

Bab 1:

Mengenal Bahasa Pemrograman Java

1.1 Kopetensi Dasar

Pada pembahasan Bab 1 ini penulis mengajak mendiskusikan mengenai apa itu bahasa pemrograman Java. Kopetensi dasar secara umum, agar pembaca bisa mendeskripsikan dapat mendeskripsikan penggunaan konsep pemrograman berbasis objek.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- a. Mengenal Program Java dan Membuat Program Java pada Editor Java.
- b. Menterjemahkan dan Menjalankan Program Java
- c. Konsep Pemrograman Berbasis Objek
- d. Karakteristik Pemrograman Berbasis Objek.

1.2 Mengenal Program Java

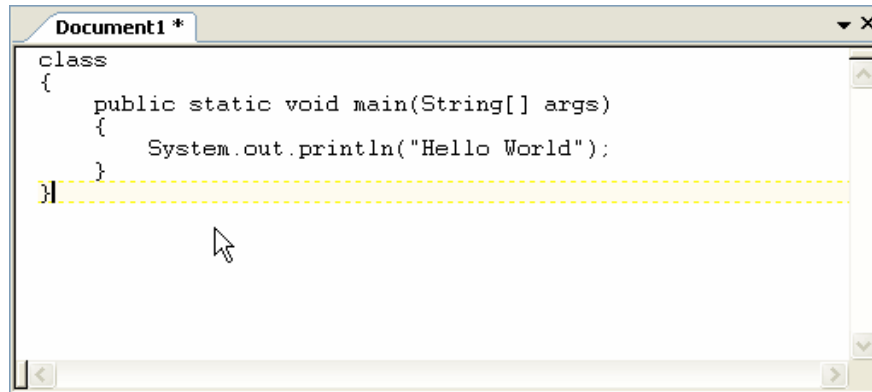
Dalam subbab ini, kita akan membuat dan mengenal program java yang ada sekarang ini. Secara umum, program java dibagi menjadi 2(dua) macam, yaitu Java Application dan Java Applet. Lebih lanjut akan dibahas pada penjelasan dibawah ini dan sekaligus diterapkan pada editor Java. Untuk saat ini penulis menggunakan editor TextPad yang bisa didownload secara gratis.

File Program Java merupakan File Program yang dapat dikompile, dan dijalankan untuk menampilkan hasilnya serta mempunyai ekstensi .java.

Langkah untuk mengaktifkannya adalah :

- a. Klik Menu File
- b. Klik New

- c. Kemudian tampil Dokumen Baru. Selanjutnya anda bisa mengetikkan kode-kode program seperti dibawah ini



Gambar 1.1. Jendela Text pada TextPad

Didalam pemrograman Java, terdapat 2 (dua) bentuk program java, yaitu Java Application dan Java Applets

1.2.1 Java Application

Java Application adalah program yang dapat dijalankan secara langsung, bersifat aplikasi, tanpa perangkat tambahan untuk menjalankannya.

Berikut contoh program Java Application sederhana seperti contoh dibawah ini :

```
1 class Lat101
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello World !");
6     }
7 }
```

Berikut penjelasan mengenai contoh program diatas.

1. *Class Lat101*, merupakan mengawali pendefenisian *Class*, dan diikuti dengan nama *Class* yaitu : *Lat101*.
2. *main* adalah metode yang digunakan untuk mengawali segala bentuk eksekusi pada program java. Metode *main* ini, didefinisikan sebagai *public static void*, yang memiliki arti ;
 - a. *public*, yang berarti metode ini bisa dipanggil dan digunakan didalam *Class* atau diluar *Class*.
 - b. *static*, yang berarti memiliki sifat yang sama disemua instant *Class*.

- c. void, yang berarti bahwa metode ini tidak mengirimkan nilai balik
- 3. Didalam metode main, terdapat String[] args, yang memiliki pengertian ;
 - a. String[], adalah tipe data objek yang menangani serangkaian karakter-karakter yang berjenis array.
 - b. args, adalah variabel objek.
- 4. System.out.println("Hello World !"); , yang memiliki pengertian;
 - a. System.out, adalah stream yang digunakan untuk menangani keluaran standar java.
 - b. println(); , merupakan metode yang digunakan untuk menampilkan teks dilayar.
 - c. Tanda titik koma (;), digunakan untuk mengakhiri pernyataan.

1.2.2 Java Applet

Java Applet adalah program java bisaa diletakan diwebserver dan dijalankannya menggunakan web browser.

Berikut contoh program Java Applet sederhana seperti contoh dibawah ini:

```
1 import java.awt.*;  
2 import java.applet.*;  
3  
4 public Class Lat102 extends Applet  
5 {  
6     public void paint(Graphics g)  
7     {  
8         g.drawString("Hello World", 20, 20);  
9     }  
10 }
```

Berikut penjelasan mengenai contoh program diatas.

- 1. public Class Lat102 extends Applet, merupakan mengawali pendefenisian Class, dan diikuti dengan nama Class yaitu : Lat102. Bersifat public yang merupakan turunan dari Applet.
- 2. public void paint(Graphics g) adalah menggunakan metode paint untuk menggambar semua graphic applet didrawing area, dengan parameter Class abstrak untuk merepresentasikan area applet.
- 3. g.drawString("Hello World", 20, 20) adalah untuk mencetak text Hello Word, pada posisi baris 20 dan posisi kolom 20.

Buatlah sebuah file html seperti dibawah ini, yang nantinya untuk menampilkan hasilnya diweb browser.

```
1 <HTML>
2 <HEAD>
3   <APPLET
4     CODE = "Lat102.Class"
5     HEIGHT = 300 WIDTH = 300>
6   </APPLET>
7 </HEAD>
8 </HTML>
```



1.3 Menyimpan File Java

Perlu diketahui bahwa, untuk menyimpan file program java, kita tidak boleh sembarangan. Untuk nama file program java, harus sesuai dengan nama *Class*. Jika nama *Class* Lat101, harus menyimpan dengan nama file Lat101.java, dan tidak diperbolehkan lat101.java atau yang lain. Perlu diingat, huruf besar atau huruf kecil sangat berpengaruh pada penyimpanan file.

Setelah selesai mengetikan naskah program yang baru pada jendela Text Edit, maka selanjutnya disimpan dengan cara :

1. Kik Menu File → Save
2. Atau Menekan HotKey Ctrl + S.

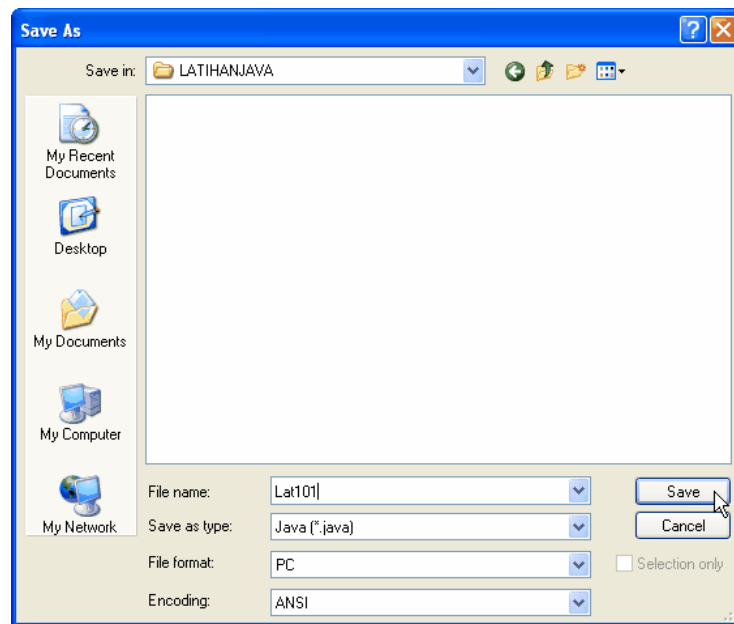
Sebagai latihan anda, buatlah folder kerja anda, anda dapat ikuti langkah berikut :

1. Klik icon Up One Level [] sampai ke drive C: , atau dengan cara klik combobox Look In kemudian pilih drive C:
2. Klik icon Create New Folder [], maka akan tampil New Folder seperti berikut :



3. Kemudian tuliskan nama foldernya : LATIHANJAVA. Tekan tombol Enter.
4. Kemudian tekan tombol Enter lagi, sehingga Look In, menunjukan folder LATIHANJAVA.
5. Tuliskan pada kotak isian File Name, dengan nama LAT101. Tekan tombol Enter atau Klik tombol Open. Maka selanjutnya file anda telah tersimpan.

File Editor memiliki ekstensi file adalah .java. Maka file yang anda simpan menjadi LAT101.java



Gambar 1.2. Menyimpan file pada Folder yang telah ditentukan

Pada Textpad terdapat tiga cara menyimpan file editor, diantaranya yaitu :

1. Save digunakan untuk menyimpan File Program pada jendela yang sedang aktif kedalam disk. Hotkey yang ada bisa gunakan untuk menyimpan dengan menekan tombol Ctrl + S.
2. Save As digunakan untuk menyimpan File Program pada jendela yang sedang aktif kedalam disk dengan nama file yang berbeda.
3. Save All digunakan untuk menyimpan semua File Program pada jendela yang sedang aktif kedalam disk.

Setelah itu simpan juga file yang berisi program Java Applet, dengan nama : Lat102.java dan file berjenis html, dengan nama BrowseLat102.html

1.4 Menterjemahkan Program

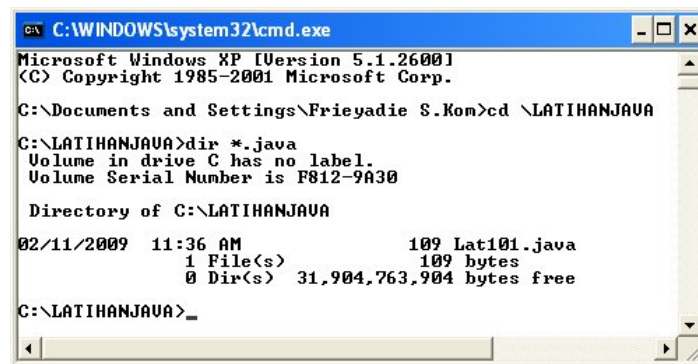
Proses Compile merupakan suatu proses menterjemahkan program dari bahasa manusia kedalam bahasa yang dimengerti oleh komputer yaitu bahasa mesin. Proses Compile program Java Application atau Java Applet sama, tidak dibedakan. Berikut langkah yang dapat anda ikuti untuk menterjemahkan program adalah :

1.4.1 Melalui Command Prompt

Untuk melakukan kompilasi program Java, anda bisa melakukannya melalui Command Prompt atau jendela Command

Terlebih dulu aktifkan jendela Command melalui Run dengan menuliskan perintah cmd, kemudian klik tombol OK. Kemudian tampil jendela Command.

Aktifkan folder / direktori tempat file java anda disimpan, seperti pada gambar dibawah ini:

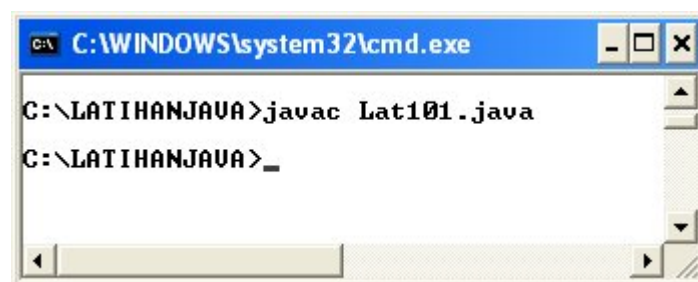


Gambar 1.3. Jendela Command

Untuk kompilasi perintah yang digunakan yaitu javac diikuti dengan nama_file_java.java. Perintah ini akan menghasilkan file bytecode dengan ekstensi .Class, nama file ini sama dengan nama file java, hanya ekstensinya yang berbeda. Berikut bentuk umum perintah compile seperti dibawah ini :

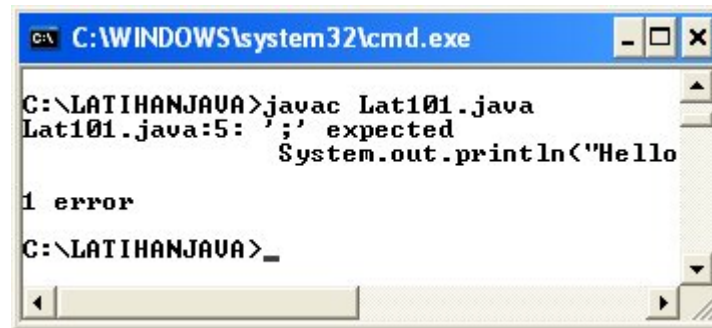
```
javac nama_file_java_application.java
```

Lakukan kompilasi file Lat101.java, seperti gambar dibawah ini :



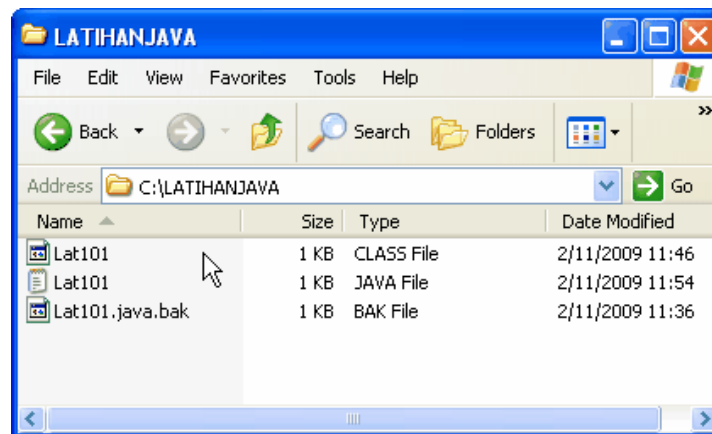
Gambar 1.4. Kompilasi File Java

Jika tidak ada kesalahan dalam proses kompilasi maka, tidak ada pesan apa-apa, hanya kembali ke cursor saja. Tetapi jika terjadi kesalahan maka, akan diberitahukan code program mana yang salah dan pada baris keberapa yang salah, maka dengan cara ini anda akan bisa mengetahui posisi kesalahan pada program anda. Contoh seperti dibawah ini :



Gambar 1.5. Terjadi Kesalahan pada Kompilasi

Hasil Penterjemahan seperti dijelaskan diatas, menghasilkan file bytecode dengan ekstensi **.Class**, nama file ini sama dengan nama file java.

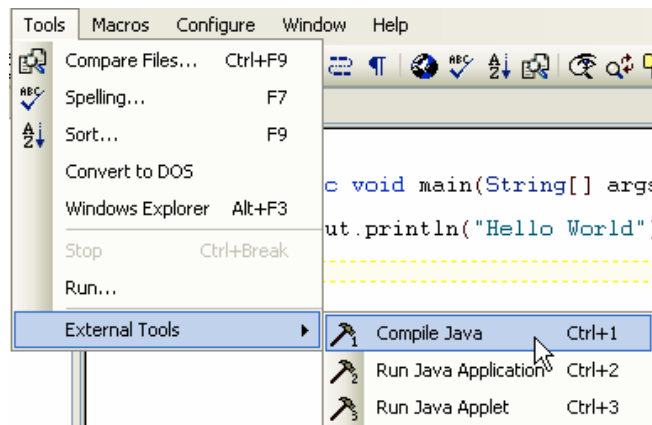


Gambar 1.6. File .CLASS hasil dari Compile

1.4.2 Melalui Tools

Jika anda menggunakan TextPad, anda bisa melakukannya melalui hotkey atau perintah yang sudah disediakan. Ikuti langkah-langkah seperti dibawah ini :

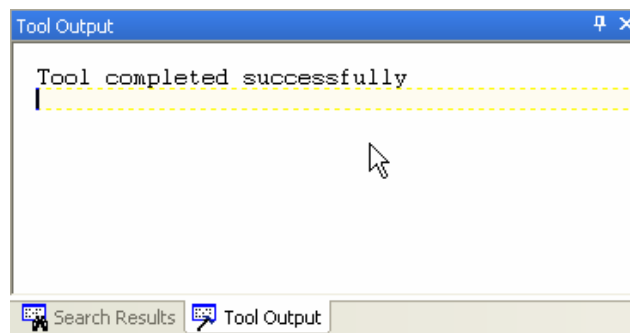
1. Klik Menu Tools
2. Klik External Tools
3. Pilihlah salah satu sesuai dengan kebutuhan proses yang akan dilaksanakan.



Gambar 1.7 Mengubah User ToolGroup

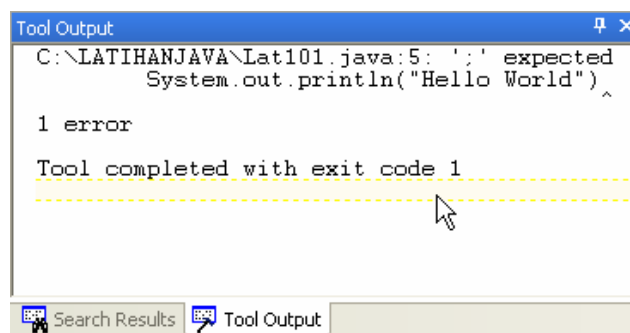
Compile Java, untuk melakukan proses penterjemahan program java. Anda bisa langsung menekan tombol Ctrl + 1

Sekarang coba anda lakukan compile (tekan tombol Ctrl dan tombol 1), program Lat101.java, maka jika benar, maka akan tampil seperti gambar dibawah ini :



Gambar 1.8. Proses Compile yang Berhasil

Maka jika terdapat kesalahan (misal: kurang titik koma diakhir perintah), maka akan tampil seperti gambar dibawah ini :



Gambar 1.9. Proses Compile yang Gagal

1.5 Menjalankan Program

Setelah melakukan proses compiling, maka selanjutnya melihat hasil yang telah dcompile, berikut beberapa langkah melalui command prompt dan tools :

1.5.1 Melalui Command Prompt

Berikut langkah-langkah running program melalui command prompt:

Aktifkan jendela command

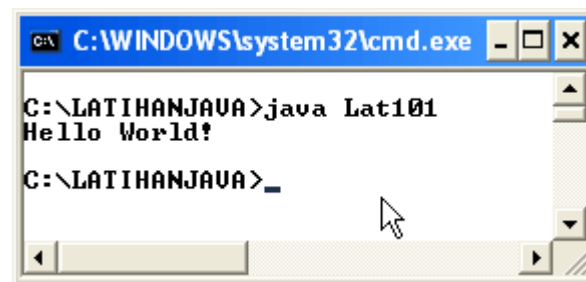
Berikut perintah-perintah untuk menjalankan program, disesuaikan dengan jenisnya java, yaitu :

a. Menjalankan Java Application

Perintah yang digunakan untuk menjalankan java application, yaitu :

```
javac nama_file_Class
```

Pada penulisannya nama_file_Class yang akan digunakan tidak perlu menuliskan extensinya dibelakang nama file.



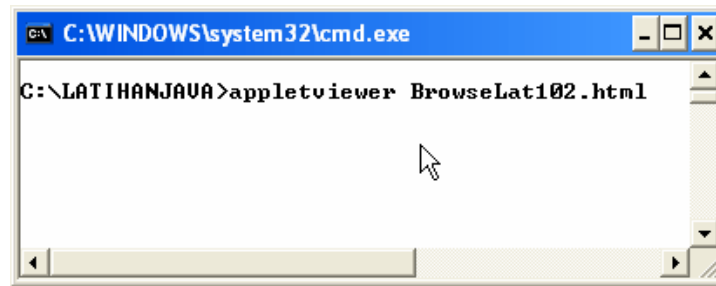
Gambar 1.10. Menjalankan Java Application

b. Menjalankan Java Applet

Perintah yang digunakan untuk menjalankan java application, yaitu :

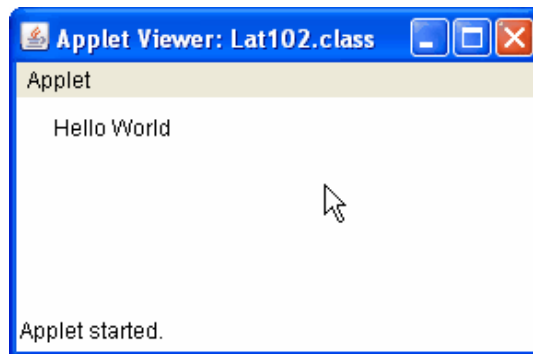
```
appletviewer nama_file_html
```

Pada penulisannya nama_file_Class yang akan digunakan tidak perlu menuliskan extensinya dibelakang nama file.



Gambar 1.11. Menjalankan Java Applet

Setelah anda tekan tombol Enter, maka tampilan applet yang dihasilkan seperti dibawah ini.



Gambar 1.12. Menjalankan Java Applet

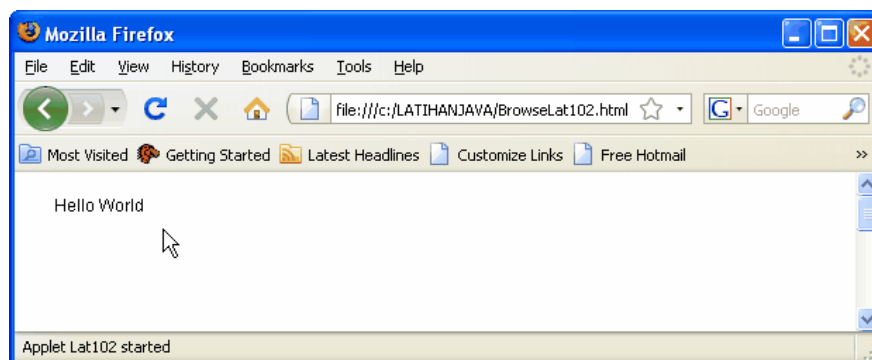
Bisa juga dijalankan diweb browser, dengan cara :

Klik menu File pada web browser | klik dan pilih OpenFile

Arahkan ke nama file yang terdapat file yang telah anda buat diatas.

file:///c:/LATIHANJAVA/BrowseLat102.html

hasilnya akan seperti gambar dibawah ini



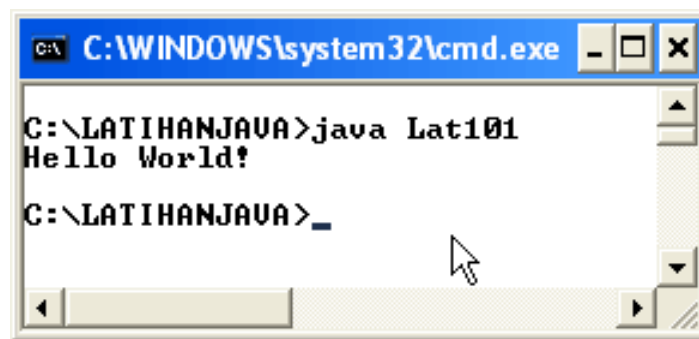
Gambar 1.13. Menjalankan Java Applet pada WebBrowser

1.5.2 Melalui Tools

Jika anda menggunakan TextPad, anda bisa melakukannya melalui hotkey atau perintah yang sudah disediakan. Ikuti langkah-langkah seperti dibawah ini :

a. Menjalankan Java Application

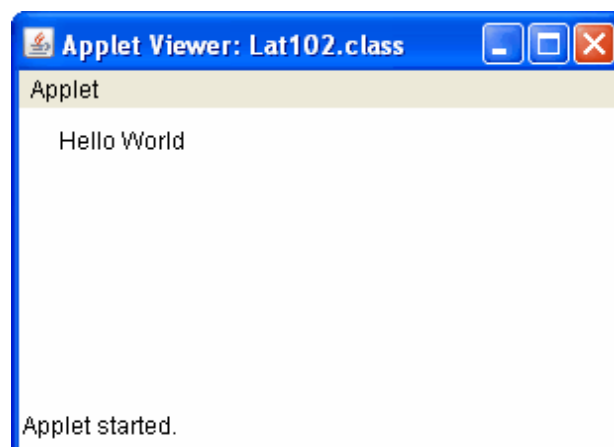
Untuk menjalankan atau Run Java Application, anda bisa langsung menekan tombol Ctrl + 2. Maka akan tampil seperti gambar dibawah ini :



Gambar 1.14. Menjalankan Java Application

b. Menjalankan Java Application

Untuk menjalankan atau Run Java Application, anda bisa langsung menekan tombol Ctrl + 3. Maka akan tampil seperti gambar dibawah ini :



Gambar 1.15. Menjalankan Java Applet

1.6 Konsep Pemrograman Berbasis Objek

Pada konsep pemrograman berbasis objek ini, kita akan membahas mengenai objek, *Class* dan *Method*. Berikut penjelasan singkat dan padat mengenai ketiga konsep diatas.

1.6.1 *Object* dan *Class*

Dalam Pemrograman Berorientasi Objek melihat atau memandang sesuatu berdasarkan objek. Objek sebenarnya mencerminkan pola kerja manusia dalam kehidupan sehari-hari.

Pada suatu objek dapat dilihat menjadi 2 (dua) hal, yaitu :

1. *Atributte*

Atribut merupakan segala sesuatu yang melekat pada *Object*. Didalam penerapan didalam program, atribut adalah Variabel atau Member.

Misalkan pada *Object* Burung. Atribut-atribut yang melekat pada burung, misalnya paruh, ekor, sayap, kaki, mata, dan lain-lain.

2. *Behaviour*

Behaviour merupakan pola tingkah laku atau perilaku yang dimiliki oleh objek. Misalnya pada objek Burung memiliki perilaku diantaranya terbang, mengepakkan sayap, berjalan dan lain-lain. Didalam penerapan didalam program, Behaviour adalah *Method* atau Fungsi.

Bentuk penulisan *Class*, seperti dibawah ini :

```
[public | private] [abstract] Class Nama_Class
{
    ... daftar property...
    ... daftar Method ...
}
```

Bentuk penulisan mendeklarasikan *Object*, dengan menggunakan **new**, seperti dibawah ini :

```
nama_Class nama_objek = new nama_Class();
```

- `nama_Class`, merupakan nama *Class* yang akan dijadikan objek.
- `nama_objek`, merupakan nama objek baru.

Contoh pembuatan *Class* sederhana :

```
Class burung
{
    String jenis, warna;
    int usia;
```

```

}
Class burung_terbang
{
    public static void main(String[] args)
    {
        //membuat objek
        burung burung_elang = new burung();
        .....
        .....
    }
}

```

1.6.2 Method

Method adalah implementasi operasi yang bisa dilakukan oleh *Class* dan *Object*. Operasi-operasi yang dilakukan oleh *Method*, diantaranya, yaitu :

1. Suatu *Method* bisa menerima dan memanipulasi data atau field didalam diri *Method* tersebut.
2. Suatu *Method* bisa mempengaruhi nilai suatu *Object* lain.

Berikut bentuk penulisan deklarasi *Method*:

```

Tipe_Akses Tipe_Return NamaMethod(Argumen1, Argumen2, ..., Argumen-N)
{
    ... Badan / Tubuh Method ..
}

```

Berikut penjelasan deklarasi *Method* diatas :

1. Tipe Akses, menyatakan tingkatan akses untuk memproteksi akses terhadap data-data didalam *Method*, tipe akses ini bersifat opsional.
2. Tipe Return, menyatakan nilai hasil yang diolah oleh *Method* akan dikembalikan atau akan mengirimkan kepada objek yang memanggil *Method*. Bentuk Tipe Return, bisa berupa tipe data primitive yaitu integer, float, double dan lain-lain.

Apabila *Method* tidak akan mengembalikan nilai kepada objek yang memanggilnya, maka bisa dituliskan didepan nama *Method* dengan perintah void.

1.7 Karakteristik Pemrograman Berbasis Objek

Sekarang ini dalam tahap mempelajari pemrograman berbasis objek, anda harus mengenal karakteristik yang dimiliki pemrograman berbasis objek. Adapun ketiga karakteristik tersebut, yaitu :

1.7.1 Enkapsulasi (Encapsulation)

Karakteristik ini merupakan suatu cara bagaimana menyembunyikan sedemikian rupa suatu proses kedalam sistem, hal ini berguna untuk menghindari interferensi dari luar sistem dan juga lebih untuk menyederhakan sistem itu sendiri.

Kita ambil contoh, pada saat anda mengganti chanel TV menggunakan remote TV, apakah anda mengetahui proses yang terjadi didalam TV tersebut ?, maka jawabannya tidak tau, dan anda pun sebagai pembeli TV tidak mau dipusingkan dengan proses yang terjadi. Maka hal tersebut menyederhakan sistem.

1.7.2 Pewarisan (Inheritance)

Pewarisan, bahasa kerennya *Inheritance*. Dalam pemrograman berbasis objek, dimungkinkan suatu *Class* bisa mewariskan atribut dan *Method* kepada *Class* yang lainnya atau *subClass*, sehingga membentuk *Class* hirarki.

Sebagai contoh, pada saat kita bicara mengenai bus, maka bus tersebut bisa mewarsikan kepada bus yang lain berupa, nomor trayek, body besar, jumlah penumpang banyak dan lain sebagainya.

1.7.3 Polymorphism

Karakteristik dari polymorphism yaitu memungkinkan suatu objek dapat memiliki berbagai bentuk atau banyak bentuk. Bentuk dari objek ini bisa sebagai *Object* dari *Class*nya sendiri atau *Object* dari *superClass*nya.

Pada polymorphism kita akan sering menjumpai 2 (dua) istilah yang sering digunakan dalam pemrograman berbasis objek, istilah tersebut yaitu :

a. *Overloading*.

Overloading yaitu menggunakan 1 (satu) nama objek untuk beberapa *Method* yang berbeda ataupun bisa juga beda parameternya.

b. *Overriding*

Overriding akan terjadi apabila ketika pendeklarasian suatu *Method* *subClass* dengan nama objek dan parameter yang sama dengan *Method* dari *superClass*nya.

1.7.4 Abstrak

Abstrak didalam pemrograman berbasis objek, yaitu dimaksudkan untuk melihat suatu sistem, menjadi lebih sederhana atau simple.

Apabila kita melihat suatu sistem, misalnya motor, maka bisa kita lihat ada apa saja disistem motor ?, yang pasti ada sistem pengapian, sistem rem, sistem oper gigi dan lain sebagainya. Maka kesemua sistem-sistem tersebut kalau kita lihat menjadi satu sistem yang lebih sederhana yaitu sistem motor.

1.7.5 Modularity

Setiap objek didalam pemrograman berbasis objek, memungkinkan bisa dituliskan atau dibuat secara terpisah-pisah dari objek lainnya. Sehingga program bisa lebih mudah dikembangkan dan dimodifikasi.

Kita ambil contoh pada sistem motor, bisa anda bayangkan seandainya sistem rem terebut langsung menyatu pada objek utama pada motor, apabila seandainya ada perbaikan atau mengubah, maka akan membongkar objek utamanya, baru keobjek tujuan, maka hal ini akan makan waktu yang lama. Maka dengan adanya modularity, apabila ada objek yang akan diperbaiki atau dimodifikasi, langsung keobjek tujuannya saja.

1.8 Latihan

Sebagai latihan untuk pemahaman anda mengenai bahasa pemrograman java. Sebagai tugas anda, buatlah artikel pada blog anda masing-masing mengenai Bahasa Pemrograman Java. Gunakan bahasa yang releks, setelah itu kirimkan URL blog anda ke email dosen anda.

Lembar ini sengaja dikosongkan

Bab 2:

Aktifitas Dasar Pemrograman Java

2.1 Kopetensi Dasar

Pada pembahasan Bab 2 ini penulis mengajak mendiskusikan mengenai aktifitas dasar bahasa pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami aktifitas dasar pemrograman java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- a. Menenal Tipe Data Primitif.
- b. Membuat dan Menggunakan Variabel
- c. Penggunaan Operasi I/O Stream

2.2 Menenal Tipe Data Primitif

Didalam pemrograman Java, kita bisa mengklasifikasikan tipe data primitif menjadi beberapa tipe data, yaitu :

- 1 Bertipe Integer terdapat 4 (empat) Tipe Data.
- 2 Bertipe Floating Point sebanyak 2 (dua) Tipe Data
- 3 Satu Tipe Data berjenis Character
- 4 Satu Tipe Data berjenis Boolean yaitu tipe untuk nilai logika.

Berikut kita bahas secara singkat dan padat mengenai keempat kategori tipe data diatas.

2.2.1 Java Integer

Tipe data integer digunakan untuk operasi data bilangan bulat dan perhitungan aritmatika. Berikut keempat tipe data yang tercakup kedalam kategori integer.

Table 2.1. Kategori Integer

Nama Tipe Data	Keyword	Ukuran	Jangkauan Nilai
Byte-Length Integer	byte	8 bit	–128 s.d 127
Short Integer	short	16 bit	–32768 s.d 32767
Integer	int	32 bit	–2147483648 s.d 2147483647
Long Integer	long	64 bit	–223372036854775808 s.d 223372036854775807

2.2.2 Java Floating Point

Floating-point dasarnya digunakan ketika kita mempunyai situasi dimana mendapatkan hasil atau output dalam bentuk desimal dan seluruh angka yang tidak disebutkan dalam tipe data integers. Tipe data yang termasuk kategori ini yaitu float dan double.

Table 2.2. Kategori Floating Point

Nama Tipe Data	Keyword	Ukuran	Jangkauan Nilai
Single-precision Floating Point	float	32 bit, Presisi 6-7 bit	–3.4E38 s.d 3.4E38
Double-precision Floating Point	double	64 bit Presisi 14-15 bit	–1.7E308 s.d 1.7E308

2.2.3 Java Character

Tipe data Character digunakan untuk mendefinisikan sebuah karakter yang merupakan simbol dalam karakter Set, seperti huruf dan angka. Keyword tipe data Character ini yaitu char, dengan ukuran 16 bit.

2.2.4 Java Boolean

Tipe data boolean digunakan untuk menyebut variabel yang hanya mengandung nilai-nilai True atau False, dengan ukuran 1 bit.

Selain tipe data – tipe data Primitive yang dimiliki oleh Java. Java memiliki tipe data class Object. Tipe data class Object yang sering digunakan yaitu String. String disediakan untuk menampung sejumlah character.

2.3 Mengenal Variabel

Variabel adalah suatu tempat menampung data atau konstanta dimemori yang mempunyai nilai atau data yang dapat berubah-ubah selama proses program.

Dalam pemberian nama variabel, mempunyai ketentuan-ketentuan antara lain ;

Tidak boleh ada spasi (cth : gaji bersih) dan dapat menggunakan tanda garis bawah (_) sebagai penghubung (cth : gaji_bersih).

Tidak boleh diawali oleh angka dan menggunakan operator aritmatika.

2.3.1 Deklarasi Variabel

Deklarasi Variabel adalah proses memperkenalkan variabel kepada java dan pendeklarasian tersebut bersifat mutlak karena jika tidak diperkenalkan terlebih dulu maka java tidak menerima variabel tersebut.

Deklarasi Variabel ini meliputi tipe variabel, seperti : integer atau character dan nama variabel itu sendiri. Setiap kali pendeklarasian variabel harus diakhiri oleh tanda titik koma (;).

Bentuk penulisannya :

```
Tipe data nama variabel;
```

Contoh Deklarasi :

```
String nama_mahasiswa;
```

```
char grade;
```

```
float rata_rata ;
```

```
int nilai1, nilai2;
```

2.3.2 Menempatkan Nilai kedalam Variabel

Setelah pendeklarasian Variabel dilaksanakan, selanjutnya variabel tadi bisa anda masukan nilai kedalam variabel. Berikut cara yang mudah untuk menempatkan nilai kedalam variabel.

Berikut Bentuk penulisannya :

```
nama variabel = nilai;
```

Contoh Penempatan Nilai kedalam Variabel :

```
nama_mahasiswa = "Irvan Y. Ardiansyah";  
grade = 'A';  
rata_rata = 95.75;  
nilai1 = 90; nilai2 = 95;
```

Java bisa juga memperbolehkan memberikan nilai yang sama kebeberapa nama variabel yang berbeda. Seperti contoh dibawah ini:

```
a = c = d = 7;
```

Pada contoh diatas variabel a, c, dan d masing-masing berisi nilai 7.

2.4 Membuat Komentar Program

Pada bahasa pemrograman manapun komentar program biasa digunakan untuk memberikan penjelasan baris atau blok program supaya pembaca program atau programmer lainnya supaya bisa mengerti bagian-bagian program tersebut. Ada 3 (tiga) cara memberikan komentar program pada Java, yaitu :

a. End Of Line Comment (//)

Komentar dengan tanda slash ganda (//) disebut dengan end-of-line comment, karena semua perintah program, komentar-komentar atau penjelasan program berada setelah tanda slash ganda, semua dianggap sebagai komentar dan komentar hanya satu baris saja. Sebagai contoh:

```
// isi komentar program
```

```
// a = b + c;
```

b. Multiple Line Comment (/* */)

Komentar dengan tanda slash dan asterik (/ * */) disebut dengan Multiple-Line Comment, karena ini, perintah program, komentar-komentar atau penjelasan program berada dalam apitan tanda slash dan asterik, semua dianggap sebagai komentar, dan komentar bisa lebih dari satu baris. Sebagai contoh :

```
/* isi komentar program bisa
```

```
terdiri dari beberapa baris komentar
```

```
atau informasi */
```

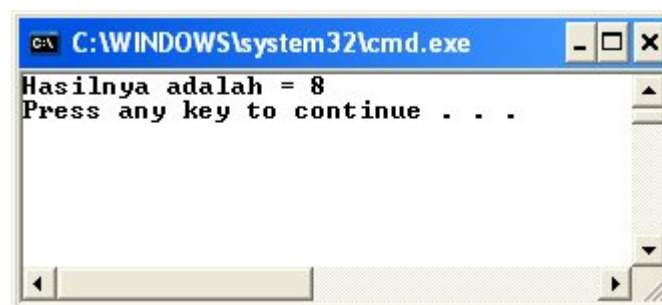
c. Javadoc Comment (/** */)

Komentar dengan tanda slash ganda didepan dan asterik (/** */) disebut dengan Javadoc Comment. Penggunaannya sama seperti Multiple Line Comment, akan tetapi penggunaannya untuk dokumentasi-dokumentasi didalam program. Sebagai contoh :

Semua komentar program atau penjelasan program pada saat program java dicompile tidak ikut serta dicompile, karena tidak dianggap sebagai suatu baris program.

```
1  /* -----  
2      Nama File : Lat201.java  
3      Author   : Frieyadie  
4  ----- */  
5  class Lat201  
6  {  
7      public static void main(String[] args)  
8      {  
9          // deklarasi variabel  
10         int a, b, c;  
11  
12         // memberikan nilai  
13         a = 3;  
14         b = 5;  
15  
16         // proses  
17         c = a + b;  
18  
19         // cetak variabel  
20         System.out.println("Hasilnya = " + c);  
21     }  
22 }
```

Berikut hasil dari program Lat210.java diatas. Maka terlihat komentar atau penjelasan program tidak tampak pada hasil running program.



Gambar 2.1. Hasil Lat201.java

2.5 Perintah Keluaran

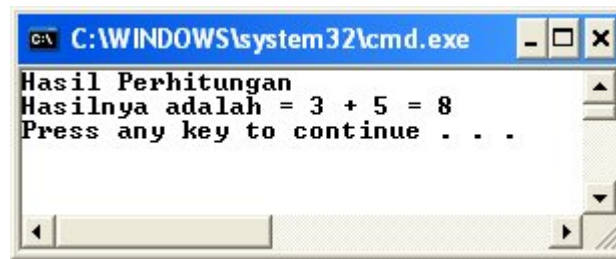
Pada saat melakukan aktivitas dasar pemrograman, pasti tidaklah terlepas dari menampilkan data atau hasil kelayar. Dalam hal ini kita butuh sebuah standard output yaitu stream yang digunakan untuk mengirimkan keluaran kelayar. Stream tersebut yaitu System.out. Pada saat kita akan menampilkan kelayar, kita butuh method print atau println. Perbedaan antara print dan println yaitu :

- System.out.print(Statement); , maka informasi yang ditampilkan dilayar tidak pindah baris.
- System.out.println(Statement); , maka informasi yang ditampilkan dilayar pindah baris (line new). Statement bisa berupa pesan dan argument atau variabel. Statement biasanya diapit dengan tanda kutip ganda (" "), untuk memisahkan antara statement dan argumen atau variabel dipisah dengan tanda plus (+).

Berikut contoh program sederhana perintah keluaran.

```
1  /* -----  
2      Nama File : Lat202.java  
3      Author   : Frieyadie  
4  ----- */  
5  class Lat202  
6  {  
7      public static void main(String[] args)  
8      {  
9          // deklarasi variabel  
10         int a, b, c;  
11  
12         // memberikan nilai  
13         a = 3;  
14         b = 5;  
15  
16         // proses  
17         c = a + b;  
18  
19         // cetak variabel  
20         System.out.println("Hasil Perhitungan");  
21         System.out.print("Hasilnya adalah = " + a);  
22         System.out.print(" + " + b);  
23         System.out.println(" = " + c);  
24     }  
25 }
```

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.2. Hasil Lat201.java

2.6 Perintah Masukan

Untuk melakukan perintah masukan, kita akan menggunakan 3 (tiga) cara, yaitu `InputStream`, `BufferedInputStream` dan `Scanner`.

2.6.1 `InputStream`

`InputStream` adalah subclass `Object`, yang menjadi landasan untuk class-class yang biasa digunakan untuk membaca data dan menampilkan kelayar. Untuk penggunaan `InputStream` ini, harus menyertakan package **java.io**.

Berikut penggunaan `InputStream`, untuk masukan data dari keyboard..

```

1  /* -----
2     Nama File : Lat203.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.io.*;
7
8  class Lat203
9  {
10     public static void main(String[] args)
11     {
12         String kata = "";
13         boolean akhir = false;
14         int huruf;
15
16         System.out.print("Masukkan Kata - Kata Anda : ");
17
18         while(!akhir)
19         {
20             try
21             {
22                 huruf = System.in.read();
23                 if(huruf < -1 || huruf == '\n')
24                     akhir = true;
25                 kata = kata + (char) huruf;
26             }
27             catch (IOException e)
28             {
29                 System.err.println("Mengalami Salah?");
30                 akhir = true;

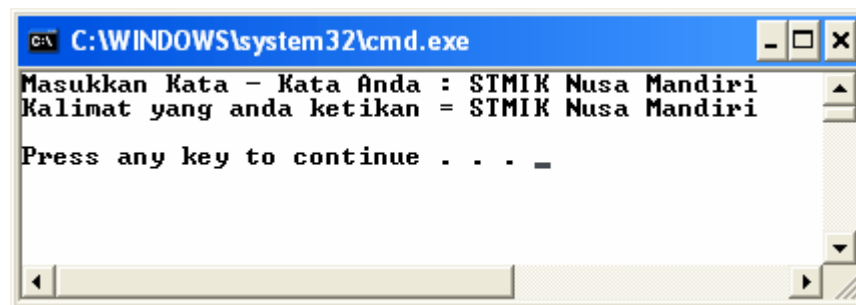
```

```
31     }  
32     }  
33     System.out.println("Kalimat yang anda ketikan = " + kata);  
34     }  
35 }
```

Penjelasan Program :

- Perintah baris 6 : `import java.io.*;`
Digunakan untuk menyertakan package `java.io`.
- Perintah baris 18 sampai 34, akan mengulan terus sampai selesai pengetikan dan menekan tombol enter. Semua karakter yang diketikan dibaca oleh `System.in.read()`, kemudian disimpan didalam variabel huruf. Semua karakter yang diketikan disimpan kembali didalam variabel kata, sehingga menjadi dalam bentuk string.
- Apabila ada kesalahan pengetikan, maka kesalahan tersebut dilempar ke `IOException`, kemudian Standar error akan bekerja sehingga proses dihentikan.
- Setelah selesai pengetikan, menjalankan perintah pada baris 33, selanjutnya menampilkan karakter-karakter yang diketikan dalam bentuk string.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.3. Hasil Lat203.java

2.6.2 InputStreamReader dan BufferedReader

`InputStreamReader` digunakan membaca arus byte stream dan mengkonversi byte-byte ke dalam nilai-nilai bilangan bulat yang merepresentasikan karakter-karakter Unicode.

Kelas `BufferedReader` membaca masukan Stream karakter dan penyangga tersebut untuk efisiensi. Pada penggunaannya harus mempunyai Reader Object untuk membuat versi buffered. Berikut konstrutor yang digunakan untuk membuat `BufferedReader`.

- `BufferedReader(Reader)` digunakan untuk membuat penyangga karakter stream yang berhubungan dengan Reader Objek yang ditetapkan.
- `BufferedReader (Reader, int)` digunakan untuk membuat penyangga karakter stream yang berhubungan dengan Reader Objek yang ditetapkan dan dengan penyangga ukuran integer.

Penyangga Karakter Stream dapat dibaca menggunakan metoda `read()` dan `read(char[], int, int)` untuk menguraikan `FileReader`, serta dapat juga membaca baris dari teks dengan menggunakan `readLine ()`.

metoda `readLine ()`, digunakan untuk kembalikan String objek yang berisi next line dari teks pada stream, tidak termasuk karakter atau karakter-karakter yang merepresentasikan end-of-line.

Jika akhir stream dicapai, maka nilai string yang ingin dikembalikan bernilai dengan null.

end-of-line ditandai dengan beberapa pernyataan berikut :

- newline karakter (`'\n'`)
- carriage return character (`'\r'`)
- carriage return character yang diikuti oleh satu newline (`"\n\r"`)

Berikut penggunaan `InputStreamReader` dan `Buffered`, untuk masukan data dari keyboard.

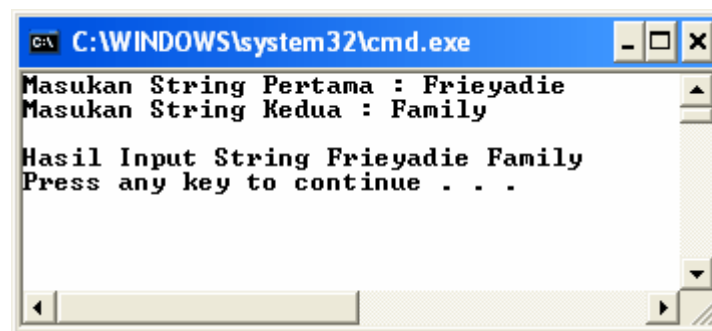
```
1  /* -----
2     Nama File : Lat204.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.io.*;
7
8  class Lat204
9  {
10     public static void main(String args[])throws Exception
11     {
12         //membuat objek baru
13
14         InputStreamReader keyreader = new InputStreamReader(System.in);
15         BufferedReader input = new BufferedReader(keyreader);
16
17         //deklarasi variabel
18         String s1, s2;
19
20         System.out.print("Masukan String Pertama : ");
21         s1 = input.readLine();
22     }
```

```
23      System.out.print("Masukan String Kedua : ");
24      s2 = input.readLine();
25
26      System.out.println("\nHasil Input String " + s1 + " " + s2);
27  }
28 }
```

Penjelasan Program :

- Perintah baris 6 : `import java.io.*;`
Digunakan untuk menyertakan package `java.io`.
- Pada baris 10, terdapat perintah `throws Exception` yang digunakan untuk melempar jika terjadi kesalahan-kesalahan, maka `Excetion` akan bekerja dengan sendirinya.
- Perintah baris 14 dan 15, mendeklarasikan `InputStreamReader` dan `BufferedReader` membuat objek baru.
- Perintah baris 20 dan 24, menginput string.
- Perintah baris 26, menampilkan hasil input data string.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.4. Hasil Lat204.java

2.6.3 Scanner

Class `Scanner` digunakan secara ekstensif untuk memasukan data dari keyboard. Tidak seperti perintah masukan lainnya, data-data yang dimasukan misalnya berupa angka, tidak perlu dilakukan konversi dari string ke integer atau tipe data lainnya. Untuk menggunakan Class `Scanner`, harus menyertakan package **`java.util`**.

Untuk membaca baris dari text yang diinputkan dengan menggunakan metoda `next()`. Supaya string yang dibaca utuh termasuk spasi, dengan menggunakan `nextLine()`. Untuk membaca data

berupa nilai integer atau tipe data angka lainnya, sebagai contoh bisa menggunakan metoda, seperti dibawah ini :

- `readInt()` untuk membaca nilai integer
- `readDouble()` untuk membaca nilai double.
- `readFloat()` untuk membaca nilai float.

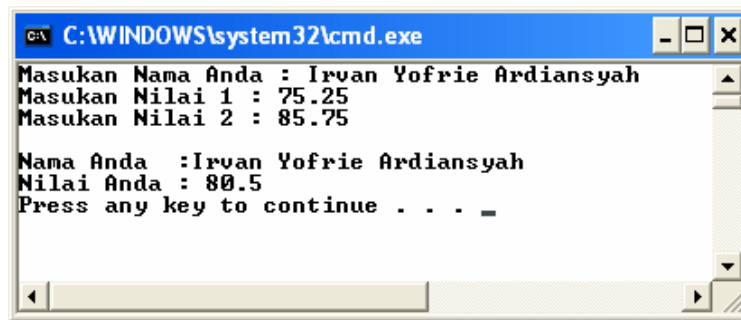
Berikut penggunaan `InputStreamReader` dan `Buffered`, untuk masukan data dari keyboard.

```
1  /* -----
2     Nama File : Lat205.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat205
9  {
10     public static void main(String args[])
11     {
12         //membuat objek baru
13         Scanner input = new Scanner(System.in);
14
15         //deklarasi variabel
16         String nama;
17         int n2;
18         double n1, n3;
19
20         System.out.print("Masukan Nama Anda : ");
21         nama = input.nextLine();
22
23         System.out.print("Masukan Nilai 1 : ");
24         n1 = input.nextDouble();
25
26         System.out.print("Masukan Nilai 2 : ");
27         n2 = input.nextInt();
28
29         n3 = n1 + n2;
30
31         System.out.println("\nNama Anda : " + nama);
32         System.out.println("Nilai Anda : " + n3);
33     }
34 }
```

Penjelasan Program :

- Perintah baris 6 : `import java.util.*;` .Digunakan untuk menyertakan package `java.util`.
- Pada baris 13, membuat objek dengan standar masukan.
- Perintah baris 20 dan 27, menginput string, nilai integer dan double
- Perintah baris 29, proses nilai
- Perintah baris 31 dan 32, menampilkan hasil input data string dan proses perhitungan.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.5. Hasil Lat205.java

2.7 Perintah Konversi Data

Bentuk data yang diinputkan melalui keyboard, secara umum berupa nilai string, maka dalam hal proses perhitungan matematika tidak bisa diproses, maka supaya bisa digunakan, harus dikonversi kebentuk tipe data yang diinginkan.

2.7.1 Konversi String to Integer

Untuk melakukan konversi String ke Integer, dengan menggunakan bentuk penulisan seperti dibawah ini :

```
var_penampung = Integer.parseInt(nilai_string);
```

Untuk lebih jelasnya anda bisa lihat contoh dibawah ini :

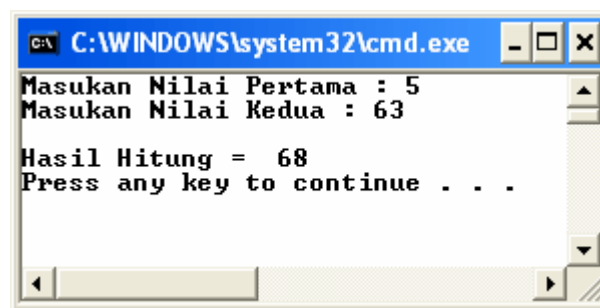
```
1  /* -----
2     Nama File : Lat206.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.io.*;
7
8  class Lat206
9  {
10     public static void main(String args[])throws Exception
11     {
12         //membuat objek baru
13         InputStreamReader keyreader = new InputStreamReader(System.in);
14         BufferedReader input = new BufferedReader(keyreader);
15
16         //deklarasi variabel
17         String s1, s2;
18         int n1, n2, n3;
19     }
```

```
20 System.out.print("Masukan Nilai Pertama : ");
21 s1 = input.readLine();
22
23 System.out.print("Masukan Nilai Kedua : ");
24 s2 = input.readLine();
25
26 //konversi
27 n1 = Integer.parseInt(s1);
28 n2 = Integer.parseInt(s2);
29
30 n3 = n1 + n2;
31
32 System.out.println("\nHasil Hitung = " + n3);
33 }
34 }
```

Penjelasan Program :

- Perintah baris 27 dan 28. Digunakan untuk melakukan konversi nilai String ke Integer.
- Nilai String diambil dari string s1 dan s2 yang diinputkan dari keyboard. Hasil konversi, ditampung kemasing-masing variabel penampung.
- Perintah baris 30. Melakukan pengetesan apakah nilai sudah terkonversi dengan baik. Jika ya, maka nilai n1 dan n2 bisa diproses penambahan.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.6. Hasil Lat206.java

2.7.2 Konversi String to Float

Untuk melakukan konversi String ke Float, dengan menggunakan bentuk penulisan seperti dibawah ini :

```
var_penampung = Float.parseFloat(nilai_string);
```

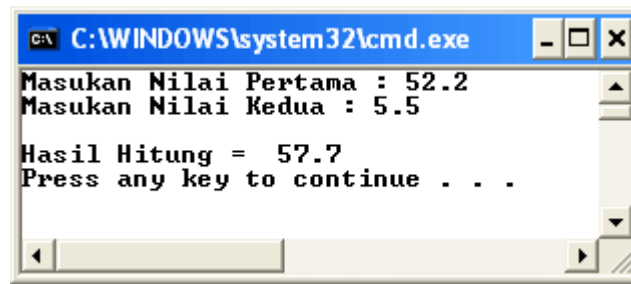
Untuk lebih jelasnya anda bisa lihat contoh dibawah ini :

```
1  /* -----  
2      Nama File : Lat207.java  
3      Author   : Frieyadie  
4  ----- */  
5  
6  import java.io.*;  
7  
8  class Lat206  
9  {  
10     public static void main(String args[])throws Exception  
11     {  
12         //membuat objek baru  
13         InputStreamReader keyreader = new InputStreamReader(System.in);  
14         BufferedReader input = new BufferedReader(keyreader);  
15  
16         //deklarasi variabel  
17         String s1, s2;  
18         float n1, n2, n3;  
19  
20         System.out.print("Masukan Nilai Pertama : ");  
21         s1 = input.readLine();  
22  
23         System.out.print("Masukan Nilai Kedua : ");  
24         s2 = input.readLine();  
25  
26         //konversi  
27         n1 = Float.parseFloat(s1);  
28         n2 = Float.parseFloat(s2);  
29  
30         n3 = n1 + n2;  
31  
32         System.out.println("\nHasil Hitung = " + n3);  
33     }  
34 }
```

Penjelasan Program :

- Perintah baris 27 dan 28. Digunakan untuk melakukan konversi nilai String ke Float.
- Nilai String diambil dari string s1 dan s2 yang diinputkan dari keyboard. Hasil konversi, ditampung ke masing-masing variabel penampung.
- Perintah baris 30. Melakukan pengetesan apakah nilai sudah terkonversi dengan baik. Jika ya, maka nilai n1 dan n2 bisa diproses penambahan.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.7. Hasil Lat207.java

2.7.3 Konversi String to Double

Untuk melakukan konversi String ke Double, dengan menggunakan bentuk penulisan seperti dibawah ini :

```
var_penampung = Double.parseDouble(nilai_string);
```

Untuk lebih jelasnya anda bisa lihat contoh dibawah ini :

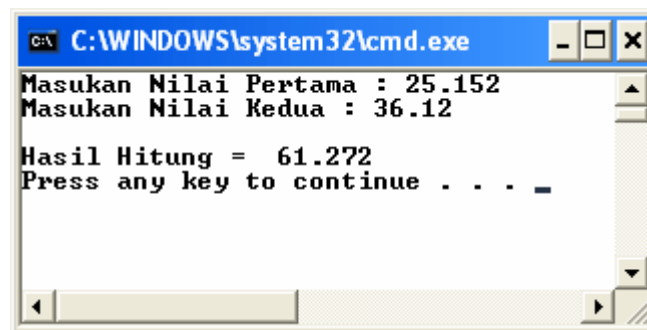
```

1  /* -----
2     Nama File : Lat208.java
3     Author   : Frieyadie
4  ----- */
5  import java.io.*;
6
7  class Lat208
8  {
9      public static void main(String args[])throws Exception
10     {
11         //membuat objek baru
12
13         InputStreamReader keyreader = new InputStreamReader(System.in);
14         BufferedReader input = new BufferedReader(keyreader);
15
16         //deklarasi variabel
17         String s1, s2;
18         double n1, n2, n3;
19
20         System.out.print("Masukan Nilai Pertama : ");
21         s1 = input.readLine();
22
23         System.out.print("Masukan Nilai Kedua : ");
24         s2 = input.readLine();
25
26         //konversi
27         n1 = Double.parseDouble(s1);
28         n2 = Double.parseDouble(s2);
29
30         n3 = n1 + n2;
31
32         System.out.println("\nHasil Hitung = " + n3);
33     }
34 }
```

Penjelasan Program :

- Perintah baris 27 dan 28. Digunakan untuk melakukan konversi nilai String ke Double.
- Nilai String diambil dari string s1 dan s2 yang diinputkan dari keyboard. Hasil konversi, ditampung kemasing-masing variabel penampung.
- Perintah baris 30. Melakukan pengecekan apakah nilai sudah terkonversi dengan baik. Jika ya, maka nilai n1 dan n2 bisa diproses penambahan.

Maka hasil pada saat dieksekusi, seperti dibawah ini :



Gambar 2.7. Hasil Lat207.java

2.8 Latihan

1. Buatlah program untuk menghitung konversi dari derajat Celcius ke derajat Fahrenheit dan Reamur. Diketahui nilai Celcius diinput melalui keyboard

Masukan Nilai Derajat Celcius : __

Hasil Konversi :

Derajat Fahrenheit : ____

Derajat Reamur : ____

2. Buatlah program menghitung Luas dan Keliling Lingkaran. Dengan Layar masukan dan keluaran seperti dibawah ini :

Masukan Nilai Radius : __

Hasil Perhitungan

Luas Lingkaran : ____

Keliling Lingkaran : ____

3. Buatlah program menghitung Nilai Akhir Siswa Informatika. Dengan Ketentuan seperti dibawah ini :
- Nilai UTS, UAS dan Tugas Mandiri diinput melalui keyboard
 - Menghitung Nilai Murni
 - Nilai Murni UTS = Nilai UTS dikali dengan 35%
 - Nilai Murni UAS = Nilai UAS dikali dengan 45%
 - Nilai Murni Tugas Mandiri = Nilai Tugas Mandiri dikali dengan 20%
 - Nilai Akhir adalah perhitungan Nilai Murni - Nilai Murni
 - Layar masukan dan keluaran seperti dibawah ini :

Masukan Nama Siswa : _____
Nilai UTS : _____
Nilai UAS : _____
Nilai Tugas Mandiri : _____

Nilai Murni yang diperoleh :
Nilai Murni UTS : _____
Nilai Murni UAS : _____
Nilai Murni Tugas : _____

Nilai Akhir yang diperoleh yaitu : _____

Lembar ini sengaja dikosongkan

Bab 3:

Operator Operator pada Bahasa Java

3.1 Kopetensi Dasar

Pada pembahasan Bab 3 ini penulis mengajak mendiskusikan mengenai penggunaan operator-operator yang disediakan oleh Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami penggunaan operator-operator pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- a. Penggunaan Operator Aritmatika.
- b. Penggunaan Operasi Pemberi Nilai
- c. Penggunaan Operator Penambah dan Pengurang
- d. Penggunaan Operator Logika dan Operator Bitwise

3.2 Operator Aritmatika

Operator adalah simbol atau karakter yang biasa dilibatkan dalam program untuk melakukan sesuatu operasi atau manipulasi, seperti penjumlahan, pengurangan dan lain-lain.

- a. Unary, yaitu hanya melibatkan sebuah operand pada suatu ekspresi aritmatika.

Contoh : -7

- b. Binary, yaitu melibatkan dua buah operand pada suatu ekspresi aritmatika.

Contoh : $13 + 5$

- c. Ternary, yaitu melibatkan tiga buah operand pada suatu operasi aritmatika.

Contoh : $(8 + 4) * 7 - 5$.

Operator Aritmatika yang tergolong sebagai operator binary adalah :

Table 3.1. Operator Aritmatika

Operasi pada Java	Operator	Contoh Ekspresi
Perkalian	*	4 * 5
Pembagian	/	7 / 4
Sisa Pembagian	%	5 % 2
Penjumlahan	+	7 + 3
Pengurangan	-	6 - 4

Pembagian bilangan bulat menghasilkan suatu hasil bagi bilangan bulat juga, sebagai contoh, ungkapan $7 / 4$ menghasilkan nilai 1, dan misalkan ungkapan $17 / 5$ akan menghasilkan nilai 3. Karena bagian sisa pembagian bilangan bulat dibuang.

Java menyediakan operator Sisa Pembagian (%), yaitu nilai hasil pembagian dari ungkapan pembagian nilai. Misalnya pada ungkapan $7 / 4$, maka akan menghasilkan sisa 3.

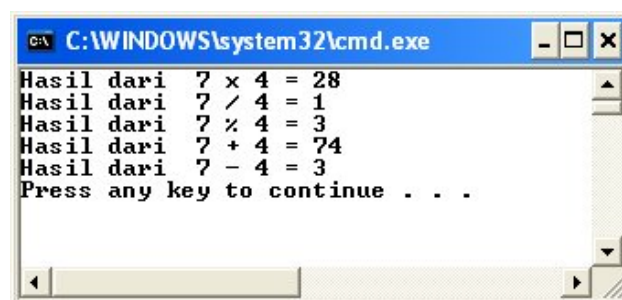
Berikut contoh penggunaan operator aritmatika, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat301.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat301
7  {
8      public static void main(String[] args)
9      {
10         System.out.println("Hasil dari 7 x 4 = " + (7 * 4));
11         System.out.println("Hasil dari 7 / 4 = " + (7 / 4));
12         System.out.println("Hasil dari 7 % 4 = " + (7 % 4));
13         System.out.println("Hasil dari 7 + 4 = " + (7 + 4));
14         System.out.println("Hasil dari 7 - 4 = " + (7 - 4));
15     }
16 }

```

Output yang dihasilkan dari program Lat301.java diatas, seperti dibawah ini :

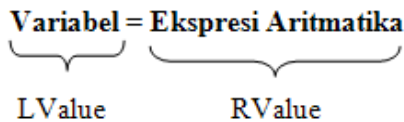


Gambar 3.1 Hasil Program Lat301.java

3.2.1 Ekspresi Aritmatika

Penulisan suatu ekspresi aritmatika pada program java, sangat berkaitan dengan pernyataan pemberi nilai. Karena hasil dari ekspresi aritmatika akan ditampung kedalam suatu variabel.

Bentuk umum penulisan ekspresi aritmatika, seperti dibawah ini :



- LValue (Left Value), merupakan berupa variabel tunggal sebagai penampung hasil dari ekspresi Aritmatika
- RValue (Right Value), merupakan Ekspresi Aritmatika, bisa berupa unary, binary atau ternary dan variabel lainnya.

Tanda = (sama dengan), dikenal sebagai operator pemberi nilai (Assignment Operator)

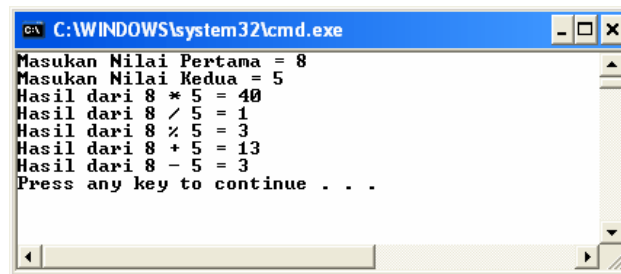
Berikut contoh ekspresi aritmatika, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat302.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat302
8  {
9      public static void main(String[] args)
10     {
11         int nil1, nil2;
12         int a, b, c, d, e, f;
13         Scanner input = new Scanner(System.in);
14
15         System.out.print("Masukan Nilai Pertama = ");
16         nil1 = input.nextInt();
17         System.out.print("Masukan Nilai Kedua = ");
18         nil2 = input.nextInt();
19
20         //operasi aritmatika
21         a = nil1 * nil2;
22         b = nil1 / nil2;
23         c = nil1 % nil2;
24         d = nil1 + nil2;
25         e = nil1 - nil2;
26
27         System.out.println("Hasil " + nil1 + " * " + nil2 + " = " + a);
28         System.out.println("Hasil " + nil1 + " / " + nil2 + " = " + b);
29         System.out.println("Hasil " + nil1 + " % " + nil2 + " = " + c);
30         System.out.println("Hasil " + nil1 + " + " + nil2 + " = " + d);
31         System.out.println("Hasil " + nil1 + " - " + nil2 + " = " + e);
32     }
33 }

```

Output yang dihasilkan dari program Lat302.java diatas, seperti dibawah ini :



```

C:\WINDOWS\system32\cmd.exe
Masukan Nilai Pertama = 8
Masukan Nilai Kedua = 5
Hasil dari 8 * 5 = 40
Hasil dari 8 / 5 = 1
Hasil dari 8 % 5 = 3
Hasil dari 8 + 5 = 13
Hasil dari 8 - 5 = 3
Press any key to continue . . .

```

Gambar 3.2 Hasil Program Lat302.java

3.2.2 Hierarki Operator Aritmatika

Didalam suatu penulisan ekspresi aritmatika sering kita jumpai menggunakan beberapa operator aritmatika yang berbeda secara bersamaan. Maka dalam prosenya akan berbeda, tergantung dari urutan atau tingkatan operator tersebut. Berikut urutan operator aritmatika, seperti dibawah ini :

Table 3.2. Operator Aritmatika

Operator	Penjelasan Operator
*	Ketiga operator ini memiliki tingkatan yang akan diproses lebih dulu. Tingkatan operator sama dan penggunaannya tergantung letak yang didepan akan diproses lebih dulu.
/	
%	
+	Kedua operator ini akan diproses kemudian. Tingkatan operator sama dan penggunaannya tergantung letak yang yang didepan akan diproses lebih dulu.
-	

Contoh kasus ekspresi aritmatika, seperti berikut ini :

$$A = 8 + 2 * 3 / 6$$

Maka langkah-langkah perhitungannya :

Langkah 1 : $A = 2 * 3$ hasilnya 6

$$A = 8 + 6 / 6$$

Langkah 2 : $A = 6 / 6$ hasilnya 1

$$A = 8 + 1$$

Langkah 3 : $A = 9$

Tingkatan operator-operator ini, bisa diabaikan dengan menggunakan tanda kurung buka " (" dan ") ". Jika suatu ekspresi terdapat didalam tanda kurung, maka proses ekspresi tersebut akan diproses terlebih dahulu, tanpa melihat tingkatan operator.

Contoh : $A = (8 + 2) * 3 / 6$

Maka langkah-langkah perhitungannya :

Langkah 1 : $A = 8 + 2$ hasilnya 10

$$A = 10 * 3 / 6$$

Langkah 2 : $A = 10 * 3$ hasilnya 30

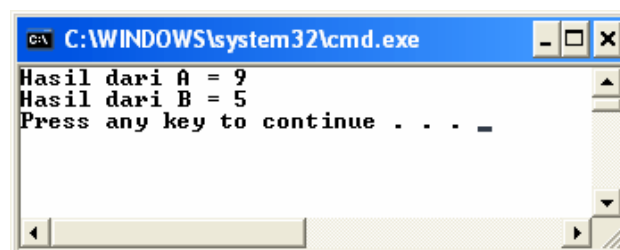
$$A = 30 / 6$$

Langkah 3 : $A = 5$

Berikut contoh penggunaan hierarki operator aritmatika, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2      Nama File : Lat303.java
3      Author   : Frieyadie
4  ----- */
5
6  class Lat303
7  {
8      public static void main(String[] args)
9      {
10         int A, B;
11
12         A = 8 + 2 * 3 / 6;
13         B = (8 + 2) * 3 / 6;
14
15         System.out.println("Hasil dari A = " + A);
16         System.out.println("Hasil dari B = " + B);
17     }
18 }
```

Output yang dihasilkan dari program Lat303.java diatas, seperti dibawah ini :



Gambar 3.3 Hasil Program Lat303.java

3.3 Operator Pemberi Nilai Aritmatika

Sebelumnya, kita telah mengenal operator pemberi nilai (Assignment Operator), yaitu menggunakan tanda sama dengan "=", sebagai contoh $A = A + 1$

Dari ekspresi $A = A + 1$, bisa disederhanakan bentuk penulisan ekspresinya, yaitu menjadi $A += 1$.

Notasi $+=$, ini dikenal dengan operator pemberi nilai aritmatika. Java menyediakan beberapa notasi pemberi nilai.

Table 3.3. Operator Pemberi Nilai

Operasi pada Java	Operator Pemberi Nilai	Contoh Ekspresi	Penggunaan Operator Pemberi Nilai
Perkalian	<code>*=</code>	$A = A * 5$	$A *= 5$
Pembagian	<code>/=</code>	$A = A / 5$	$A /= 5$
Sisa Pembagian	<code>%=</code>	$A = A \% 2$	$A \% = 2$
Penjumlahan	<code>+=</code>	$A = A + 1$	$A += 1$
Pengurangan	<code>-=</code>	$A = A - 4$	$A -= 4$

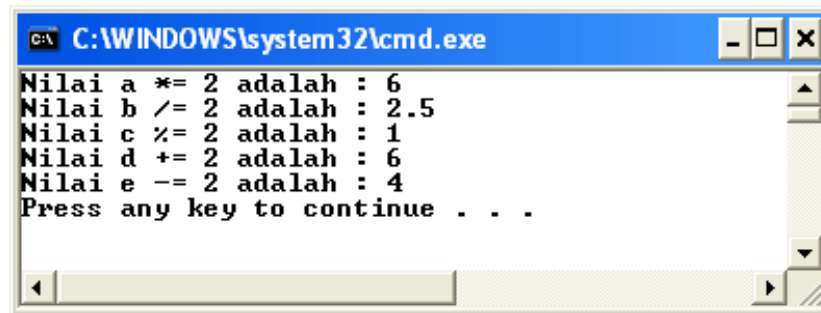
Berikut contoh penggunaan operator pemberi nilai aritmatika.

```

1  /* -----
2     Nama File : Lat304.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat304
7  {
8      public static void main(String[] args)
9      {
10         int a, c, d, e;
11         double b;
12
13         a = 3; b = 5; c = 13; d = 4; e = 6;
14
15         //penggunaan assignment operator
16
17         a *= 2;
18         b /= 2;
19         c %= 2;
20         d += 2;
21         e -= 2;
22
23         System.out.println("Nilai a *= 2 adalah : "+ a);
24         System.out.println("Nilai b /= 2 adalah : "+ b);
25         System.out.println("Nilai c %= 2 adalah : "+ c);
26         System.out.println("Nilai d += 2 adalah : "+ d);
27         System.out.println("Nilai e -= 2 adalah : "+ e);
28     }
29 }

```


Output yang dihasilkan dari program Lat304.java diatas, seperti dibawah ini :



```

C:\WINDOWS\system32\cmd.exe
Nilai a *= 2 adalah : 6
Nilai b /= 2 adalah : 2.5
Nilai c %= 2 adalah : 1
Nilai d += 2 adalah : 6
Nilai e -= 2 adalah : 4
Press any key to continue . . .

```

Gambar 3.4 Hasil Program Lat304.java

3.4 Operator Penambah dan Pengurang

Masih berkaitan dengan operator pemberi nilai, Java menyediakan operator penambah dan pengurang, yaitu digunakan untuk menambah satu dan mengurangi satu dari nilai pada dirinya sendiri. Dari contoh penulisan operator pemberi nilai sebagai penyederhanaannya dapat digunakan operator penambah dan pengurang.

Tabel. 3.4. Tabel Operator Penambah dan Pengurang

Operator	Keterangan
++	Penambahan
--	Pengurangan

Sebagai contoh, terdapat ungkapan aritmatika seperti dibawah ini :

$A = A + 1$ atau $A = A - 1$; maka bentuk ekspresi tersebut bisa disederhanakan menjadi $A += 1$ atau $A -= 1$; hal ini masih dapat disederhanakan menjadi $A++$ atau $A--$.

Notasi “++” atau “--” dapat diletakan didepan atau di belakang variabel. Bentuk penulisannya seperti dibawah ini :

$A++$ atau $++A$ dan $A--$ atau $--A$

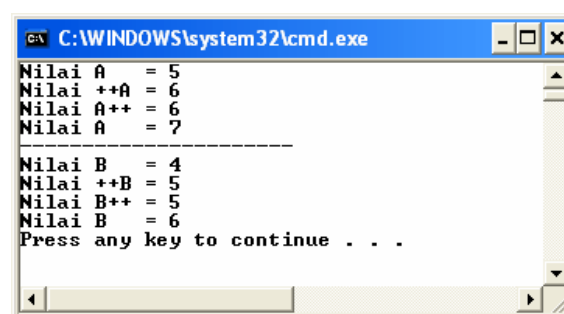
Kedua bentuk penulisan notasi ini mempunyai arti yang berbeda. Perbedaan penulisan tersebut, yaitu :

- a. Jika diletakan didepan variabel, maka proses penambahan atau pengurangan akan dilakukan sesaat sebelum atau langsung pada saat menjumpai ekspresi ini, sehingga nilai variabel tadi akan langsung berubah begitu ekspresi ini ditemukan, sedangkan
- b. Jika diletakan dibelakang variabel, maka proses penambahan atau pengurangan akan dilakukan setelah ekspresi ini dijumpai atau nilai variabel akan tetap pada saat ekspresi ini ditemukan.

Berikut contoh penggunaan operator penambah dan pengurang, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2     Nama File : Lat305.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat305
7  {
8      public static void main(String[] args)
9      {
10         int a, b;
11
12         a = 5;
13         b = 4;
14         System.out.println("Nilai A   = " + a);
15         System.out.println("Nilai ++A = " + ++a);
16         System.out.println("Nilai A++ = " + a++);
17         System.out.println("Nilai A   = " + a);
18         System.out.println("-----");
19         System.out.println("Nilai B   = " + b);
20         System.out.println("Nilai ++B = " + ++b);
21         System.out.println("Nilai B++ = " + b++);
22         System.out.println("Nilai B   = " + b);
23     }
24 }
```

Output yang dihasilkan dari program Lat305.java diatas, seperti dibawah ini :



```
C:\WINDOWS\system32\cmd.exe
Nilai A   = 5
Nilai ++A = 6
Nilai A++ = 6
Nilai A   = 7
-----
Nilai B   = 4
Nilai ++B = 5
Nilai B++ = 5
Nilai B   = 6
Press any key to continue . . .
```

Gambar 3.5 Hasil Program Lat305.java

3.5 Operator Pembandingan (Comparison)

Java menyediakan beberapa operator yang digunakan untuk membuat perbandingan-perbandingan antar variabel-variabel, variabel dan literal atau tipe informasi lainnya didalam program.

Operator Pembading (Comparasion) digunakan untuk membandingkan dua buah nilai. Hasil perbandingan operator ini menghasilkan nilai Boolean yaitu True atau False.

Tabel. 3.5. Tabel Operator Pembandingan

Operator	Keterangan
==	Sama Dengan (bukan pemberi nilai)
!=	Tidak Sama dengan
>	Lebih Dari
<	Kurang Dari
>=	Lebih Dari sama dengan
<=	Kurang Dari sama dengan

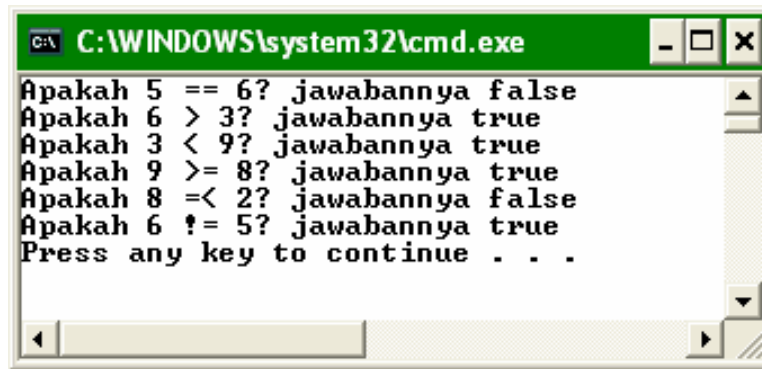
Berikut contoh penggunaan operator pembandingan, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat306.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat306
7  {
8      public static void main(String[] args)
9      {
10         int a, b, c, d, e, f;
11         boolean aa, bb, cc, dd, ee, ff;
12
13         a = 5; b = 6; c = 3; d = 9; e = 8; f = 2;
14
15         aa = a == b;
16         bb = b > c;
17         cc = c < d;
18         dd = d >= e;
19         ee = e <= f;
20         ff = b != a;
21
22         System.out.println("Apakah "+ a +" == "+ b +"? jawabannya " + aa);
23         System.out.println("Apakah "+ b +" > "+ c +"? jawabannya " + bb);
24         System.out.println("Apakah "+ c +" < "+ d +"? jawabannya " + cc);
25         System.out.println("Apakah "+ d +" >= "+ e +"? jawabannya " + dd);
26         System.out.println("Apakah "+ e +" <= "+ f +"? jawabannya " + ee);
27         System.out.println("Apakah "+ b +" != "+ a +"? jawabannya " + ff);
28     }
29 }

```

Output yang dihasilkan dari program Lat306.java diatas, seperti dibawah ini :



Gambar 3.6 Hasil Program Lat306.java

3.6 Operator Logika

Operator Relasi digunakan untuk menghubungkan dua buah operasi relasi menjadi sebuah ungkapan kondisi. Hasil dari operator logika ini menghasilkan nilai boolean True atau False.

Tabel. 3.6. Tabel Operator Logika

Operator	Keterangan
&&	Operator Logika AND
	Operator Logika OR
!	Operator Logika NOT

3.6.1 Operator Logika AND

Operator logika AND digunakan untuk menghubungkan dua atau lebih ekspresi relasi, akan dianggap BENAR, bila semua ekspresi relasi yang dihubungkan bernilai BENAR.

Contoh :

Ekspresi Relasi-1 $\rightarrow A + 4 < 10$

Ekspresi Relasi-2 $\rightarrow B > A + 5$

Ekspresi Relasi-3 $\rightarrow C - 3 \geq 4$

Penggabungan ketiga ekspresi relasi diatas menjadi ;

$A + 4 < 10 \ \&\& \ B > A + 5 \ \&\& \ C - 3 \geq 4$

Jika nilai $A = 3$; $B = 3$; $C = 7$, maka ketiga ekspresi tersebut mempunyai nilai :

Ekspresi Relasi-1 $\rightarrow A + 4 < 10 \rightarrow 3 + 4 < 10 \rightarrow \text{BENAR}$

Ekspresi Relasi-2 $\rightarrow B > A + 5 \rightarrow 3 > 3 + 5 \rightarrow \text{SALAH}$

Ekspresi Relasi-3 $\rightarrow C - 3 \geq 4 \rightarrow 7 - 3 \geq 4 \rightarrow \text{BENAR}$

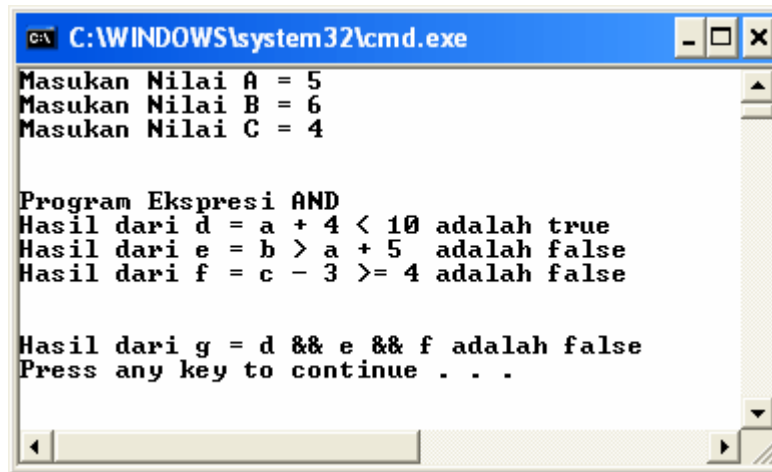
Dari ekspresi relasi tersebut mempunyai nilai BENAR, maka

$A + 4 < 10 \ \&\& \ B > A + 5 \ \&\& \ C - 3 \geq 4 \rightarrow \text{SALAH}$

Berikut contoh penggunaan operator Logika AND, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2     Nama File : Lat307.java
3     Author    : Frieyadie
4     ----- */
5
6  import java.util.Scanner;
7
8  class Lat307
9  {
10     public static void main(String[] args)
11     {
12         Scanner input = new Scanner(System.in);
13
14         int a, b, c ;
15         boolean d, e, f, g;
16
17         System.out.print("Masukan Nilai A = ");
18         a = input.nextInt();
19
20         System.out.print("Masukan Nilai B = ");
21         b = input.nextInt();
22
23         System.out.print("Masukan Nilai C = ");
24         c = input.nextInt();
25
26         // Proses
27
28         d = a + 4 < 10;
29         e = b > a + 5;
30         f = c - 3 >= 4;
31         g = d && e && f;
32
33         System.out.println("\n");
34         System.out.println("Program Ekspresi AND");
35
36         System.out.println("Hasil dari d = a + 4 < 10 adalah " + d);
37         System.out.println("Hasil dari e = b > a + 5 adalah " + e);
38         System.out.println("Hasil dari f = c - 3 >= 4 adalah " + f);
39         System.out.println("\n");
40         System.out.println("Hasil dari g = d && e && f adalah " + g);
41     }
42 }
```

Output yang dihasilkan dari program Lat307.java diatas, seperti dibawah ini :



```

C:\WINDOWS\system32\cmd.exe
Masukan Nilai A = 5
Masukan Nilai B = 6
Masukan Nilai C = 4

Program Ekspresi AND
Hasil dari d = a + 4 < 10 adalah true
Hasil dari e = b > a + 5 adalah false
Hasil dari f = c - 3 >= 4 adalah false

Hasil dari g = d && e && f adalah false
Press any key to continue . . .

```

Gambar 3.7 Hasil Program Lat307.java

3.6.2 Operator Logika OR

Operator logika OR digunakan untuk menghubungkan dua atau lebih ekspresi relasi, akan dianggap BENAR, bila salah satu ekspresi relasi yang dihubungkan bernilai BENAR dan bila semua ekspresi relasi yang dihubungkan bernilai SALAH, maka akan bernilai SALAH.

Contoh :

Ekspresi Relasi-1 $\rightarrow A + 4 < 10$

Ekspresi Relasi-2 $\rightarrow B > A + 5$

Ekspresi Relasi-3 $\rightarrow C - 3 > 4$

Penggabungan ketiga ekspresi relasi diatas menjadi ;

$A + 4 < 10 \parallel B > A + 5 \parallel C - 3 > 4$

Jika nilai $A = 3$; $B = 3$; $C = 7$, maka ketiga ekspresi tersebut mempunyai nilai :

Ekspresi Relasi-1 $\rightarrow A + 4 < 10 \rightarrow 3 + 4 < 10 \rightarrow \text{BENAR}$

Ekspresi Relasi-2 $\rightarrow B > A + 5 \rightarrow 3 > 3 + 5 \rightarrow \text{SALAH}$

Ekspresi Relasi-3 $\rightarrow C - 3 > 4 \rightarrow 7 - 3 > 4 \rightarrow \text{SALAH}$

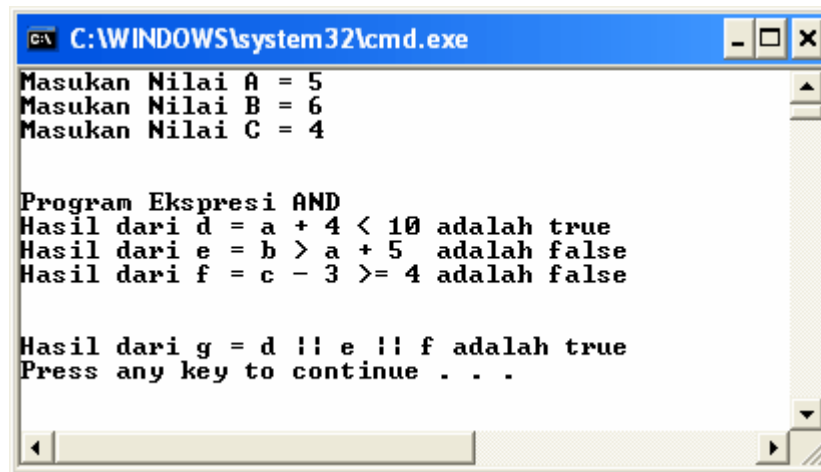
Dilihat ekspresi diatas salah satu ekspresi tersebut mempunyai nilai BENAR, maka ekspresi tersebut tetap bernilai BENAR.

$A+4 < 10 \parallel B>A+5 \parallel C-3 > 4 \rightarrow \text{BENAR}$

Berikut contoh penggunaan operator Logika OR, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2      Nama File : Lat308.java
3      Author    : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat308
9  {
10     public static void main(String[] args)
11     {
12         Scanner input = new Scanner(System.in);
13
14         int a, b, c ;
15         boolean d, e, f, g;
16
17         System.out.print("Masukan Nilai A = ");
18         a = input.nextInt();
19
20         System.out.print("Masukan Nilai B = ");
21         b = input.nextInt();
22
23         System.out.print("Masukan Nilai C = ");
24         c = input.nextInt();
25
26         // Proses
27
28         d = a + 4 < 10;
29         e = b > a + 5;
30         f = c - 3 >= 4;
31         g = d || e || f;
32
33         System.out.println("\n");
34         System.out.println("Program Ekspresi OR");
35
36         System.out.println("Hasil dari d = a + 4 < 10 adalah " + d);
37         System.out.println("Hasil dari e = b > a + 5  adalah " + e);
38         System.out.println("Hasil dari f = c - 3 >= 4 adalah " + f);
39         System.out.println("\n");
40         System.out.println("Hasil dari g = d || e || f adalah " + g);
41     }
42 }
```

Output yang dihasilkan dari program Lat308.java diatas, seperti dibawah ini :



Gambar 3.8 Hasil Program Lat308.java

3.6.3 Operator Logika NOT

Operator logika NOT akan memberikan nilai kebalikkan dari ekspresi yang disebutkan. Jika nilai yang disebutkan bernilai BENAR maka akan menghasilkan nilai SALAH, begitu pula sebaliknya.

Contoh :

Ekspresi Relasi $\rightarrow A + 4 < 10$

Penggunaan Operator Logika NOT diatas menjadi : $!(A+4 < 10)$

Jika nilai $A = 3$; maka ekspresi tersebut mempunyai nilai :

Ekspresi Relasi-1 $\rightarrow A + 4 < 10 \rightarrow 3 + 4 < 10 \rightarrow \text{BENAR}$

Dilihat ekspresi diatas salah satu ekspresi tersebut mempunyai nilai BENAR dan jika digunakan operator logika NOT, maka ekspresi tersebut akan bernilai SALAH

$!(A+4 < 10) \rightarrow !(\text{BENAR}) = \text{SALAH}$

Berikut contoh penggunaan operator Logika NOT, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2  Nama File : Lat309.java
3  Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat309
9  {
10     public static void main(String[] args)

```

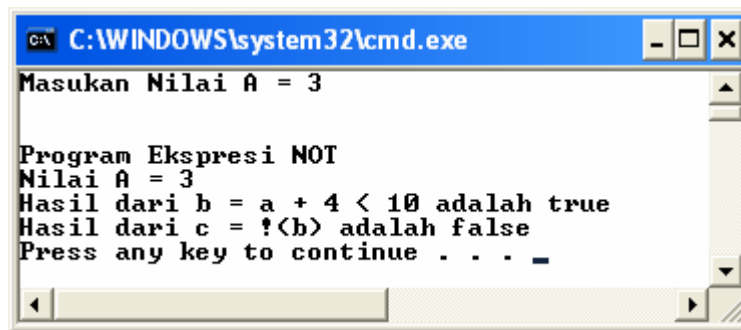


```

11  {
12      Scanner input = new Scanner(System.in);
13
14      int a;
15      boolean b, c;
16
17      System.out.print("Masukan Nilai A = ");
18      a = input.nextInt();
19
20      // Proses
21
22      b = a + 4 < 10;
23      c = !(b);
24
25      System.out.println("\n");
26      System.out.println("Program Ekspresi NOT");
27      System.out.println("Nilai A = " + a);
28      System.out.println("Hasil dari b = a + 4 < 10 adalah " + b);
29      System.out.println("Hasil dari c = !(b) adalah " + c);
30  }
31  }

```

Output yang dihasilkan dari program Lat309.java diatas, seperti dibawah ini :



Gambar 3.9 Hasil Program Lat309.java

3.7 Operator Bitwise

Operator Bitwise digunakan untuk memanipulasi data dalam bentuk bit. Pemrograman Java menyediakan enam buah operator bitwise.

Tabel. 3.8. Tabel Operator Bitwise

Operator	Keterangan
~	Bitwise NOT
<<	Bitwise Shift Left
>>	Bitwise Shift Right
&	Bitwise AND
^	Bitwise XOR
	Bitwise OR

3.7.1 Operator Bitwise << (Shift Left)

Operator Bitwise Shift Left digunakan untuk menggeser sejumlah bit kekiri.

Contoh :

0 0 1 1 0 0 1 0 0 1 = 201
 ↙ ↘ ↙ ↘ ↙ ↘ ↙ ↘ ↙ ↘ ← digeser 1 bit ke kiri
 0 1 1 0 0 1 0 0 1 0 = 402

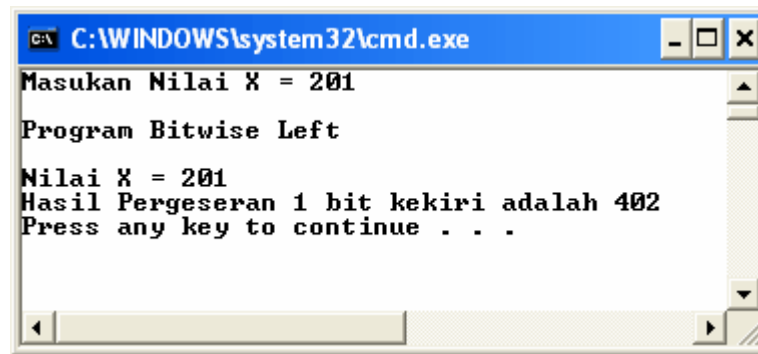
Dibagian kanan yang kosong disisipkan 0, dan sebanyak 1 bit yang digeser kekiri

Berikut contoh penggunaan operator Bitwise Left, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat310.java
3     Author   : Frieyadie
4     ----- */
5
6  import java.util.Scanner;
7
8  class Lat310
9  {
10     public static void main(String[] args)
11     {
12         Scanner input = new Scanner(System.in);
13
14         int x, hasil;
15
16         System.out.print("Masukan Nilai X = ");
17         x = input.nextInt();
18
19         // Proses
20         hasil = x << 1;
21
22         System.out.println("\nProgram Bitwise Left\n");
23         System.out.println("Nilai X = " + x);
24         System.out.println("Hasil Pergeseran 1 bit kekiri adalah " + hasil);
25     }
26 }
  
```

Output yang dihasilkan dari program Lat310.java diatas, seperti dibawah ini :



Gambar 3.10 Hasil Program Lat310.java

3.7.2 Operator Bitwise >> (Shift Right)

Operator Bitwise Shift Right digunakan untuk menggeser sejumlah bit kanan.

Contoh :

0 0 1 1 0 0 1 0 0 1 = 201
 ↙ ↘ ↙ ↘ ↙ ↘ ↙ ↘ ↙ ↘ ← digeser 1 bit ke kanan
 0 0 0 1 1 0 0 1 0 0 = 100

Dibagian kiri yang kosong disisipkan 0, sebanyak satu bit yang digeser kekanan

Berikut contoh penggunaan operator Bitwise Right, bisa anda ikuti seperti dibawah ini.

```

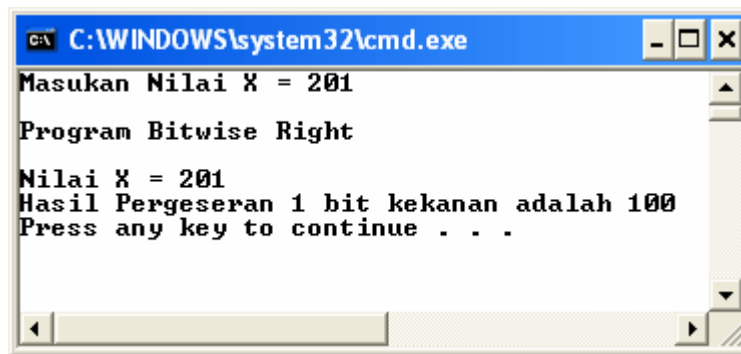
1  /* -----
2     Nama File : Lat311.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat311
9  {
10     public static void main(String[] args)
11     {
12         Scanner input = new Scanner(System.in);
13
14         int x, hasil;
15
16         System.out.print("Masukan Nilai X = ");
17         x = input.nextInt();
18
19         // Proses
20         hasil = x >> 1;
21
22         System.out.println("\nProgram Bitwise Right\n");
23         System.out.println("Nilai X = " + x);
  
```

```

24      System.out.println("Hasil Pergeseran 1 bit kekanan adalah " + hasil);
25      }
26  }

```

Output yang dihasilkan dari program Lat311.java diatas, seperti dibawah ini :



Gambar 3.11 Hasil Program Lat311.java

3.7.3 Operator Bitwise & (And)

Operator Bitwise & (And) digunakan untuk membandingkan bit dari dua operand. Akan bernilai benar (true) jika semua operand yang digabungkan bernilai benar (true). Berikut anda dapat melihat ilustrasi untuk membandingkan bit dari 2 operand.

Tabel. 3.9. Tabel Operator Bitwise And

Bit Operand 1	Bit Operand 2	Hasil Operand
0	0	0
0	1	0
1	0	0
1	1	1

Contoh :

11001001 = 201

01100100 = 100

————— AND

01000000 = 64

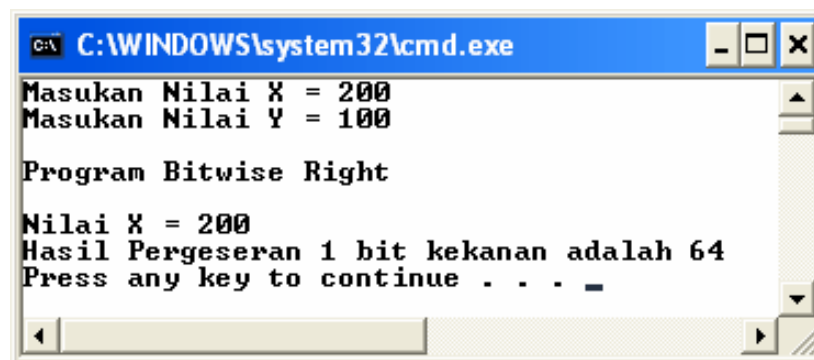
Berikut contoh penggunaan operator Bitwise AND, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat312.java
3     Author   : Frieyadie
4  ----- */
5  import java.util.Scanner;
6
7  class Lat311
8  {
9      public static void main(String[] args)
10     {
11         Scanner input = new Scanner(System.in);
12
13         int x, y, hasil;
14
15         System.out.print("Masukan Nilai X = ");
16         x = input.nextInt();
17
18         System.out.print("Masukan Nilai Y = ");
19         y = input.nextInt();
20
21         // Proses
22         hasil = x & y;
23
24         System.out.println("\nProgram Bitwise And\n");
25         System.out.println("Nilai X = " + x);
26         System.out.println("Nilai Y = " + y);
27         System.out.println("Hasil Bitwise AND adalah " + hasil);
28     }
29 }

```

Output yang dihasilkan dari program Lat312.java diatas, seperti dibawah ini :



Gambar 3.12. Hasil Program Lat312.java

3.7.4 Operator Bitwise | (Or)

Operator Bitwise | (Or) digunakan untuk membandingkan bit dari dua operand. Akan bernilai benar jika ada salah satu operand yang digabungkan ada yang bernilai benar (1). Berikut anda dapat melihat ilustrasi untuk membandingkan bit dari 2 operand.

Tabel. 3.10. Tabel Operator Bitwise Or

Bit Operand 1	Bit Operand 2	Hasil Operand
0	0	0
0	1	1
1	0	1
1	1	1

Contoh :

11001001 = 201

01100100 = 100

OR

11101101 = 237

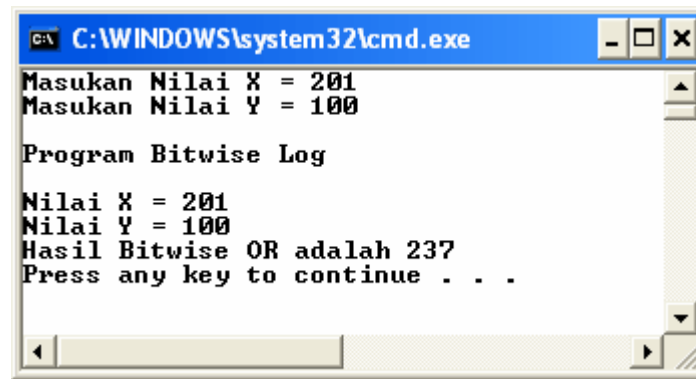
Berikut contoh penggunaan operator Bitwise OR, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat313.java
3     Author   : Frieyadie
4  ----- */
5  import java.util.Scanner;
6
7  class Lat313
8  {
9      public static void main(String[] args)
10     {
11         Scanner input = new Scanner(System.in);
12
13         int x, y, hasil;
14
15         System.out.print("Masukan Nilai X = ");
16         x = input.nextInt();
17
18         System.out.print("Masukan Nilai Y = ");
19         y = input.nextInt();
20
21         // Proses
22         hasil = x | y;
23
24         System.out.println("\nProgram Bitwise Log\n");
25         System.out.println("Nilai X = " + x);
26         System.out.println("Nilai Y = " + y);
27         System.out.println("Hasil Bitwise OR adalah " + hasil);
28     }
29 }

```

Output yang dihasilkan dari program Lat313.java diatas, seperti dibawah ini :



Gambar 3.13. Hasil Program Lat313.java

3.7.5 Operator Bitwise ^ (eXclusive Or)

Operator Bitwise ^ (XOr) digunakan untuk membandingkan bit dari dua operand. Akan bernilai benar (1) jika dari dua bit yang dibandingkan hanya sebuah bernilai benar (1). Berikut anda dapat melihat ilustrasi untuk membandingkan bit dari 2 operand.

Tabel. 3.11. Tabel Operator Bitwise XOR

Bit Operand 1	Bit Operand 2	Hasil Operand
0	0	0
0	1	1
1	0	1
1	1	0

Contoh :

11001001 = 201

01100100 = 100

————— XOR

10101101 = 137

Berikut contoh penggunaan operator Bitwise XOR, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat314.java
3     Author   : Frieyadie
4  ----- */
5  import java.util.Scanner;
6
7  class Lat314
8  {
9      public static void main(String[] args)
10     {

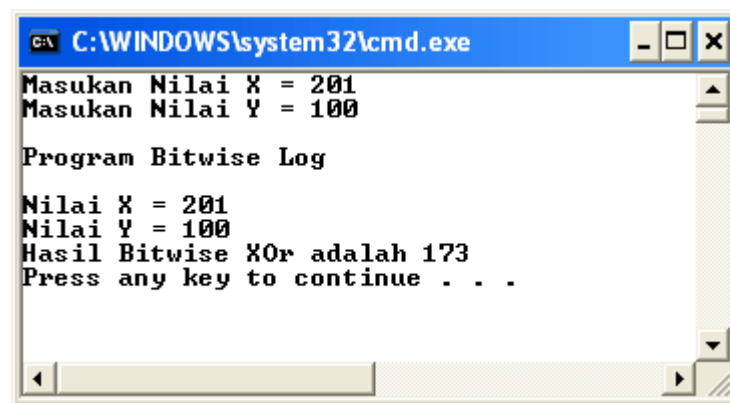
```

```

11     Scanner input = new Scanner(System.in);
12
13     int x, y, hasil;
14
15     System.out.print("Masukan Nilai X = ");
16     x = input.nextInt();
17
18     System.out.print("Masukan Nilai Y = ");
19     y = input.nextInt();
20
21     // Proses
22     hasil = x ^ y;
23
24     System.out.println("\nProgram Bitwise Log\n");
25     System.out.println("Nilai X = " + x);
26     System.out.println("Nilai Y = " + y);
27     System.out.println("Hasil Bitwise XOR adalah " + hasil);
28 }
29

```

Output yang dihasilkan dari program Lat314.java diatas, seperti dibawah ini :



Gambar 3.14. Hasil Program Lat313java

3.7.6 Operator Bitwise ~ (Not)

Operator Bitwise ~ (Not) digunakan membalik nilai bit dari suatu operand. Berikut anda dapat melihat ilustrasi untuk membandingkan bit dari 2 operand.

Tabel. 3.12. Tabel Operator Bitwise Not

Bit Operand	Hasil
0	1
1	0

Contoh :

$$\begin{array}{cccccccccccccc}
 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & = 8 \\
 \downarrow & \downarrow & \downarrow & \downarrow & & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
 1 & 1 & 1 & 1 & \dots & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & = \sim 8 = -9
 \end{array}$$

Pada operasi Bitwise Not ini, hasil dari pertukaran nilai bit, akan menjadi tidak terhingga, maka nilai yang dihasilkan adalah $-(\text{nilai operan} + 1)$. Jika nilai operan yang dimasukan adalah 8, maka akan menghasilkan nilai tak terhingga ~ 8 , hasilnya adalah $-(8+1) \rightarrow -9$

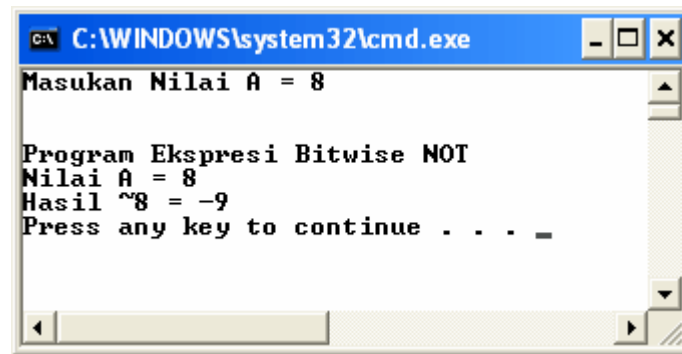
Berikut contoh penggunaan operator Bitwise NOT, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat315.java
3     Author   : Frieyadie
4  ----- */
5  import java.util.Scanner;
6
7  class Lat315
8  {
9      public static void main(String[] args)
10     {
11         Scanner input = new Scanner(System.in);
12
13         int a, b;
14
15         System.out.print("Masukan Nilai A = ");
16         a = input.nextInt();
17
18         // Proses
19
20         b = ~a;
21
22         System.out.println("\n");
23         System.out.println("Program Ekspresi Bitwise NOT");
24         System.out.println("Nilai A = " + a);
25         System.out.println("Hasil ~" + a + " = " + b);
26     }
27 }

```

Output yang dihasilkan dari program Lat315.java diatas, seperti dibawah ini :



Gambar 3.15. Hasil Program Lat315.java

3.8 Latihan

1. Tentukan apa hasil logkanya dari ekspresi relasi dan logika dibawah ini. Diberikan nilai
 $A = 3; B = 6; C = 2; K = 5; L = 4; M = 3$
 $D = (4 + 2 > A \ \&\& \ B - 2 > 3 + 2 \ || \ B + 2 <= 6 + 2)$
 $K + 5 < M \ || \ (C * M < L \ \&\& \ 2 * M - L > 0)$
 $L + 5 < M \ || \ C * K < L \ \&\& \ 2 * K - L > 0$
 $A * 4 <= 3 * M + B$
 $K + 10 > A \ \&\& \ L - 2 > 4 * C$
2. Dari program dibawah ini, analisa bagaimanakah keluaran yang dihasilkan dan ada kesalahan apa ?

```
class Tugas302
{
    public static void main(args)
    {
        int a = 21;

        System.out.println("Nilai a = " + a);
        System.out.println("Nilai ++a = " + ++a);
        System.out.println("Nilai ++a = " + ++a);
        System.out.println("Nilai a = " + a);

        a+=3

        System.out.println("\n\nNilai a = " + a);
        System.out.println("Nilai ++a = " + ++a);
        System.out.println("Nilai ++a = " + ++a);
        System.out.println("Nilai --a = " + --a);
        System.out.println("Nilai a = " + a--);
    }
}
```

3. Dari program dibawah ini, bagaimanakah keluaran yang dihasilkan

```
class Tugas303
{
    public static void main(String[] args)
    {
        int a = 25;

        System.out.println("Nilai a = " + a);
        System.out.println("Nilai a++ = " + a++);
        System.out.println("Nilai a = " + ++a);
        System.out.println("Nilai a-- = " + a--);
        System.out.println("Nilai a = " + a);

        a*=2;

        System.out.println("\n\nNilai a = " + a);
        System.out.println("Nilai a++ = " + a++);
        System.out.println("Nilai ++a = " + ++a);
        System.out.println("Nilai --a = " + --a);
        System.out.println("Nilai a-- = " + a--);
    }
}
```

Lembar ini sengaja dikosongkan

Bab 4:

Operator Koondisi Bahasa Java

4.1 Kopetensi Dasar

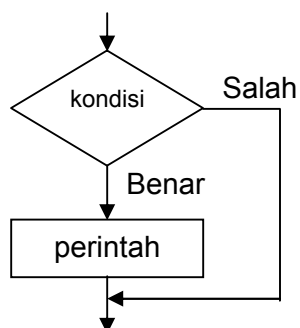
Pada pembahasan Bab 4 ini penulis mengajak mendiskusikan mengenai penggunaan operasi kondisi yang pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami operasi kondisi pada bahasa pemrograman Java. Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Penggunaan Pernyataan If, If – Else, Nested If, dan If Majemuk
- Penggunaan Pernyataan Case
- Conditional Operator

Untuk keperluan pengambilan keputusan, Java menyediakan beberapa perintah antara lain.

4.2 Pernyataan IF

Pernyataan if mempunyai pengertian, “ Jika kondisi bernilai benar, maka perintah akan dikerjakan dan jika tidak memenuhi syarat maka akan diabaikan”. Dari pengertian tersebut dapat dilihat dari diagram alir berikut:



Gambar 4.1. Flow Chart Pernyataan IF

Penulisan kondisi harus didalam tanda kurung dan berupa ekspresi relasi dan penulisan pernyataan dapat berupa sebuah pernyataan tunggal, pernyataan majemuk atau pernyataan kosong. Jika pemakaian if diikuti dengan pernyataan majemuk, bentuk penulisannya sebagai berikut :

```
if (kondisi)
    pernyataan;
```

Jika lebih dari satu pernyataan harus diapit dengan tanda kurung kurawal

```
if (kondisi)
{
    pernyataan;
    .....
}
```

Contoh

Menentukan besarnya potongan dari pembelian barang yang diberikan seorang pembeli, dengan kriteria :

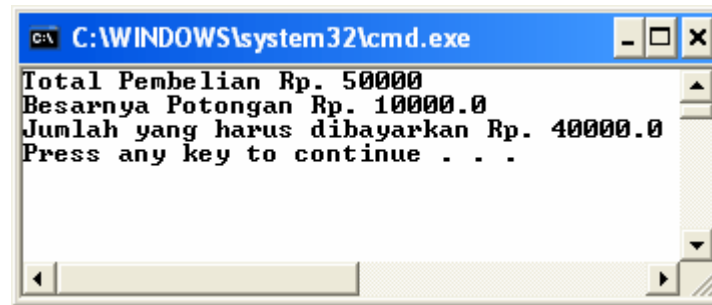
Tidak ada potongan jika total pembelian kurang dari Rp. 50.000,-

Jika total pembelian lebih dari atau sama dengan Rp. 50.000,- potongan yang diterima sebesar 20% dari total pembelian.

Berikut contoh penggunaan pernyataan if sederhana, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2     Nama File : Lat401.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat401
8  {
9      public static void main(String[] args)
10     {
11         double tot_beli, potongan=0, jum_bayar=0;
12         Scanner input = new Scanner(System.in);
13
14         System.out.print("Total Pembelian Rp. ");
15         tot_beli = input.nextDouble();
16
17         if (tot_beli >= 50000)
18             potongan = 0.2 * tot_beli;
19
20         System.out.println("Besarnya Potongan Rp. " + potongan);
21
22         jum_bayar = tot_beli - potongan;
23
24         System.out.println("Jumlah yang harus dibayarkan Rp. " + jum_bayar);
25     }
26 }
```

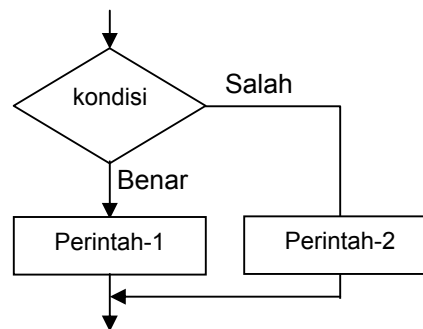
Output yang dihasilkan dari program Lat401.java diatas, seperti dibawah ini :



Gambar 4.2 Hasil Program Lat401.java

4.2.1 Pernyataan IF - ELSE

Pernyataan if mempunyai pengertian, “ Jika kondisi bernilai benar, maka perintah-1 akan dikerjakan dan jika tidak memenuhi syarat maka akan mengerjakan perintah-2”. Dari pengertian tersebut dapat dilihat dari diagram alir berikut :



Gambar 4.3. Diagram Alir if-else

Perintah-1 dan perintah-2 dapat berupa sebuah pernyataan tunggal, pernyataan majemuk atau pernyataan kosong. Jika pemakaian if-else diikuti dengan pernyataan majemuk, bentuk penulisannya sebagai berikut:

```
if (kondisi)
    pernyataan-1;
else
    pernyataan-1;
```

Jika lebih dari satu pernyataan harus diapit dengan tanda kurung kurawal

```

if (kondisi)
{
    perintah-1;
    ...
}
else
{
    perintah-2;
    ...
}

```

Contoh :

Menentukan besarnya potongan dari pembelian barang yang diberikan seorang pembeli, dengan kriteria :

- a. jika total pembelian kurang dari Rp. 50.000,- potongan yang diterima sebesar 5% dari total pembelian.
- b. Jika total pembelian lebih dari atau sama dengan Rp. 50.000,- potongan yang diterima sebesar 20% dari total pembelian.

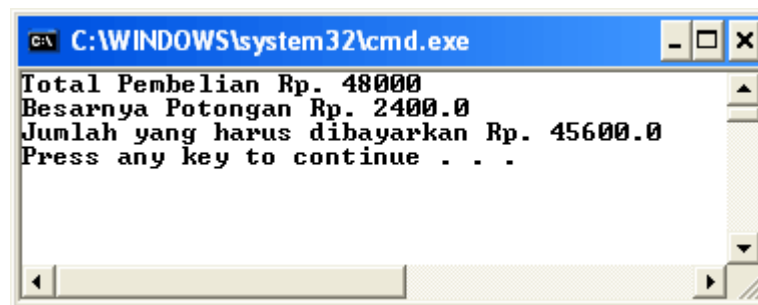
Berikut contoh penggunaan pernyataan if - else, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat402.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat402
8  {
9      public static void main(String[] args)
10     {
11         double tot_beli, potongan=0, jum_bayar=0;
12         Scanner input = new Scanner(System.in);
13
14         System.out.print("Total Pembelian Rp. ");
15         tot_beli = input.nextDouble();
16
17         if (tot_beli >= 50000)
18             potongan = 0.2 * tot_beli;
19         else
20             potongan = 0.05 * tot_beli;
21
22         System.out.println("Besarnya Potongan Rp. " + potongan);
23
24         jum_bayar = tot_beli - potongan;
25
26         System.out.println("Jumlah yang harus dibayarkan Rp. " + jum_bayar);
27     }
28 }

```


Output yang dihasilkan dari program Lat402.java diatas, seperti dibawah ini :



Gambar 4.4 Hasil Program Lat402.java

4.2.2 Pernyataan NESTED IF

Nested if merupakan pernyataan if berada didalam pernyataan if yang lainnya. Bentuk penulisan pernyataan Nested if adalah :

```
if(syarat)
    if(syarat)
        ... perintah;
    else
        ... perintah;
else
    if(syarat)
        ... perintah;
    else
        ... perintah;
```

Contoh :

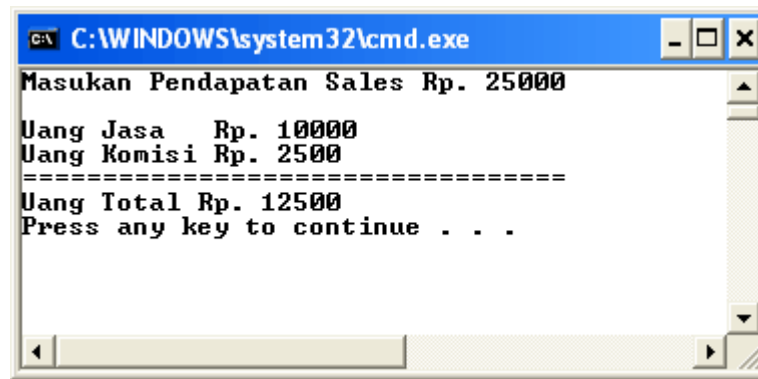
Suatu perusahaan memberikan komisi kepada para salesman dengan ketentuan sebagai berikut:

- Bila salesman dapat menjual barang hingga Rp. 20.000 ,- , akan diberikan uang jasa sebesar Rp. 10.000 ditambah dengan uang komisi Rp. 10% dari pendapatan yang diperoleh hari itu.
- Bila salesman dapat menjual barang diatas Rp. 20.000 ,- , akan diberikan uang jasa sebesar Rp. 20.000 ditambah dengan uang komisi Rp. 15% dari pendapatan yang diperoleh hari itu.
- Bila salesman dapat menjual barang diatas Rp. 50.000 ,- , akan diberikan uang jasa sebesar Rp. 30.000 ditambah dengan uang komisi Rp. 20% dari pendapatan yang diperoleh hari itu.

Berikut contoh penggunaan pernyataan nested - if, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2     Nama File : Lat403.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat403
8  {
9      public static void main(String[] args)
10     {
11         int pendapatan;
12         double jasa, komisi, total;
13
14         /deklarasi objek
15         Scanner input = new Scanner(System.in);
16
17         System.out.print("Masukan Pendapatan Sales Rp. ");
18         pendapatan = input.nextInt();
19
20         if (pendapatan >= 0 && pendapatan <= 200000)
21         {
22             jasa=10000;
23             komisi=0.1*pendapatan;
24         }
25         else
26         {
27             if(pendapatan<=500000)
28             {
29                 jasa=20000;
30                 komisi=0.15*pendapatan;
31             }
32             else
33             {
34                 jasa=30000;
35                 komisi=0.2*pendapatan;
36             }
37         }
38
39         /* menghitung total */
40         total = komisi+jasa;
41
42         System.out.println("\nUang Jasa   Rp. " + (int) jasa);
43         System.out.println("Uang Komisi Rp. " + (int) komisi);
44         System.out.println("=====");
45         System.out.println("Uang Total Rp. " + (int) total);
46
47     }
48 }
49
50
```

Output yang dihasilkan dari program Lat403.java diatas, seperti dibawah ini :



Gambar 4.5 Hasil Program Lat403.java

4.2.3 Pernyataan IF – ELSE Majemuk

Bentuk dari if-else bertingkat sebenarnya mirip dengan nested if, keuntungan penggunaan if-else bertingkat dibanding dengan nested if adalah penggunaan bentuk penulisan yang lebih sederhana.

Bentuk Penulisannya

```
if (syarat)
{
    ... perintah;
    ... perintah;
}
else if (syarat)
{
    ... perintah;
    ... perintah;
}
else
{
    ... perintah;
    ... perintah;
}
```

Contoh :

Suatu perusahaan memberikan komisi kepada para salesman dengan ketentuan sebagai berikut:

- a. Bila salesman dapat menjual barang hingga Rp. 200.000 ,- , akan diberikan uang jasa sebesar Rp. 10.000 ditambah dengan uang komisi Rp. 10% dari pendapatan yang diperoleh hari itu.

- b. Bila salesman dapat menjual barang diatas Rp. 200.000 ,- , akan diberikan uang jasa sebesar Rp. 20.000 ditambah dengan uang komisi Rp. 15% dari pendapatan yang diperoleh hari itu.
- c. Bila salesman dapat menjual barang diatas Rp. 500.000 ,- , akan diberikan uang jasa sebesar Rp. 30.000 ditambah dengan uang komisi Rp. 20% dari pendapatan yang diperoleh hari itu.

Berikut contoh penggunaan pernyataan if – else majemuk, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat404.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat404
8  {
9      public static void main(String[] args)
10     {
11         int pendapatan;
12         double jasa, komisi, total;
13
14         //deklarasi objek
15         Scanner input = new Scanner(System.in);
16
17         System.out.print("Masukan Pendapatan Sales Rp. ");
18         pendapatan = input.nextInt();
19
20         if (pendapatan >= 0 && pendapatan <= 200000)
21         {
22             jasa=10000;
23             komisi=0.1*pendapatan;
24         }
25         else if(pendapatan<=500000)
26         {
27             jasa=20000;
28             komisi=0.15*pendapatan;
29         }
30         else
31         {
32             jasa=30000;
33             komisi=0.2*pendapatan;
34         }
35
36         /* menghitung total */
37         total = komisi+jasa;
38
39         System.out.println("\nUang Jasa   Rp. " + (int) jasa);
40         System.out.println("Uang Komisi Rp. " + (int) komisi);
41         System.out.println("=====");
42         System.out.println("Uang Total Rp. " + (int) total);
43     }
44 }
45
46

```

Output yang dihasilkan dari program Lat404.java diatas, seperti dibawah ini :



Gambar 4.6 Hasil Program Lat404.java

4.3 Pernyataan switch - case

Bentuk dari switch - case merupakan pernyataan yang dirancang khusus untuk menangani pengambilan keputusan yang melibatkan sejumlah atau banyak alternatif. Pernyataan switch - case ini memiliki kegunaan sama seperti if – else bertingkat, tetapi penggunaannya hanya untuk memeriksa data yang bertipe primitif integer saja. Bentuk penulisan perintah ini sebagai berikut :

```
switch (ekspresi integer)
{
    case konstanta-1 :
        ... perintah;
        ... perintah;
        break;
    case konstanta-2 :
        ... perintah;
        ... perintah;
        break;

    .....
    .....
    default :
        ... perintah;
        ... perintah;
```

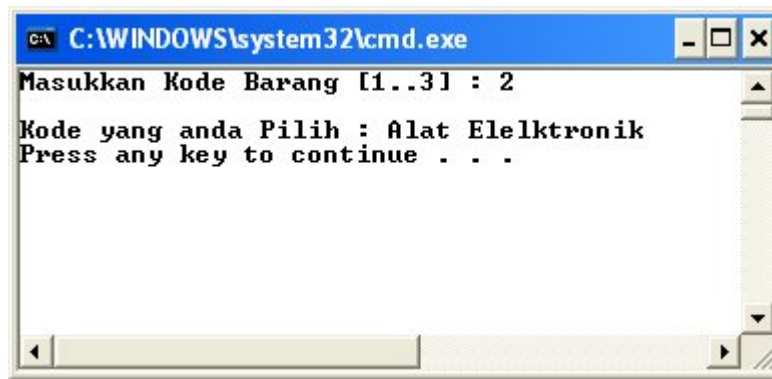
Setiap cabang akan dijalankan jika syarat nilai konstanta tersebut dipenuhi dan default akan dijalankan jika semua cabang diatasnya tidak terpenuhi.

Pernyataan break menunjukkan bahwa perintah siap keluar dari switch. Jika pernyataan ini tidak ada, maka program akan diteruskan kecabang – cabang yang lainnya.

Berikut contoh penggunaan pernyataan switch - case, bisa anda ikuti seperti dibawah ini.

```
1  /* -----
2     Nama File : Lat405.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7
8  class Lat405
9  {
10     public static void main(String[] args)
11     {
12         int kode;
13         String nmbarang;
14
15         Scanner input = new Scanner(System.in);
16
17         System.out.print("Masukkan Kode Barang [A..C] : ");
18         kode = input.nextInt();
19
20         switch(kode)
21         {
22             case 1 :
23                 nmbarang = "Alat Olah Raga";
24                 break;
25             case 2 :
26                 nmbarang = "Alat Elelektronik";
27                 break;
28             case 3 :
29                 nmbarang = "Alat Masak";
30                 break;
31             default:
32                 nmbarang = "Anda Salah Memasukan kode";
33                 break;
34         }
35         System.out.println("\nKode yang anda Pilih : " + nmbarang);
36     }
37 }
```

Output yang dihasilkan dari program Lat405.java diatas, seperti dibawah ini :



Gambar 4.7 Hasil Program Lat405.java

4.4 Operator ?:

Operator ?: disebut dengan Conditional Operator atau Operator Kondisi yang digunakan untuk menyeleksi nilai untuk mendapatkan hasil dari kondisi yang diseleksi. Operator ?: ini tergolong kedalam operator ternary.

Bentuk Penulisan :

Ekspresi Logika-OR ? Ekspresi : Ekspresi Kondisi

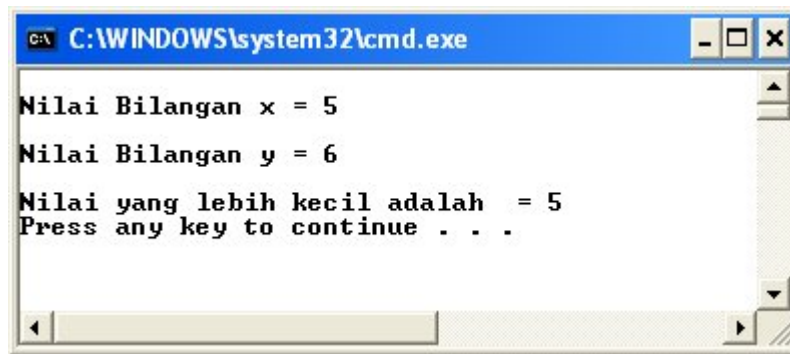
Berikut contoh penggunaan pernyataan Conditional Operator, bisa anda ikuti seperti dibawah ini.

```

1  /* -----
2     Nama File : Lat406.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat406
7  {
8      public static void main(String[] args)
9      {
10         int x, y , z ;
11
12         x = 5;
13         y = 6;
14
15         z = (x < y) ? x : y;
16
17         System.out.println("\nNilai Bilangan x = " + x);
18         System.out.println("\nNilai Bilangan y = " + y);
19         System.out.println("\nNilai yang lebih kecil adalah = " + z);
20     }
21 }

```

Output yang dihasilkan dari program Lat406.java diatas, seperti dibawah ini :

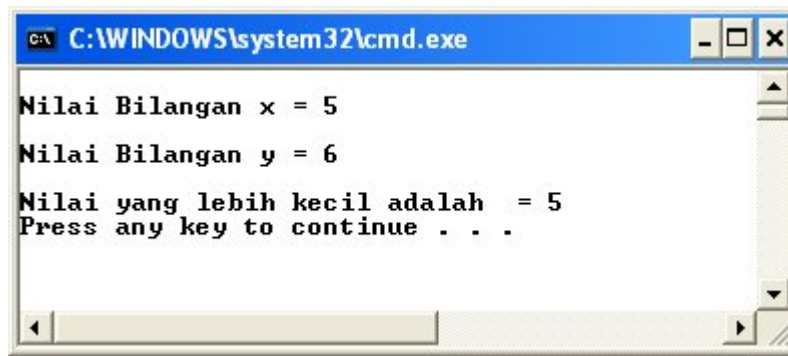


Gambar 4.8 Hasil Program Lat406.java

Pada contoh program – 7 diatas, merupakan pengaplikasian dari perintah if – else berikut :

```
1  /* -----
2     Nama File : Lat407.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat407
7  {
8      public static void main(String[] args)
9      {
10         int x, y , z ;
11
12         x = 5;
13         y = 6;
14
15         if (x < y)
16             z = x;
17         else
18             z = y;
19
20         System.out.println("\nNilai Bilangan x = " + x);
21         System.out.println("\nNilai Bilangan y = " + y);
22         System.out.println("\nNilai yang lebih kecil adalah = " + z);
23     }
24 }
```

Output yang dihasilkan dari program Lat407.java diatas, seperti dibawah ini :



Gambar 4.9 Hasil Program Lat407.java

4.5 Latihan

1. Buatlah program untuk menghitung nilai rata-rata dari seorang siswa, dengan ketentuan sebagai berikut :
 - a. Nama Siswa, Nilai Pertandingan I, Nilai Pertandingan II, Nilai Pertandingan III diinput.
 - b. Nilai Rata-rata merupakan hasil dari Nilai Pertandingan I, II dan III dibagi dengan 3.
 - c. Ketentuan Mendapat Hadiah dari pertandingan :
 - 1) Jika Nilai Rata-Rata ≥ 85 , maka mendapat hadiah Seperangkat Komputer P4
 - 2) Jika Nilai Rata-Rata ≥ 70 , maka mendapat hadiah Seperangkat Uang sebesar Rp. 500,000
 - 3) Jika Nilai Rata-Rata < 70 , maka mendapat hadiah Hiburan
 - d. Tampilan yang diinginkan sebagai berikut :
 - 1) Layar Masukan

PROGRAM HITUNG NILAI RATA-RATA

```
Nama Siswa           : ... <diinput>
Nilai Pertandingan I  : ... <diinput>
Nilai Pertandingan II : ... <diinput>
Nilai Pertandingan III : ... <diinput>
```

2) Layar Keluaran

```
Siswa yang bernama ... <tampil data>
Memperoleh nilai rata-rata <hasil proses> dari hasil perlombaan
yang diikutinya.
```

Hadiah yang didapat adalah ... <hasil proses>

2. Buatlah program untuk menghitung nilai akhir seorang siswa dari kursus yang diikutinya.

Dengan ketentuan sebagai berikut :

- a. Nama Siswa, Nilai Keaktifan, Nilai Tugas dan Nilai Ujian diinput.
- b. Proses yang dilakukan untuk mendapatkan nilai murni dari masing-masing nilai, adalah
 - 1) Nilai Murni Keaktifan = Nilai Keaktifan dikalikan dengan 20%.
 - 2) Nilai Murni Tugas = Nilai Tugas dikalikan dengan 30%
 - 3) Nilai Murni Ujian = Nilai Ujian dikalikan dengan 50%
 - 4) Nilai Akhir adalah Nilai Murni Keaktifan + Nilai Murni Tugas + Nilai Murni Ujian
- c. Ketentuan untuk mendapatkan grade nilai :
 - 1) Nilai Akhir ≥ 80 mendapat Grade A
 - 2) Nilai Akhir ≥ 70 mendapat Grade B
 - 3) Nilai Akhir ≥ 59 mendapat Grade C
 - 4) Nilai Akhir ≥ 50 mendapat Grade D
 - 5) Nilai Akhir < 50 mendapat Grade E
- d. Tampilan yang diinginkan sebagai berikut :
 - 1) Layar Masukkan

PROGRAM HITUNG NILAI AKHIR

```
Nama Siswa : .....<diinput>
Nilai Keaktifan : ..... <diinput>
Nilai Tugas      : ..... <diinput>
Nilai Ujian      : ..... <diinput>
```

2) Layar Keluaran

```
Siswa yang bernama <tampil data>
Dengan Nilai Persentasi Yang dihasilkan.
Nilai Keaktifan * 20% : ...<hasil proses>
Nilai Tugas      * 30% : ...<hasil proses>
Nilai Ujian      * 50% : ...<hasil proses>
```

```
Jadi Siswa yang bernama <tampil data> memperoleh nilai akhir
sebesar ... <hasil proses>
Grade nilai yang didapat adalah ... <hasil proses>
```

3. Buatlah program untuk menghitung total pembayaran dari sebuah penjualan agen susu di kota besar ini.. Dengan ketentuan sebagai berikut :

a. Jenis susu diinput diinput berdasarkan kode yang sudah ditentukan, yaitu :

- Jika kode A adalah Dancow
- Jika kode B adalah Bendera
- Jika kode A adalah SGM

b. Ukuran kaleng susu diinput berdasarkan kode yang sudah ditentukan.

- Jika kode 1 adalah Kecil
- Jika kode 2 adalah Sedang
- Jika kode 3 adalah Besar

c. Harga susu sesuai dengan jenis susu dan ukuran kaleng susu

JENIS SUSU	HARGA BERDASARKAN UKURAN KALENG SUSU		
	KECIL	SEDANG	BESAR
DANCOW	25000	20000	15000
BENDERA	20000	17500	13500
SGM	22000	18500	15000

d. Proses yang dilakukan untuk mendapatkan Total Pembayaran

Total Bayar = Harga Susu per ukuran dan Jenis dikali dengan banyak beli

e. Tampilan Layar Masukkan dan Keluaran yang diinginkan sebagai berikut :

TOKO KELONTONG KERONCONGAN

- A. Susu Dancow
 - 1. Ukuran Kecil
 - 2. Ukuran Sedang
 - 3. Ukuran Besar
- B. Susu Bendera
 - 1. Ukuran Kecil
 - 2. Ukuran Sedang
 - 3. Ukuran Besar
- C. Susu SGM
 - 1. Ukuran Kecil
 - 2. Ukuran Sedang
 - 3. Ukuran Besar

Masukan Merk Susu [Dancow | Bendera | SGM] : < diinput >

Masukan Ukuran Kaleng [Kecil|Sedang|Besar] : < diinput >

Harga Satuan Barang Rp.< tampil harga satuan >

Jumlah Yang dibeli : ... < diinput >

Harga Yang Harus dibayar Sebesar Rp. <hasil proses>

- 4 DINGIN DAMAI, memberikan Honor tetap kepada karyawan kontraknya sebesar Rp. 300,000,- per bulan, dengan memperoleh tujangan-tunjangan sebagai berikut :

a. Tunjangan Jabatan

Golongan	Persentase
1	5%
2	10%
3	15%

Sebagai contoh : Jika seorang karyawan tersebut dengan golongan 3, maka mendapatkan tunjangan sebesar 15% * Rp. 300,000,-

b. Tunjangan Pendidikan

Kode	Pendidikan	Persentase
1	SMU	2,5%
2	D3	5%
3	S1	7,5%

c. Honor Lembur

Jumlah jam kerja normal sebanyak 8 Jam Kerja. Honor lembur diberikan jika jumlah jam kerja lebih dari 8 jam, maka kelebihan jam kerja tersebut dikalikan dengan honor lembur perjam sebesar Rp. 2,500 untuk setiap kelebihan jam kerja perharinya.

d. Tampilan yang diinginkan sebagai berikut :

- Layar Masukkan

Program Hitung Honor Karyawan Kontrak
PT. DINGIN DAMAI

Nama Karyawan : ... <di input>
Golongan : ... <di input>
Pendidikan (SMU/D3/S1) : ... <di input>
Jumlah Jam Kerja: ... <di input>

- **Layar Keluaran**

Karyawan yang bernama : ... <tampil data>
Honor yang diterima

Honor Tetap	Rp.	<hasil proses>
Tunjangan jabatan	Rp.	<hasil proses>
Tunjangan Pendidikan	Rp.	<hasil proses>
Honor Lembur	Rp.	<hasil proses>
		+
Honor Yang Diterima	Rp.	<hasil proses>

Lembar ini sengaja dikosongkan

Bab 5:

Perintah Perulangan Pada Pemrograman Java

5.1 Kopetensi Dasar

Pada pembahasan Bab 5 ini penulis mengajak mendiskusikan mengenai penggunaan operasi kondisi yang pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami perintah perulangan pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Penggunaan Pernyataan Perulangan for, nested for dan perulangan tak berhingga dengan perulangan for.
- Penggunaan Pernyataan Perulangan While dan Do While
- Perintah Break dan Continue

Pernyataan Perulangan digunakan untuk melakukan proses yang sifatnya mengulang pada pemrograman java. Untuk keperluan perulangan proses, Java menyediakan beberapa perintah perulangan, yaitu: for, while dan do-while.

5.2 Pernyataan for

Perulangan yang pertama adalah for. Bentuk umum pernyataan for sebagai berikut :

```
for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah )
```

Bila pernyataan didalam for lebih dari satu maka pernyataan-pernyataan tersebut harus diletakan didalam tanda kurung.

```
for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah )
{
    pernyataan / perintah;
    pernyataan / perintah;
    pernyataan / perintah;
}
```

Kegunaan dari masing-masing argumen for diatas adalah :

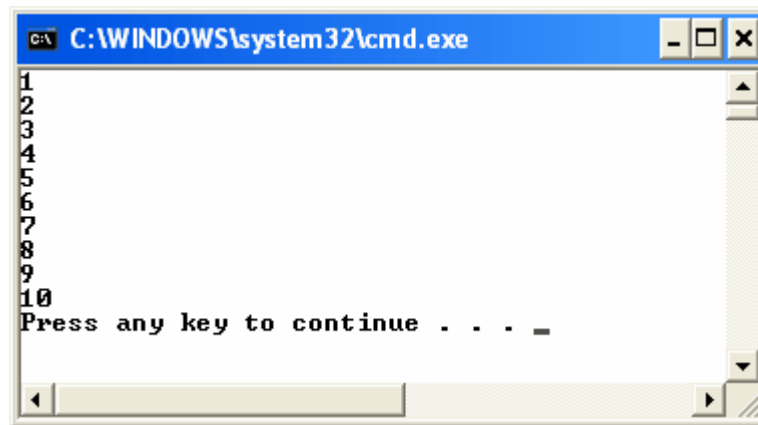
- Inisialisasi : merupakan bagian untuk memberikan nilai awal untuk variabel-variabel tertentu.
- Syarat Pengulangan : memegang kontrol terhadap pengulangan, karena bagian ini yang akan menentukan suatu pengulangan diteruskan atau dihentikan.
- Pengubah Nilai Pencacah : mengatur kenaikan atau penurunan nilai pencacah.

Contoh :

Sebagai contoh program untuk mencetak bilangan dari 1 hingga 10 secara menaik, secara selengkapnya seperti dibawah ini:

```
1  /* -----
2     Nama File : Lat501.java
3     Author   : Frieyadie
4     ----- */
5
6  class Lat501
7  {
8      public static void main(String[] args)
9      {
10         int a;
11
12         for(a = 1; a <= 10; ++a)
13             System.out.println(a);
14     }
15 }
```

Output yang dihasilkan dari program Lat501.java diatas, seperti dibawah ini :

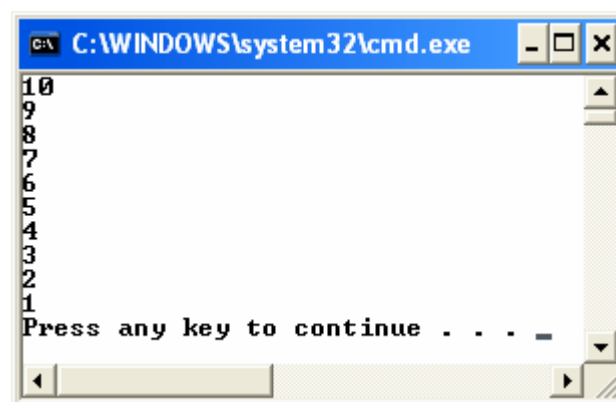


Gambar 5.1 Hasil Program Lat501.java

Sebagai contoh program untuk mencetak bilangan dari 10 hingga 1 secara menurun, selengkapnya seperti dibawah ini:

```
1  /* -----  
2  Nama File : Lat502.java  
3  Author   : Frieyadie  
4  ----- */  
5  
6  class Lat502  
7  {  
8      public static void main(String[] args)  
9      {  
10         int a;  
11  
12         for(a = 10; a >= 1; --a)  
13             System.out.println(a);  
14     }  
15 }
```

Output yang dihasilkan dari program Lat502.java diatas, seperti dibawah ini :



Gambar 5.2 Hasil Program Lat502.java

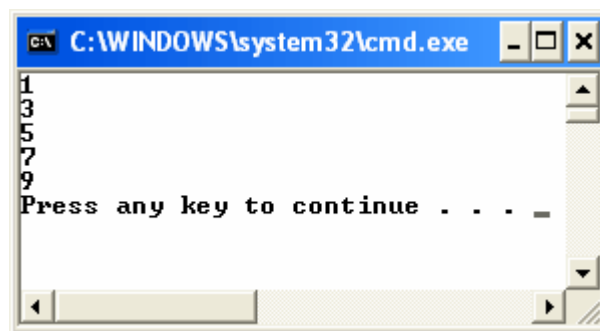
Sebagai contoh program untuk mencetak bilangan dari 1 hingga 10, dengan loncat dua angka secara menaik, selengkapnya seperti dibawah ini::

```

1  /* -----
2     Nama File : Lat503.java
3     Author    : Frieyadie
4  ----- */
5
6  class Lat503
7  {
8      public static void main(String[] args)
9      {
10         int a;
11
12         for(a = 1; a <= 10; a+=2)
13             System.out.println(a);
14     }
15 }

```

Output yang dihasilkan dari program Lat503.java diatas, seperti dibawah ini :



Gambar 5.3 Hasil Program Lat503.java

5.1.1 Pernyataan nested - for

Pernyataan Nested for adalah suatu perulangan for didalam perulangan for yang lainnya. Bentuk umum pernyataan Nested for sebagai berikut :

```

for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah )
{
    for ( inisialisasi; syarat pengulangan; pengubah nilai pencacah)
    {
        pernyataan / perintah;
    }
}

```



Didalam penggunaan nested-for, perulangan yang didalam terlebih dahulu dihitung hingga selesai, kemudian perulangan yang diluar diselesaikan.

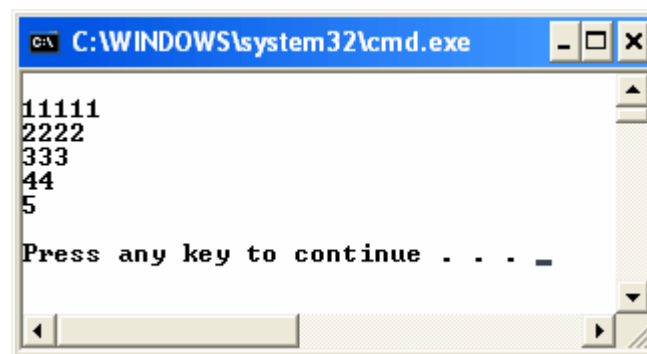
Sebagai contoh program menerapkan perintah nested - for, selengkapnya seperti dibawah ini::

```

1  /* -----
2     Nama File : Lat504.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat504
7  {
8      public static void main(String[] args)
9      {
10         int a, b;
11
12         for(a = 1; a <= 5; a++)
13         {
14             System.out.println();
15             for(b = a; b <= 5; b++)
16             {
17                 System.out.print(a);
18             }
19             System.out.println("\n");
20         }
21     }
22 }

```

Output yang dihasilkan dari program Lat504.java diatas, seperti dibawah ini :



Gambar 5.4 Hasil Program Lat504.java

Dapat anda analisa kenapa bisa hasilnya menjadi seperti gambar 5.6, diatas ?, berikut dapat dilihat pada tabel proses berikut ini :

a	a<=5	? \n	b=a	b<=5	? a	b++	a++
1	1<=5	✓	1	1<=5	1	2	
			2	2<=5	1	3	
			3	3<=5	1	4	
			4	4<=5	1	5	
			5	5<=5	1	6	

			6	6<=5	-	-	2
2	2<=5	✓	2	2<=5	2	3	
			3	3<=5	2	4	
			4	4<=5	2	5	
			5	5<=5	2	6	
			6	6<=5	-	-	3
3	3<=5	✓	3	3<=5	3	4	
			4	4<=5	3	5	
			5	5<=5	3	6	
			6	6<=5	-	-	4
4	4<=5	✓	4	4<=5	4	5	
			5	5<=5	4	6	
			6	6<=5	-	-	5
5	5<=5	✓	5	5<=5	5	6	
	6<=5	Proses perulangan Selesai					

Maka tampilan hasil yang diperoleh dari perulangan diatas bisa anda lihat pada variabel a, variabel ini yang mencetak, semua nilai pada variabel a tersebut.

5.1.2 Perulangan Tidak Berhingga

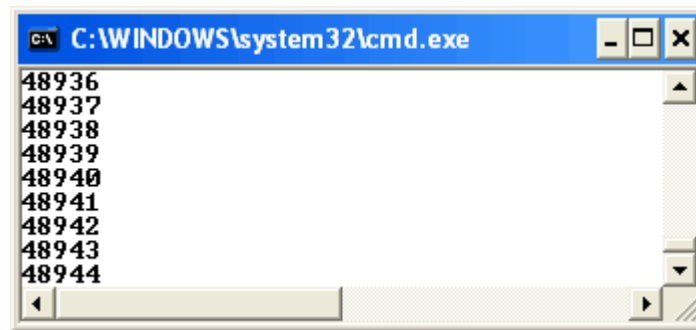
Perulangan tak berhingga merupakan perulangan (loop) yang tak pernah berhenti atau mengulang terus, hal ini sering terjadi disebabkan adanya kesalahan penanganan kondisi yang dipakai untuk keluar dari loop. Sebagai contoh, jika penulisan perintah sebagai berikut:

Sebagai contoh program menerapkan perintah perulangan tak berhingga, selengkapnya seperti dibawah ini::

```

1  /* -----
2     Nama File : Lat505.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat505
7  {
8      public static void main(String[] args)
9      {
10         int bil;
11
12         for (bil = 60; bil >=10; bil++)
13             System.out.println(bil);
14     }
15 }
```

Output yang dihasilkan dari program Lat505.java diatas, seperti dibawah ini :



Gambar 5.5 Hasil Program Lat505.java

Pada pernyataan ini tidak akan berhenti untuk menampilkan bilangan menurun, kesalahan terjadi pada pengubah nilai pencacah, seharusnya penulisan yang benar berupa

`bil --`

Akan tetapi yang ditulis adalah :

`bil ++`

Oleh karena kondisi `bil >= 1` selalu bernilai benar (karena bil bernilai 60), maka pernyataan `System.out.println(bil);` akan terus dijalankan.

5.3 Pernyataan while

Pernyataan perulangan while merupakan instruksi perulangan yang mirip dengan perulangan for. Bentuk perulangan while dikendalikan oleh syarat tertentu, yaitu perulangan akan terus dilaksanakan selama syarat tersebut terpenuhi.

Bentuk umum perulangan while, sebagai berikut :

```
while ( syarat )  
  
    Pernyataan / perintah ;
```

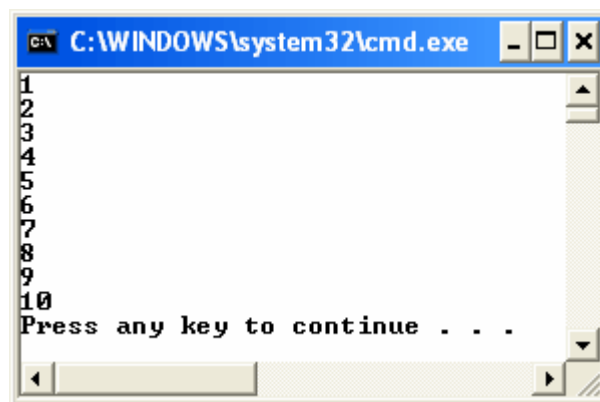
Bentuk umum perulangan while, dengan lebih dari perintah / pernyataan, sebagai berikut :

```
while ( syarat )  
{  
    Pernyataan / perintah ;  
    Pernyataan / perintah ;  
}
```

Sebagai contoh program menerapkan perintah perulangan while, selengkapnya seperti dibawah ini::

```
1  /* -----
2     Nama File : Lat506.java
3     Author   : Frieyadie
4     ----- */
5
6  class Lat506
7  {
8      public static void main(String[] args)
9      {
10         int bil=1;
11
12         while(bil<=10)
13         {
14             System.out.println(bil);
15             ++bil;
16         }
17     }
18 }
```

Output yang dihasilkan dari program Lat506.java diatas, seperti dibawah ini :



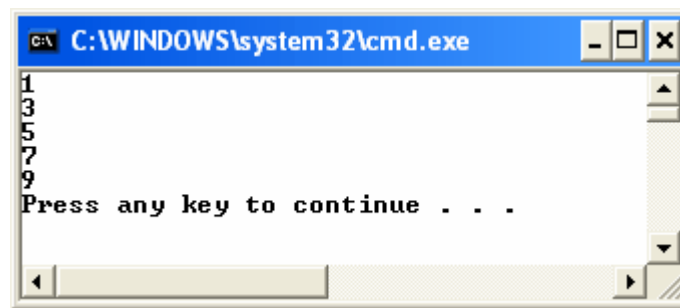
Gambar 5.6 Hasil Program Lat506.java

Sebagai contoh program menerapkan perintah perulangan while dengan penambahan pencacah sebanyak 2, selengkapnya seperti dibawah ini::

```
1  /* -----
2     Nama File : Lat506.java
3     Author   : Frieyadie
4     ----- */
5
6  class Lat506
7  {
8      public static void main(String[] args)
9      {
10         int bil=1;
11
12         while(bil<=10)
```

```
13 {  
14     System.out.println(bil);  
15     bil+=2;  
16 }  
17 }  
18 }
```

Output yang dihasilkan dari program Lat507.java diatas, seperti dibawah ini :



Gambar 5.7 Hasil Program Lat507.java

5.4 Pernyataan do - while

Pernyataan perulangan do - while merupakan bentuk perulangan yang melaksanakan perulangan terlebih dahulu dan pengujian perulangan dilakukan dibelakang.

Bentuk umum perulangan do - while, sebagai berikut :

```
do  
    pernyataan / perintah ;  
while ( syarat );
```

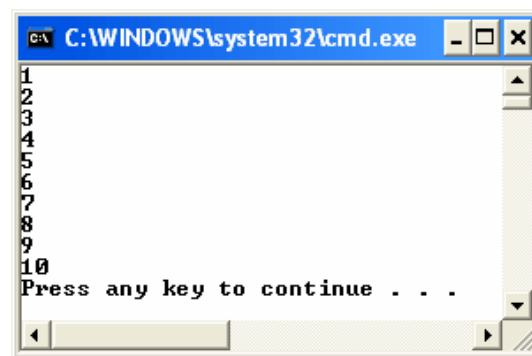
Bentuk umum perulangan do - while, dengan lebih dari perintah / pernyataan, sebagai berikut:

```
do  
{  
    Pernyataan / perintah ;  
    Pernyataan / perintah ;  
}  
while ( syarat );
```

Sebagai contoh program menerapkan perintah perulangan do - while, selengkapnya seperti dibawah ini:

```
1  /* -----  
2      Nama File : Lat508.java  
3      Author    : Frieyadie  
4  ----- */  
5  
6  class Lat508  
7  {  
8      public static void main(String[] args)  
9      {  
10         int bil=1;  
11  
12         do  
13         {  
14             System.out.println(bil);  
15             ++bil;  
16         }  
17         while (bil<=10);  
18     }  
19 }
```

Output yang dihasilkan dari program Lat508.java diatas, seperti dibawah ini :

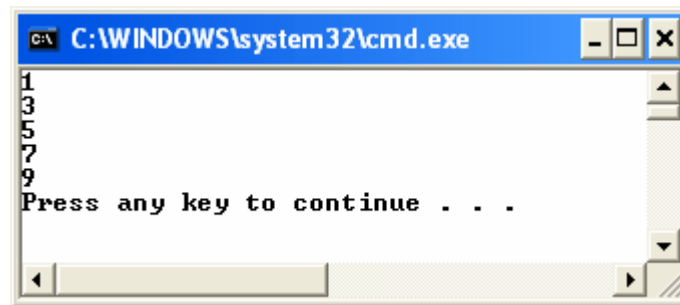


Gambar 5.8 Hasil Program Lat508.java

Sebagai contoh program menerapkan perintah perulangan do - while dengan penambahan pencacah sebanyak 2, selengkapnya seperti dibawah ini:

```
1  /* -----  
2      Nama File : Lat509.java  
3      Author    : Frieyadie  
4  ----- */  
5  
6  class Lat509  
7  {  
8      public static void main(String[] args)  
9      {  
10         int bil=1;  
11  
12         do  
13         {  
14             System.out.println(bil);  
15             bil+=2;  
16         }  
17         while (bil<=10);  
18     }  
19 }
```


Output yang dihasilkan dari program Lat509.java diatas, seperti dibawah ini :



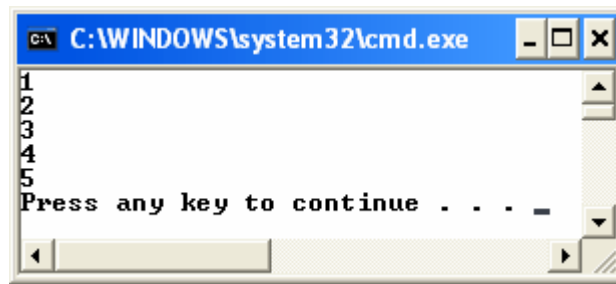
Gambar 5.9 Hasil Program Lat509.java

5.5 Pernyataan break

Pernyataan break telah dibahas pada pernyataan pengambilan keputusan switch. Pernyataan break ini berfungsi untuk keluar dari struktur switch. Selain itu pernyataan break berfungsi keluar dari perulangan (for, while dan do-while). Jika pernyataan break dikerjakan, maka eksekusi akan dilanjutkan ke pernyataan yang terletak sesudah akhir dari badan perulangan (loop). Sebagai contoh program menerapkan perintah perulangan do - while dengan menggunakan break, selengkapnya seperti dibawah ini:

```
1  /* -----
2     Nama File : Lat510.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat510
7  {
8      public static void main(String[] args)
9      {
10         int bil=1;
11
12         do
13         {
14             if(bil >= 6)
15                 break;
16
17             System.out.println(bil);
18
19             bil++;
20         }
21         while (bil<=10);
22     }
23 }
```

Output yang dihasilkan dari program Lat510.java diatas, seperti dibawah ini :



Gambar 5.10 Hasil Program Lat510.java



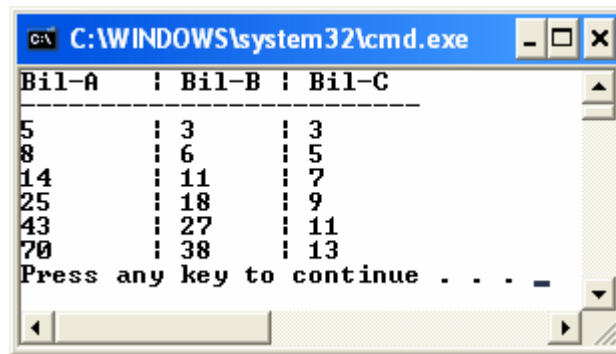
Pada saat nilai $bil = 6$, maka proses seleksi $bil \geq 6$, maka bernilai True, dan perintah `break` dijalankan, selanjutnya proses menuju ke akhir perulangan.

Sebagai contoh program menerapkan perintah perulangan `for` dengan menggunakan `break`, selengkapnya seperti dibawah ini:

```

1  /* -----
2     Nama File : Lat511.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat511
7  {
8      public static void main(String[] args)
9      {
10         int a=3, b=2, c=1, bil;
11
12         System.out.println("Bil-A   | Bil-B | Bil-C");
13         System.out.println("-----");
14
15         for(bil=1; bil<=10; ++bil)
16         {
17             a+=b; b+=c; c+=2;
18             System.out.println(a + "\t" + "|" + b + "\t" + "|" + c);
19
20             if(c==13)
21                 break;
22         }
23     }
24 }
```

Output yang dihasilkan dari program Lat511.java diatas, seperti dibawah ini :



Gambar 5.11 Hasil Program Lat511.java

5.6 Pernyataan continue

Pernyataan continue digunakan untuk mengarahkan eksekusi ke iterasi (proses) berikutnya pada loop yang sama, dengan kata lain mengembalikan proses yang sedang dilaksanakan ke-awal loop lagi, tanpa menjalankan sisa perintah dalam loop tersebut.

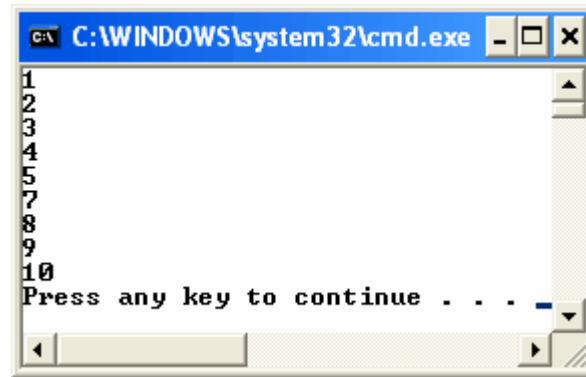
Sebagai contoh program menerapkan perintah perulangan for dengan menggunakan continue, selengkapnya seperti dibawah ini:

```

1  /* -----
2     Nama File : Lat512.java
3     Author   : Frieyadie
4  ----- */
5
6  class Lat512
7  {
8      public static void main(String[] args)
9      {
10         int bil;
11
12         for(bil=1; bil<=10; ++bil)
13         {
14             if(bil==6)
15                 continue;
16
17             System.out.println(bil);
18         }
19     }
20 }

```

Output yang dihasilkan dari program Lat512.java diatas, seperti dibawah ini :



Gambar 5.12 Hasil Program Lat512.java



Pada saat nilai `bil = 6`, maka proses seleksi `bil==6`, maka bernilai `True`, dan perintah `continue` dijalankan, selanjutnya proses menuju ke pernyataan `++bil`, pada perintah perulangan, dan proses pencetakan tidak dilakukan.

5.7 Latihan

1. Buatlah program untuk menghitung 10 deret bilangan genap dengan hasilnya :
 $2 + 4 + 6 + 8 + 10 + 12 + 14 + 16 + 18 + 20 = 110$
2. Buatlah program untuk menghitung 10 deret bilangan ganjil dengan hasilnya :
 $1 + 3 + 5 + 7 + 9 + 11 + 13 + 15 + 17 + 19 = 100$
3. Buatlah program untuk menampilkan deret fibonanci, seperti dibawah ini :
1, 1, 2, 3, 5, 8, 13, 21
4. Buatlah program untuk menampilkan bilangan prima, seperti dibawah ini :
2, 3, 5, 7, 11, 13, 17, 19

Bab 6:

Array Pada Pemrograman Java

6.1 Kopetensi Dasar

Pada pembahasan Bab 6 ini penulis mengajak mendiskusikan mengenai penggunaan array yang pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami menggunakan array pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Penggunaan pendeklarasian array berdimensi satu, memasukan nilai ke array dimensi satu dan mengambil nilai dari array dimensi satu.
- Penggunaan pendeklarasian array berdimensi dua, memasukan nilai ke array dimensi dua dan mengambil nilai dari array dimensi dua
- Penggunaan pendeklarasian array berdimensi tiga, memasukan nilai ke array dimensi tiga dan mengambil nilai dari array dimensi tiga.

6.2 Array Berdimensi Satu

Variabel Larik atau lebih dikenal dengan ARRAY adalah adalah Tipe terstruktur yang terdiri dari sejumlah komponen-komponen yang mempunyai tipe yang sama. Sebelum digunakan, variabel array perlu dideklarasikan terlebih dahulu. Cara mendeklarasikan variabel array sama seperti deklarasi variabel yang lainnya, hanya saja diikuti oleh suatu indek yang menunjukkan jumlah maksimum data yang disediakan.

Deklarasi Array Bentuk Umum pendeklarasian array :

```
tipe_data[] nama_var_array;  
nama_var_array = new tipe_data[ukuran];
```

Atau bisa juga secara langsung seperti dibawah ini :

```
tipe_data[] nama_var_array = new tipe_data[ukuran];
```

Keterangan :

Type Data : Untuk menyatakan type data yang digunakan.

Ukuran : Untuk menyatakan jumlah maksimum elemen array.

Berikut contoh pendeklarasian array dan membuat objek array;

```
int[] nil_akhir;  
nil_akhir = new int[6];
```

Contoh Pendeklarasian Array sekaligus membuat objek array.

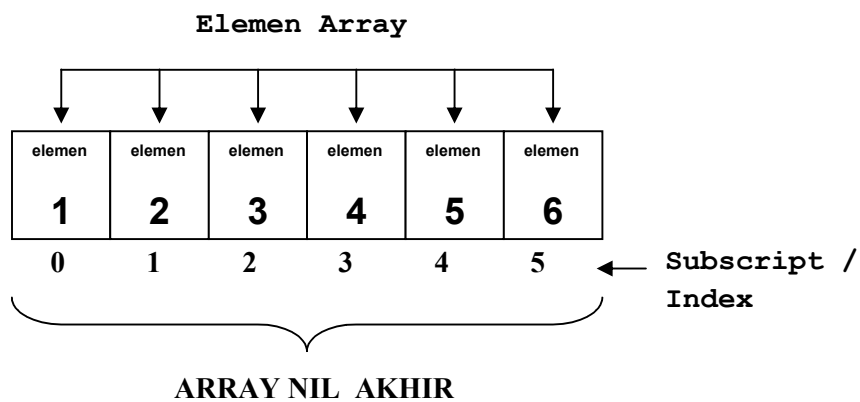
```
int[] nil_akhir = new int[6];
```

Diagram illustrating the components of the array declaration `int[] nil_akhir = new int[6];`:

- `int`: Tipe data elemen array
- `nil_akhir`: Nama Array
- `6`: Jumlah Elemen Array

Pada pendeklarasian diatas, variabel `nil_akhir` sebagai array-of-int, banyak elemen yang dapat ditampung sebanyak 6 (enam) elemen.

Suatu array dapat digambarkan sebagai kotak panjang yang berisi kotak-kotak kecil didalam kotak panjang tersebut.



Subscript atau Index array pada Java, selalu dimulai dari Nol (0)

6.2.1. Mengakses Array Berdimensi Satu

Suatu array, dapat diakses dengan menggunakan subscript atau indexnya. Bentuk umum pengaksesan dengan bentuk :

```
nama_var_array[nomor_index_array];
```

Contoh :

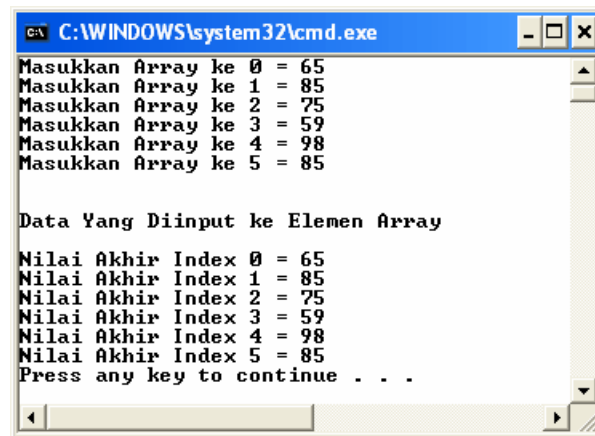
```
Nil_Akhir[3];
```

```
Nil_Akhir[1];
```

Sebagai contoh program memasukan data kedalam elemen array dan mengambil data dari dalam elemen array, secara selengkapnya seperti dibawah ini:

```
1  /* -----  
2      Nama File : Lat601.java  
3      Author   : Frieyadie  
4  ----- */  
5  
6  import java.util.*;  
7  class Lat601  
8  {  
9      public static void main(String[] args)  
10     {  
11         int i;  
12         int[] nil_akhir;      // deklarasi variabel array  
13         nil_akhir = new int[6]; // membuat objek array  
14  
15         Scanner input = new Scanner(System.in);  
16  
17         for(i=0; i<6; i++)  
18         {  
19             System.out.print("Masukkan Array ke " + i + " = ");  
20             nil_akhir[i] = input.nextInt();  
21         }  
22  
23         System.out.println("\n\nData Yang Diinput ke Elemen Array \n");  
24  
25         for(i=0; i<6; i++)  
26         {  
27             System.out.print("Nilai Akhir Index " + i );  
28             System.out.println(" = " + nil_akhir[i]);  
29         }  
30     }  
31 }
```

Output yang dihasilkan dari program Lat601.java diatas, seperti dibawah ini :



Gambar 6.1 Hasil Program Lat601.java

6.2.2. Inisialisasi Array Berdimensi Satu

Inisialisasi adalah memberikan nilai awal terhadap suatu variabel. Bentuk pendefinisian suatu array dapat dilihat dari contoh berikut :

```
tipe_data[] nama_array = { nilai array };
```

Contoh:

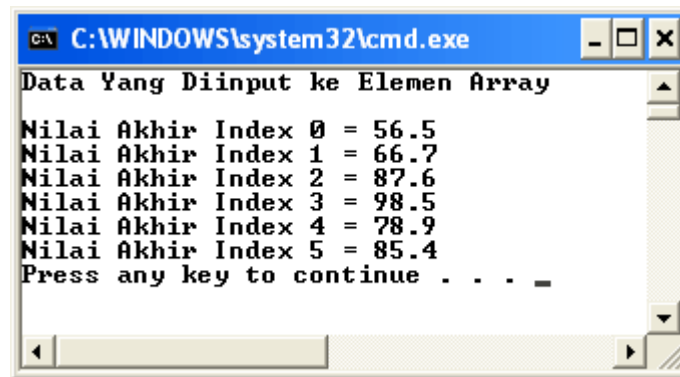
```
float[] nilai= {56.5, 66.7, 87.45, 98,5, 78.9 };
```

Sebagai contoh program memasukan data kedalam elemen array dengan cara inisialisasi objek array dan mengambil data dari dalam elemen array, secara selengkapnya seperti dibawah ini:

```

1  /* -----
2     Nama File : Lat602.java
3     Author    : Frieyadie
4  ----- */
5
6  class Lat602
7  {
8      public static void main(String[] args)
9      {
10         int i;
11         double[] nil_akhir = {56.5, 66.7, 87.6, 98,5, 78.9, 85.4};
12
13         System.out.println("\nData Yang Diinput ke Elemen Array \n");
14         //menampilkan data dari elemen array
15         for(i=0; i<6; i++)
16         {
17             System.out.print("Nilai Akhir Index " + i );
18             System.out.println(" = " + nil_akhir[i]);
19         }
20     }
21 }
```


Output yang dihasilkan dari program Lat602.java diatas, seperti dibawah ini :



Gambar 6.2 Hasil Program Lat602.java

6.3 Array Berdimensi Dua

Array dimensi dua tersusun dalam bentuk baris dan kolom, dimana indeks pertama menunjukkan baris dan indeks kedua menunjukkan kolom. Array dimensi dua dapat digunakan seperti pendataan penjualan, pendataan nilai dan lain sebagainya.

Bentuk Umum pendeklarasian array :

Deklarasi Array Bentuk Umum pendeklarasian array :

```
tipe_data[][] nama_var_array;  
nama_var_array = new tipe_data[ukuran_baris][ukuran_kolom];
```

Atau bisa juga secara langsung seperti dibawah ini :

```
tipe_data[][] nama_var_array = new tipe_data[ukuran_baris][ukuran_kolom];
```

Keterangan :

Type Data : Untuk menyatakan type data yang digunakan.

Ukuran_baris : Untuk menyatakan jumlah maksimum elemen baris dalam array.

Ukuran_kolom : Untuk menyatakan jumlah maksimum elemen kolom dalam array.

Berikut contoh pendeklarasian array dan membuat objek array;

```
int[][] nil_akhir;  
nil_akhir = new int[6][3];
```

Contoh Pendeklarasian Array sekaligus membuat objek array.

```
int[][] nil_akhir = new int[6][3];
```

Diagram illustrating the components of the array declaration `int[][] nil_akhir = new int[6][3];`:

- `int`: Tipe data elemen array
- `nil_akhir`: Nama Array
- `6`: Jumlah Elemen Baris
- `3`: Jumlah Elemen Array

6.3.1 Mengakses Array Berdimensi Dua

Sebagai contoh pendeklarasian dan pengaksesan data, kita gunakan adalah pengolahan data penjualan, berikut dapat anda lihat pada tabel berikut :

Tabel 6.1. Tabel Data Penjualan Pertahun

No	Tahun Penjualan		
	2001	2002	2003
1	150	159	230
2	100	125	150
3	210	125	156

Jika anda lihat dari tabel 6.1 diatas maka dapat dituliskan kedalam array dimensi dua berikut :

```
int[][] data_jual = new int[3][3];
```

Diagram illustrating the components of the array declaration `int[][] data_jual = new int[3][3];`:

- `int`: Tipe data elemen array
- `data_jual`: Nama variabel Array
- `3`: Jumlah Baris
- `3`: Jumlah Kolom

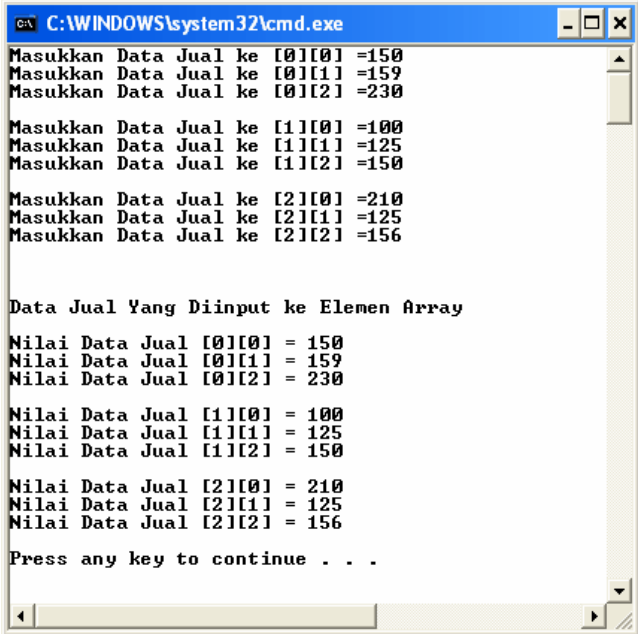
Sebagai contoh program memasukan data kedalam elemen array berdimensi dua dan mengambil data dari dalam elemen array berdimensi dua, secara selengkapnya seperti dibawah ini:

```

1  /* -----
2     Nama File : Lat603.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat603
8  {
9      public static void main(String[] args)
10     {
11         int i, j;
12         int[][] data_jual;
```

```
13 data_jual = new int[3][3];
14
15 Scanner input = new Scanner(System.in);
16
17 for(i=0; i<3; i++)
18 {
19     for(j=0; j<3; j++)
20     {
21         System.out.print("Masukkan Data Jual ke ["+i+"] ["+j+"] =");
22         data_jual[i][j] = input.nextInt();
23     }
24     System.out.println();
25 }
26
27 System.out.println("\n\nData Jual Yang Diinput ke Elemen Array \n");
28
29 for(i=0; i<3; i++)
30 {
31     for(j=0; j<3; j++)
32     {
33         System.out.print("Nilai Data Jual ["+i+"] ["+j+"]");
34         System.out.println(" = " + data_jual[i][j]);
35     }
36     System.out.println();
37 }
38 }
```

Output yang dihasilkan dari program Lat603.java diatas, seperti dibawah ini :



```
C:\WINDOWS\system32\cmd.exe
Masukkan Data Jual ke [0][0] =150
Masukkan Data Jual ke [0][1] =159
Masukkan Data Jual ke [0][2] =230

Masukkan Data Jual ke [1][0] =100
Masukkan Data Jual ke [1][1] =125
Masukkan Data Jual ke [1][2] =150

Masukkan Data Jual ke [2][0] =210
Masukkan Data Jual ke [2][1] =125
Masukkan Data Jual ke [2][2] =156

Data Jual Yang Diinput ke Elemen Array

Nilai Data Jual [0][0] = 150
Nilai Data Jual [0][1] = 159
Nilai Data Jual [0][2] = 230

Nilai Data Jual [1][0] = 100
Nilai Data Jual [1][1] = 125
Nilai Data Jual [1][2] = 150

Nilai Data Jual [2][0] = 210
Nilai Data Jual [2][1] = 125
Nilai Data Jual [2][2] = 156

Press any key to continue . . .
```

Gambar 6.3 Hasil Program Lat603.java

6.3.2 Inisialisasi Array Berdimensi Dua

Suatu array, dapat diakses dengan menggunakan subscript atau indexnya. Bentuk umum pengaksesan dengan bentuk :

```
tipe_data[][] nama_array = { nilai array };
```

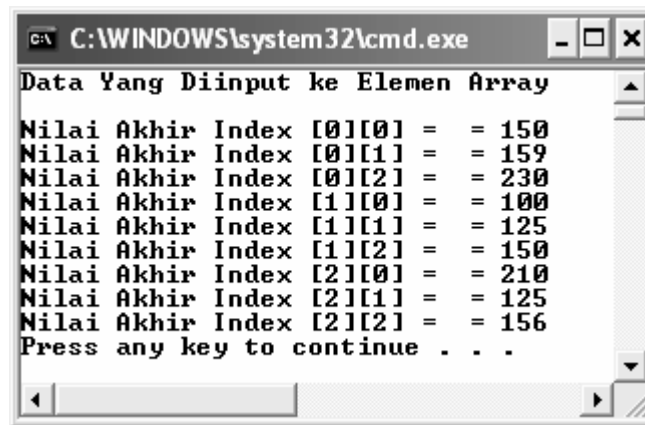
Contoh:

```
int[][] nil_akhir = {{150, 159, 230}, {100,125,150}, {210,125,156}};
```

Sebagai contoh program memasukan data kedalam elemen array dengan cara inisialisasi objek array dan mengambil data dari dalam elemen array, secara selengkapnya seperti dibawah ini:

```
1  /* -----
2     Nama File : Lat604.java
3     Author   : Frieyadie
4     ----- */
5
6  class Lat604
7  {
8      public static void main(String[] args)
9      {
10         int i, j;
11         int[][] nil_akhir = {{150, 159, 230}, {100,125,150}, {210,125,156}};
12
13         System.out.println("Data Yang Diinput ke Elemen Array \n");
14         //menampilkan data dari elemen array
15         for(i=0; i<3; i++)
16         {
17             for(j=0; j<3; j++)
18             {
19                 System.out.print("Nilai Akhir Index ["+i+"] ["+j+"] = ");
20                 System.out.println(" = " + nil_akhir[i][j]);
21             }
22         }
23     }
24 }
```

Output yang dihasilkan dari program Lat604.java diatas, seperti dibawah ini :



```
C:\WINDOWS\system32\cmd.exe
Data Yang Diinput ke Elemen Array
Nilai Akhir Index [0][0] = = 150
Nilai Akhir Index [0][1] = = 159
Nilai Akhir Index [0][2] = = 230
Nilai Akhir Index [1][0] = = 100
Nilai Akhir Index [1][1] = = 125
Nilai Akhir Index [1][2] = = 150
Nilai Akhir Index [2][0] = = 210
Nilai Akhir Index [2][1] = = 125
Nilai Akhir Index [2][2] = = 156
Press any key to continue . . .
```

Gambar 6.4 Hasil Program Lat604.java

6.4 Array Berdimensi Tiga

Array dimensi dua tersusun dalam bentuk baris, kolom dan isi dari baris, dimana indeks pertama menunjukan baris, indeks kedua menunjukan kolom dan indeks ketiga menunjukan isi dari baris.

Bentuk Umum pendeklarasian array :

```
tipe-data[][][] Nama_Variabel = new tipe-data[index-1][index-2][index-3];
```

Keterangan :

Type Data : Untuk menyatakan type data yang digunakan.

Index-1 : Untuk menyatakan jumlah baris

Index-2 : Untuk menyatakan jumlah isi dari baris

Index-3 : Untuk menyatakan jumlah kolom

Sebagai contoh pendeklarasian yang akan kita gunakan adalah pengolahan data penjualan, berikut dapat anda lihat pada tabel berikut :

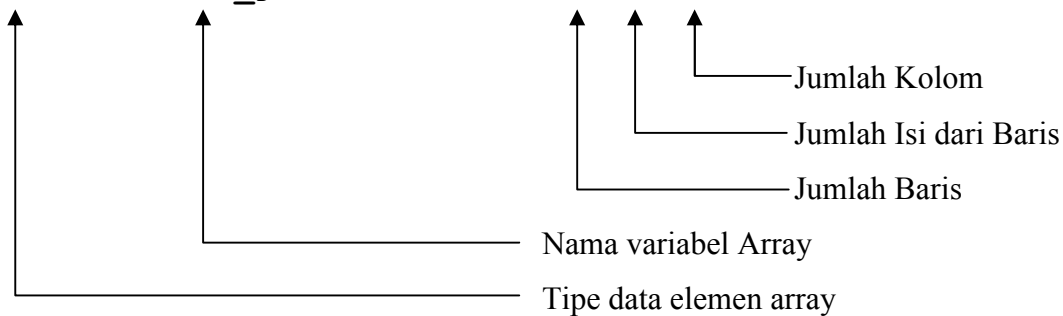
Tabel 6.2. Tabel Data Penjualan Pertahun

	Tahun Ke	Hasil Ke	Tahun Penjualan Ke.	
			1	2
Indeks ke.1 {	1	1	150	159
		2	200	400
{	2	1	100	125
		2	210	125

Indeks ke.3

Jika anda lihat dari tabel 6.2 diatas maka dapat dituliskan kedalam array dimensi dua berikut :

```
int[][][] data_jual = new int[3][3][3];
```



6.4.1 Mengakses Array Berdimensi Tiga

Suatu array, dapat diakses dengan menggunakan subscript atau indexnya. Bentuk umum pengaksesan dengan bentuk :

```
Nama_Array[index-1][index-2][index-3]
```

Contoh: `data_jualan[1][1][1];`

`data_jualan[1][0][1];`

Sebagai contoh program memasukan data kedalam elemen array tiga dimensi dan mengambil data dari dalam elemen array, secara selengkapnya seperti dibawah ini:

```

1  /* -----
2     Nama File : Lat605.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7  class Lat605

```

```

8  {
9      public static void main(String[] args)
10     {
11         Scanner input = new Scanner(System.in);
12         int i, j, k;
13         int[][][] data_jual;
14         data_jual = new int[2][3][2];
15
16         for(i=0;i<2;i++)
17         {
18             for(j=0;j<3;j++)
19             {
20                 for(k=0;k<2;k++)
21                 {
22                     System.out.println("Data Tahun Ke - " + (i+1) );
23                     System.out.println("Data Ke - "+ (j+1) + " " + (k+1));
24                     System.out.print("Jumlah Penjualan    : ");
25                     data_jual[i][j][k] = input.nextInt();
26                 }
27                 System.out.println();
28             }
29             System.out.println();
30         }
31
32         System.out.println();
33         System.out.println("Data Penjualan Pertahun");
34         System.out.println("-----");
35         System.out.println();
36         System.out.println("Tahun    Hasil    Tahun Penjualan Ke. ");
37         System.out.println();
38         System.out.println("Ke.      Ke.      -----");
39         System.out.println();
40         System.out.println("                1        2        ");
41         System.out.println();
42         System.out.println("-----");
43         System.out.println();
44
45         for(i=0;i<2;i++)
46         {
47             for(j=0;j<3;j++)
48             {
49                 System.out.print((i+1) + "\t");
50                 System.out.print((j+1) + "\t\t");
51                 for(k=0;k<2;k++)
52                 {
53                     System.out.print(data_jual[i][j][k]);
54                     System.out.print("\t");
55                 }
56                 System.out.println();
57             }
58             System.out.println();
59         }
60
61         System.out.println("-----");
62         System.out.println();
63     }
64 }

```

Output yang dihasilkan dari program Lat605.java diatas, seperti dibawah ini :

```

C:\WINDOWS\system32\cmd.exe
Data Tahun Ke - 1
Data Ke - 1 1
Jumlah Penjualan : 100
Data Tahun Ke - 1
Data Ke - 1 2
Jumlah Penjualan : 200

Data Tahun Ke - 1
Data Ke - 2 1
Jumlah Penjualan : 300
Data Tahun Ke - 1
Data Ke - 2 2
Jumlah Penjualan : 400

Data Tahun Ke - 1
Data Ke - 3 1
Jumlah Penjualan : 500
Data Tahun Ke - 1
Data Ke - 3 2
Jumlah Penjualan : 600

Data Tahun Ke - 2
Data Ke - 1 1
Jumlah Penjualan : 700
Data Tahun Ke - 2
Data Ke - 1 2
Jumlah Penjualan : 800

Data Tahun Ke - 2
Data Ke - 2 1
Jumlah Penjualan : 900
Data Tahun Ke - 2
Data Ke - 2 2
Jumlah Penjualan : 950

Data Tahun Ke - 2
Data Ke - 3 1
Jumlah Penjualan : 560
Data Tahun Ke - 2
Data Ke - 3 2
Jumlah Penjualan : 850

```

Gambar 6.5 Hasil Program Lat605.java

Output yang akan dihasilkan, dari Input Data ke Array dari program Lat605.java, diatas adalah :

```

C:\WINDOWS\system32\cmd.exe
Data Penjualan Pertahun
-----
Tahun    Hasil    Tahun Penjualan Ke.
Ke.      Ke.      -----
                1      2
-----
1        1        100    200
1        2        300    400
1        3        500    600

2        1        700    800
2        2        900    950
2        3        560    850

-----
Press any key to continue . . .

```

Gambar 6.6 Hasil Program Lat605.java

6.4.2 Inisialisasi Array Berdimensi Tiga

Suatu array, dapat diakses dengan menggunakan subscript atau indexnya. Bentuk umum pengaksesan dengan bentuk :

```
tipe_data[][][] nama_array = { nilai array
```

Contoh:

```
int[][][] data_jual = { {100, 200, 300},
                        {150, 240, 360},
                        {250, 340, 460},
                        {250, 340, 460}},
                      { {160, 250, 365},
                        {175, 275, 375},
                        {275, 375, 575},
                        {380, 480, 580}}
};
```

Sebagai contoh program memasukan data kedalam elemen array dengan cara inisialisasi objek array dan mengambil data dari dalam elemen array, secara selengkapnya seperti dibawah ini:

```
1  /* -----
2     Nama File : Lat604.java
3     Author    : Frieyadie
4     ----- */
5
6  class Lat606
7  {
8      public static void main(String[] args)
9      {
10         int i, j, k;
11         int[][][] data_jual = {
12             {100, 200, 300},
13             {150, 240, 360},
14             {250, 340, 460},
15             {250, 340, 460}},
16         { {160, 250, 365},
17           {175, 275, 375},
18           {275, 375, 575},
19           {380, 480, 580}}
20         };
21
22         System.out.println();
23         System.out.println("Data Penjualan Pertahun");
24         System.out.println("-----");
25         System.out.println();
26         System.out.println("Tahun    Hasil    Tahun Penjualan Ke. ");
27         System.out.println();
28         System.out.println("Ke.      Ke.      -----");
29         System.out.println();
```

```

30      System.out.println("                                1      2      ");
31      System.out.println();
32      System.out.println("-----");
33      System.out.println();
34
35      for(i=0;i<2;i++)
36      {
37          for(j=0;j<3;j++)
38          {
39              System.out.print((i+1) + "\t");
40              System.out.print((j+1) + "\t\t");
41              for(k=0;k<2;k++)
42              {
43                  System.out.print(data_jual[i][j][k]);
44                  System.out.print("\t");
45              }
46              System.out.println();
47          }
48          System.out.println();
49      }
50
51      System.out.println("-----");
52      System.out.println();
53  }
54  }

```

Output yang dihasilkan dari program Lat606.java diatas, seperti dibawah ini :

```

C:\WINDOWS\system32\cmd.exe

Data Penjualan Pertahun
-----
Tahun   Hasil   Tahun Penjualan Ke.
Ke.     Ke.      -----
                                1      2
-----
1       1       100    200
1       2       150    240
1       3       250    340
2       1       160    250
2       2       175    275
2       3       275    375
-----
Press any key to continue . . .

```

Gambar 6.4 Hasil Program Lat604.java

Inisialisasi adalah memberikan nilai awal terhadap suatu variabel. Bentuk pendefinisian suatu array dapat dilihat dari contoh berikut :

6.5 Latihan

1. Buatlah sebuah class dengan nama AngkaTerbesar, yang digunakan untuk melakukan pencarian sebuah nilai terbesar diantara nilai-nilai yang dimasukan.
2. Sebuah perusahaan ayam goreng dengan nama “GEROBAK FRIED CHICKEN” yang telah lumayan banyak pelanggannya, ingin dibantu dibuatkan program untuk membantu kelancaran usahanya. “GEROBAK FRIED CHICKEN” mempunyai daftar harga ayam sebagai berikut:

Kode	Jenis	Harga
D	Dada	Rp. 2500
P	Paha	Rp. 2000
S	Sayap	Rp. 1500

Buatlah programnya dengan ketentuan :

- a. Setiap pembeli dikenakan pajak sebesar 10% dari pembayaran.
- b. Banyak Jenis, Jenis Potong dan Banyak Beli diinput.
- c. Tampilan yang diinginkan sebagai berikut :

Layar Masukkan

GEROBAK FRIED CHICKEN		
Kode	Jenis	Harga
D	Dada	Rp. 2500
P	Paha	Rp. 2000
S	Sayap	Rp. 1500

Banyak Jenis : ... <diinput>

Jenis Ke - ... <proses counter>

Jenis Potong [D/P/S] : ... <diinput>

Banyak Potong : ... <diinput>

<<Terus berulang tergantung Banyak Jenis>>

Layar Keluaran

GEROBAK FIRED CHICHEN				
No.	Jenis Potong	Harga Satuan	Bayak Beli	Jumlah Harga
...	Rp
...	Rp
Jumlah Bayar				Rp
Pajak 10%				Rp
Total Bayar				Rp

3. Buatlah program untuk menghitung nilai akhir seorang siswa dari kursus yang diikutinya. Dengan ketentuan sebagai berikut :
 - a. Nama Mahasiswa, Nilai Tugas, Nilai UTS dan Nilai UAS diinput.
 - b. Proses yang dilakukan untuk mendapatkan nilai murni dari masing-masing nilai, adalah
 - Nilai Murni Tugas = Nilai Tugas dikalikan dengan 30%
 - Nilai Murni UTS = Nilai UTS dikalikan dengan 30%
 - Nilai Murni UAS = Nilai UAS dikalikan dengan 40%
 - Nilai Akhir adalah Nilai Murni Tugas + Nilai Murni UTS + Nilai Murni UAS
 - c. Ketentuan untuk mendapatkan grade nilai :
 - Nilai Akhir ≥ 80 mendapat Grade A
 - Nilai Akhir ≥ 70 mendapat Grade B
 - Nilai Akhir ≥ 59 mendapat Grade C
 - Nilai Akhir ≥ 50 mendapat Grade D
 - Nilai Akhir < 50 mendapat Grade E
 - d. Tampilan yang diinginkan sebagai berikut :

Layar Masukkan

PROGRAM HITUNG NILAI AKHIR
MATERI PEMROGRAMMAN C++

Masukkan Jumlah Mahasiswa : ... <diinput>

```

Mahasiswa Ke - ... <proses counter>
Nama Mahasiswa : ..... <diinput>
Nilai Tugas    : ..... <diinput>
Nilai UTS      : ..... <diinput>
Nilai UAS      : ..... <diinput>

```

```
<<Terus berulang tergantung Jumlah Mahasiswa>>
```

Layar Keluaran

```

DAFTAR NILAI
MATERI : PEMROGRAMMAN C++

```

```

-----
No.   Nama                               Nilai                               Grade
      Mahasiswa
      Tugas    UTS    UAS    Akhir
-----
...   .....   ....   ....   ....   ....
...   .....   ....   ....   ....   ....
-----

```

4. PT. EASY, memberikan Honor tetap kepada karyawan kontraknya sebesar Rp. 700,000,- per bulan, dengan memperoleh tunjangan-tunjangan sebagai berikut :

a. Tunjangan Jabatan.

Golongan	Persentase
1	5%
2	10%
3	15%

Sebagai contoh : Jika seorang karyawan tersebut dengan golongan 3, maka mendapatkan tunjangan sebesar 15% * Rp. 700,000,-

b. Tunjangan Pendidikan

Kode	Pendidikan	Persentase
1	SMU	2,5%
2	D3	5%
3	S1	7,5%

c. Honor Lembur

Jumlah jam kerja normal dalam satu bulan sebanyak 240 Jam Kerja. Honor lembur diberikan jika jumlah jam kerja lebih dari 240 jam, maka kelebihan jam kerja tersebut dikalikan dengan honor lembur perjam sebesar Rp. 2,500 untuk setiap kelebihan jam kerja dalam satu bulannya.

d. Tampilan yang diinginkan sebagai berikut :

Layar Masukan

Program Hitung Honor Karyawan Kontrak
PT. EASY

Masukkan Jumlah Karyawan : ... <diinput>

Karyawan Ke - ... <proses counter>

Nama Karyawan : ... <di input>

Golongan (1/2/3) : ... <di input>

Pendidikan (1=SMU/2=D3/3=S1) : ... <di input>

Jumlah Jam Kerja : ... <di input>

<<Terus berulang tergantung Jumlah Karyawan>>

Layar Keluaran

PT. EASY

No.	Nama Karyawan	Tunjangan	Honor	Gaji
		Jabatan	Pendidikan	Lembur
				Pajak Bersih
...
...
Total Gaji yang dikeluarkan Rp.				

Bab 7:

Class dan Object

7.1 Kopetensi Dasar

Pada pembahasan Bab7 ini penulis mengajak mendiskusikan mengenai penggunaan Class dan Object yang ada pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami menggunakan Class dan Object pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- a. Membuat pendeklarasian Class dan Penggunaan Class.
- b. Membuat Object dari Class

7.2 Class

Disetiap pemrograman Java, wajib memiliki minimal satu buah class agar program tersebut bisa berjalan. Untuk mendefinisikan Class pada Pemrograman Java dengan diawali dengan kata class, dan diikuti dengan nama class.

Berikut bentuk umum pendeklarasian class pada pemrograman Java.

```
[public | private | protected] Class Nama_Class
{
    ... daftar property...
    ... daftar Method ...
}
```

Terdapat class modifier pada pemrograman java, yaitu :

- a. Public : Pengaksesan suatu variabel instan atau *method*, bisa diakses dari luar class secara langsung.

- b. Private : Pengaksesan suatu variabel instan atau *method*, tidak bisa diakses dari luar class secara langsung.
- c. Protected : Tingkat pengaksesan antara public dan private. Pengaksesan dapat dilakukan oleh anggota package class dan subclass – subclass yang lainnya didalam package.

Pada penggunaan class modifier, dapat digunakan tergantung kebutuhan keamanan aplikasi tersebut. Berikut contoh pembuatan suatu class :

```
public class burung
{
    String jenis, warna;
    int usia;
}
```

7.3 Object

Dalam Pemrograman Berorientasi Objek melihat atau memandang sesuatu berdasarkan objek. Objek sebenarnya mencerminkan pola kerja manusia dalam kehidupan sehari-hari.

Pada suatu objek dapat dilihat menjadi 2 (dua) hal, yaitu :

1. Atributte

Atribut merupakan segala sesuatu yang melekat pada *Object*. Didalam penerapan didalam program, atribut adalah Variabel atau Member.

Misalkan pada *Object* Burung. Atribut-atribut yang melekat pada burung, misalnya paruh, ekor, sayap, kaki, mata, dan lain-lain.

2. Behaviour

Behaviour merupakan pola tingkah laku atau perilaku yang dimiliki oleh objek. Misalnya pada objek Burung memiliki perilaku diantaranya terbang, mengepakkan sayap, berjalan dan lain-lain. Didalam penerapan didalam program, Behaviour adalah *Method* atau Fungsi.

7.3.1 Membuat Object

Bentuk penulisan mendeklarasikan *Object*, dengan menggunakan **new**, seperti dibawah ini :

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

- nama_class, merupakan nama *Class* yang akan dijadikan objek.
- nama_objek, merupakan nama objek baru.

Sebagai contoh, kita membuat sebuah class, seperti dibawah ini :

```
public class burung
{
    String jenis, warna;
    int usia;
}
```

Bentuk penulisan mendeklarasikan *Object*, dengan menggunakan **new**, seperti dibawah ini :

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

Maka untuk membuat objeknya, dengan cara seperti contoh berikut :

```
burung burung_elang = new burung();
```

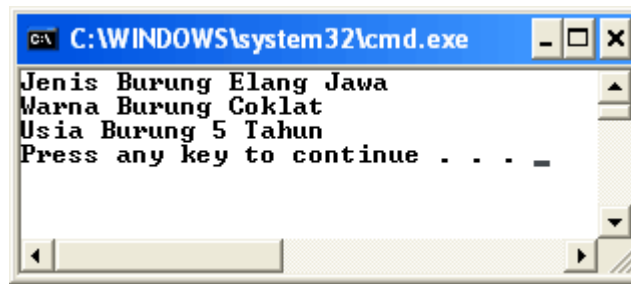
burung_elang sebagai nama objek yang berdasarkan dari class burung.

7.3.2 Menerapkan Class dengan Object

Sebagai contoh program membuat class sederhana dan membentuk objek, secara selengkapanya seperti dibawah ini:

```
1  /* -----
2     Nama File : lat701.java
3     Author    : Frieyadie
4  ----- */
5
6  class burung
7  {
8      String jenis, warna;
9      int usia;
10 }
11
12 class Lat701
13 {
14     public static void main(String[] args)
15     {
16         //membuat objek
17         burung burung_elang = new burung();
18
19         burung_elang.jenis = "Elang Jawa";
20         burung_elang.warna = "Coklat";
21         burung_elang.usia = 5;
22
23         System.out.println("Jenis Burung : " + burung_elang.jenis);
24         System.out.println("Warna Burung : " + burung_elang.warna);
25         System.out.println("Usia Burung : " + burung_elang.usia + " Tahun");
26     }
27 }
```

Output yang dihasilkan dari program burung_terbang.java diatas, seperti dibawah ini :



Gambar 7.1 Hasil Program Lat702.java

Berikut penjelasan program Lat701.java, diatas :

- a. Pada baris 6 sampai 10, mendeklarasikan pembuatan class dengan nama burung, didalamnya memiliki anggota class. Pada class burung ini tidak digunakan *access modifier*, yang berarti dapat diakses oleh class lain.

- b. Pada baris 17, terdapat pernyataan
`burung burung_elang = new burung();`

digunakan untuk membuat objek dengan nama `burung_elang`, yang berdasarkan class `burung`.

- c. Pada baris 23 sampai 25, terdapat pernyataan

```
burung_elang.jenis = "Elang Jawa";
burung_elang.warna = "Coklat";
burung_elang.usia = 5
```

Pernyataan tersebut digunakan untuk memberikan nilai kepada variabel instan class `burung`.

- d. Pada baris 19 sampai 21, terdapat pernyataan

```
burung_elang.jenis = "Elang Jawa";
burung_elang.warna = "Coklat";
burung_elang.usia = 5
System.out.println("Jenis Burung : " + burung_elang.jenis);
System.out.println("Warna Burung : " + burung_elang.warna);
System.out.println("Usia Burung      : " + burung_elang.usia + "
Tahun");
```

Pernyataan tersebut digunakan untuk menampilkan nilai variabel instan kelayar.

7.4 Method

Method adalah implementasi operasi yang bisa dilakukan oleh *Class* dan *Object*. Operasi-operasi yang dilakukan oleh *Method*, diantaranya, yaitu :

1. Suatu *Method* bisa menerima dan memanipulasi data atau field didalam diri *Method* tersebut.
2. Suatu *Method* bisa mempengaruhi nilai suatu *Object* lain.

7.4.1 Membuat *Method*

Untuk membuat atau menciptakan *method* terdapat 4(empat) bagian yang mendasar, yaitu :

1. Nama *method*
2. Daftar Parameter – paramater
3. Tipe objek atau tipe primitif (tipe data) yang dikembalikan oleh *method*
4. Badan program *method*

Berikut bentuk penulisan deklarasi *Method*:

```
Tipe_Akses Tipe_Return NamaMethod(Parmater1,...,Argumen-N)
{
    ... Badan / Tubuh Method ..
}
```

Berikut penjelasan deklarasi *Method* diatas :

1. Tipe Akses, menyatakan tingkatan akses untuk memproteksi akses terhadap data-data didalam *Method*, tipe akses ini bersifat opsional.
2. Tipe Return, menyatakan nilai hasil yang diolah oleh *Method* akan dikembalikan atau akan mengirimkan kepada objek yang memanggil *Method*. Bentuk Tipe Return, bisa berupa tipe data primitive yaitu integer, float, double dan lain-lain.

Apabila *Method* tidak akan mengembalikan nilai kepada objek yang memanggilnya, maka bisa dituliskan didepan nama *Method* dengan perintah void.

7.4.2 Menerapkan *Method* pada Class

Berikut contoh program membuat class dengan penggunaan *method*, secara selengkapny seperti dibawah ini:

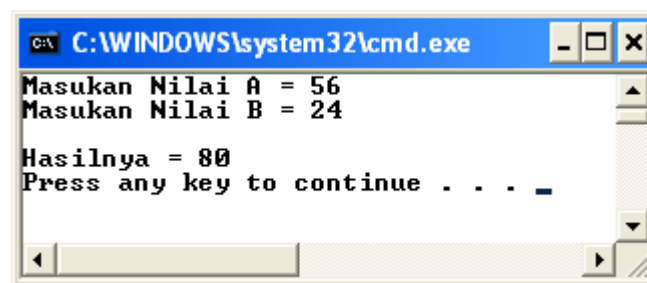
```
1  /* -----
2     Nama File : Lat702.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.*;
7
8  class Hitung
9  {
10     int tambah;
11     int setHitung(int nil1, int nil2)
12     {
```

```

13     tambah = nil1 + nil2 ;
14     return tambah;
15 }
16 }
17
18 class Lat702
19 {
20     public static void main(String[] args)
21     {
22         int a, b, hasil;
23
24         Hitung ngitung = new Hitung();
25         Scanner input = new Scanner(System.in);
26
27         System.out.print("Masukan Nilai A = ");
28         a = input.nextInt();
29         System.out.print("Masukan Nilai B = ");
30         b = input.nextInt();
31
32         hasil = ngitung.setHitung(a, b);
33
34         System.out.println("Hasilnya " + hasil);
35     }
36 }

```

Output yang dihasilkan dari program Lat702.java diatas, seperti dibawah ini :



Gambar 7.2 Hasil Program Lat702.java

Berikut penjelasan program Lat702.java, diatas :

- a. Pada baris 8 sampai 16, mendeklarasikan pembuatan class dengan nama Hitung, didalamnya memiliki sebuah *method* dengan nama setHitung untuk menampung nilai-nilai kiriman dari class 702 yang bertipe integer, sekaligus melakukan proses perhitungan.

- b. Pada baris 24, terdapat pernyataan
`Hitung ngitung = new Hitung();`

digunakan untuk membuat objek dengan nama ngitung, yang berdasarkan class Hitung.

- c. Pada baris 27 sampai 30, terdapat pernyataan
`burung_elang.jenis = "Elang Jawa";`

```

burung_elang.warna = "Coklat";
burung_elang.usia = 5

System.out.print("Masukan Nilai A = ");
    a = input.nextInt();
System.out.print("Masukan Nilai B = ");
    b = input.nextInt();

```

Pernyataan tersebut digunakan untuk memberikan menginput nilai kepada variabel a dan b.

- d. Pada baris 32, terdapat pernyataan

```
hasil = ngitung.setHitung(a, b);
```

Pernyataan tersebut digunakan untuk memanggil *method* setHitung serta mengirimkan nilai variabel a dan b, dan hasil return dari *method* setHitung akan ditampung pada variabel hasil.

7.5 this

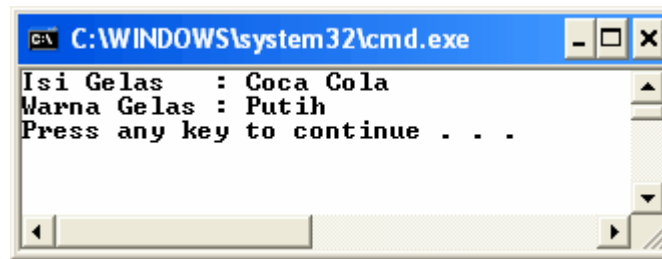
this adalah sebuah *keyword* yang digunakan untuk menunjukan class instan saat ini, *this* dapat dipergunakan pada variabel atau *method*.. Berikut contoh penggunaan kata kunci *this*, selengkapnya seperti program dibawah ini :

```

1  /* -----
2     Nama File : Lat703.java
3     Author    : Frieyadie
4     ----- */
5
6  class Gelas
7  {
8      String isigelas, warnagelas;
9
10     void setGelas(String isi, String warna)
11     {
12         this.isigelas = isi;
13         this.warnagelas = warna;
14     }
15 }
16
17 class Lat703
18 {
19     public static void main(String[] args)
20     {
21         Gelas Mug = new Gelas();
22
23         Mug.setGelas("Coca Cola", "Putih");
24
25         System.out.println("Isi Gelas    : " + Mug.isigelas);
26         System.out.println("Warna Gelas : " + Mug.warnagelas);
27     }
28 }

```

Output yang dihasilkan dari program Lat703.java diatas, seperti dibawah ini :



Gambar 7.3 Hasil Program Lat703.java

Penjelasan Program :

- Pada baris 12 dan 13, terdapat perintah :

```
this.isigelas = isi;  
this.warnagelas = warna;
```

Digunakan untuk menyatakan variabel instan saat ini yaitu isigelas dan warnagelas

- Pada baris 25 dan 26, terdapat perintah :

```
System.out.println("Isi Gelas : " + Mug.isigelas);  
System.out.println("Warna Gelas : " + Mug.warnagelas);
```

Digunakan untuk menampilkan nilai yang ada didalam variabel instan tersebut.

7.6 Latihan

1. Buatlah program menghitung luas dan keliling lingkaran dengan menggunakan class dan method. Method yang harus dibuat, yaitu : luas() untuk menghitung luas lingkaran dan keliling() untuk menghitung keliling lingkaran.
2. Buatlah program untuk menghitung besarnya diskon yang diberikan atas besarnya sejumlah pembelian, dengan ketentuan sebagai berikut :
 - a. Jika belanja dibawah Rp. 1,000,000 , maka tidak mendapat diskon.
 - b. Jika belanja dimulai dari Rp. 1,000,000 , sampai dengan Rp. 5.000.000, maka mendapat diskon sebesar 20%.
 - c. Jika belanja diatas Rp. 5.000.000, maka mendapat diskon sebesar 35%.

Method yang harus dibuat, yaitu `potong()` untuk menghitung besar potongan yang akan diberikan. Dengan tampilan yang diinginkan sebagai berikut :

Layar Masukkan dan Keluaran

Besar pembelian barang Rp. <di input >

Besar diskon yang diberikan Rp. ...< hasil proses >

Besar harga yang harus dibayarkan Rp. ...< hasil proses >

3. Buatlah program untuk menghitung konversi dari derajat fahrenheit ke celcius. Buatlah class dan method baru untuk mengolah data konversi.

Rumus konversi yang digunakan adalah

$$c = (f - 32.0) * 5 / 9;$$

Contoh :

Jika nilai Fahrenheit = 100

$$c = (100 - 32) * 5 / 9;$$

$$c = (68) * 5 / 9;$$

$$c = 37,7778$$

4. Buatlah program untuk menghitung jumlah pembayaran pada perpustakaan "Kecil-Kecilan".

Mempunyai ketentuan sebagai berikut:

Kode Jenis Buku	Jenis Buku	Tarif Buku
C	CerPen (Kumpulan Cerita Pendek)	500
K	Komik	700
N	Novel	1000

Petunjuk Proses :

- Buatlah Method Tarif untuk menentukan tarif penyewaan
- Gunakan Pernyataan If – Else

Tampilan Masukan yang diinginkan :

Perpustakaan ".Kecil-Kecilan".

Nama Penyewa Buku : <diinput>

Kode Buku [C/K/N] : <diinput>

Banyak Pinjam : <diinput>

Tampilan Keluaran yang diinginkan :

Tarif Sewa Rp. <hasil proses>
 Jenis Buku : < hasil proses >

Penyewa dengan Nama <hasil proses>
 Jumlah Bayar Penyewaan Sebesar Rp. <hasil proses>

- 5 **Buatlah program untuk menghitung proses pada perpustakaan rakyat pedesaan, menyewakan 3 golongan buku, yaitu A, B dan C. Harga sewa buku per 7 hari adalah:**

Golongan Harga Sewa per 7 hari

A Rp. 200

B Rp. 250

C Rp. 150

Jika meminjam lebih dari 7 hari, maka setiap harinya didenda sebesar Rp. 100

Buatlah Program untuk menghitung pembayarannya.

Buatlah Method untuk Menghitung Harga Sewa

Buatlah Method untuk Menghitung Denda

Buatlah Method untuk Menghitung Total Bayar

Bentuk Rancangan Masukan

Perpustakaan Rakyat Pedesaan

Nama Peminjam : _____
 Golongan Buku [A/B/C] : _____
 Lama Peminjaman : _____

Bentuk Rancangan Keluaran

Perpustakaan Rakyat Pedesaan

Pembayaran Peminjaman Buku

Nama Peminjam :<hasil proses>
 Harga Sewa Buku :<hasil proses>
 Lama Peminjaman : Hari <hasil proses>
 Jumlah Bayar :<hasil proses>
 Besar Denda :<hasil proses>

Jumlah yang Harus dibayar Rp.<hasil proses>

Bab 8:

Constructor, Inheritance dan Polymorphism

8.1 Kopetensi Dasar

Pada pembahasan Bab 8 ini penulis mengajak mendiskusikan mengenai penggunaan Constructor pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami menggunakan Constructor dan Inheritance pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- a. Membuat pendeklarasian dan menggunakan *Constructor*
- b. Melakukan *overloading* terhadap *constructor* dan *method*.
- c. Melakukan pewarisan Class
- d. Penggunaan Pewarisan Class seperti Menggunakan Superclass, Constructor Superclass dan lain sebagainya
- e. Melakukan Pendeklarasian Polymorphism dan Penggunaan Polymorphism

8.2 Constructor

Constructor merupakan suatu method yang akan memberikan nilai awal pada saat suatu objek dibuat. Pada saat program dijalankan, constructor akan langsung memberikan nilai awal pada saat perintah *new*, membuat suatu objek.

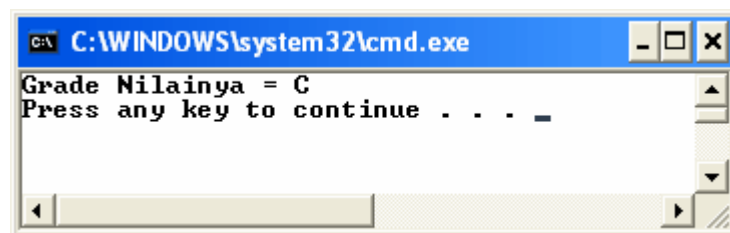
Pada saat kita bekerja dengan constructor, hal mendasar yang perlu diperhatikan, yaitu :

- a. Nama Constructor sama dengan nama Class.
- b. Tidak ada *return type* yang diberikan kedalam *Constructor Signature*.
- c. Tidak ada *return statement*, didalam tubuh *constructor*.

Untuk lebih jelasnya sekarang bisa anda lihat pada program seperti dibawah ini :

```
1  /* -----
2  Nama File : lat801.java
3  Author    : Frieyadie
4  ----- */
5
6  class Lat801
7  {
8      float nilakhir;
9
10     Lat801(int nilai_akhir)
11     {
12         nilakhir = nilai_akhir;
13     }
14
15     public String grade()
16     {
17         String nilgrade;
18         if(nilakhir >= 80)
19             nilgrade = "A";
20         else if(nilakhir >= 68)
21             nilgrade = "B";
22         else if(nilakhir >= 56)
23             nilgrade = "C";
24         else if(nilakhir >= 49)
25             nilgrade = "D";
26         else
27             nilgrade = "E";
28
29         return nilgrade;
30     }
31
32     public void cetak()
33     {
34         System.out.println("Grade Nilainya = " + grade());
35     }
36
37     public static void main(String[] args)
38     {
39         Lat801 hasil = new Lat801(67);
40         hasil.cetak();
41     }
42 }
```

Output yang dihasilkan dari program Lat801.java diatas, seperti dibawah ini :



Gambar 8.1 Hasil Program Lat801.java

Berikut penjelasan program Lat801.java, diatas :

- a. Pada baris 10 sampai 13, mendeklarasikan pembuatan Constructor. Bisa anda lihat bahwa nama methodnya sama dengan nama class yaitu sama-sama Lat801. Pada constructor Lat801 ini menangkap argumen yang dikirimkan dari main method.

- b. Pada baris 39, terdapat pernyataan

```
Lat801 hasil = new Lat801(67);
```

digunakan untuk membuat objek dengan nama hasil, sekaligus mengirimkan argumen ke constructor.

8.2.1 Constructor Overloading

Overloading adalah suatu cara membuat lebih dari *constructor* pada suatu *class*. Supaya pengaksesan *constructor* tersebut lancar, maka sebagai pembedanya adalah tipe parameter dan atau jumlah parameternya. Untuk lebih jelasnya anda bisa lihat program dibawah ini :

```

1  /* -----
2     Nama File : lat802.java
3     Author   : Frieyadie
4  ----- */
5  class Lat802
6  {
7      float nilakhir, a;
8
9      Lat802(int nilai_akhir){
10         nilakhir = nilai_akhir;
11     }
12
13     Lat802(int nil1, int nil2){
14         nilakhir = ab + ac;
15     }
16
17     public String grade()
18     {
19         String nilgrade;
20         if(nilakhir >= 80)
21             nilgrade = "A";
22         else if(nilakhir >= 68)
23             nilgrade = "B";
24         else if(nilakhir >= 56)
25             nilgrade = "C";
26         else if(nilakhir >= 49)
27             nilgrade = "D";
28         else
29             nilgrade = "E";
30
31         return nilgrade;
32     }
33
34     public void cetak()
35     {

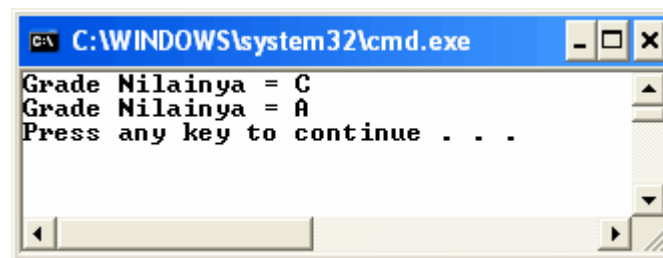
```

```

36      System.out.println("Grade Nilainya = " + grade());
37  }
38
39  public static void main(String[] args)
40  {
41      Lat802 hasil = new Lat802(67);
42      hasil.cetak();
43      Lat802 hasilnya = new Lat802(45, 35);
44      hasilnya.cetak();
45  }
46  }

```

Output yang dihasilkan dari program Lat802.java diatas, seperti dibawah ini :



Gambar 8.2 Hasil Program Lat802.java

Berikut penjelasan program Lat802.java, diatas :

- a. Pada baris 10 sampai 18, mendeklarasikan pembuatan 2 (dua) buah Constructor. Bisa anda lihat bahwa nama methodnya masing-masing sama dengan nama class yaitu sama-sama Lat802. Perbedaan pada masing-masing constructor hanya pada parameternya saja. Pada constructor Lat802 ini menangkap argumen yang dikirimkan dari main method.
- b. Pada baris 44 dan 46, terdapat pernyataan

```

Lat801 hasil = new Lat801(67);
Lat801 hasilnya = new Lat801(45, 35);

```

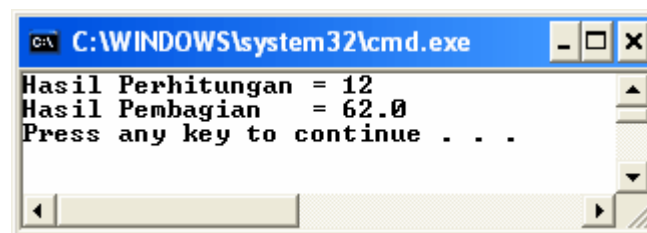
digunakan untuk membuat objek dengan nama hasil, sekaligus mengirimkan argumen ke constructor.

8.2.2 Method Overloading

Method Overloading adalah suatu cara membuat lebih dari *method* pada suatu *class*. Supaya pengaksesan *method* tersebut lancar, maka sebagai pembedanya adalah tipe parameter dan atau jumlah parameternya. Untuk lebih jelasnya anda bisa lihat program dibawah ini :

```
1  /* -----
2     Nama File : lat803.java
3     Author    : Frieyadie
4     ----- */
5
6  class Perhitungan
7  {
8      static public int hitung(int a, int b)
9      {
10         return a + b;
11     }
12
13     static public double hitung(double a, double b, double c)
14     {
15         return (a + b)/c;
16     }
17 }
18
19 public class Lat803
20 {
21     public static void main(String[] args)
22     {
23         Perhitungan Ngitung = new Perhitungan();
24
25         int hitung;
26         double bagi;
27
28         hitung = Ngitung.hitung(4, 8);
29         bagi = Ngitung.hitung(55, 69, 2);
30
31         System.out.println("Hasil Perhitungan = " + hitung);
32         System.out.println("Hasil Pembagian    = " + bagi);
33     }
34 }
```

Output yang dihasilkan dari program Lat803.java diatas, seperti dibawah ini :



Gambar 8.3 Hasil Program Lat803.java

Berikut penjelasan program Lat802.java, diatas :

- c. Pada baris 8 sampai 16, mendeklarasikan pembuatan 2 (dua) buah Method. Bisa anda lihat bahwa nama methodnya sama dengan nama method yang lainnya, yaitu sama-sama hitung. Perbedaan pada masing-masing method hanya pada parameternya saja. Pada method hitung ini menangkap argumen yang dikirimkan dari main method.

d. Pada baris 44 dan 46, terdapat pernyataan

```
hitung = Ngitung.hitung(4, 8);  
bagi = Ngitung.hitung(55, 69, 2);
```

digunakan untuk mengirimkan nilai parameter ke masing-masing method yang dituju, dan kemudian nilai baliknya ditampung kemasing-masing variabel, dimana parameter tersebut dikirimkan.

8.3 Inheritance

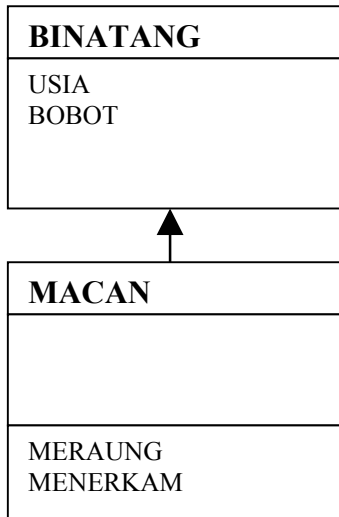
Pewarisan, bahasa kerennya *Inheritance*. Dalam pemrograman berbasis objek, *Inheritance* memungkinkan suatu *Class* bisa mewariskan atribut dan *Method* kepada *Class* yang lainnya atau *subClass*, sehingga membentuk *Class* hirarki.

Inheritance merupakan suatu aspek atau pengarah pokok kecerdasan manusia untuk mencari, mengenali, dan menciptakan hubungan antar konsep. Kita membangun hirarki, matriks, jaringan, dan hubungan timbal balik lain untuk menjelaskan dan memahami tata cara di mana hal-hal saling berhubungan.

Tanpa *Inheritance*, kelas-kelas merupakan sebuah unit berdiri sendiri. *Inheritance*, akan membentuk suatu konsep dimana jika konsep yang diatas berubah, maka perubahan akan berlaku dibawahnya.

Inheritance sangat mirip dengan hubungan orang tua dengan anak. Manakala suatu kelas menerima warisan dari semua anggota data dan fungsi menerima warisan, walaupun tidak semua di antara mereka akan dapat diakses oleh anggota fungsi dari kelas

Sebagai contoh, pada saat kita bicara mengenai Binatang, maka Binatang tersebut bisa mewarsikan kepada Binatang yang lain berupa, usia, bobot, makan, tidur dan lain sebagainya.



Gambar 8.4. Merwariskan Class

Dilihat pada gambar diatas, Class Macan, akan mewariskan seluruh method dan variabel instan dari Class Binatang yang bersifat public. Berarti Class Binatang disebut sebagai SuperClass atau Class Induk dan Class Macan disebut sebagai SubClass atau Class Anak atau Class Turunan.

Terkadang akan menemukan dimana kondisi suatu nama method pada class anak sama dengan nama method pada SuperClass nya, hal seperti ini disebut dengan *Override*.

8.3.1 Penentu Akses Pada Inheritance

Penentuan akses pada Inheritance ada tiga macam, yaitu :

a. Public

Penentuan akses berbasis Public, menyebabkan anggota dari public dari sebuah class utama akan menjadi anggota public class turunan dan menyebabkan juga anggota protect class utama menjadi anggota protect class turunan, tetapi untuk anggota class private tetap pada private class utama.

b. Private

Penentu akses berbasis Private, menyebabkan anggota dari anggota public dari class utama akan menjadi anggota protect class turunan, dan menyebabkan anggota dari class utama menjadi anggota private dari class turunan. Anggota Private dari class utama tidak dapat diakses kecuali oleh class utama.

c. Protected

Penentu akses berbasis Protect, menyebabkan anggota dari anggota protect dan public dari class utama akan menjadi anggota private dari class turunan. Anggota Private dari class utama selalu menjadi anggota private class utama.

8.3.2 Mulai Melakukan Pewarisan

Untuk melakukan pewarisan suatu class, yang perlu diperhatikan, yaitu kita harus menggunakan *keyword extends*. *Extends* digunakan untuk mendeklarasikan suatu class adalah turunan dari class lainnya. Terpenting yang perlu diingat, sebuah class tidak diperbolehkan mempunyai lebih dari satu class induk atau superclass.

Supaya lebih jelas bagaimana melakukan pewarisan suatu class, sekarang perhatikan beberapa program dibawah ini :

```
1  /* -----  
2      Nama File : Matematika.java  
3      Author   : Frieyadie  
4  ----- */  
5  
6  class Matematika  
7  {  
8      private int a, b;  
9  
10     public Matematika()  
11     {  
12         a = 1;  
13         b = 2;  
14     }  
15  
16     public int tambah()  
17     {  
18         return a + b;  
19     }  
20  
21     public int kali()  
22     {  
23         return b * 3;  
24     }  
25 }
```

Setelah selesai, membuat program diatas, simpan dengan nama : Matematika.java. Pada class Matematika, mempunyai 2 (dua) buah method, yaitu tambah dan kali dan 2 (dua) buah variabel yaitu a dan b. Kemudian *compile* program Matematika.java tersebut.

Kemudian, berdasarkan class Matematika diatas, sekarang kita akan buat sebuah class baru yang merupakan class turunan dari class Matematika. Bentuk penulisan class turunan seperti dibawah ini:


```

class Nama_Class_Turunan extends Class_Induk
{
    Badan class turunan
}

```

Penjelasan :

- Nama_Class_Turunan, adalah nama class yang akan dibuat sebagai class turunan atau subclass
- Class_Induk, adalah class yang ditunjuk sebagai SuperClass untuk class turunan

Berikut contoh pembuatan subclass Hitungan

```

1  /* -----
2      Nama File : Hitungan.java
3      Author   : Frieyadie
4  ----- */
5
6  class Hitungan extends Matematika
7  {
8      private int x, y;
9
10     public Hitungan()
11     {
12         x = 1;
13         y = 2;
14     }
15
16     public Hitungan(int i, int j)
17     {
18         x = i;
19         y = j;
20     }
21
22     public int tambah()
23     {
24         return x + y;
25     }
26
27     public int kali()
28     {
29         return y * 3;
30     }
31 }

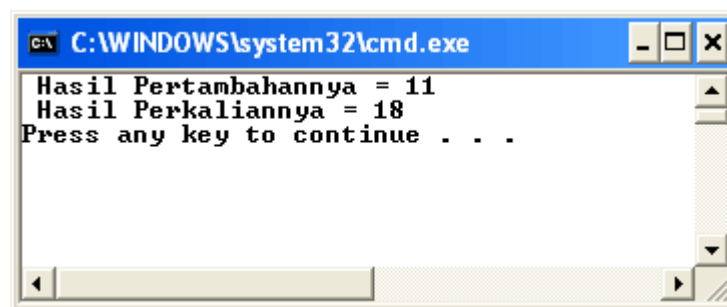
```

Pada program Hitungan.java, diperuntukan untuk melakukan proses perhitungan, jika dilihat disini terdapat 2 (dua) constructor dan penggunaan method yang sama dengan method class induknya. Lakukan compile program Hitungan.java.

Selanjutnya kita akan membuat program utama, untuk menampilkan hasil proses yang dilakukan oleh class Matematika dan class Hitungan. Berikut program utamanya.

```
1  /* -----  
2      Nama File : Lat804.java  
3      Author   : Frieyadie  
4  ----- */  
5  
6  class Lat804  
7  {  
8      public static void main(String[] args)  
9      {  
10  
11          Hitungan ngitung = new Hitungan(5, 6);  
12  
13          System.out.println(" Hasil Pertambahannya = " + ngitung.tambah());  
14          System.out.println(" Hasil Perkaliannya = " + ngitung.kali());  
15      }  
16  }
```

Program Lat804.java, kita bisa melakukan pengiriman nilai kedalam class Hitungan. Selanjutnya memanggil method tambah() dan kali() untuk ditampilkan hasilnya.



Gambar 8.5. Hasil Program La804.java

a. Menggunakan Method SuperClass

Method yang berada didalam SuperClass juga bisa digunakan atau dipanggil dari SubClass nya. Hal tersebut bisa dilakukan dengan menggunakan *keyword* **super**, yang memiliki pengertian SuperClass. Bentuk penulisan penggunaan method SuperClass, seperti dibawah ini :

```
super.nama_method();
```

Untuk lebih jelasnya, sekarang bukalah program Hitungan.java, kemudian tambahkan, beberapa perintah, seperti dibawah ini :

```
1  /* -----
2     Nama File : Hitungan.java
3     Author   : Frieyadie
4     ----- */
5
6  class Hitungan extends Matematika
7  {
8     private int x, y;
9
10     public Hitungan()
11     {
12         x = 1;
13         y = 2;
14     }
15     public Hitungan(int i, int j)
16     {
17         x = i;
18         y = j;
19     }
20     public int tambah()
21     {
22         return (x + y) + super.kali();
23     }
24     public int kali()
25     {
26         return (y * 3 - super.tambah());
27     }
28 }
```

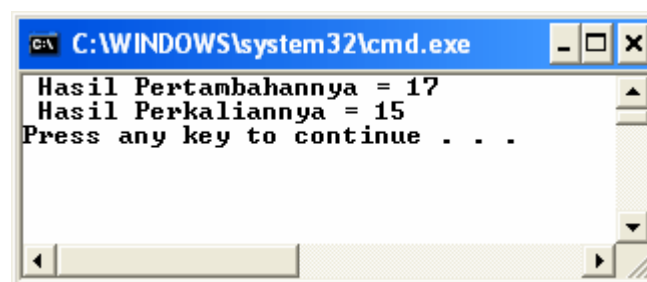
Sudah dijelaskan sebelumnya class Hitungan, merupakan class turunan dari class Matematika, sekarang perhatikan pada baris 24, perintahnya ditambah menjadi

```
return (x + y) + super.kali();
```

Pada baris 29, juga tambahkan pula, menjadi :

```
return (y * 3) - super.tambah();
```

Kedua penambahan tersebut, yaitu memanggil method kali dan method tambah, yang berada didalam SuperClass. Sekarang compile kembali program Hitungan.java dan running kembali program Lat901.java.



Gambar 8.6. Hasil Program Lat804.java

b. Menggunakan Constructor SuperClass

Constructor yang berada didalam SuperClass juga bisa digunakan atau dipanggil dari SubClass nya. Hal tersebut bisa dilakukan dengan menggunakan *keyword* **super**, yang digunakannya seperti pada saat menggunakan method. Bentuk penulisan menggunakan Constructor SuperClass, seperti dibawah ini :

```
super(parameter1, parameter2, ..., parameter-n);
```

Untuk lebih jelasnya, sekarang bukalah program Hitungan.java, kemudian tambahkan, beberapa perintah, seperti dibawah ini :

```

1  /* -----
2     Nama File : Aritmatika.java
3     Author   : Frieyadie
4  ----- */
5
6  class Aritmatika
7  {
8      private int a, b;
9
10     public Aritmatika(int x, int y)
11     {
12         this.a = x;
13         this.b = y;
14     }
15
16     public int kali()
17     {
18         return a * b;
19     }
20
21     public void hasil()
22     {
23         System.out.println("Nilai A = " + this.a);
24         System.out.println("Nilai B = " + this.b);
25     }
26 }
```

Program pertama diatas, kita anggap sebagai class induk yang akan dipergunakan pada subclass pada program dibawah ini. Pada program diatas, terdapat sebuah constructor yang mendeklarasikan nilai awal a dan b. Setelah itu *compile* lah program Aritmatika.java.

```

1  /* -----
2     Nama File : Perhitungan.java
3     Author    : Frieyadie
4  ----- */
5
6  class Perhitungan extends Aritmatika
7  {
```

```

8      protected int z;
9
10     public Perhitungan(int z, int x, int y)
11     {
12         super(x, y);
13         this.z = z;
14     }
15
16     public int Hitung()
17     {
18         return z + super.kali();
19     }
20
21     public void hasil()
22     {
23         System.out.println("Nilai Z = " + this.z);
24         super.hasil();
25     }
26 }

```

Program Perhitungan.java, sebagai class turunan dari class Aritmatika. Bisa anda perhatikan pada baris 12, terdapat pemanggilan constructor SuperClass, yang berfungsi mengirimkan nilai paramater x danb y, ke SuperClass Aritmatika.

Pada baris 18, terdapat pemanggilan method kali() dari SuperClass, pada proses ini nilai z akan ditambahkan dari hasil perkalian yang berada didalam method kali().

```

1  /* -----
2     Nama File : Lat805.java
3     Author    : Frieyadie
4  ----- */
5
6  public class Lat805
7  {
8      public static void main(String[] args)
9      {
10         Perhitungan matematika = new Perhitungan(5, 4, 3);
11         matematika.hasil();
12         System.out.println("Hasil Perhitungannya = " + matematika.Hitung());
13     }
14 }

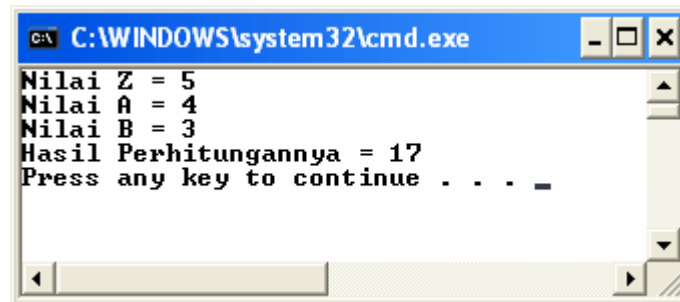
```

Pada program utama, membuat objek berdasarkan dari class perhitungan, dikarenakan yang akan diproses adalah class turunan yaitu class Perhitungan, dan sekaligus mengirimkan nilai parameter ke class Perhitungan.

Pada baris 11, juga terdapat pemanggilan method hasil() dari class Perhitungan, yang gunakanya untuk menampilkan nilai-nilai yang ada pada class Perhitungan.

Pada baris 12, juga melakukan pemanggilan method Hitung(), yang akan ditampilkan hasil perhitungannya.

Sekarang simpanlah program Lat805.java diatas pada folder kerja anda masing-masing. Kemudian compile dan running, maka akan tampil hasil program Lat805.java seperti dibawah ini:



Gambar 8.7. Hasil Program Lat805.java

8.4 Polymorphism

Karakteristik dari polymorphism yaitu memungkinkan suatu objek dapat memiliki berbagai bentuk atau banyak bentuk. Bentuk dari objek ini bisa sebagai *Object* dari *Class*nya sendiri atau *Object* dari *superClass*nya.

Pada polymorphism kita akan sering menjumpai 2 (dua) istilah yang sering digunakan dalam pemrograman berbasis objek, istilah tersebut yaitu :

a. *Overloading*.

Overloading yaitu menggunakan 1 (satu) nama objek untuk beberapa *Method* yang berbeda ataupun bisa juga beda parameternya.

b. *Overriding*

Overriding akan terjadi apabila ketika pendeklarasian suatu *Method* subClass dengan nama objek dan parameter yang sama dengan *Method* dari *superClass*nya.

```
1  /* -----  
2  Nama File : Binatang.java  
3  Folder    : c:\LatihanJava  
4  Author    : Frieyadie  
5  ----- */  
6  class Binatang  
7  {  
8      String namaBinatang;  
9      Binatang()  
10     {  
11     }
```

```
12
13     Binatang(String namaBinatang)
14     {
15         this.namaBinatang = namaBinatang;
16     }
17
18     public void cetakjenis()
19     {
20         System.out.println("Nama Binatang : "+ namaBinatang);
21     }
22 }
```

Pada program Binatang.java, terdapat constructor, untuk memberikan nilai awal. Berikutnya buatlah class baru seperti dibawah ini:

```
1  /* -----
2  Nama File : suara.java
3  Folder    : c:\LatihanJava
4  Author    : Frieyadie
5  ----- */
6  class suara extends Binatang
7  {
8      String suara;
9      suara()
10     {
11         super();
12     }
13
14     public void cetakjenis()
15     {
16         suara="Mengaum";
17         System.out.println("Suara : " + suara);
18     }
19 }
```

Pada program suara.java, sebagai subclass dari class binatang, juga terdapat constructor suara, untuk memberikan nilai awal, dan terdapat overriding cetakjenis(). Berikutnya buatlah class baru seperti dibawah ini:

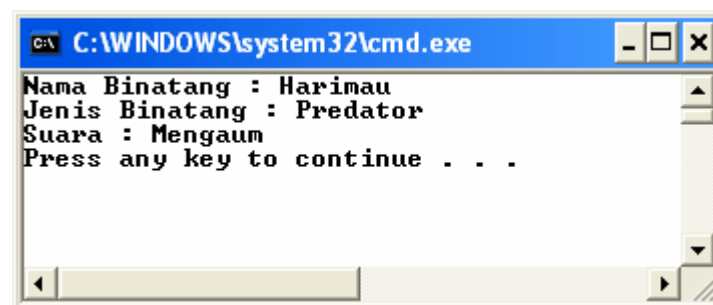
```
1  /* -----
2  Nama File : Binatang.java
3  Folder    : c:\LatihanJava
4  Author    : Frieyadie
5  ----- */
6  class jenisBinatang extends Binatang
7  {
8      String jenisBinatang;
9      jenisBinatang(String jenisBinatang)
10     {
11         super(jenisBinatang);
12     }
13
14     public void cetakjenis()
```

```
15 {  
16     super.cetakjenis();  
17     jenisBinatang="Predator";  
18     System.out.println("Jenis Binatang : " + jenisBinatang);  
19 }  
20 }
```

Pada program `jenisBinatang.java`, sebagai subclass dari class `binatang`, juga terdapat constructor `jenisBinatang`, untuk memberikan nilai awal, dan terdapat overriding `cetakjenis()`. Berikutnya buatlah class utama seperti dibawah ini

```
1  /* -----  
2  Nama File : Lat806.java  
3  Folder   : c:\LatihanJava  
4  Author    : Frieyadie  
5  ----- */  
6  class Harimau  
7  {  
8      public static void main(String[] args)  
9      {  
10         suara binatang = new suara();  
11         jenisBinatang suara = new jenisBinatang("Harimau");  
12         suara.cetakjenis();  
13         binatang.cetakjenis();  
14     }  
15 }
```

Berikut hasil program `Lat806`, seperti dibawah ini :



Gambar 8.8. Hasil Program Lat806.java

8.5 Latihan

Kerjakanlah soal latihan mengenai Constructor

1. Buatlah program menghitung luas dan keliling lingkaran, dengan menggunakan cara constructor. Nilai Radius (R), dikirimkan ke constructor.
2. Buatlah program menghitung luas trapesium, dengan menggunakan cara constructor. Rumus Luas Trapesium = $\frac{1}{2} \times (\text{panjang atas} + \text{panjang bawah}) \times \text{tinggi}$
Gunakan Method Constructor untuk mengerjakannya.
3. Kembangkanlah program constructor diatas. Nilai Akhir didapat dari hasil perhitungan nilai UTS, nilai UAS, nilai Tugas dan Nilai Absensi. Adapun nilai UTS, nilai UAS, nilai Tugas dan Nilai Absensi diinput dari keyboard.
4. Buatlah program untuk menghitung volume Silinder. Gunakan class lingkaran sebagai Super Class dan Class Silinder sebagai SubClass. Nilai Radius diinput melalui keyboard.
5. Buatlah program untuk menghitung penjumlahan, pengurangan, perkalian dan pembagian yang diturunkan dari class Matematika. Nilai Satu dan Dua diinput.
6. Buatlah program untuk menghitung nilai akhir dan mencari nilai grade dengan cara Pewarisan. Gunakan class Ujian sebagai SuperClass dan Class UTS dan Class UAS sebagai SubClass. Ketentuan lainnya nilai UTS dikali dengan 40% untuk mendapatkan nilai murni UTS dan untuk mendapatkan nilai murni UAS yaitu 60% dikali dengan nilai UAS.

Lembar ini Sengaja Dikosongkan

Bab 9:

Enkapsulasi

9.1 Kopetensi Dasar

Pada pembahasan Bab 9 ini penulis mengajak mendiskusikan mengenai pengkapsulan pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami menggunakan pengkalpsulan pada bahasa pemrograman Java. Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Melakukan Enkapsulasi terhadap properti Class
- Penggunaan Kontrol Akses Encapsulation

9.2 Apa Itu Enkapsulasi

Enkapsulasi, bahasa kerennya *Encapsulation*. Dalam pemrograman berbasis objek, Encapsulation merupakan suatu cara bagaimana menyembunyikan sedemikian rupa suatu proses kedalam sistem, hal ini berguna untuk menghindari interferensi dari luar sistem dan juga lebih untuk menyederhakanan sistem itu sendiri.

Kita ambil contoh, pada saat anda mengganti chanel TV menggunakan remote TV, apakah anda mengetahui proses yang terjadi didalam TV tersebut ?, maka jawabannya tidak tau, dan anda pun sebagai pembeli TV tidak mau dipusingkan dengan proses yang terjadi. Maka hal tersebut menyederhakan sistem.

Pada Enkapsulasi, segala jenis pengkasesan properti class, harus melalui method, nah inilah cara kerja encapsulasi, sehingga proses yang mencegah class variabel - class variabel yang sedang dibaca, dimodifikasi oleh class lainnya.

9.3 Kontrol Akses Pada Enkapsulasi

9.3.1 Akses Default

Variabel atau metode bisa dideklarasikan tanpa kontrol akses modifier yang tersedia untuk setiap kelas lainnya dalam satu paket. Java Class Library pada Java disusun dalam paket seperti javax.swing seperti class window, terutama untuk digunakan dalam pemrograman GUI (*Graphical User Interface*), dan java.util, yang berguna kelompok Class utilitas.

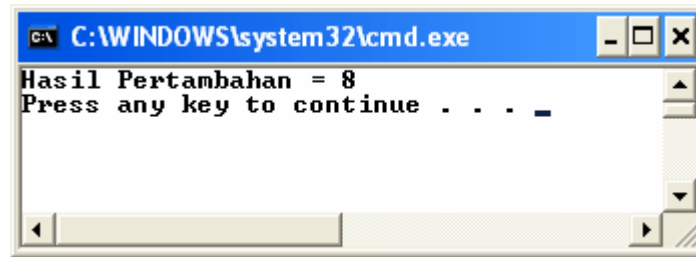
Setiap variabel dideklarasikan tanpa *modifier* dapat dibaca atau diubah oleh yang lain di kelas satu paket. Metode apapun dapat dinyatakan dengan cara yang sama dan dapat dipanggil oleh kelas lain dalam satu paket.

```
1  /* -----  
2     Nama File : Lat1001.java  
3     Author   : Frieyadie  
4  ----- */  
5  
6  class hitung  
7  {  
8      int c = 8;  
9  }  
10  
11  class Lat1001  
12  {  
13      public static void main(String[] args)  
14      {  
15          int hasil;  
16          hitung tambah = new hitung();  
17  
18          System.out.println("Hasil Pertambahan = " + tambah.c);  
19      }  
20  }
```

Penjelasan :

Pada class hitung diatas tidak terdapat akses kontrol modifier yang digunakan. Kemudian pada class Lat1001, variabel c, juga bisa langsung digunakan.

Berikut output yang dihasilkan dari program Lat1001.java



Gambar 9.1. Hasil Program Lat901.java

9.3.2 Akses Private

Secara utuh menyembunyikan variabel atau metoda yang akan digunakan oleh kelas-kelas lain, dengan menggunakan modifier private. Satu-satunya tempat dimana variabel-variabel atau metoda-metoda ini dapat diakses yaitu dari dalam kelasnya sendiri.

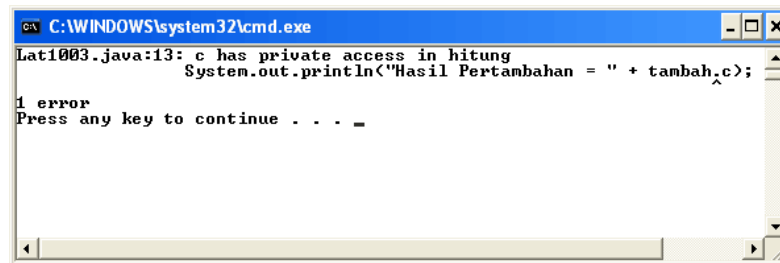
Sebagai contoh variabel instan private, bisa digunakan oleh method-method didalam kelas kepunyaan nya, akan tetapi bukan oleh object dari Class lainnya. Method Private dapat dipanggil oleh metoda-metoda lain di dalam kelas mereka sendiri tetapi tidak bisa dipanggil oleh lainnya.

Variabel-variabel private akan bermanfaat pada 2 (dua) keadaan:

1. Apabila kelas-kelas lain tidak punya alasan untuk menggunakan variabelnya.
2. Apabila kelas lain bisa menghadirkan nilai dengan mengubah variabel pada satu cara yang tidak sesuai.

```
1  /* -----  
2  Nama File : Lat1002.java  
3  Author   : Frieyadie  
4  ----- */  
5  
6  class hitung  
7  {  
8      private int c = 8;  
9  }  
10  
11  class Lat1002  
12  {  
13      public static void main(String[] args)  
14      {  
15          int hasil;  
16          hitung tambah = new hitung();  
17  
18          System.out.println("Hasil Pertambahan = " + tambah.c);  
19      }  
20  }
```

Di contoh kode ini, format variabel class hitung yaitu `private`, maka sama sekali class-class lainnya tidak bisa untuk mendapat kembali atau menetapkan nilainya secara langsung. Maka jika di compile, akan terjadi kesalahan seperti gambar dibawah ini :



Gambar 9.2. Pesan Kesalahan Akses Data Private

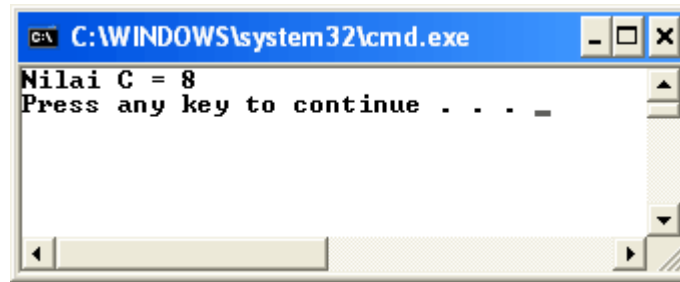
Cara untuk mengakses variabel `c` yang bersifat `private`, maka dengan cara menambahkan metode yang digunakan untuk mengakses variabel tersebut. Modifikasi programnya seperti dibawah ini :

```

1  /* -----
2  Nama File : Lat1003.java
3  Author   : Frieyadie
4  ----- */
5
6  class hitung
7  {
8      private int c = 8;
9      public int getVAR()
10     {
11         return this.c;
12     }
13 }
14
15 class Lat1003
16 {
17     public static void main(String[] args)
18     {
19         int hasil;
20         hitung tambah = new hitung();
21
22         System.out.println("Nilai C = " + tambah.getVAR());
23     }
24 }

```

Maka dengan cara menambahkan method `getVAR`, sekarang variabel `c`, yang bersifat `public` bisa diakses.



Gambar 9.3. Hasil Pengaksesan Variabel Private

Menggunakan modifier private adalah cara utama suatu object mengencapsulasi dirinya.

9.3.3 Akses Public

Dalam beberapa hal, anda mungkin ingin variabel atau metoda dalam suatu kelas untuk dengan sepenuhnya tersedia bagi kelas lain yang ingin menggunakan. Sebagai contoh, class warna di dalam paket java.awt, mempunyai variabel-variabel publik untuk warna-warna umum seperti hitam. Variabel ini digunakan apabila suatu class ingin gunakan warna hitam, dan tidak perlu punya kendali akses sama sekali.

proses kedalam sistem, hal ini berguna untuk menghindari interferensi dari luar sistem dan juga lebih untuk menyederhakanan sistem itu sendiri.

Modifier public membuat suatu variabel atau metoda dengan sepenuhnya dapat diakses oleh semua kelas. Kita sudah sering menggunakannya di dalam tiap-tiap aplikasi yang kita tertulis sejauh ini, dengan statemen seperti berikut:

```
public static void main(String[] args)
{
    //code here ...
}
```

Method main(), haruslah bersifat public, karena tidak akan bisa dipanggil oleh Java Interpreter untuk menjalankan class. Oleh karena itu subclass menerima warisan, semua variabel dan method dari suatu class.

9.3.4 Akses Protected

Protected digunakan untuk membatasi variabel dan metoda untuk digunakan oleh dua kelompok berikut, yaitu :

- Sub Class dari satu Class
- Class-class lain di dalam paket yang sama

Tingkat kendali akses ini bermanfaat jika kita ingin membuat itu lebih mudah untuk subclass untuk diterapkan sendiri. Akses protected memberi subclass untuk menolong variabel atau metoda, ketika suatu class yang bukan hubungannya berusaha untuk menggunakan hal tersebut.

```
1  /* -----
2     Nama File : Paket1.java
3     Folder   : c:\LatihanJava\Latpack
4     Author   : Frieyadie
5  ----- */
6  Package Latpack
7
8  public class Paket1
9  {
10     protected int a = 75;
11     public void info()
12     {
13         System.out.println("Ini Kelas Paket1");
14     }
15 }
```

Simpan dengan nama Packet1.java, kemudian compilelah file Paket1.java. Selanjutnya buatlah class yang kedua, seperti dibawah ini :

```
1  /* -----
2     Nama File : Paket2.java
3     Folder   : c:\LatihanJava\Latpack
4     Author   : Frieyadie
5  ----- */
6  Package Latpack
7
8  public class Paket2
9  {
10     public void info()
11     {
12         System.out.println("Ini Kelas Paket2");
13     }
14 }
```

Simpan dengan nama Packet2.java, kemudian compilelah file Paket2.java. Selanjutnya buatlah class yang ketiga, seperti dibawah ini:


```
1  /* -----  
2  Nama File : Lat1004.java  
3  Folder    : c:\LatihanJava  
4  Author    : Frieyadie  
5  ----- */  
6  import Latpack.Paket1;  
7  import Latpack.Paket2;  
8  
9  public class Lat1004  
10 {  
11     public static void main(String[] args)  
12     {  
13         Paket1 objekPaket1 = new Paket1();  
14         objekPaket1.info();  
15  
16         System.out.println("Nilai A = " + objekPaket1.a);  
17  
18         Paket2 objekPaket2 = new Paket2();  
19         objekPaket2.info();  
20     }  
21 }
```

Penjelasan :

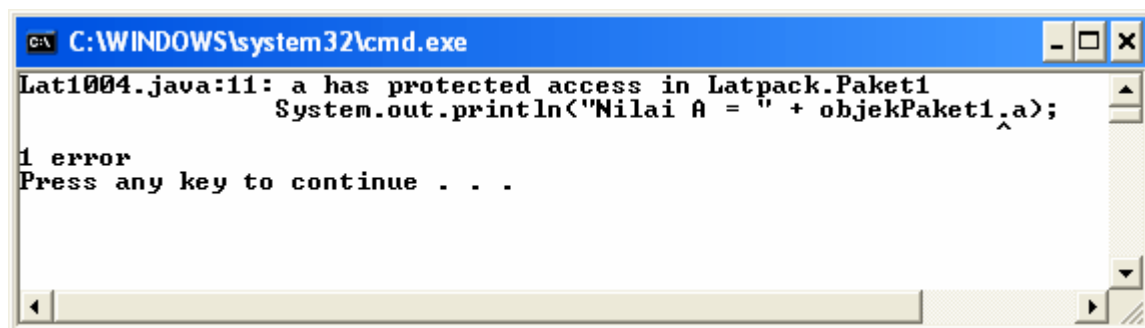
Pada program Paket1.java, terdapat deklarasi variabel

```
protected int a = 75;
```

Kemudian pada program Lat1004.java, terdapat pemanggilan secara langsung variabel a, dari program Lat1004.java,

```
System.out.println("Nilai A = " + objekPaket1.a);
```

maka dalam hal ini, jika dicompile, maka akan terjadi error, seperti dibawa ini :



Gambar 9.4. Error pada Pengaksesan Langsung

Solusi yang bisa dilakukan yaitu dengan cara membuat method untuk pengaksesan variabel yang menggunakan modifier protected();

9.4 Latihan

1. Buatlah program untuk menghitung volume Silinder. Gunakan konsep Enkapsulasi. Nilai Radius diinput melalui keyboard.
2. Buatlah program untuk menghitung penjumlahan, pengurangan, perkalian dan pembagian dengan menggunakan konsep Enkapsulasi. Nilai Satu dan Dua diinput.
3. Buatlah program untuk menghitung nilai akhir dan mencari nilai grade dengan cara Pewarisan. Gunakan Konsep Enkapsulasi. Ketentuan lainnya nilai UTS dikali dengan 40% untuk mendapatkan nilai murni UTS dan untuk mendapatkan nilai murni UAS yaitu 60% dikali dengan nilai UAS.

Bab 10:

Package dan Interface

10.1 Kopetensi Dasar

Pada pembahasan Bab 10 ini penulis mengajak mendiskusikan mengenai Package dan Interface pada Bahasa Pemrograman Java. Kompetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami menggunakan Package dan Interface pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Melakukan Pendeklarasian pembuatan Package dan Interface
- Penggunaan Package dan Interface.

10.2 Package

Package berisi beberapa class-class yang terkait dalam tujuan, lingkup, atau yang disebabkan oleh inheritance. Package akan lebih bermanfaat apabila, digunakan pada proyek-proyek aplikasi yang besar dengan banyak class yang mungkin saling dihubungkan dengan inheritance. Sehingga akan lebih menguntungkan dikelola didalam package.

10.2.1 Memulai Membuat Package

Untuk membuat package, diawali dengan kata kunci package kemudian diikuti dengan nama packagenya. Berikut bentuk umum penulisan package :

```
package nama_package;  
public class nama_class  
{  
    //tubuh class  
}
```

Dalam pembuatan package ini, yang perlu diperhatikan semua file yang dikelompokkan dalam package, harus disimpan didalam folder yang nama foldernya sama dengan nama packagenya. Misalnya nama package nya LatPackage, berarti nama foldernya juga LatPackage.

Untuk latihan pertama pembuatan package, anda bisa mengikuti beberapa langkah seperti dibawah ini :

1. Buatlah folder dengan nama LatPackage didalam folder latihan anda. Misalnya :
c:\LatihanJava\LatPackage.
2. Selanjutnya buatlah program, seperti dibawah ini :

```
1  /* -----  
2     Nama File   : Paket1.java  
3     Folder      : c:\LatihanJava\LatPackage  
4     Author       : Frieyadie  
5  ----- */  
6  package LatPackage;  
7  
8  public class Paket1  
9  {  
10     protected int a = 85;  
11  
12     public int getnilai()  
13     {  
14         return a;  
15     }  
16  
17     public void info()  
18     {  
19         System.out.println("Ini Class Paket1");  
20     }  
21 }
```

Simpanlah file Paket1.java ini didalam folder LatPackage, yang telah anda buat diatas.
(c:\LatihanJava\LatPackage\Paket1.java)

```
1  /* -----  
2     Nama File   : Paket2.java  
3     Folder      : c:\LatihanJava\LatPackage  
4     Author       : Frieyadie  
5  ----- */  
6  package LatPackage;  
7  
8  public class Paket2  
9  {  
10     public void info()  
11     {  
12         System.out.println("Ini Class Paket2");  
13     }  
14 }
```

Simpanlah file Paket2.java ini didalam folder LatPackage, yang telah anda buat diatas. (c:\LatihanJava\LatPackage\Paket2.java).

Jika dilihat dari kedua contoh program diatas, diawal penulisan program terdapat perintah:

```
packate LatPackage;
```

Perintah ini menyatakan bahwa program Paket1.java dan Paket2.java, dikelompokkan dalam package bernama LatPackage dan kedua program tersebut dideklarasikan secara public.

10.2.2 Menggunakan Package

Untuk menggunakan anggota-anggota paket, hanya paket yang dideklarasikan secara public yang dapat diakses dari luar paket di mana mereka itu buat.

Untuk menggunakan kelas dalam sebuah paket, anda dapat menggunakan salah satu dari tiga teknik berikut:

1. Jika class yang akan digunakan didalam paket java.lang, misalnya System atau Date, maka kita bisa secara langsung menggunakan nama class sebagai rujukan keclass tersebut.
2. Jika kelas yang akan menggunakan beberapa paket, anda dapat merujuk ke kelas dengan memberi nama secara lengkap, termasuk nama paket, misalnya, java.awt.Font.
3. Untuk kelas-kelas yang sering digunakan dari paket lainnya, kita dapat mengimpor setiap kelas atau seluruh paket kelas. Setelah kelas atau paket telah diimpor, kita dapat merujuk ke kelas dengan nama kelasnya.

Untuk menggunakan package-package yang telah dibuat diatas, dengan menggunakan perintah :

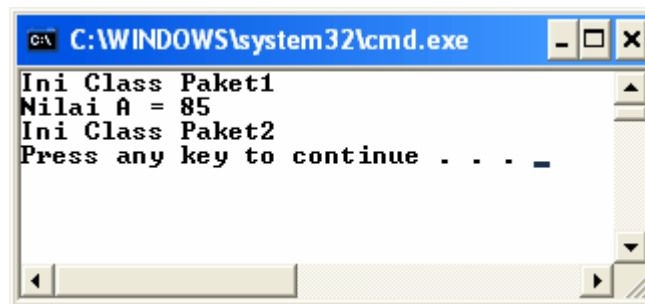
```
import nama_package;
```

Berikut penggunaan package yang telah kita buat diatas.

```
1  /* -----  
2      Nama File   : Lat1001.java  
3      Folder    : c:\LatihanJava  
4      Author     : Frieyadie  
5  ----- */  
6  import LatPackage.Paket1;  
7  import LatPackage.Paket2;  
8  
9  public class Lat1001  
10 {  
11     public static void main(String[] args)
```

```
12 {  
13     Paket1 objekPaket1 = new Paket1();  
14     objekPaket1.info();  
15  
16     System.out.println("Nilai A = " + objekPaket1.getnilai());  
17  
18     Paket2 objekPaket2 = new Paket2();  
19     objekPaket2.info();  
20 }  
21 }
```

Kemudian simpanlah dengan nama Lat1001.java didalam folder LatihanJava pada driver C: (c:\LatihanJava\Lat1001.java). Compile dan jalankan program Lat1001.java, maka akan tampil seperti dibawah ini.



Gambar 10.1. Hasil Program Lat1001.java

10.3 Interface

Interface adalah satu kontrak dalam wujud koleksi metoda dan deklarasi-deklarasi tetap. Bila suatu kelas menerapkan interface, maka hal tersebut untuk menerapkan semua metoda yang dideklarasikan didalam interface.

Interface sering digunakan sebagai pengganti kelas abstrak walaupun tidak ada implementasi secara default untuk menerima inheritance yaitu, tidak ada field-field dan tidak ada implementasi-implementasi metoda secara default. Seperti kelas-kelas abstrak publik, interface adalah jenis-jenis publik secara khusus, sehingga secara normal dideklarasikan di dalam file-file dengan sendirinya, yaitu dengan nama yang sama sebagai interface dan nama file java.

Inteface pada java merupakan satu koleksi perilaku abstrak yang dapat diadopsi oleh kelas manapun tanpa menerima inheritance dari suatu superclass.

10.3.1 Membuat Interface

Untuk membuat interface, dengan cara mendeklarasikannya sama seperti mendeklarasikan class. Pendeklarasian interface terlebih dahulu dengan kata kunci interface, seperti dibawah ini:

```
public interface nama_interface
{
    //tubuh interface
}
```

Penjelasan :

- Public bertujuan bahwa interface bisa diakses oleh class manapun, apabila tidak dideklarasikan secara public, maka hanya bisa diakses oleh class-class yang berada didalam satu paket yang sama saja.
- Tubuh Interface, berisikan konstanta dan method yang berada didalam interface.

Berikut contoh mendeklarasikan dan membuat interface.

```
interface Trayek
{
    public static final String PERGI = "Blok M";
    public static final String PULANG = "Ciledug";

    public abstract void tarikpergi();
    public abstract void tarikpulang();
}
```

10.3.2 Menggunakan Interface

Untuk menggunakan interface pada sebuah class, anda bisa dengan menggunakan kata kunci implement, dengan bentuk penulisan seperti dibawah ini:

```
1  /* -----
2     Nama File   : Lat1002.java
3     Folder     : c:\LatihanJava
4     Author      : Frieyadie
5  ----- */
6  interface Trayek
7  {
8      public static final String PERGI = "Blok M";
9      public static final String PULANG = "Ciledug";
10
11      public abstract void tarikpergi();
12      public abstract void tarikpulang();
```

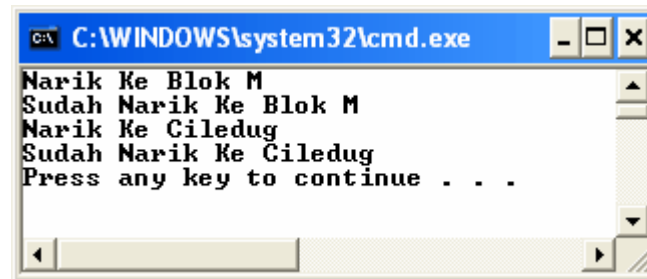
```
13 }
14
15 class Bus implements Trayek
16 {
17     private String statusBus = "Ciledug";
18
19     public void tarikpergi()
20     {
21         if(this.statusBus.equals(PULANG))
22         {
23             this.statusBus = PERGI;
24             System.out.println("Narik Ke Blok M");
25         }
26         else
27         {
28             System.out.println("Sudah Narik Ke Blok M");
29         }
30     }
31
32
33     public void tarikpulang()
34     {
35         if(this.statusBus.equals(PERGI))
36         {
37             this.statusBus = PULANG;
38             System.out.println("Narik Ke Ciledug");
39         }
40         else
41         {
42             System.out.println("Sudah Narik Ke Ciledug");
43         }
44     }
45 }
46 class Lat1002
47 {
48     public static void main(String[] args)
49     {
50         Bus MetroMini = new Bus();
51         MetroMini.tarikpergi();
52         MetroMini.tarikpergi();
53         MetroMini.tarikpulang();
54         MetroMini.tarikpulang();
55     }
56 }
```

Penjelasan Program :

- Pada baris 6 – 13, telah dideklarasikan sebuah interface dengan nama Trayek. Pada interface Trayek ini, terdapat pendeklarasian konstanta, yaitu PERGI dan PULANG. Konstanta ini dapat dipergunakan secara langsung pada class Bus, seperti contoh pada pengimpelentasian method tarikpergi() :
`this.statusBus = PERGI;`
- Pada barus 15 – 45, telah dideklarasikan sebuah class dengan nama Bus menggunakan interface Trayek.


```
class Bus implements Trayek
```

- Kompilasi dan Jalankan program Lat1002.java diatas, maka akan tampil seperti gambar dibawah ini :



Gambar 10.2. Hasil program Lat1002.java

10.4 Latihan

1. Buatlah Program hitung luas dan keliling lingkaran. Buatlah sebuah package, yang digunakan untuk melakukan proses perhitungan luas dan keliling lingkaran tersebut. Nilai Radius diinput dari main method (program utamanya).
2. Buatlah program untuk menghitung nilai akhir. Buatlah sebuah package, yang akan digunakan untuk menghitung nilai akhir tersebut. Nilai-Nilai yang diinput yaitu : Nilai Tugas Mandiri, Nilai Quis dan Nilai Ujian. Adapun ketentuan persentasi nilai sebagai berikut :
 - Nilai Murni Tugas Mandiri didapat dari 25% Nilai Tugas Mandiri
 - Nilai Murni Quis didapat dari 30% Nilai Quis
 - Nilai Murni Ujian didapat dari 45% Nilai Ujian

Lembar ini sengaja dikosongkan

Bab 11:

Penanganan Eksepsi

11.1 Kopetensi Dasar

Pada pembahasan Bab 11 ini penulis mengajak mendiskusikan mengenai penanganan Eksepsi pada Bahasa Pemrograman Java. Kopetensi dasar secara umum, agar mahasiswa/i atau pembaca bisa mendeskripsikan dapat memahami penanganan Eksepsi pada bahasa pemrograman Java.

Penulis berharap, diakhir pembahasan, para pembaca bisa :

- Melakukan Penanganan Eksepsi
- Menggunakan try, throw dan catch untuk mendeteksi, menandai adanya dan penanganan eksepsi secara berturut-turut.

11.2 Apa Itu Penanganan Eksepsi

Eksepsi adalah suatu indikasi masalah yang terjadi selama pelaksanaan program atau eksekusi program. Nama "Exception" menyiratkan bahwa masalah jarang terjadi jika “aturan” suatu statemen secara normal dilaksanakan dengan tepat, maka eksepsi digunakan untuk “mengatur” masalah yang terjadi.

Penanganan eksepsi memberdayakan para programmer untuk membuat aplikasi-aplikasi yang dapat memecahkan atau menangani eksepsi-eksepsi. Dalam banyak kesempatan, menangani Eksepsi memungkinkan suatu program untuk melanjutkan jalannya program, seolah-olah tidak ada masalah.

11.3 Memulai Penanganan Eksepsi

Didalam penanganan eksepsi ini, ada hal yang perlu diketahui, yaitu menangkap eksepsi. Untuk menangkap eksepsi dengan menggunakan pernyataan try. Bentuk Penulisan memiliki 2(dua) bentuk penulisan.

a. Bentuk penulisan Penanganan Eksepsi yang pertama :

```
try
{
    //blok penangkap eksepsi
}
catch(parameter Eksepsi)
{
    //blok jika terjadi eksepsi
}
```

Berikut contoh Penanganan Eksepsi seperti dibawah ini :

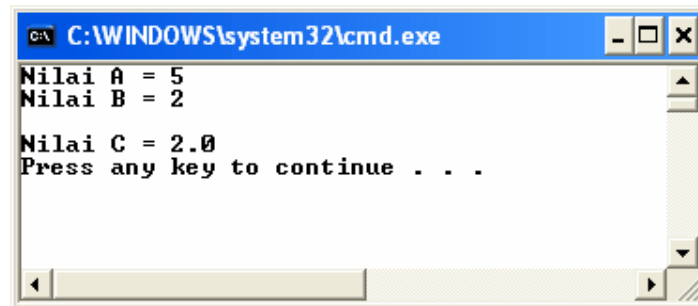
```
1  /* -----
2     Nama File : Lat1101.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat1101
9  {
10     public static void main(String[] args)
11     {
12         int a, b;
13         double c=0;
14
15         Scanner input = new Scanner(System.in);
16
17         System.out.print("Nilai A = ");
18         a = input.nextInt();
19
20         System.out.print("Nilai B = ");
21         b = input.nextInt();
22
23         try
24         {
25             c = a / b;
26         }
27         catch (Exception e)
28         {
29             System.err.println("\nTidak Bisa Dibagi 0");
30         }
31
32         System.out.println("\nNilai C = " + c);
33     }
34 }
```

Penjelasan Program :

Pada blok try, terdapat proses perhitungan, dimana untuk menangkap apakah pembagian tersebut sesuai atau tidak, disini nilai pembagi tidak diperkenankan bernilai 0 (nol).

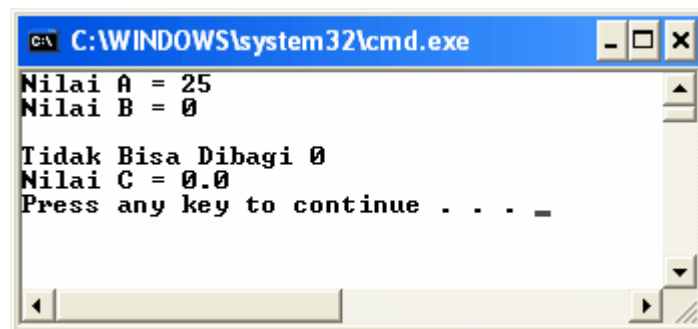
Abila nilai pembagi bernilai 0 (nol), maka terjadi eksepsi. Selanjutnya pada blok catch yang akan dijalankan dengan menampilkan pesan “Tidak Bisa dibagi 0”.

Berikutnya kompilasilah dan jalankan program diatas, maka akan tampil seperti dibawah ini :



Gambar 11.1. Pemasukan Data Secara Normal

Berikutnya coba anda masukan nilai b sebagai pembagi dengan nilai 0 (nol), maka akan tampil seperti dibawah ini:



Gambar 11.2. Terjadi Eksepsi pada saat nilai b dimasukan dengan 0 (nol)

b. Bentuk penulisan Penanganan Eksepsi yang pertama :

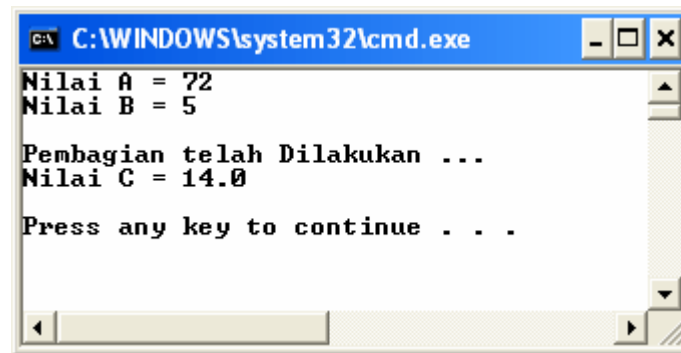
```
try
{
    //blok penangkap eksepsi
}
catch(tipeEksepsi ObjekEksepsi)
{
    //blok jika terjadi eksepsi
}
```

```
finally
{
    //blok akhir yang akan dieksekusi setelah blok try dan catch
}
```

Pernyataan finally bermanfaat diluar eksepsi. Hal ini dapat melakukan clean-up kode setelah pernyataan return, break atau continue didalam perulangan. Berikut contoh program dengan pemakaian pernyataan finally.

```
1  /* -----
2     Nama File : Lat1102.java
3     Author    : Frieyadie
4     ----- */
5
6  import java.util.Scanner;
7
8  class Lat1101
9  {
10     public static void main(String[] args)
11     {
12         int a, b;
13         double c=0;
14
15         Scanner input = new Scanner(System.in);
16
17         System.out.print("Nilai A = ");
18         a = input.nextInt();
19
20         System.out.print("Nilai B = ");
21         b = input.nextInt();
22
23         try
24         {
25             c = a / b;
26         }
27         catch (Exception e)
28         {
29             System.err.print("\nTidak Bisa Dibagi 0 atau ");
30             System.err.println(e.getMessage());
31         }
32         finally
33         {
34             System.err.print("\nPembagian telah Dilakukan ... ");
35         }
36
37         System.out.println("\nNilai C = " + c);
38     }
39 }
```

Berikut hasil eksekusi program Lat1102.java.



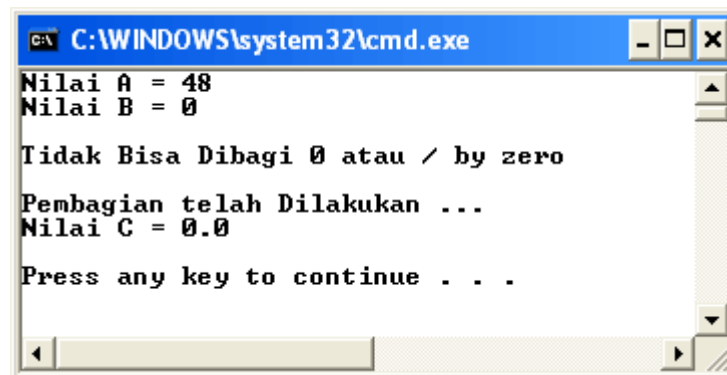
```
C:\WINDOWS\system32\cmd.exe
Nilai A = 72
Nilai B = 5

Pembagian telah Dilakukan ...
Nilai C = 14.0

Press any key to continue . . .
```

Gambar 11.3. Hasil Pembagian Pertama

Dilihat dari eksekusi program Lat1102.java, diatas proses pembagian dilaksanakan secara normal, dan blok finally juga dijalankan dan ditampilkan. Berikut hasil eksekusi dengan pemasukan data, yang tidak diijinkan.



```
C:\WINDOWS\system32\cmd.exe
Nilai A = 48
Nilai B = 0

Tidak Bisa Dibagi 0 atau / by zero

Pembagian telah Dilakukan ...
Nilai C = 0.0

Press any key to continue . . .
```

Gambar 11.4. Hasil Pembagian Kedua

Dari pemasukan data Nilai A = 48 dan Nilai B = 0, maka blok catch, dijalankan dan blok finally juga tetap dilaksanakan. Hal ini menunjukkan bahwa finally, akan selalu dijalankan, walaupun waktu terjadi eksepsi maupun tidak terjadi eksepsi.

11.4 Multi Catch

Java mengijinkan penggunaan multi catch, yang bisa mengetahui adanya kemungkinan-kemungkinan kesalahan dan jenis kesalahan yang akan terjadi. Class yang digunakan untuk penanganan eksepsi yaitu Throwable, class ini adalah class tertinggi untuk penanganan eksepsi. Class Throwable terdapat 2(dua) subclass, yaitu Error dan Exception.

Subclass Error, biasanya digunakan untuk penanganan kesalahan seperti :

- a. `OutOfMemoryError` (Error karena Kehabisan Memori)
- b. `StackOverflowError` (Error karena Tumpukan Melebihi Kapasitas)

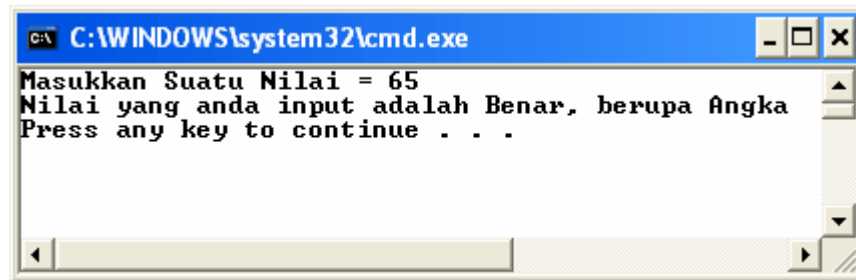
Subclass `Exception`, biasanya digunakan untuk penanganan kesalahan seperti :

- a. `IOException` (Kesalahan pada Input/Output)
- b. `InterruptedException` (Kesalahan adanya Interupsi Program).
- c. `NumberFormatException` (Kesalahan pada Format Angka)

Masih banyak sekali subclass `Exception`, mungkin anda bisa baca lebih lanjut pada `Java Document`. Berikut contoh penerapan `Multi Catch`.

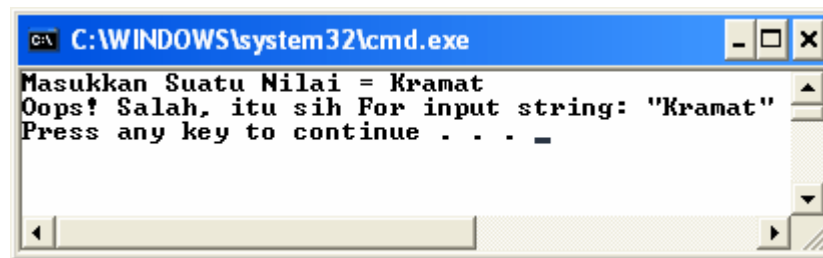
```
1  /* -----
2     Nama File : Lat1103.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat1103
9  {
10     public static void main(String[] args)
11     {
12         String a;
13         int b;
14         Scanner input = new Scanner(System.in);
15
16         System.out.print("Masukan Suatu Nilai = ");
17         a = input.next();
18
19         try
20         {
21             b = Integer.parseInt(a);
22             System.out.println("Nilai yang anda input adalah Benar, berupa
Angka");
23         }
24         catch (NumberFormatException e)
25         {
26             System.out.println("Oops! Salah, itu sih " + e.getMessage());
27         }
28         catch (Exception e)
29         {
30             System.out.println("Input/output error");
31             System.out.println(e.getMessage());
32         }
33         catch (Throwable e)
34         {
35             System.out.println("Program interrupted");
36             System.out.println(e.getMessage());
37         }
38     }
39 }
```


Pada program diatas, digunakan untuk melakukan pengecekan, apakah nilai yang diinputkan tersebut berupa angka atau String. Terlihat pada catch terakhir terdapat class Throwable, yang bertujuan untuk menangkap eksepsi lain yang tidak tertangkap pada eksepsi yang diletakkan atasnya. Berikut hasil eksekusi program jika nilai yang dimasukan benar.



Gambar 11.5. Hasil Pemasukan Data Benar

Berikut hasil eksekusi program jika nilai yang dimasukan salah.



Gambar 11.6. Hasil Pemasukan Data Salah

11.5 Melempar Eksepsi

Untuk keperluan melempar eksepsi ini java menyediakan pernyataan throw, untuk keperluan ini. Berikut bentuk penulisan pernyataan throw.

```
throw variabelObjek;
```

Variabel Objek adalah variabel objek yang berupa suatu class eksepsi. Pada saat penggunaan throw, maka pembacaannya dengan menggunakan try dan catch. Berikut contoh penerapannya:

```
1  /* -----
2     Nama File : Lat1104.java
3     Author   : Frieyadie
4  ----- */
5
6  import java.util.Scanner;
7
8  class Lat1104
```

```
9 {
10     public static void main(String[] args)
11     {
12         String a;
13         int b;
14         Scanner input = new Scanner(System.in);
15
16         System.out.print("Masukkan Suatu Nilai = ");
17         a = input.next();
18
19         try
20         {
21             b = Integer.parseInt(a);
22             System.out.println("Nilai yang anda input Sudah Benar ");
23         }
24         catch (NumberFormatException nf)
25         {
26             nf = new NumberFormatException();
27             throw(nf);
28         }
29     }
30 }
```

Penjelasan Program :

Pada baris program 21, terdapat pernyataan :

```
b = Integer.parseInt(a);
```

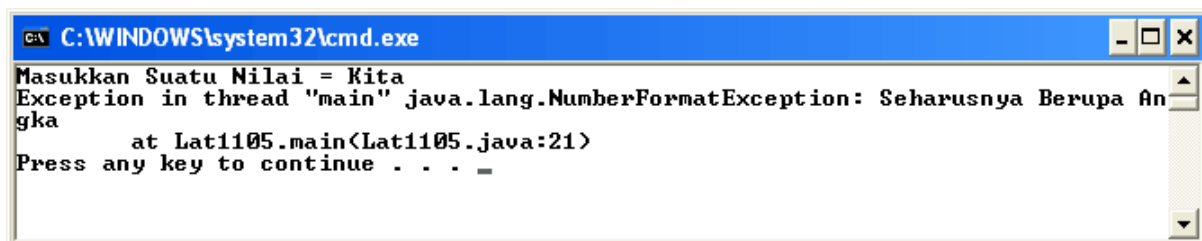
Pernyataan tersebut adalah menyatakan penkonversiaan nilai String ke integer. Eksepsi yang ada pada catch yaitu `NumberFormatException`.

Pada baris 21, terdapat pernyataan :

```
nf = new NumberFormatException();
```

Pernyataan tersebut untuk mengganti keterangan eksepsi, dan pernyataan `throw(nf);`

Berguna untuk melakukan melemparkan eksepsi. Hasil dari program `Lat1104.java` diatas, seperti gambar dibawah ini, jika nilai yang dimasukan berupa string.



Gambar 11.7. Hasil Pemasukan Berupa Data String

11.6 Penggunaan throws Pada Method

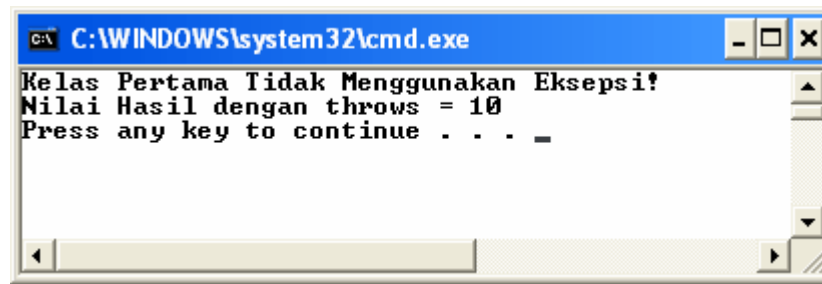
Kata kunci throws, digunakan untuk menginformasikan, kemungkinan method akan terjadi eksepsi. Berikut bentuk penulisan penggunaan throws pada method.

```
public namaMethod(parameter) throws jenisException
{
    Blok method dengan eksepsi
}
```

Beriku contoh program pennggunaan throws pada mehtod.

```
1  /* -----
2     Nama File : Lat1105.java
3     Author   : Frieyadie
4     ----- */
5
6  class KelasAja
7  {
8      public void MetodePertama()
9      {
10         System.out.println("Kelas Pertama Tidak Menggunakan Eksepsi!");
11     }
12
13     public int MetodeKedua(int f) throws NumberFormatException
14     {
15         return f + 5;
16     }
17 }
18
19 class Lat1105
20 {
21     public static void main(String[] args)
22     {
23         int hasil=0;
24         KelasAja ObjekKelas = new KelasAja();
25
26         ObjekKelas.MetodePertama();
27
28         //penggunaan metode dengan eksepsi
29         try
30         {
31             hasil = ObjekKelas.MetodeKedua(5);
32         }
33         catch (NumberFormatException e)
34         {
35             System.out.println("Adanya Kesalahan");
36         }
37
38         System.out.println("Nilai Hasil dengan throws = "+hasil);
39     }
40 }
```

Hasil dari program Lat1105.java diatas, seperti gambar dibawah ini:



Gambar 11.8. Hasil program Lat1105.java