

Nama : Kelvianto Pratama Harum

NIM : 200210500016

MK : Pemrograman Lanjut

Dosen : Muhammad Fajar B, S.Pd., M.Cs.

PERTEMUAN III

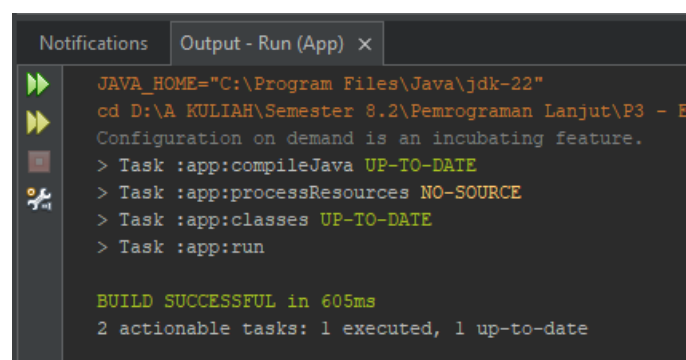
ENKAPSULASI

A. Contoh 3.1 – Method

Source code:

```
1 package com.example.Pertemuan3;
2
3 class mahasiswa {
4     private int nim;
5     public void setNim(int n) {
6         nim=n;
7     }
8 }
9
10 public class App {
11     public static void main(String[] args) {
12         mahasiswa mhs = new mahasiswa();
13         mhs.setNim(5122);
14     }
15 }
```

Output:



The screenshot shows the 'Output - Run (App)' window in an IDE. It displays the following text:

```
JAVA_HOME="C:\Program Files\Java\jdk-22"
cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 - E
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE
> Task :app:run
BUILD SUCCESSFUL in 605ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **App.java** bergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **mahasiswa1**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **8**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **mahasiswa1**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **nim** dengan tipe data **integer**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan method tanpa nilai balik (**void**) dengan nama **setNim**, dalam method ini parameter yang digunakan adalah variabel **n** dengan tipe data **integer**. Scope method ini mulai dari baris **5** hingga baris **7**. Style coding ini biasa disebut **setter** pada style **setter-getter**.

Pada baris **6**, diberikan nilai pada field **nim** dengan nilai dari argumen variabel **n** pada saat pemanggilan method **setNim**.

Pada baris **10**, dideklarasikan Class bernama **App**, class ini mencakup semua kode yang ada pada baris **10** hingga baris **15**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **App**.

Pada baris **11**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **11** hingga baris **14**.

Pada baris **12**, dibuat sebuah object dari class **mahasiswa** dengan nama **mhs** tanpa argumen konstruktor.

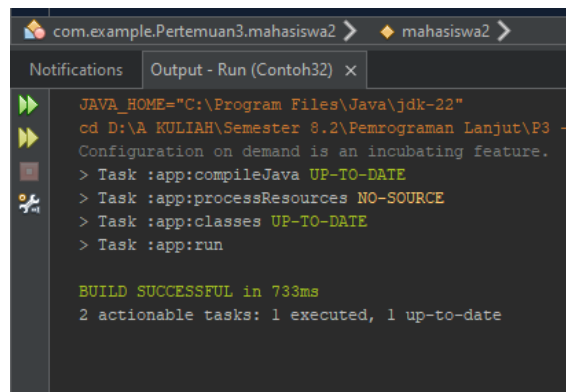
Pada baris **13**, dipanggil method **setNim** pada object **mhs** dengan argumen nilai **5122**. Pemanggilan method ini berarti akan mengubah nilai **field nim** pada object **mhs** dengan nilai **5122**.

B. Contoh 3.2 – Konstruktor

Source code:

```
1 package com.example.Pertemuan3;
2
3 class mahasiswa2 {
4     private int nim;
5     private String nama;
6     public mahasiswa2(int n, String m){
7         nim = n;
8         nama = m;
9     }
10 }
11 public class Contoh32 {
12     public static void main(String args[]){
13         mahasiswa2 mhs = new mahasiswa2(5122, "Kelvin Pratama");
14     }
15 }
```

Output:



```
com.example.Pertemuan3.mahasiswa2 > mahasiswa2 >
Notifications Output - Run (Contoh32) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 -
>> Configuration on demand is an incubating feature.
>> > Task :app:compileJava UP-TO-DATE
>> > Task :app:processResources NO-SOURCE
>> > Task :app:classes UP-TO-DATE
>> > Task :app:run
>>
BUILD SUCCESSFUL in 733ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Contoh32.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **mahasiswa2**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **10**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **mahasiswa2**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **nim** dengan tipe data **integer**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **private** bernama **nama** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **6**, dideklarasikan **konstruktor** untuk class ini (**mahasiswa2**). Konstruktor adalah bagian kode yang paling pertama dipanggil pada saat Object Initialization atau pembentukan object. Konstruktor ini dapat menggunakan parameter atau tidak. Pada dasarnya, konstruktor adalah method yang pertama kali dijalankan pada saat object creation/loading. Konstruktor ini memiliki parameter variabel **n** dengan tipe data **integer**, dan variabel **m** dengan tipe data **String**.

Pada baris **7**, diberikan nilai pada field **nim** dengan nilai dari argumen variabel **n** pada saat konstruktor dijalankan.

Pada baris **8**, diberikan nilai pada field **nama** dengan nilai dari argumen variabel **m** pada saat konstruktor dijalankan.

Pada baris **11**, dideklarasikan Class bernama **Contoh32**, class ini mencakup semua kode yang ada pada baris **11** hingga baris **15**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Contoh32**.

Pada baris **12**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **11** hingga baris **14**.

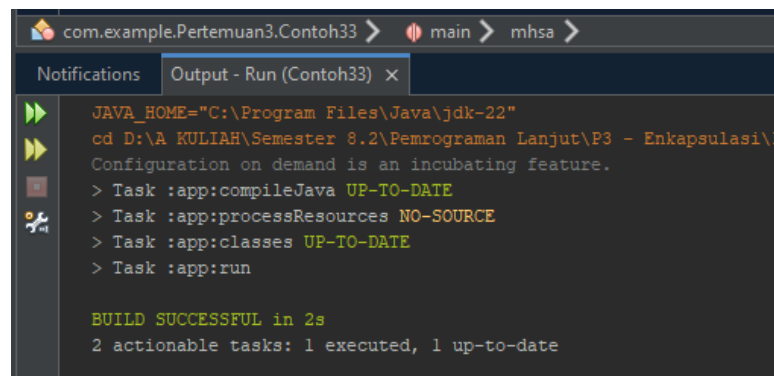
Pada baris **13**, dibuat sebuah object dari class **mahasiswa2** dengan nama **mhs** dan argumen konstruktor nilai **5122** pada parameter **n**, dan nilai **Kelvin Pratama** pada parameter **m**.

C. Contoh 3.3 – Konstruktor

Source code:

```
1 package com.example.Pertemuan3;
2
3 class mahasiswa3 {
4     private int nim;
5     private String nama;
6     public mahasiswa3(String m){
7         nim = 0;
8         nama = "";
9     }
10
11     public mahasiswa3(int n, String m){
12         nim = n;
13         nama = m;
14     }
15 }
16
17 public class Contoh33 {
18     public static void main(String args[]){
19         mahasiswa3 mhsa = new mahasiswa3("Kelvin");
20         mahasiswa3 mhsb = new mahasiswa3(5122, "Kelvin");
21     }
22 }
```

Output:



```
com.example.Pertemuan3.Contoh33 > main > mhsa >
Notifications Output - Run (Contoh33) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 - Enkapsulasi\
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE
> Task :app:run
BUILD SUCCESSFUL in 2s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Contoh33.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **mahasiswa3**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **15**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **mahasiswa3**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **nim** dengan tipe data **integer**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **private** bernama **nama** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **6**, dideklarasikan **konstruktor** untuk class ini (**mahasiswa2**). Konstruktor adalah bagian kode yang paling pertama dipanggil pada saat Object Initialization atau pembentukan object. Konstruktor ini dapat menggunakan parameter atau tidak. Pada dasarnya, konstruktor adalah method yang pertama kali dijalankan pada saat object creation/loading. Konstruktor ini memiliki parameter variabel **m** dengan tipe data **String**.

Pada baris **7**, diberikan nilai **0** pada field **nim** pada saat konstruktor dijalankan.

Pada baris **8**, diberikan nilai **""** (**string kosong**) pada field **nama** pada saat konstruktor dijalankan.

Pada baris **11**, dibuat sebuah konstruktor untuk class ini (**mahasiswa3**). Konstruktor dapat dideklarasikan lebih dari satu dengan catatan memiliki parameter yang berbeda dengan konstruktor lainnya, konsep ini dinamakan **constructor overloading**. Pada saat **runtime**, program akan menyesuaikan konstruktor mana yang akan digunakan pada saat pembuatan object tergantung pada argumen yang diberikan pada saat pembuatan object. Konstruktor ini memiliki parameter variabel **n** dengan tipe data **integer**, dan variabel **m** dengan tipe data **String**.

Pada baris **12**, diberikan nilai pada field **nim** dengan nilai dari argumen variabel **n** pada saat konstruktor dijalankan.

Pada baris **13**, diberikan nilai pada field **nama** dengan nilai dari argumen variabel **m** pada saat konstruktor dijalankan.

Pada baris **16**, dideklarasikan Class bernama **Contoh33**, class ini mencakup semua kode yang ada pada baris **16** hingga baris **21**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Contoh33**.

Pada baris **17**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **17** hingga baris **20**.

Pada baris **21**, dibuat sebuah object dari class **mahasiswa3** dengan nama **mhsa** dan argumen konstruktor nilai **Kelvin** pada parameter **m**. Pembuatan object ini akan menggunakan konstruktor ke-1 (berdasarkan urutan deklarasi)

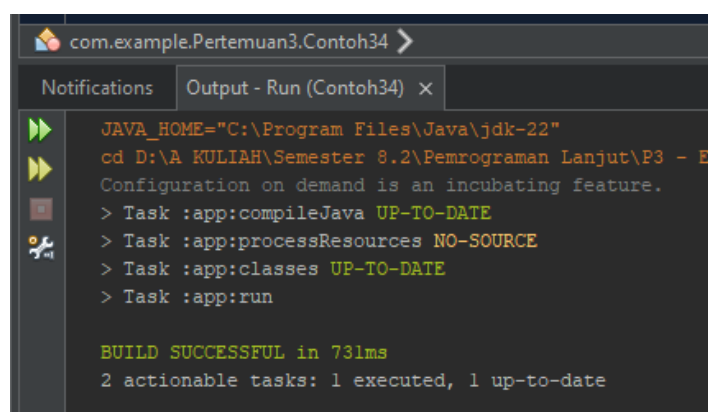
Pada baris **22**, dibuat sebuah object dari class **mahasiswa3** dengan nama **mhsb** dan argumen konstruktor nilai **5122** pada parameter **n**, dan nilai **Kelvin** pada parameter **m**. Pembuatan object ini akan menggunakan konstruktor ke-2 (berdasarkan urutan deklarasi)

D. Contoh 3.4

Source code:

```
1 package com.example.Pertemuan3;
2
3 class mahasiswa4{
4     private String nama;
5     private String getNama() {
6         return nama;
7     }
8 }
9 public class Contoh34 {
10     public static void main(String main[]){
11         mahasiswa4 mhs = new mahasiswa4();
12     }
13 }
```

Output:



```
com.example.Pertemuan3.Contoh34 >
Notifications Output - Run (Contoh34) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 - E
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE
> Task :app:run
BUILD SUCCESSFUL in 731ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Contoh34.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **mahasiswa4**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **8**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **mahasiswa4**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **nama** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan method dengan modifier **private**, tipe data balik **String**, nama **getNim**, dan tidak menggunakan parameter. Scope method ini mulai dari baris **5** hingga baris **7**. Modifier **private** dalam deklarasi ini berarti method ini hanya dapat dipanggil **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya. Style coding ini biasa disebut **getter** pada style **setter-getter**.

Pada baris **6**, nilai **field nama** dimuat dan dikembalikan kepada pemanggil method.

Pada baris **9**, dideklarasikan Class bernama **Contoh34**, class ini mencakup semua kode yang ada pada baris **9** hingga baris **13**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Contoh34**.

Pada baris **10**, dideklarasikan method **main** dengan parameter **String args[]**, method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **10** hingga baris **12**.

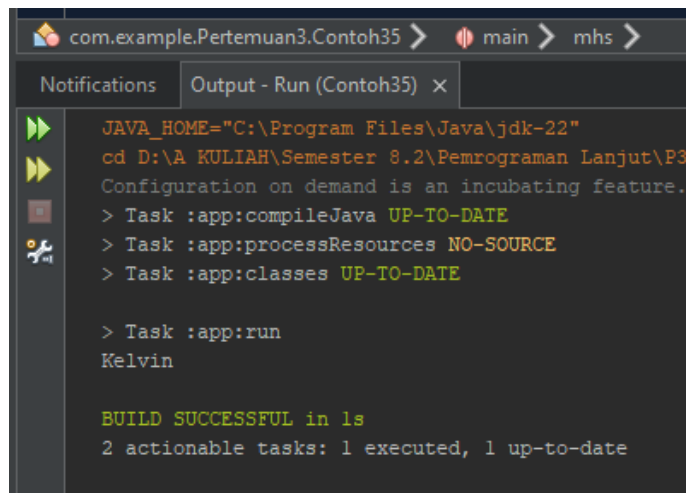
Pada baris **11**, dibuat sebuah object dari class **mahasiswa4** dengan nama **mhs** tanpa argumen konstruktor.

E. Contoh 3.5

Source code:

```
1 package com.example.Pertemuan3;
2
3 class mahasiswa5{
4     String nama;
5     String getNama(){
6         return nama;
7     }
8 }
9 public class Contoh35 {
10     public static void main(String args[]){
11         mahasiswa5 mhs = new mahasiswa5();
12         mhs.nama = "Kelvin";
13         System.out.println(mhs.getNama());
14     }
15 }
```

Output:



```
com.example.Pertemuan3.Contoh35 > main > mhs >
Notifications Output - Run (Contoh35) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE

> Task :app:run
Kelvin

BUILD SUCCESSFUL in 1s
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Contoh35.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **mahasiswa5**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **8**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **mahasiswa5**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **default** (tanpa keyword modifier seperti **public**, **private**, **protected**) bernama **nama** dengan tipe data **String**. Modifier **default** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** package yang sama dengan class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan method dengan modifier **default** (tanpa keyword modifier seperti **public**, **private**, **protected**), tipe data balik **String**, nama **getNama**, dan tidak menggunakan parameter. Scope method ini mulai dari baris **5** hingga baris **7**. Modifier **default** dalam deklarasi ini berarti method ini hanya dapat dipanggil **didalam** package yang sama dengan class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya. Style coding ini biasa disebut **getter** pada style **setter-getter**.

Pada baris **6**, nilai **field nama** dimuat dan dikembalikan kepada pemanggil method.

Pada baris **9**, dideklarasikan Class bernama **Contoh35**, class ini mencakup semua kode yang ada pada baris **9** hingga baris **15**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Contoh35**.

Pada baris **10**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **10** hingga baris **12**.

Pada baris **11**, dibuat sebuah object dari class **mahasiswa5** dengan nama **mhs** tanpa argumen konstruktor.

Pada baris **12**, diubah field **nama** dalam object **mhs**, dengan nilai “**Kelvin**”.

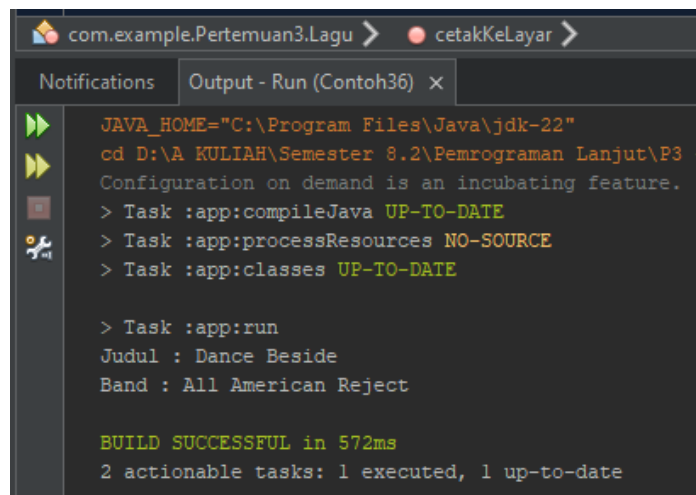
Pada baris **13**, digunakan untuk menampilkan nilai balik dari pemanggilan method **getNama()** dari object **mhs**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

F. Contoh 3.6

Source code:

```
1 package com.example.Pertemuan3;
2
3 class Lagu {
4     private String band;
5     private String judul;
6     public void IsiParam(String judul, String band){
7         this.judul = judul;
8         this.band = band;
9     }
10    public void cetakKeLayar(){
11        if(judul==null && band==null) return;
12        System.out.println("Judul : " + judul + "\nBand : " + band);
13    }
14 }
15
16 public class Contoh36 {
17     public static void main(String args[]){
18         Lagu song = new Lagu();
19         song.IsiParam("Dance Beside","All American Reject");
20         song.cetakKeLayar();
21     }
22 }
```

Output:



```
com.example.Pertemuan3.Lagu > cetakKeLayar >
Notifications Output - Run (Contoh36) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 -
>> Configuration on demand is an incubating feature.
>> > Task :app:compileJava UP-TO-DATE
>> > Task :app:processResources NO-SOURCE
>> > Task :app:classes UP-TO-DATE

>> > Task :app:run
>> Judul : Dance Beside
>> Band : All American Reject

>> BUILD SUCCESSFUL in 572ms
>> 2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Contoh36.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **Lagu**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **14**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Lagu**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **band** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **private** bernama **judul** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **6**, dideklarasikan sebuah method tanpa nilai balik (**void**), dengan nama **IsiParam** dan parameter variabel **judul** dengan tipe data **String**, dan parameter variabel **band** dengan tipe data **String**. Scope method ini dari baris **6** hingga baris **9**.

Pada baris **7**, diberikan nilai pada field **judul** dengan nilai dari argumen variabel **judul** pada saat method dipanggil.

Pada baris **8**, diberikan nilai pada field **band** dengan nilai dari argumen variabel **band** pada saat method dipanggil.

Pada baris **10**, dideklarasikan sebuah method tanpa nilai balik (**void**), dengan nama **cetakKeLayar**. Scope method ini dari baris **10** hingga baris **13**.

Pada baris **11**, dideklarasikan seleksi kondisi dengan kondisi jika **member variabel judul** tidak memiliki nilai **dan member variabel band** tidak memiliki nilai maka akhiri eksekusi method ini.

Pada baris **12**, digunakan untuk string **“Judul: “** diikuti dengan nilai **member variable judul** lalu baris baru, kemudian string **“Band : “** dan nilai **member variabel band**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **16**, dideklarasikan Class bernama **Contoh36**, class ini mencakup semua kode yang ada pada baris **16** hingga baris **22**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Contoh36**.

Pada baris **17**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **17** hingga baris **21**.

Pada baris **18**, dibuat sebuah object dari class **Lagu** dengan nama **song** tanpa argumen konstruktor.

Pada baris **19**, dipanggil method **IsiParam** dari object **song** dengan argumen **“Dance Beside”** dan **“All American Reject”**. Pemanggilan ini akan mengubah nilai field **judul** pada object **song** dengan nilai **“Dance Beside”** dan nilai field **band** pada object **song** dengan nilai **“All American Reject”**.

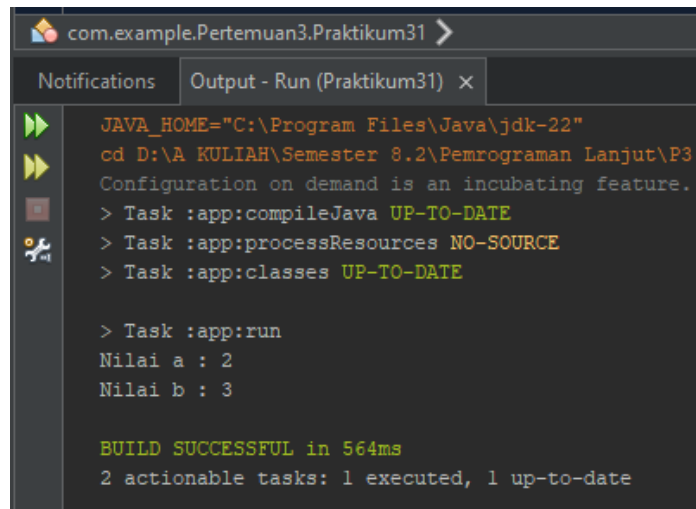
Pada baris **20**, dipanggil method **cetakKeLayar** dari object **song**.

G. Praktikum 3.1

Source code:

```
1 package com.example.Pertemuan3;
2
3 class atas{
4     public int a;
5     protected int b;
6     private int c;
7 }
8 public class Praktikum31 {
9     public static void main(String args[]){
10         atas objek = new atas();
11         objek.a = 2;
12         objek.b = 3;
13         System.out.println("Nilai a : " + objek.a);
14         System.out.println("Nilai b : " + objek.b);
15     }
16 }
```

Output:



```
com.example.Pertemuan3.Praktikum31 >
Notifications Output - Run (Praktikum31) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3
>> Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE

> Task :app:run
Nilai a : 2
Nilai b : 3

BUILD SUCCESSFUL in 564ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Praktikum31.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **atas**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **7**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **atas**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **public** bernama **a** dengan tipe data **int**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class maupun package lain.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **protected** bernama **b** dengan tipe data **int**. Modifier **protected** dalam deklarasi ini berarti field ini hanya dapat diakses **oleh** class yang sama maupun class/object turunannya dalam package yang sama.

Pada baris **6**, dideklarasikan sebuah field dengan modifier **private** bernama **c** dengan tipe data **int**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **8**, dideklarasikan Class bernama **Praktikum31**, class ini mencakup semua kode yang ada pada baris **8** hingga baris **16**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Praktikum31**.

Pada baris **9**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **9** hingga baris **15**.

Pada baris **10**, dibuat object dari class **atas** dengan nama **objek** tanpa argumen konstruktor.

Pada baris **11**, dimodifikasi nilai field **a** pada object **objek** dengan nilai **2**.

Pada baris **12**, dimodifikasi nilai field **b** pada object **objek** dengan nilai **3**.

Pada baris **13**, digunakan untuk string **“Nilai a: “** diikuti dengan nilai field **a** pada object **objek**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

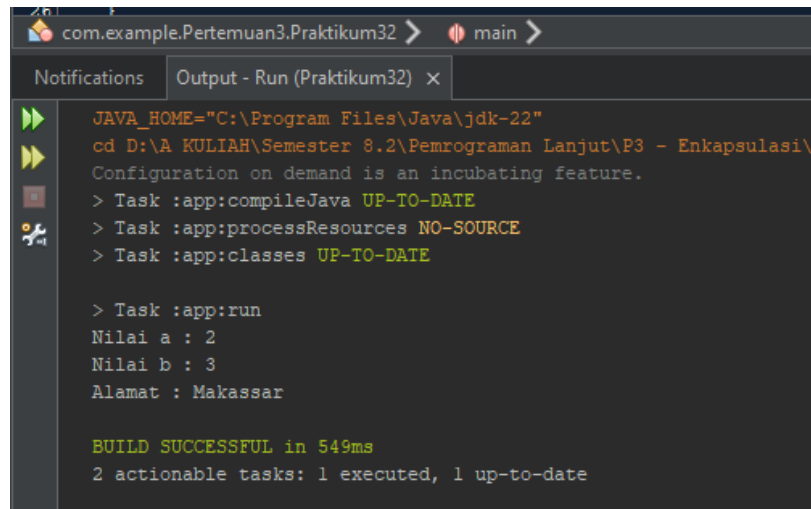
Pada baris **14**, digunakan untuk string **“Nilai b: “** diikuti dengan nilai field **b** pada object **objek**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

H. Praktikum 3.2

Source code:

```
1 package com.example.Pertemuan3;
2
3 class atas2 {
4     public int a;
5     protected int b;
6     private String alamat;
7
8     public String getAlamat(){
9         return alamat;
10    }
11
12    public void setAlamat(String tempString){
13        alamat = tempString;
14    }
15 }
16 public class Praktikum32 {
17     public static void main(String args[]){
18         atas2 objek = new atas2();
19         objek.a = 2;
20         objek.b = 3;
21         objek.setAlamat("Makassar");
22         System.out.println("Nilai a : " + objek.a);
23         System.out.println("Nilai b : " + objek.b);
24         System.out.println("Alamat : " + objek.getAlamat());
25     }
26 }
```

Output:



```
com.example.Pertemuan3.Praktikum32 > main >
Notifications Output - Run (Praktikum32) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 - Enkapsulasi\
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE

> Task :app:run
Nilai a : 2
Nilai b : 3
Alamat : Makassar

BUILD SUCCESSFUL in 549ms
2 actionable tasks: 1 executed, 1 up-to-date
```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Praktikum32.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **atas2**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **15**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **atas2**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **public** bernama **a** dengan tipe data **int**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class maupun package lain.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **protected** bernama **b** dengan tipe data **int**. Modifier **protected** dalam deklarasi ini berarti field ini hanya dapat diakses **oleh** class yang sama maupun class/object turunannya dalam package yang sama.

Pada baris **6**, dideklarasikan sebuah field dengan modifier **private** bernama **alamat** dengan tipe data **String**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **8**, dideklarasikan method dengan nama **getAlamat** dan tipe data balik **String** tanpa parameter. Scope method ini mulai dari baris **8** hingga baris **10**.

Pada baris **9**, dimuat nilai field **alamat** pada class ini lalu digunakan sebagai nilai balik method ini.

Pada baris **12**, dideklarasikan method dengan nama **setAlamat** tanpa tipe data balik dan parameter variabel **tempString** dengan tipe data **String**. Scope method ini mulai dari baris **12** hingga baris **14**.

Pada baris **13**, dilakukan modifikasi nilai field **alamat** pada class ini dengan nilai variabel argumen **tempString**.

Pada baris **16**, dideklarasikan Class bernama **Praktikum32**, class ini mencakup semua kode yang ada pada baris **16** hingga baris **26**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Praktikum32**.

Pada baris **17**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **17** hingga baris **25**.

Pada baris **18**, dibuat object dari class **atas2** dengan nama **objek** tanpa argumen konstruktor.

Pada baris **19**, dimodifikasi nilai field **a** pada object **objek** dengan nilai **2**.

Pada baris **20**, dimodifikasi nilai field **b** pada object **objek** dengan nilai **3**.

Pada baris **21**, dipanggil method **setAlamat** dari object **objek** dengan argumen nilai **“Makassar”**. Pemanggilan method ini akan mengubah nilai field **alamat** pada object **objek** dengan nilai **“Makassar”**.

Pada baris **22**, digunakan untuk string **“Nilai a: “** diikuti dengan nilai field **a** pada object **objek**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **23**, digunakan untuk string **“Nilai b: “** diikuti dengan nilai field **b** pada object **objek**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **24**, digunakan untuk string **“Alamat: “** diikuti dengan nilai balik pemanggilan method **getAlamat** pada object **objek**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

I. **Praktikum 3.3**

Source code:

```

1 package com.example.Pertemuan3;
2
3 public class Praktikum33 {
4     public static void main(String args[]){
5         enkapsulasi ob = new enkapsulasi();
6         ob.setAlas(5);
7         ob.setTinggi(7);
8         System.out.println("Alas Segitiga : " + ob.getAlas());
9         System.out.println("Tinggi Segitiga : " + ob.getTinggi());
10        ob.setLuasSegitiga(ob.getAlas(), ob.getTinggi());
11        System.out.println("Luas Segitiga : " + ob.getLuasSegitiga());
12    }
13 }

```

```

1 package com.example.Pertemuan3;
2 //file ini bagian dari Praktikum 3.3
3 public class enkapsulasi {
4     private int alas, tinggi;
5     private double luasSegitiga;
6
7     public void setAlas(int alas){
8         this.alas = alas;
9     }
10
11    public int getAlas(){
12        return this.alas;
13    }
14
15    public void setTinggi(int tinggi){
16        this.tinggi = tinggi;
17    }
18
19    public int getTinggi(){
20        return this.tinggi;
21    }
22
23    public void setLuasSegitiga(int alas, int tinggi){
24        this.luasSegitiga = 0.5 * (double)(alas * tinggi);
25    }
26
27    public double getLuasSegitiga(){
28        return this.luasSegitiga;
29    }
30 }

```

Output:

```

JAVA_HOME="C:\Program Files\Java\jdk-22"
cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 -
Configuration on demand is an incubating feature.
> Task :app:compileJava UP-TO-DATE
> Task :app:processResources NO-SOURCE
> Task :app:classes UP-TO-DATE

> Task :app:run
Alas Segitiga : 5
Tinggi Segitiga : 7
Luas Segitiga : 17.5

BUILD SUCCESSFUL in 781ms
2 actionable tasks: 1 executed, 1 up-to-date

```

Penjelasan:

Praktikum33.java

Pada baris 1, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Praktikum33.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **Praktikum33**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **13**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Praktikum33**.

Pada baris **4**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **4** hingga baris **12**.

Pada baris **5**, dibuat object dari class **enkapsulasi** dengan nama **ob** tanpa argumen konstruktor.

Pada baris **6**, dipanggil method **setAlas** dari object **ob** dengan argumen nilai **5**. Pemanggilan method ini akan mengubah nilai field **alas** pada object **ob** dengan nilai **5**.

Pada baris **7**, dipanggil method **setTinggi** dari object **ob** dengan argumen nilai **7**. Pemanggilan method ini akan mengubah nilai field **tinggi** pada object **ob** dengan nilai **7**.

Pada baris **8**, digunakan untuk string “**Alas Segitiga:** “ diikuti dengan nilai balik pemanggilan method **getAlas** pada object **ob**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **9**, digunakan untuk string “**Tinggi Segitiga:** “ diikuti dengan nilai balik pemanggilan method **getTinggi** pada object **ob**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **10**, dipanggil method **setLuasSegitiga** dari object **ob** dengan argumen nilai balik pemanggilan method **getAlas** pada object **ob** dan nilai balik pemanggilan method **getTinggi** pada object **ob**.

Pada baris **11**, digunakan untuk string “**Luas Segitiga:** “ diikuti dengan nilai balik pemanggilan method **getLuasSegitiga** pada object **ob**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

enkapsulasi.java

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **enkapsulasi.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **enkapsulasi**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **30**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **enkapsulasi**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **private** bernama **alas** dan **tinggi** dengan tipe data **integer**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **private** bernama **luasSegitiga** dengan tipe data **double**. Modifier **private** dalam deklarasi ini berarti field ini hanya dapat diakses **didalam** class ini saja, tidak pada object turunannya maupun class lain atau inheritancenya.

Pada baris **7**, dideklarasikan method dengan nama **setAlas** tanpa tipe data balik dan parameter variabel **alas** dengan tipe data **int**. Scope method ini mulai dari baris **7** hingga baris **9**.

Pada baris **8**, dilakukan modifikasi nilai field **alas** pada class ini dengan nilai variabel argumen **alas**.

Pada baris **11**, dideklarasikan method dengan nama **getAlamat** dan tipe data balik **int** tanpa parameter. Scope method ini mulai dari baris **11** hingga baris **13**.

Pada baris **12**, dimuat nilai field **alas** pada class ini lalu digunakan sebagai nilai balik method ini.

Pada baris **15**, dideklarasikan method dengan nama **setTinggi** tanpa tipe data balik dan parameter variabel **tinggi** dengan tipe data **int**. Scope method ini mulai dari baris **15** hingga baris **17**.

Pada baris **16**, dilakukan modifikasi nilai field **tinggi** pada class ini dengan nilai variabel argumen **tinggi**.

Pada baris **19**, dideklarasikan method dengan nama **getTinggi** dan tipe data balik **int** tanpa parameter. Scope method ini mulai dari baris **19** hingga baris **21**.

Pada baris **20**, dimuat nilai field **tinggi** pada class ini lalu digunakan sebagai nilai balik method ini.

Pada baris **23**, dideklarasikan method dengan nama **setLuasSegitiga** tanpa tipe data balik dan parameter variabel **alas** dan **tinggi** dengan tipe data **int**. Scope method ini mulai dari baris **23** hingga baris **25**.

Pada baris **24**, dilakukan modifikasi nilai field **luasSegitiga** pada class ini dengan hasil perhitungan **0.5 * (alas * tinggi)**.

Pada baris **27**, dideklarasikan method dengan nama **getTinggi** dan tipe data balik **int** tanpa parameter. Scope method ini mulai dari baris **27** hingga baris **29**.

Pada baris **28**, dimuat nilai field **luasSegitiga** pada class ini lalu digunakan sebagai nilai balik method ini.

J. Soal Tantangan

Buatlah program enkapsulasi sederhana yang dapat menampilkan biodata Anda, berupa : nama, nim, jurusan, fakultas, universitas, alamat, email, hobi, dan keahlian.

Source code:

```
1 package com.example.Pertemuan3;
2
3 class Biodata {
4     public String nama;
5     public long nim;
6     public String jurusan;
7     public String fakultas;
8     public String universitas;
9     public String alamat;
10    public String email;
11    public String hobi;
12    public String keahlian;
13
14    public Biodata(String nama,
15                    long nim,
16                    String jurusan,
17                    String fakultas,
18                    String universitas,
19                    String alamat,
20                    String email,
21                    String hobi,
22                    String keahlian)
23    {
24        this.nama = nama;
25        this.nim = nim;
26        this.jurusan = jurusan;
27        this.fakultas = fakultas;
28        this.universitas = universitas;
29        this.alamat = alamat;
30        this.email = email;
31        this.hobi = hobi;
32        this.keahlian = keahlian;
33    }
34 }
```

```

37 public class Tantangan {
38     public static void main(String args[]){
39         Biodata kelvin = new Biodata(
40             "Kelvianto Pratama Harum",
41             200210500016L,
42             "Teknik Informatika dan Komputer",
43             "Teknik",
44             "Universitas Negeri Makassar",
45             "di Rumah",
46             "kelvin@kelvin-pratama.my.id",
47             "Billiard",
48             "Network Technician"
49         );
50         System.out.println("Nama: " + kelvin.nama);
51         System.out.println("NIM: " + kelvin.nim);
52         System.out.println("Jurusan: " + kelvin.jurusan);
53         System.out.println("Fakultas: " + kelvin.fakultas);
54         System.out.println("Universitas: " + kelvin.universitas);
55         System.out.println("Alamat: " + kelvin.alamat);
56         System.out.println("Email: " + kelvin.email);
57         System.out.println("Hobi: " + kelvin.hobi);
58         System.out.println("Keahlian: " + kelvin.keahlian);
59     }
60 }

```

Output:

```

com.example.Pertemuan3.Biodata > Biodata >
Notifications Output - Run (Tantangan) x
>> JAVA_HOME="C:\Program Files\Java\jdk-22"
>> cd D:\A KULIAH\Semester 8.2\Pemrograman Lanjut\P3 - Enkapsulasi\
>> Configuration on demand is an incubating feature.
>> Task :app:compileJava UP-TO-DATE
>> Task :app:processResources NO-SOURCE
>> Task :app:classes UP-TO-DATE

>> Task :app:run
Nama: Kelvianto Pratama Harum
NIM: 200210500016
Jurusan: Teknik Informatika dan Komputer
Fakultas: Teknik
Universitas: Universitas Negeri Makassar
Alamat: di Rumah
Email: kelvin@kelvin-pratama.my.id
Hobi: Billiard
Keahlian: Network Technician

BUILD SUCCESSFUL in 557ms
2 actionable tasks: 1 executed, 1 up-to-date

```

Penjelasan:

Pada baris **1**, dideklarasikan package dimana file source ini bergabung pada contoh ini file **Tantangan.java** tergabung pada package **com.example.Pertemuan3**

Pada baris **3**, dideklarasikan Class bernama **Biodata**, class ini mencakup semua kode yang ada pada baris **3** hingga baris **34**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Biodata**.

Pada baris **4**, dideklarasikan sebuah field dengan modifier **public** bernama **nama** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **5**, dideklarasikan sebuah field dengan modifier **public** bernama **nim** dengan tipe data **long**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **6**, dideklarasikan sebuah field dengan modifier **public** bernama **jurusan** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **7**, dideklarasikan sebuah field dengan modifier **public** bernama **fakultas** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **8**, dideklarasikan sebuah field dengan modifier **public** bernama **universitas** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **9**, dideklarasikan sebuah field dengan modifier **public** bernama **alamat** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **10**, dideklarasikan sebuah field dengan modifier **public** bernama **email** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **11**, dideklarasikan sebuah field dengan modifier **public** bernama **hobi** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **12**, dideklarasikan sebuah field dengan modifier **public** bernama **keahlian** dengan tipe data **String**. Modifier **public** dalam deklarasi ini berarti field ini dapat diakses **semua** class baik dalam package yang sama maupun berbeda.

Pada baris **14**, dideklarasikan konstruktor untuk class ini dengan modifier **public** dan parameter variabel **nama** tipe data **String**, variabel **nim** tipe data **long**, variabel **jurusan** tipe data **String**, variabel **fakultas** tipe data **String**, variabel **universitas** tipe data **String**, variabel **alamat** tipe data **String**, variabel **email** tipe data **String**, variabel **hobi** tipe data **String**, dan variabel **keahlian** tipe data **String**.

Pada baris **24**, dilakukan modifikasi pada field **nama** dengan nilai parameter variabel **nama**.

Pada baris **25**, dilakukan modifikasi pada field **nim** dengan nilai parameter variabel **nim**.

Pada baris **26**, dilakukan modifikasi pada field **jurusan** dengan nilai parameter variabel **jurusan**.

Pada baris **27**, dilakukan modifikasi pada field **fakultas** dengan nilai parameter variabel **fakultas**.

Pada baris **28**, dilakukan modifikasi pada field **universitas** dengan nilai parameter variabel **universitas**.

Pada baris **29**, dilakukan modifikasi pada field **alamat** dengan nilai parameter variabel **alamat**.

Pada baris **30**, dilakukan modifikasi pada field **email** dengan nilai parameter variabel **email**.

Pada baris **31**, dilakukan modifikasi pada field **hobi** dengan nilai parameter variabel **hobi**.

Pada baris **32**, dilakukan modifikasi pada field **keahlian** dengan nilai parameter variabel **keahlian**.

Pada baris **37**, dideklarasikan Class bernama **Tantangan**, class ini mencakup semua kode yang ada pada baris **37** hingga baris **60**, sehingga semua kode baik method dan identifiers dalam blok ini adalah scope dari class **Tantangan**.

Pada baris **38**, dideklarasikan method **main** dengan parameter String args[], method ini adalah method yang akan pertama kali dieksekusi pada saat runtime program berjalan. Scope method ini adalah baris **38** hingga baris **59**.

Pada baris **39**, dibuat object dari class **Biodata** dengan nama **kelvin** dengan argumen konstruktor nilai “**Kelvianto Pratama Harum**”, **200210500016**, “**Teknik Informatika dan Komputer**”, “**Teknik**”, “**Universitas Negeri Makassar**”, “**di Rumah**”, kelvin@kelvin-pratama.my.id, “**Billiard**”, dan “**Network Technician**”.

Pada baris **50**, digunakan untuk menampilkan “**Nama:** “ diikuti dengan nilai field **nama** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **51**, digunakan untuk menampilkan “**NIM:** “ diikuti dengan nilai field **nim** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada

dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **52**, digunakan untuk menampilkan “**Jurusan:** “ diikuti dengan nilai field **jurusan** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **53**, digunakan untuk menampilkan “**Fakultas:** “ diikuti dengan nilai field **fakultas** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **54**, digunakan untuk menampilkan “**Universitas:** “ diikuti dengan nilai field **universitas** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **55**, digunakan untuk menampilkan “**Alamat:** “ diikuti dengan nilai field **alamat** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **56**, digunakan untuk menampilkan “**Email:** “ diikuti dengan nilai field **email** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **57**, digunakan untuk menampilkan “**Hobi:** “ diikuti dengan nilai field **hobi** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.

Pada baris **58**, digunakan untuk menampilkan “**Keahlian:** “ diikuti dengan nilai field **keahlian** pada object **kelvin**. Output ini dihasilkan menggunakan method bawaan

dari Java yang berada pada class **System**, member **out**, method **println()**. Method ini pada dasarnya akan menampilkan apapun yang diberikan pada calling argument ke runtime console.