakter {
5     public static void main(String[] args) {
6         BufferedReader bufReader = new BufferedReader
7             (new InputStreamReader(System.in));
8         char kar = ' ';
9         String kata = "";
10        System.out.println("\n\tINPUT KARAKTER DAN STRING\n");
11        try{
12            System.out.print("Inputkan karakter : ");
13            kar = (char) bufReader.read();
14            bufReader.readLine();
15            System.out.print("Inputkan string : ");
16            kata = bufReader.readLine();
17        }
18        catch(IOException exc){
19            System.out.println("\nError...!!!");
20        }
21        System.out.println("\n\tKARAKTER & STRING YANG.DIINPUTKAN\n");
22        System.out.println("Karakter : "+kar);
23        System.out.println("String : "+kata);
24    }
25 }

```



- a. Lakukan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

#### 4. Input Bilangan dengan Class **BufferedReader**.

Buatlah Package baru di Netbeans dengan nama `inputbilangan`, kemudian untuk Main Class bernama `InputBilangan`

Tulislah dan simpan kode program di bawah ini:

```
InputBilangan.java x
1 package inputbilangan;
2 import java.io.*;
3
4 public class InputBilangan {
5     public static String readInput() {
6         String temporary="";
7         InputStreamReader insReader = new InputStreamReader(System.in);
8         BufferedReader bufReader = new BufferedReader(insReader);
9         try{
10             temporary = bufReader.readLine();
11         }
12         catch(IOException e){
13             System.out.println("\nError...!!!");
14         }
15         return temporary;
16     }
17     public static void main(String[] args) {
18         int bilBulat=0;
19         double bilReal=0;
20         System.out.println("\n\tINPUT BILANGAN BULAT DAN BILANGAN REAL\n");
21         System.out.print("Inputkan Bilangan Bulat : ");
22         bilBulat = Integer.parseInt(readInput());
23         System.out.print("Inputkan Bilangan Real : ");
24         bilReal = Double.parseDouble(readInput());
25         System.out.println("\n\tBILANGAN YANG DIINPUTKAN\n");
26         System.out.println("Bilangan bulat : "+bilBulat);
27         System.out.println("Bilangan real : "+bilReal);
28     }
29 }
```

- a. Lakukan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

#### 5. Input Data dengan Class **Scanner**

Buatlah Package baru di Netbeans dengan nama `inputscanner`, kemudian untuk Main Class bernama `InputScanner`

Tulislah dan simpan kode program di bawah ini:

```

1 package inputscanner;
2 import java.util.Scanner;
3
4 public class InputScanner {
5     public static void main(String[] args) {
6         Scanner inScanner = new Scanner(System.in);
7         String kata;
8         int bilBulat;
9         double bilReal;
10        System.out.println("\n\tINPUT DATA DENGAN CLASS SCANNER\n");
11        System.out.print("Inputkan bilangan bulat : ");
12        bilBulat=inScanner.nextInt();
13        System.out.print("Inputkan bilangan real : ");
14        bilReal=inScanner.nextDouble();
15        inScanner.nextLine();
16        System.out.print("Inputkan String : ");
17        kata=inScanner.nextLine();
18        System.out.println("\n\tDATA YANG DIINPUTKAN\n");
19        System.out.println("Bilangan Bulat : "+bilBulat);
20        System.out.println("Bilangan Real : "+bilReal);
21        System.out.println("String : "+kata);
22    }
23 }

```

- a. Lakukan eksekusi program kemudian tunjukkan hasilnya!
- b. Berikan penjelasan terkait jalannya program ini!

## D. EVALUASI

### 1. Program Mencetak Kata

- Buatlah sebuah program berbasis *console* untuk mencetak kata yang dimasukkan oleh *user* sebanyak keinginan *user* dengan memanfaatkan perulangan! Gunakan *class Scanner*! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program yang Anda buat!

Contoh hasil eksekusi program:

```
PROGRAM MENCETAK KATA

Kata apa yang ingin Anda Tampilkan ? : Ayo Semangat Belajar JAVA!
Berapa kali ingin Anda tampilkan?    : 9

DATA YANG DIINPUTKAN
Mencetak kata Ayo Semangat Belajar JAVA! sebanyak 9 kali.

HASIL PERULANGAN KATA
1 Ayo Semangat Belajar JAVA!
2 Ayo Semangat Belajar JAVA!
3 Ayo Semangat Belajar JAVA!
4 Ayo Semangat Belajar JAVA!
5 Ayo Semangat Belajar JAVA!
6 Ayo Semangat Belajar JAVA!
7 Ayo Semangat Belajar JAVA!
8 Ayo Semangat Belajar JAVA!
9 Ayo Semangat Belajar JAVA!
```

### 2. Program Konversi Nilai

- Buatlah sebuah program berbasis *console* untuk melakukan konversi nilai dari angka ke huruf dengan memanfaatkan operasi kondisi, di mana nilai yang diolah merupakan input dari *user*, bukan dari kode program! Gunakan *class BufferedReader*! Lakukan kompilasi dan eksekusi program kemudian tunjukkan hasilnya!
- Berikan penjelasan terkait jalannya program yang Anda buat!

Contoh hasil eksekusi program:

```
PROGRAM KONVERSI NILAI
Masukkan Nama           : Hanif
Masukkan Nilai Matematika : 90
Masukkan Nilai Fisika    : 88
Masukkan Nilai Biologi   : 94
*****
Nama                     : Hanif

Nilai yang diperoleh
Matematika : 90.0
Fisika     : 88.0
Biologi    : 94.0
Rerata Nilai : 90.666664
Nilai Huruf : A
*****
```