



PROSEDUR dan FUNGSI

Pertemuan ke 5,6

Sub-CPMK

- *Mahasiswa mampu menggunakan prosedur dan fungsi untuk membangun method (metoda) dalam konsep pemrograman berorientasi objek. (C3, A3).*

Materi:

1. Prosedur
2. Fungsi
3. Perbedaan Prosedur Dengan Fungsi



1. Prosedur

- Pemahaman penggunaan parameter pada prosedur dan fungsi sangat penting.
- Secara umum parameter merupakan variabel yang terdapat pada prosedur dan fungsi.
- Ada dua cara pelewatan paramter yang dikenal dengan parameter passing, yaitu **by value** dan **by reference**.

- **by value:** parameter dengan by value melakukan pelewatan nilai dengan cara menyalin nilai argumen yang dilewatkan ke subrutin. Dengan demikian perubahan-perubahan yang dibuat terhadap argumen tidak mempunyai dampak apapun terhadap argumen yang digunakan untuk memanggil method.

- **by reference:** parameter dengan by reference acuan ke argumen dilewatkan ke subrutin, maka perubahan yang dibuat terhadap parameter akan juga mengubah argumen yang digunakan untuk memanggil subrutin.

- Apabila yang dilewatkan adalah objek, maka yang digunakan adalah pelewatan dengan acuan (by reference), dan bila yang dilewatkan adalah tipe sederhana ke method maka argumen dilewatkan dengan nilai (by value).
- Yang mengirim nilai disebut dengan parameter formal, dan yang menerima nilai disebut parameter aktual.

- **Catatan:**

- Pada dasarnya Java sama sekali tidak menggunakan pass by reference dan tidak dikenal istilah pass by reference pada method Java.
- Namun dengan teknik program tertentu kita dapat melakukan by reference pada Java, karena Java sendiri dalam menghemat memori menggunakan reference cara kerjanya.

- Pada bahasa pemrograman Java mendeklarasikan prosedur menggunakan kata kunci **void**.
- Prosedur pada bahasa Java juga dikenal dengan fungsi tanpa nilai balik (*return value*).
- Prosedur tidak bisa mengembalikan nilai.
- Sintaks untuk mendeklarasikan prosedur seperti berikut.

Sintaks: membuat prosedur

```
static void nama_prosedur(daftar parameter)
{
    Isi prosedur
}
```

- Membuat prosedur bisa **tanpa parameter** dan bisa **menggunakan parameter**.

1.1 Tanpa Parameter

- Berikut contoh membuat prosedur tanpa menggunakan parameter.

Prosedur (void) tanpa parameter

```
1. package latprosedur;  
2. public class LatProsedur {  
3.  
4.     public static void main(String[] args)  
5.     {  
6.         //Memanggil prosedur  
7.         tambah();  
8.     }  
9.
```

1.1 Tanpa Parameter (Lanj...)

```
10.    // membuat prosedur
11.    static void tambah()
12.    {
13.        int a,b,hasil; //mendeklar variabel lokal
14.
15.        a = 100;
16.        b = 200;
17.        hasil = a+b;
18.        System.out.println(hasil);
19.    }
20. }
```

1.1 Tanpa Parameter (Lanj...)

- Contoh keluaran programnya seperti berikut.



The screenshot shows a window titled 'Report Problems Window' with a tab labeled 'iReport output'. The output text is as follows:

```
run:  
300  
BUILD SUCCESSFUL (total time: 3 seconds)
```

1.2 Dengan Parameter

- Prosedur dapat memiliki paramter, parameter dapat berjenis by value (nilai), dan by ref (referensi).
- Parameter nilai merupakan parameter input dengan tujuan hanya nilai data yang ditransfer ke dalam prosedur yang berfungsi sebagai masukan.

1.2 Dengan Parameter (Lanj...)

- By Value
 - Parameter input hanya membawa nilai parameter yang akan diproses di dalam prosedur.

Prosedur (void) dengan parameter nilai (by value)

```
1. package latprosedur;  
2.  
3. public class LatProsedur  
4. {  
5.  
6.     public static void main(String[] args)  
7.     {  
8.         int nilaia, nilaib;
```

1.2 Dengan Parameter (Lanj...)

```
9.      nilaia = 100; //inisialisasi nilaia
10.     nilaib = 200; //inisialisasi nilaib
11.
12.     //Memanggil prosedur
13.     tambah(nilaia, nilaib);
14. }
15.
16.     // membuat prosedur dengan parameter by value
17.     static void tambah(int a, int b)
18.     {
19.         int hasil;
20.         hasil = a+b;
21.         System.out.println(hasil);
22.     }
23. }
```


1.2 Dengan Parameter (Lanj...)

- Contoh keluaran programnya seperti berikut.



The screenshot shows a window with three tabs: "Report Problems Window", "iReport output", and "Output - I". The "iReport output" tab is active, displaying the following text:

```
run:  
300  
BUILD SUCCESSFUL (total time: 3 seconds)
```

- By Ref

- Parameter by ref merupakan parameter yang dapat dijadikan input maupun output.
- Dengan parameter ref, nilai yang diproses dalam sebuah prosedur dapat dikirim keluar dengan menggunakan parameter ref sebagai output (hasil).

Catatan: *Pada dasarnya Java sama sekali tidak menggunakan pass by reference dan tidak dikenal istilah pass by reference pada method Java.*



2. Fungsi

- Fungsi dapat mengembalikan nilai, *return value* dibawa oleh nama fungsi.
- Dalam bahasa pemrograman Java mendeklarasikan fungsi menggunakan kata kunci **static**.
- Sintaks mendeklarasikan fungsi.

Sintaks: fungsi

```
static tipe_data nama_fungsi(daftar parameter)
{
    Isi fungsi
}
```

- Bila mendeklarasikan fungsi pada kelas jangan menggunakan static, supaya fungsi yang dibuat dapat diakases oleh kelas lainnya atau di main program.
- Sintaks mendeklarasikan fungsi di kelas.

Sintaks: membuat fungsi di class

```
public tipe_data nama_fungsi(daftar parameter)
{
    Isi fungsi
}
```

Atau

```
tipe_data nama_fungsi(daftar parameter)
{
    Isi fungsi
}
```

2.1 Tanpa Parameter

- Fungsi tanpa parameter.

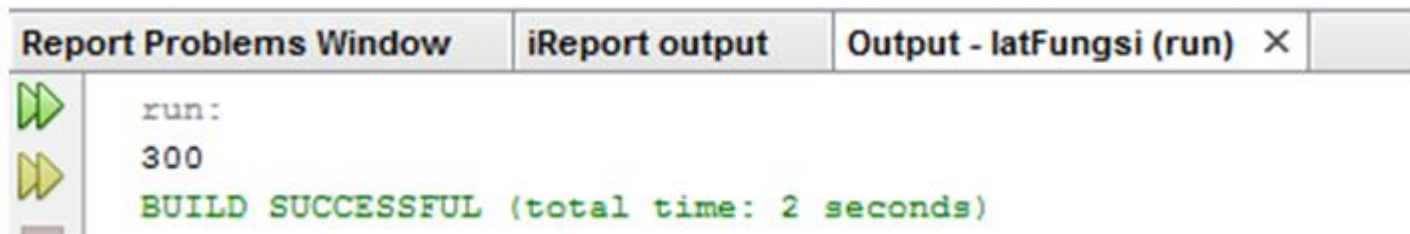
Fungsi: tanpa parameter

```
1. package latfungsi;  
2. public class LatFungsi {  
3.  
4.     public static void main(String[] args) {  
5.         //Memanggil fungsi  
6.         System.out.println(tambah());  
7.     }  
8.
```

2.1 Tanpa Parameter (Lanj...)

```
9.      // membuat fungsi
10.     static int tambah()
11.     {
12.         int a,b;
13.         a = 100;
14.         b = 200;
15.         return a+b;
16.     }
17. }
```

- Keluaran program



2.2 Dengan Parameter

- **By Value**

- Parameter input hanya membawa nilai parameter yang akan diproses di dalam fungsi.

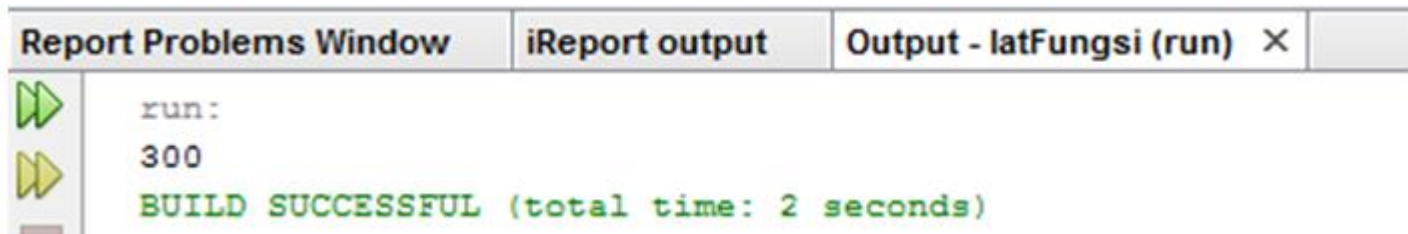
Fungsi: dengan parameter

```
1. package latfungsi;  
2. public class LatFungsi {  
3.  
4.     public static void main(String[] args) {  
5.         int nilaia;  
6.         int nilaib;  
7.  
8.         nilaia = 100; //inisialisasi nilaia  
9.         nilaib = 200; //inisialisasi nilaib
```

2.2 Dengan Parameter (Lanj...)

```
10.  
11.         //Memanggil fungsi  
12.         System.out.println(tambah(nilaia, nilaib));  
13.     }  
14.  
15.     // membuat fungsi dengan parameter by value  
16.     static int tambah(int a, int b)  
17.     {  
18.         return a+b;  
19.     }  
20. }
```

- Keluaran program



2.2 Dengan Parameter (Lanj...)

- **By Ref**

- Parameter by ref merupakan parameter yang dapat dijadikan input maupun output.
- Dengan parameter ref, nilai yang diproses dalam sebuah fungsi dapat dikirim keluar dengan menggunakan parameter ref sebagai output (hasil).

Catatan: *Pada dasarnya Java sama sekali tidak menggunakan pass by reference dan tidak dikenal istilah pass by reference pada method Java.*

- Method atau metode merupakan tingkah laku dari suatu kelas atau objek.
- Dalam konsep pemrograman berorientasi objek, method dapat dibuat dengan menggunakan prosedur dan fungsi.

- Method adalah "sekumpulan kode yang diberi nama, untuk merujuk sekumpulan kode tersebut digunakan sebuah nama yang disebut dengan nama method. Method mempunyai parameter sebagai input dan nilai kembalian sebagai output. Setiap kelas boleh mempunyai lebih dari satu method."
- Method ada yang bersifat **static** dan **non static**.

- **Method Satic** adalah method yang dapat dipanggil walaupun kelas belum diinstansiasi menjadi objek.
- Sintaks membuat method static seperti berikut.

Sintaks: method static

```
public static tipe_data nama_method(parameter)
{
    .....
}
```

Nama class: clsDokter

```
1. package dokter;
2. public class clsDokter {
3.     //mendeklarasikan atribut
4.     String IdDokter;
5.     String Nama;
6.     int Gaji;
7.     //membuat method ststic dengan fungsi
8.     public static float Tunjangan(int xGaji)
9.     {
10.         float mGaji = xGaji/100*10;
11.         return mGaji;
12.     }
13.     //membuat method static dengan fungsi
14.     public static float TotalGaji(int xGaji)
15.     {
16.         float mTotal = xGaji + Tunjangan(xGaji);
17.         return mTotal;
18.     }
19. }
```

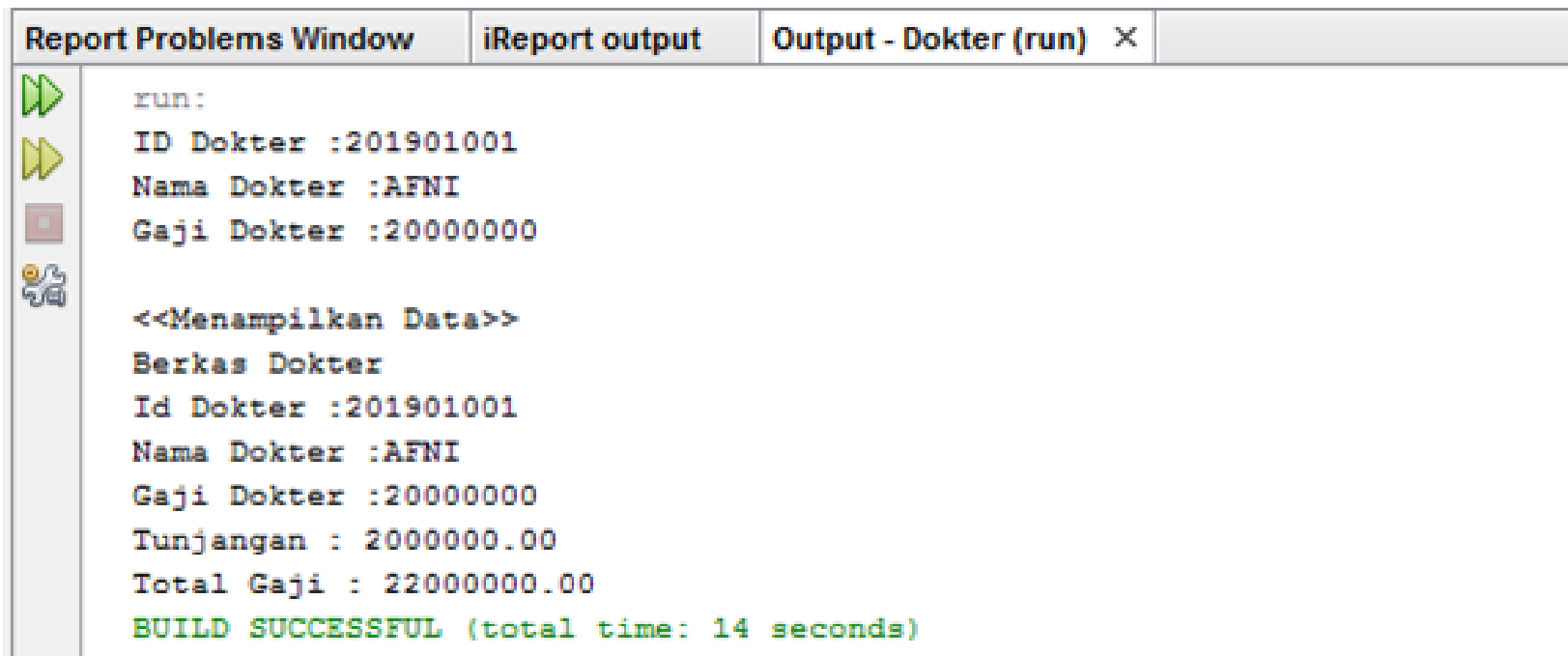
Main program: dokter

```
1.  package dokter;
2.  import static dokter.clsDokter.TotalGaji;
3.  import static dokter.clsDokter.Tunjangan;
4.  import java.util.Scanner;
5.
6.  public class Dokter {
7.      public static void main(String[] args) {
8.          //membuat scanner baru
9.          Scanner input = new Scanner(System.in);
10.         //membuat objek
11.         clsDokter objDokter = new clsDokter();
12.         System.out.print("ID Dokter :");
13.         objDokter.IdDokter = input.nextLine();
14.         System.out.print("Nama Dokter :");
15.         objDokter>Nama = input.nextLine();
16.         System.out.print("Gaji Dokter :");
17.         objDokter.Gaji = input.nextInt();
```



```
18.         //menampilkan data
19.         System.out.println();
20.         System.out.println("<<Menampilkan Data>>");
21.         System.out.println("Berkas Dokter");
22.         System.out.println("Id Dokter :"+
objDokter.IdDokter);
23.         System.out.println("Nama Dokter :"+
objDokter>Nama);
24.         System.out.println("Gaji Dokter :"+
objDokter.Gaji);
25.         //walau tidak menyebutkan nama objeknya method
tunjangan & total gaji
26.         //dapat digunakan, untuk akses atribut tetap
menggunakan nama objek
27.         System.out.printf("Tunjangan : %.2f\n", +
Tunjangan(objDokter.Gaji));
28.         System.out.printf("Total Gaji : %.2f\n", +
TotalGaji(objDokter.Gaji));
29.     }
30. }
```

- Contoh keluaran program.



The screenshot shows a software development environment's output window. The window has three tabs: 'Report Problems Window', 'iReport output', and 'Output - Dokter (run)'. The 'Output - Dokter (run)' tab is active, displaying the following text:

```
run:
ID Dokter :201901001
Nama Dokter :AFNI
Gaji Dokter :20000000

<<Menampilkan Data>>
Berkas Dokter
Id Dokter :201901001
Nama Dokter :AFNI
Gaji Dokter :20000000
Tunjangan : 2000000.00
Total Gaji : 22000000.00
BUILD SUCCESSFUL (total time: 14 seconds)
```

- Cara lain menjalankan atribut atau method static menggunakan sintaks sebagai berikut:

```
Nama_kelas.nama_atribut;
```

```
Nama_kelas.nama_method;
```

Contoh:

```
Dokter.idDokter = input.nextLine();
```

```
System.out.printf("Tunjangan \t:%.2f\n", +  
Dokter.tunjangan(Dokter.gaji));
```

Cara ini tidak perlu melakukan import.

- **Method Non-Static** adalah method yang tidak dapat dipanggil jika kelas belum diinstansiasi menjadi objek.
- Sintaks membuat method non static seperti berikut.

Sintaks: method non static

```
public tipe_data nama_method(parameter)
{
    .....
}
```

Nama class: clsDokter

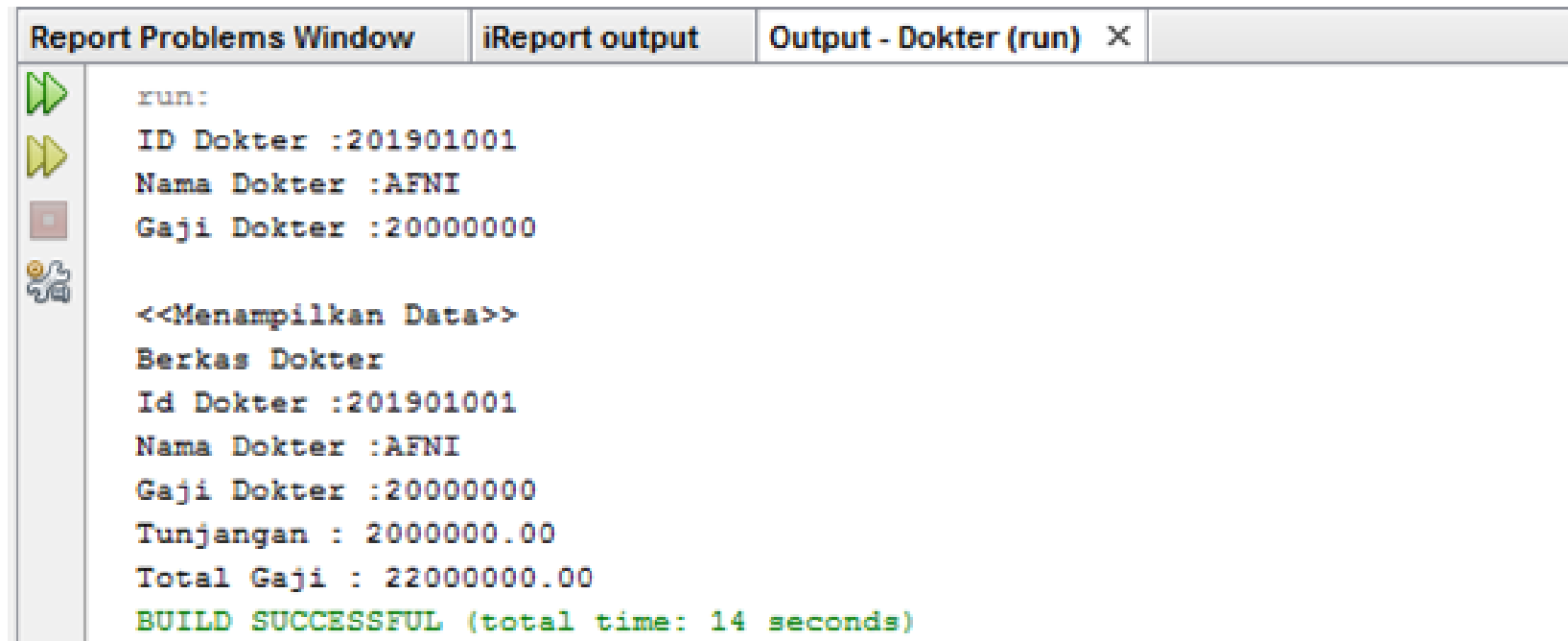
```
1. package dokter;
2. public class clsDokter {
3.     //mendeklarasikan atribut
4.     String IdDokter;
5.     String Nama;
6.     int Gaji;
7.     //membuat method ststic dengan fungsi
8.     public float Tunjangan(int xGaji)
9.     {
10.         return xGaji/100*10;
11.     }
12.     //membuat method static dengan fungsi
13.     public float TotalGaji(int xGaji)
14.     {
15.         return xGaji + Tunjangan(xGaji);
16.     }
17. }
```

Main program: dokter

```
1. package dokter;
2. import java.util.Scanner;
3. public class Dokter {
4.
5.     public static void main(String[] args) {
6.         //membuat scanner baru
7.         Scanner input = new Scanner(System.in);
8.         //membuat objek
9.         clsDokter objDokter = new clsDokter();
10.        System.out.print("ID Dokter :");
11.        objDokter.IdDokter = input.nextLine();
12.        System.out.print("Nama Dokter :");
13.        objDokter>Nama = input.nextLine();
14.        System.out.print("Gaji Dokter :");
15.        objDokter.Gaji = input.nextInt();
```

```
16.         //menampilkan data
17.         System.out.println();
18.         System.out.println("<<Menampilkan Data>>");
19.         System.out.println("Berkas Dokter");
20.         System.out.println("Id Dokter :"+
    objDokter.IdDokter);
21.         System.out.println("Nama Dokter :"+
    objDokter>Nama);
22.         System.out.println("Gaji Dokter :"+
    objDokter.Gaji);
23.         System.out.printf("Tunjangan : %.2f\n", +
    objDokter.Tunjangan(objDokter.Gaji));
24.         System.out.printf("Total Gaji : %.2f\n", +
    objDokter.TotalGaji(objDokter.Gaji));
25.     }
26. }
```

- Contoh keluaran program.



The screenshot shows an IDE window with three tabs: 'Report Problems Window', 'iReport output', and 'Output - Dokter (run)'. The 'Output - Dokter (run)' tab is active, displaying the following text:

```
run:
ID Dokter :201901001
Nama Dokter :AFNI
Gaji Dokter :20000000

<<Menampilkan Data>>
Berkas Dokter
Id Dokter :201901001
Nama Dokter :AFNI
Gaji Dokter :20000000
Tunjangan : 2000000.00
Total Gaji : 22000000.00
BUILD SUCCESSFUL (total time: 14 seconds)
```




3. Perbedaan Prosedur dan Fungsi

- Prosedur tidak bisa mengembalikan (membawa) nilai.
- Sedangkan pada fungsi dapat membawa nilai balik (return value).
- Prosedur dan fungsi dapat digunakan untuk membuat method pada konsep Pemrograman Berorientasi Objek.

Latihan Tambahan BufferReader

- Kelas BufferedReader tidak hanya untuk membaca nilai dari keyboard.
- Kelas BufferedReader dapat juga digunakan untuk membaca input dari file dan jaringan.
- Bila menggunakan class BufferedReader terdapat pada paket java.io. Buat proyek dengan nama **lat_Dokter**, buat kelas dengan nama **Dokter**. Buat program berikut di kelas Dokter.

Latihan Tambahan BufferedReader (Lanj...)

Nama kelas: Dokter

```
1. package lat_dokter;
2. public class Dokter {
3.     //mendeklarasikan atribut
4.     String idDokter;
5.     String nama;
6.     int gaji;
7.
8.     //membuat method dengan fungsi
9.     public float tunjangan()
10.    {
11.        return gaji/100*10;
12.    }
13.    //membuat method dengan fungsi
14.    public float totalGaji()
15.    {
16.        return gaji+tunjangan();
17.    }
18. }
```

Latihan Tambahan BufferedReader (Lanj...)

Main program: Lat_Dokter

```
1.  package lat_dokter;
2.  //import library
3.  import java.io.BufferedReader;
4.  import java.io.IOException;
5.  import java.io.InputStreamReader;
6.  public class Lat_Dokter {
7.
8.      public static void main(String[] args) throws IOException {
9.          // TODO code application logic here
10.         //membuat scanner baru
11.         InputStreamReader isr=new InputStreamReader(System.in);
12.         //membuat objek bufferreader
13.         BufferedReader br=new BufferedReader(isr);
14.         //membuat objek
15.         Dokter objDokter = new Dokter();
16.
```

Latihan Tambahan BufferReader (Lanj...)

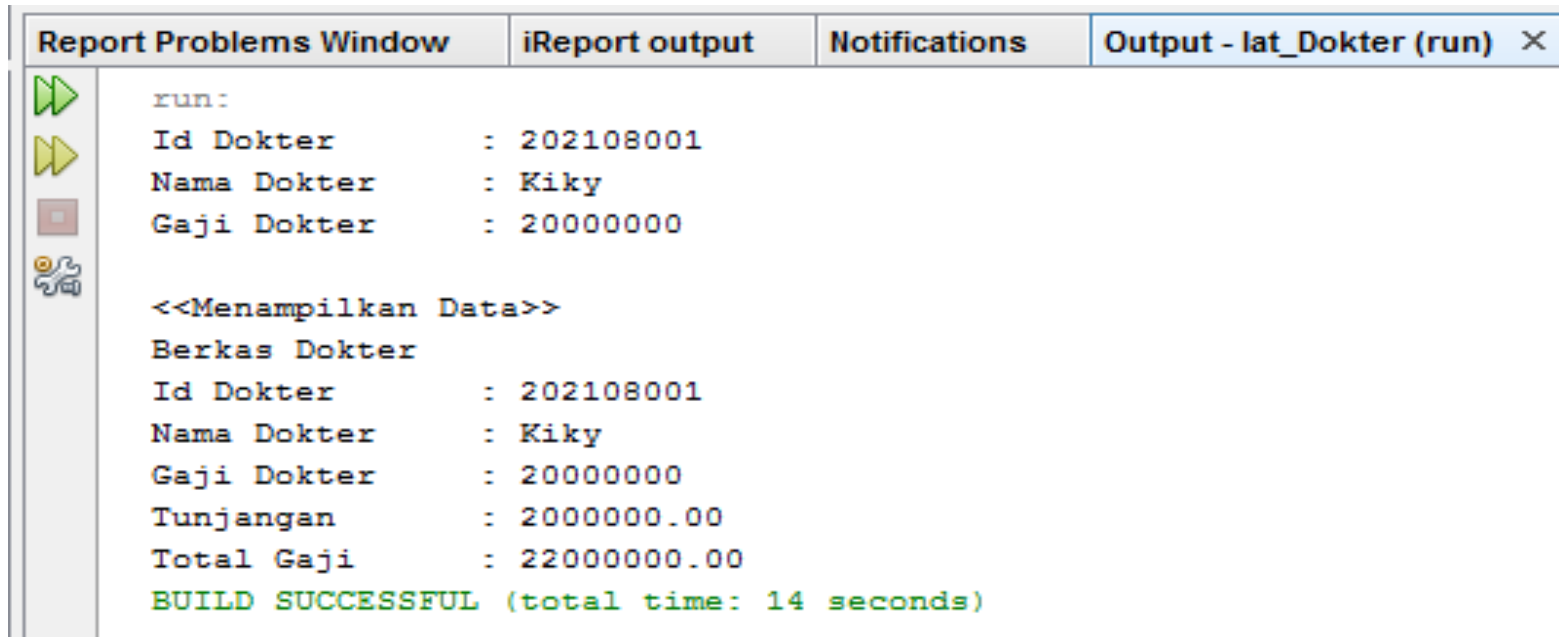
```
17.      System.out.print("Id Dokter\t: ");
18.      objDokter.idDokter = br.readLine();
19.      System.out.print("Nama Dokter\t: ");
20.      objDokter.nama = br.readLine();
21.      System.out.print("Gaji Dokter\t: ");
22.      objDokter.gaji = Integer.parseInt(br.readLine());
23.
24.      //menampilkan data
25.      System.out.println();
26.      System.out.println("<<Menampilkan Data>>");
27.      System.out.println("Berkas Dokter");
28.      System.out.println("Id Dokter\t: "+
29.          objDokter.idDokter);
```

Latihan Tambahan BufferReader (Lanj...)

```
30.      System.out.println("Nama Dokter\t: "+
31.          objDokter.nama);
32.      System.out.println("Gaji Dokter\t: "+
33.          objDokter.gaji);
34.      System.out.printf("Tunjangan\t: %.2f\n", +
35.          objDokter.tunjangan());
36.      System.out.printf("Total Gaji\t: %.2f\n", +
37.          objDokter.totalGaji());
38.  }
39. }
```

Latihan Tambahan BufferReader (Lanj...)

- Contoh keluaran program



The screenshot shows an IDE's output window with the following tabs: 'Report Problems Window', 'iReport output', 'Notifications', and 'Output - lat_Dokter (run)'. The 'Output - lat_Dokter (run)' tab is active, displaying the following text:

```
run:
Id Dokter      : 202108001
Nama Dokter    : Kiky
Gaji Dokter    : 20000000

<<Menampilkan Data>>
Berkas Dokter
Id Dokter      : 202108001
Nama Dokter    : Kiky
Gaji Dokter    : 20000000
Tunjangan      : 2000000.00
Total Gaji     : 22000000.00
BUILD SUCCESSFUL (total time: 14 seconds)
```


Latihan Tambahan

STATIC METHOD

Nama kelas: Dokter

```
1. package lat_staticmethod;
2. public class Dokter {
3.     //mendeklarasikan atribut static
4.     static String idDokter;
5.     static String nama;
6.     static int gaji;
7.
8.     //membuat static method dengan fungsi
9.     public static float tunjangan(int Gaji)
10.    {
11.        return Gaji/100*10;
12.    }
13.
14.    //membuat static method dengan fungsi
15.    public static float total_gaji(int Gaji)
16.    {
17.        return Gaji + tunjangan(Gaji);
18.    }
19. }
```

Latihan Tambahan

STATIC METHOD(Lanj...)

Main program: Lat_StaticMethod

```
1. package lat_staticmethod;
2. import java.util.Scanner;
3. public class Lat_StaticMethod {
4.     public static void main(String[] args) {
5.         // TODO code application logic here
6.         //membuat scanner
7.         Scanner input = new Scanner(System.in);
8.
9.         System.out.println("<<Masukan Data>>");
10.        System.out.print("ID Dokter \t:");
11.        Dokter.idDokter = input.nextLine();
12.        System.out.print("Nama Dokter \t:");
13.        Dokter.nama = input.nextLine();
14.        System.out.print("Gaji Dokter \t:");
15.        Dokter.gaji = input.nextInt();
16.
```

Latihan Tambahan

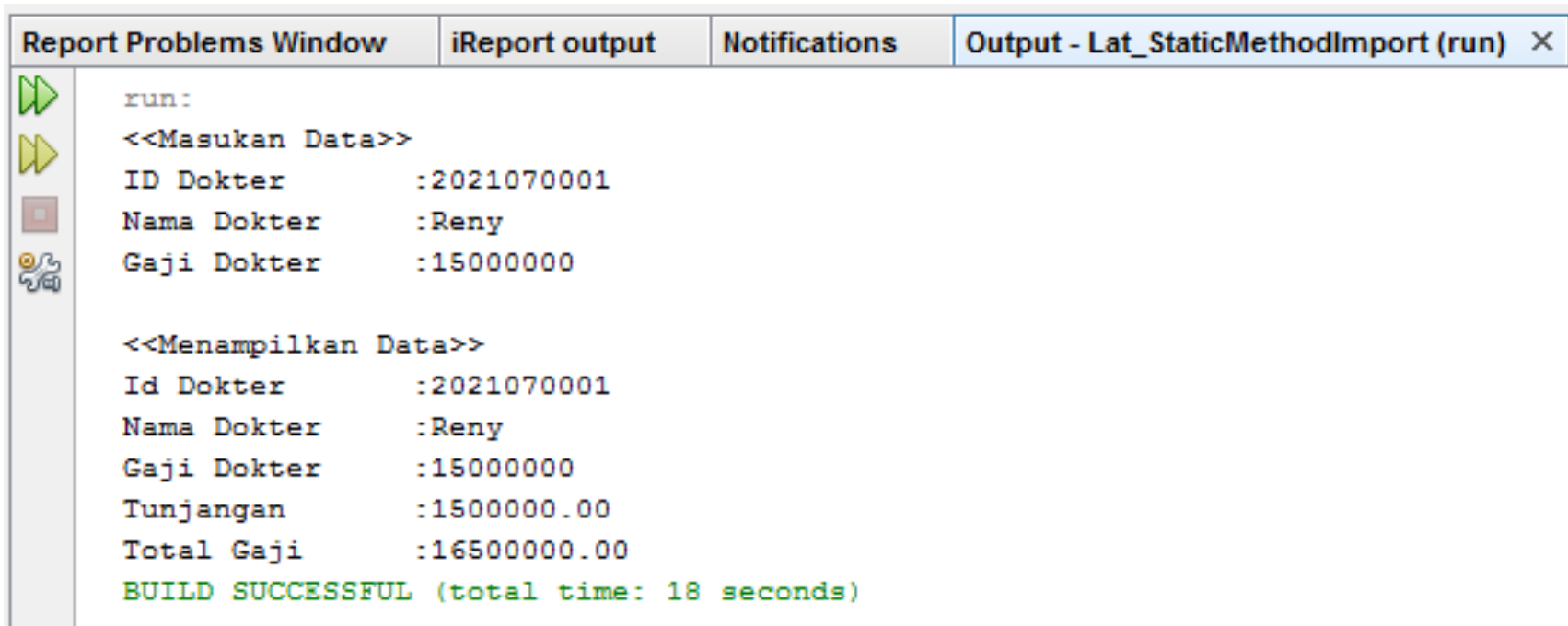
STATIC METHOD(Lanj...)

```
17.      //menampilkan data
18.      System.out.println();
19.      System.out.println("<<Menampilkan Data>>");
20.      System.out.println("Id Dokter \t:" + Dokter.idDokter);
21.      System.out.println("Nama Dokter \t:" + Dokter.nama);
22.      System.out.println("Gaji Dokter \t:" + Dokter.gaji);
23.      System.out.printf("Tunjangan \t:%.2f\n", +
24.          Dokter.tunjangan(Dokter.gaji));
25.      System.out.printf("Total Gaji \t:%.2f\n", +
26.          Dokter.total_gaji(Dokter.gaji));
27.  }
28. }
```

Latihan Tambahan

STATIC METHOD(Lanj...)

- Contoh keluaran program:



The screenshot shows the Eclipse IDE's Output Window with the 'Output - Lat_StaticMethodImport (run)' tab selected. The output text is as follows:

```
run:
<<Masukan Data>>
ID Dokter      :2021070001
Nama Dokter    :Reny
Gaji Dokter    :15000000

<<Menampilkan Data>>
Id Dokter      :2021070001
Nama Dokter    :Reny
Gaji Dokter    :15000000
Tunjangan      :1500000.00
Total Gaji     :16500000.00
BUILD SUCCESSFUL (total time: 18 seconds)
```

Ringkasan:

- Prosedur dan fungsi digunakan untuk mengerjakan tugas sesuai dengan yang ditugaskan.
- Perbedaan prosedur dan fungsi adalah: prosedur tidak dapat mengembalikan nilai, sedangkan fungsi dapat mengembalikan nilai.

Ringkasan: (Lanj...)

- Pada konsep pemrograman berorientasi objek pembuatan method dapat dilakukan dengan menggunakan prosedur dan fungsi.
- Dalam membuat method dapat menggunakan method static dan non static, akan tetapi pada kenyataanya lebih banyak menggunakan method non static dalam membuat method.

Latihan Mandiri

clsPegawai
+ NIK : String + Nama : String + Gaji : int
+ UangMakan() : float + Transport() : float + TotalGaji() : float

- Program menghitung gaji pegawai, dengan atributnya adalah: NIK, Nama, Gaji.
- Methodnya adalah: Uang_Makan, Transport. Uang_Makan sebesar 10% dari gaji, dan Transport 10% dari gaji, TotalGaji = Gaji + Uang_Makan + Transport.

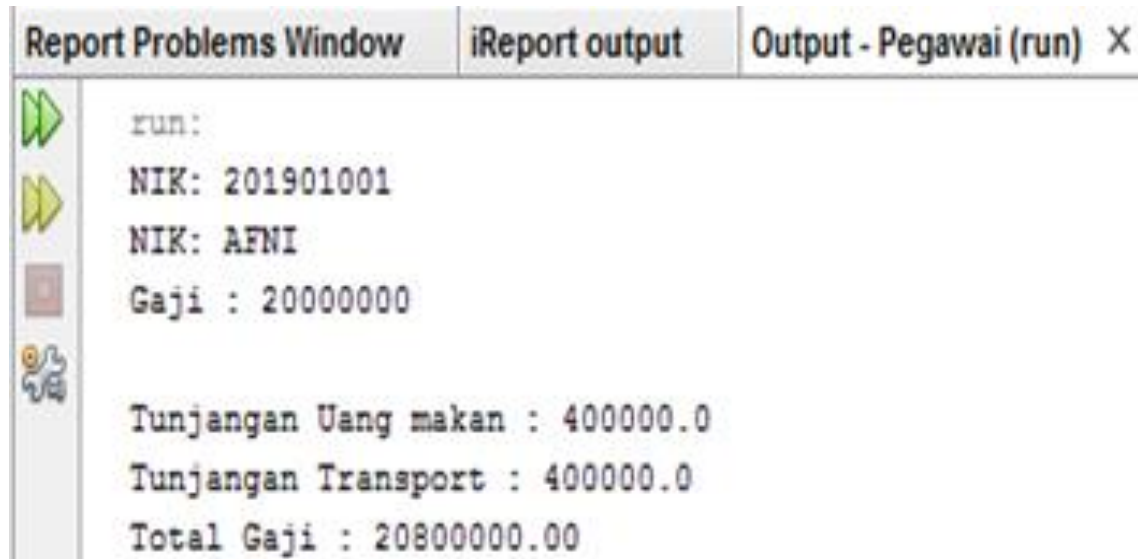
Latihan Mandiri (Lanj...)

- Contoh keluaran programnya.



The image shows three separate input dialog boxes, each titled 'Input' with a red close button. Each dialog contains a green question mark icon, a label, a text input field, and 'OK' and 'Cancel' buttons.

- Dialog 1:** Label 'NIK:', input field contains '201901001'.
- Dialog 2:** Label 'Nama:', input field contains 'AFNI'.
- Dialog 3:** Label 'Gaji:', input field contains '20000000'.



The image shows a screenshot of an IDE's output window with three tabs: 'Report Problems Window', 'iReport output', and 'Output - Pegawai (run)'. The 'Output - Pegawai (run)' tab is active, displaying the following text:

```
run:  
NIK: 201901001  
NIK: AFNI  
Gaji : 20000000  
  
Tunjangan Uang makan : 400000.0  
Tunjangan Transport : 400000.0  
Total Gaji : 20800000.00
```




TERIMA KASIH

U N I V E R S I T A S B U N D A M U L I A