

Input Output

Pemrograman Java terdiri dari 3 komponen dasar yaitu Input, Proses dan Output. Dari materi sebelumnya juga kita sudah belajar menggunakan beberapa kode yang merupakan bagian dari Proses dan Output. Sudahkah Anda menyadarinya? Pada materi ini kita akan mempelajari bagaimana mengambil hasil Input hingga bagaimana hasil Input diproses dan ditampilkan pada layar.

Lantas bagaimana Input, Proses dan Output itu?

- Input : Nilai yang dimasukkan
- Proses : Serangkaian langkah yang dilakukan untuk mengelola input yang diberikan
- Output : Menampilkan hasil olah data.

Pada bahasa pemrograman Java, *Basic Input*-nya sudah dibekali beberapa *library* untuk membantu pengambilan *Input* berbasis teks. Beberapa di antaranya:

- `BufferedReader`
- `Scanner`

Sedangkan untuk menampilkan atau Output dari program adalah:

- `Print`
- `Println`

Berikut bagaimana masing-masing kelas yang disediakan oleh Bahasa Pemrograman Java untuk mengambil input dari user.

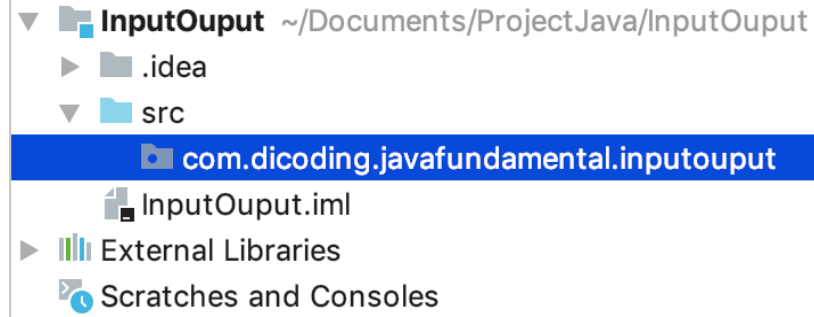
Scanner

`Scanner` merupakan kelas yang menyediakan fungsi-fungsi untuk membaca dan mengambil *input* dari pengguna. `Scanner` memiliki kemudahan yang dapat membaca teks, baik yang memiliki tipe data primitif maupun string.

Codelab Scanner

Mari kita coba kode di bawah ini untuk implementasi materi Scanner ini.

1. Buatlah proyek baru dengan nama InputOutput dengan nama package `com.dicoding.javafundamental.inputoutput` di dalamnya:

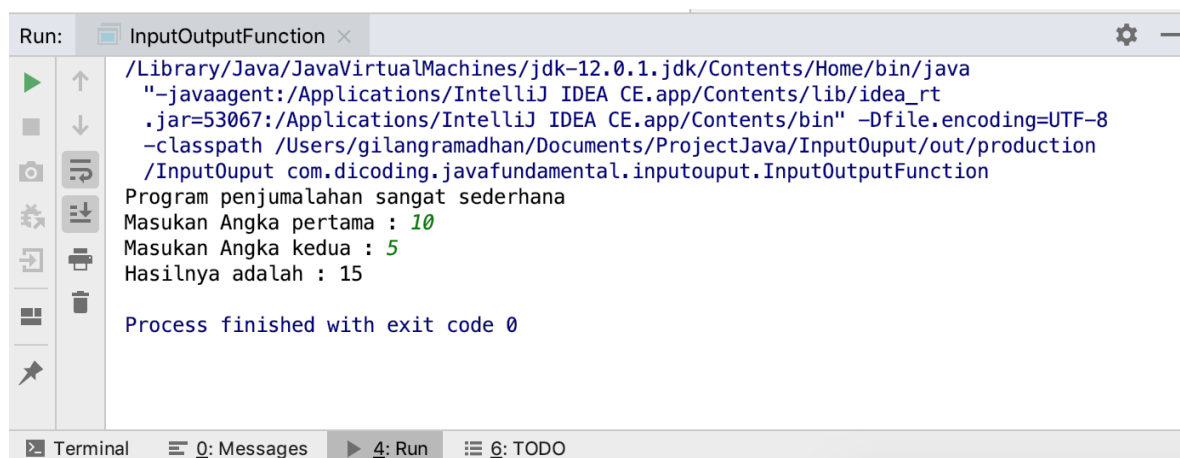


2. Buatlah sebuah kelas baru di dalamnya dengan nama `InputOutputFunction`, kemudian tambahkan kode berikut:

```
1. package com.dicoding.javafundamental.inputouput;
2.
3. import java.util.Scanner;
4.
5. public class InputOutputFunction {
6.
7.     public static void main(String[] args) {
8.         Scanner scanner = new Scanner(System.in);
9.         System.out.println("Program penjumlahan sangat sederhana");
10.        System.out.print("Masukan Angka pertama : ");
11.        int value = scanner.nextInt();
12.        System.out.print("Masukan Angka kedua : ");
13.        int anotherValue = scanner.nextInt();
14.        int result = value + anotherValue;
15.        System.out.println("Hasilnya adalah : " + result);
16.    }
17. }
```

3. Selanjutnya jalankanlah kode di atas pada IDE yang kalian gunakan. Bila sukses, seharusnya Console akan menampilkan output seperti ini.

```
Program penjumlahan sangat sederhana
Masukan Angka pertama : 10
Masukan Angka kedua : 5
Hasilnya adalah : 15
```



Bedah Code Scanner

Perhatikan kode di bawah ini:

```
1. Scanner scanner = new Scanner(System.in);
```

Kode di atas merupakan inisialisasi awal ketika akan menggunakan `Scanner`. Di bagian ini kita mencoba untuk menggunakan perintah mengambil *input* yang diberikan oleh *user*. Bila program dijalankan, setiap *input* yang diberikan akan disimpan oleh `Scanner`.

```
1. System.out.print("Masukan Angka pertama : ");
2. int value = scanner.nextInt();
```

Selanjutnya kita akan mengambil data yang telah dimasukkan melalui `Scanner`. Dengan memanggil `Scanner.nextInt()` setiap Input dari keyboard akan diberikan ke `value`.

Adapun catatan yang harus diperhatikan adalah penggunaan fungsi `Scanner`. Pengambilan data sangat bergantung pada tipe data yang dimasukan. Ini harus sesuai, misalnya untuk string, panggilah `Scanner.next()`. Jika data yang diinputkan Integer, panggilah `Scanner.nextInt()`. Jika *input* tidak sesuai, program akan menemui eror.

BufferedReader

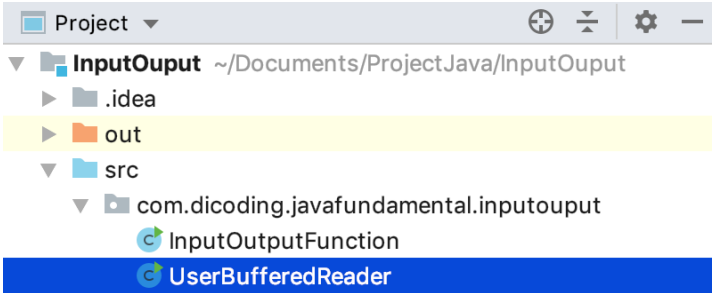
Kelas dalam Java ini merupakan paket dari `Java.io`. `BufferedReader` dapat digunakan pada materi ini sebagai *Basic Input* karena sebenarnya kelas ini tidak hanya digunakan untuk membaca input dari *keyboard* saja, melainkan juga untuk mendapatkan *input* dari *user*. Fungsi terakhir ini adalah fungsi dasar `BufferedReader` yang sama dengan `Scanner`.

Dalam implementasinya `BufferedReader` tidak dapat berjalan sendiri. Untuk mendapatkan input dibutuhkan kelas `InputStreamReader`.

Codelab BufferedReader

Pada codelab kali ini, kita akan memanfaatkan `BufferedReader` untuk mendapatkan *input* dari user.

1. Bukalah kembali proyek InputOutput dan buatlah kelas baru dengan nama `UserBufferedReader`.



2. Masukkan kode berikut ke dalam kelas `UserBufferedReader`:

```
1. package com.dicoding.javafundamental.inputouput;
2.
3. import java.io.BufferedReader;
4. import java.io.IOException;
5. import java.io.InputStreamReader;
6.
7. public class UserBufferedReader {
8.
9.     public static void main(String[] args) {
10.
11.         InputStreamReader streamReader = new InputStreamReader(System.in);
12.         BufferedReader bufferedReader = new BufferedReader(streamReader);
13.         System.out.println("Program penjumlahan sangat sederhana");
14.         int value = 0;
15.         int anotherValue = 0;
```

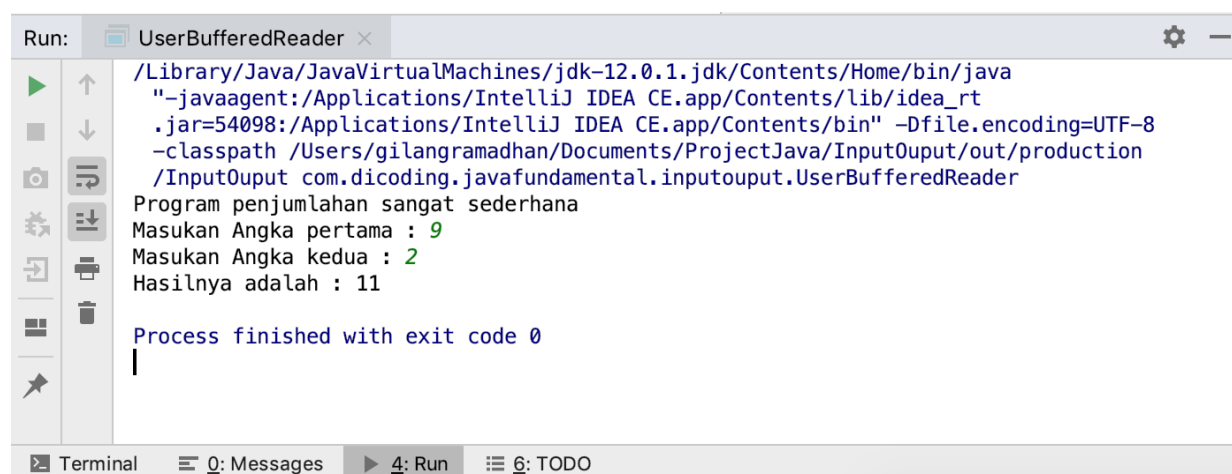
3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

Program penjumlahan sangat sederhana

Masukan Angka pertama : 9

Masukan Angka kedua : 2

Hasilnya adalah : 11



```
Run: UserBufferedReader x
/Library/Java/JavaVirtualMachines/jdk-12.0.1.jdk/Contents/Home/bin/java
"-javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/idea_rt
.jar=54098:/Applications/IntelliJ IDEA CE.app/Contents/bin" -Dfile.encoding=UTF-8
-classpath /Users/gilangramadhan/Documents/ProjectJava/InputOuput/out/production
/InputOuput com.dicoding.javafundamental.inputouput.UserBufferedReader
Program penjumlahan sangat sederhana
Masukan Angka pertama : 9
Masukan Angka kedua : 2
Hasilnya adalah : 11
Process finished with exit code 0
```

Bedah Code BufferedReader

Mari kita kupas bagaimana kode di atas berjalan.

```
1. InputStreamReader streamReader = new InputStreamReader(System.in);
2. BufferedReader bufferedReader = new BufferedReader(streamReader);
```

Tidak jauh berbeda dengan cara dari penggunaan `Scanner`, kita harus menginisialisasi `BufferedReader` yang dibantu oleh `InputStreamReader` agar dapat membaca *Input* dari *Keyboard*.

```

1. try {
2.     System.out.print("Masukan Angka pertama : ");
3.     value = Integer.parseInt(bufferedReader.readLine());
4.     System.out.print("Masukan Angka kedua : ");
5.     anotherValue = Integer.parseInt(bufferedReader.readLine());
6.
7. } catch (IOException e) {
8.     e.printStackTrace();
9. }

```

Namun sedikit berbeda dengan implementasi `Scanner`, tambahkan `IOException` sebagai penanganan *error input* pada `BufferedReader`. Selain itu untuk menyimpan pada sebuah variabel, data yang diberikan oleh `BufferedReader` harus kita parsing terlebih dahulu.

```

1. int value = Integer.parseInt(bufferedReader.readLine());

```

Seperti pada potongan kode diatas, bila ingin menyimpan data pada int maka nilai yang ada pada `bufferedReader.readLine()` harus dikonversi dahulu ke dalam bentuk integer menggunakan `Integer.parseInt`.

print dan println

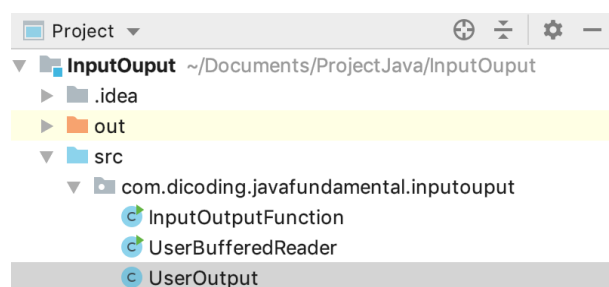
Tentunya kita sudah familiar bagaimana memberikan output dari yang sudah kita praktikkan pada beberapa contoh program sebelumnya. Pada dasarnya kedua fungsi di atas sama-sama memiliki peran untuk menampilkan teks di Console maupun dari IDE yang kita gunakan.

Apa yang berbeda diantara 2 fungsi di atas? Berikut implementasinya:

Codelab print dan println

Pada codelab kali ini, kita akan memanfaatkan print dan println:

1. Bukalah kembali proyek InputOutput dan buatlah kelas baru dengan nama `UserOutput`.



2. Masukkan kode berikut ke dalam kelas `UserOutput`:

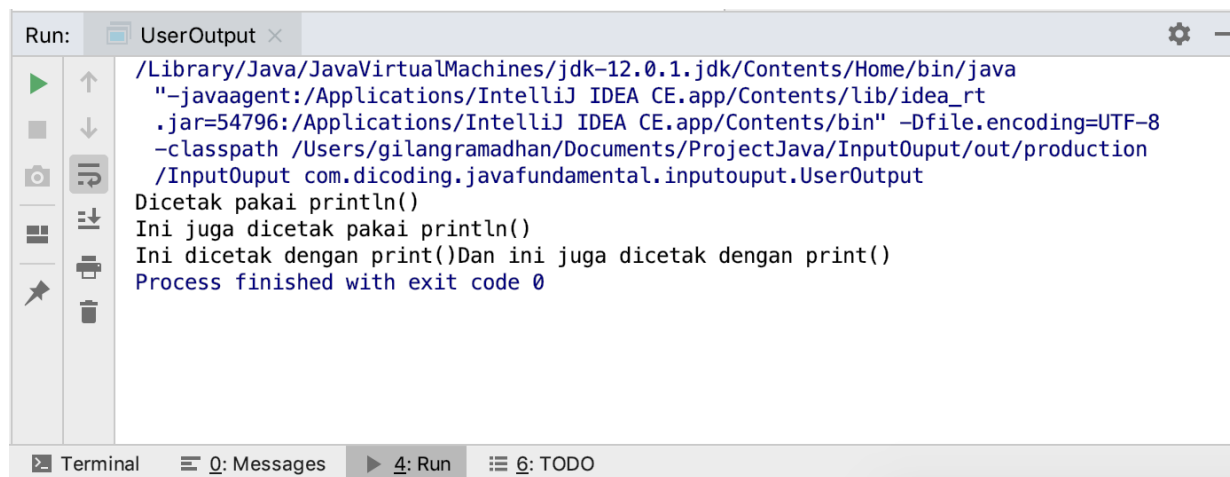
```
1. package com.dicoding.javafundamental.inputouput;
2.
3. public class UserOutput {
4.
5.     public static void main(String[] args) {
6.         System.out.println("Dicetak pakai println()");
7.         System.out.println("Ini juga dicetak pakai println()");
8.         System.out.print("Ini dicetak dengan print()");
9.         System.out.print(" dan ini juga dicetak dengan print()");
10.    }
11. }
```

3. Jalankan kode di atas maka hasilnya akan jadi seperti ini:

Dicetak pakai println()

Ini juga dicetak pakai println()

Ini dicetak dengan print() dan ini juga dicetak dengan print()



Bedah Code print dan println

Bila kode di atas kita jalankan, Ouput yang diberikan kurang lebih seperti di bawah ini:

```
1. Dicetak pakai println()
2. Ini juga dicetak pakai println()
3. Ini dicetak dengan print() dan ini juga dicetak dengan print()
```

Saat program dijalankan, semua kode yang di-*print* di atas akan keluar sesuai fungsi yang kita pakai. Kita bedah kode diatas.

```
1. System.out.println("Dicetak pakai println()");
```

Penggunaan `println` akan menampilkan teks dan tambahan baris baru.

```
1. System.out.print("Ini dicetak dengan print()");
2. System.out.print(" dan ini juga dicetak dengan print()");
```

Sedangkan fungsi `print` menampilkan keluaran berupa teks sesuai dengan yang dimasukkan.

Jika Anda kesulitan, silakan lihat hasil jadinya di [Source Code InputOuput](#).

[← Sebelumnya](#)

[Selanjutnya →](#)