Kampus Semangat PAGI

DASAR-DASAR PEMOGRAMAN JAVA
UNTUK PEMULA MENGGUNAKAN
NETBEANS

(Digunakan di Lingkungan Universitas PGRI Adi Buana Surabaya)

Penulis:

Dwi Hastuti

Yusril Arief

Semangat PAGI PEDULI AMANAH GIGIH INOVATIF

DASAR-DASAR PEMROGRAMAN JAVA UNTUK PEMULA MENGGUNAKAN NETBEANS

Oleh:
Dwi Hastuti
Yusril Arief

Adi Buana University Press
Universitas PGRI Adi Buana Surabaya
Jl. Dukuh Menanggal XII-4, Surabaya 60234
Telp. (031) 8281181

Unipa Surabaya

DASAR-DASAR PEMROGRAMAN JAVA
UNTUK PEMULA MENGGUNAKAN NETBEANS

Oleh:
Dwi Hastuti
Yusril Arief
Editor
Design Sampul : Yitno Utomo
Lay out : Adi Buana University Press

vi + 101 hlm, 14,8 x 21 cm Cetakan Pertama, Juli 2020

ISBN: 978-623-416-011-6

Penerbit :
Adi Buana University Press Universitas
PGRI Adi Buana Surabaya
Jl. Ngagel Dadi III-B/ 37 Surabaya, 60245
Telp : 031- 5041097
Fax : 031 – 5042804
Website : www. unipasby.ac.id
Email : unipasby@gmail.com

Unipa Surabaya

BUKU AJAR
SUDAH DIVALIDASI

ttd

Mengesahkan Wakil Rektor I
Universitas PGRI Adi Buana Surabaya

Unipa Surabaya

PRAKATA

Puji dan syukur penulis ucapkan kepada Tuhan Yang Maha Esa karena dapat menyelesaikan buku yang berjudul "**Dasar-Dasar Pemrograman Java Untuk Pemula Menggunakan Netbeans**". Buku ini dapat dimanfaatkan untuk pemula dalam memelajari dasar-dasar pemrograman Java. Masing-masing materi disajikan secara sistematis disertai contoh agar lebih mudah dipahami dan dipelajari. Tak lupa penulis mengucapkan terima kasih yang sebesarnya kepada pihak-pihak yang telah ikut membantu dan memberi masukan sehingga buku ini siap untuk dimanfaatkan oleh masyarakat. Saran yang membangun, sangat penulis harapkan demi perbaikan buku untuk edisi selanjutnya agar semakin lebih baik. Salam literasi.

Penulis

Unipa Surabaya

DAFTAR ISI

HALAMAN JUDUL
HALAMAN HAK CIPTA iii
HALAMAN PENGESAHAN v
PRAKATA vii
DAFTAR ISI ix
BAB 1 PERSIAPAN BELAJAR JAVA 1
Pengenalan Java 1
Integrated Development Environment 3
Instalasi Java 5
Instalasi Netbeans 9
BAB 2 ALGORITMA 13
Pengertian Algoritma 13
BAB 3 PROGRAM JAVA PERTAMA 19
Mengenal Netbeans 19
Run File 25
Shift-F4 25
Mengenal Class 26
Mengenal Main Class 28
Mengenal Blok Program 28
BAB 4 TIPE DATA DAN VARIABEL 41
Variabel 41
Tipe Data 41
Byte 42
Short 42
Int 42
Long 43

Cara Membuat Variabel 44
Konversi Data 48
BAB 5 OPERATOR DI JAVA 53
Operator Bilangan Bulat 53
Operator Bilangan Bulat Unary 53
Operator Bilangan Bulat Binary 55
Operator Bilangan Bulat Relasional 57
Operator Boolean 61
BAB 6 PERCABANGAN 63
Percabangan atau Branching 63
BAB 7 ARRAY 69
Pengenalan Array 69
Array Multi Dimensi 72
BAB 8 PENGULANGAN 75
Pengenalan Pengulangan (Looping) 75
For Loop Bersarang 76
While 81
Do While 84
BAB 9 MENGENAL CLASS 87
Class 87
Atribut Class 90
Prosedur atau Function 92
Prosedur 94
Data Diri Penulis 97

BAB 1 Persiapan Belajar Java

Pengenalan Java

Pencipta bahasa pemrograman Java adalah James Gosling. Beliau menciptakan bahasa ini saat masih bergabung di Sun Microsystem yang saat ini sudah menjadi bagaian dari Oracle dan dirilis tahun 1995. Bahasa Java dijalankan menggunakan JVM atau Java Virtual Machine, hal ini yang menyebabkan Java lebih fleksibel dan bisa dijlankan di System operasi Windows, Mac OS atau Linux. Sesuai dengan slogannya yaitu "*Write once, Run anywhere*".

Karena kelebihanya ini, pembuat aplikasi hanya perlu menulis sekali programnya dan otomatsi akan dikompilasi oleh JVM atau Java Virtual Machine. Kelebihan kedua adalah Object Oriented Programing atau disingkat OOP. Ini adalah sebuah konsep pemrograman yang berorientasi pada objek. Tujuan dari konsep ini untuk mencegah masalah pemrograman menjadi beberapa bagian kecil. Kelebihan ketiga adalah lengkapnya library atau perpustakaan (kumpulan program yang sudah dibuat untuk digunakan kembali). Hal ini memudahkan Pengguna untuk membangun aplikasinya. Kelebihan keempat adalah memiliki sintask seperti bahasa pemrograman C++ sehingga menarik banyak pengguna C++ untuk pindah ke Java. Kelebihan ketiga adalah garbage Collection. Hal ini berfungsi untuk pengaturan memori sehingga para pembuat program tidak perlu melakukan

pengaturan memori secara langsung. Ada tiga jenis pemrograman Java, antara lain :

a. Java Standart Edition

Java Standart Edition (J2SE) atau yang biasa disebut Java SE yang secara umum banyak digunakan pemrograman Bahasa Java. Java SE menggunakan JVM atau Java Virtual Machine untuk menjalankan program bersama librarynya. Java SE biasanya berupa aplikasi berbentuk text, aplikasi berbasis window atau Graphical User Interface (GUI) dan aplikasi – aplikasi berskala kecil.

b. Java Enterprise Edition

Java Enterprise Edition (JEE) banyak dimanfaatkan untuk membuat aplikasi berskala besar. Komponen JEE terdiri dari :

- Client dan Applet adalah komponen yang berjalan di sisi Client.
- Web berupa Servlet atau Java Server Pages yang berjalan di server.
- Enterprise JavaBeans k(EJB) adalah teknologi yang digunakan untuk mengembangkan komponen di sisi server. EJB

c. Java Platform Micro Edition (J2ME) adalah teknologi Java yang banyak digunakan untuk membuat aplikasi mobile atau aplikasi yang bersifat embedded, biasanya berupa aplikasi handphone atau PDA. Namun sekarang sudah mulai ditinggalkan karena aplikasi mobile sudah berpindah menjadi Android atau

IOS yang mempunyai alatnya sendiri contohnya Android SDK. Dengan seluruh kemampuan yang dimiliki Java kita dapat membuat :

- Aplikasi sederhana berbentuk konsol.
- Aplikasi window atau Graphical User Interface.
- Aplikasi yang dijalankan di browser atau berupa website menggunakan JSP
- Aplikasi berupa Applet yang dijalankan di browser.
- Aplikasi berskala enterprise dengan teknologi EJB atau Entity Java Bean, basis data (JDBC) dan mailing.

Integrated Development Environment

Integrated Development Environment atau disingkat IDE adalah sebuah aplikasi komputer yang digunakan untuk membuat sebuah perangkat lunak. IDE memiliki banyak fitur agar pembuat perangkat lunak menjadi lebih mudah dalam membuat perangkat lunaknya. IDE untuk Java banyak sekali macamnya mulai dari yang gratis dan berbayar dengan system langganan. Berikut di bawah ini adalah macam – macam IDE untuk Java :

a. Netbeans

Netbeans adalah salah satu IDE yang popular untuk membuat aplikasi Java. IDE ini sangat lengkap fiturnya dan bersifat open source. Netbeans tidak hanya mendukung Java tetapi mendukung untuk membuat perangkat lunak dengan bahasa lainya seperti

PHP, C++, HTML, Javascript dan lain lain. Netbeans juga sangat terkenal untuk mempermudah membuat aplikasi berbasis Graphical User Interface atau aplikasi desktop karean disediakan GUI editor yang membuat User cukup *drag and drop* komponen ke kanvas. Selain itu Netbeans juga multi-platform yang artinya bias dipasang di Windows, Mac atau Linux. Saat ini Netbeans sudah mencapai versi 8.2. Netbeans bias kalian unduh di https://netbeans.org/downloads/ .

b. Eclipse

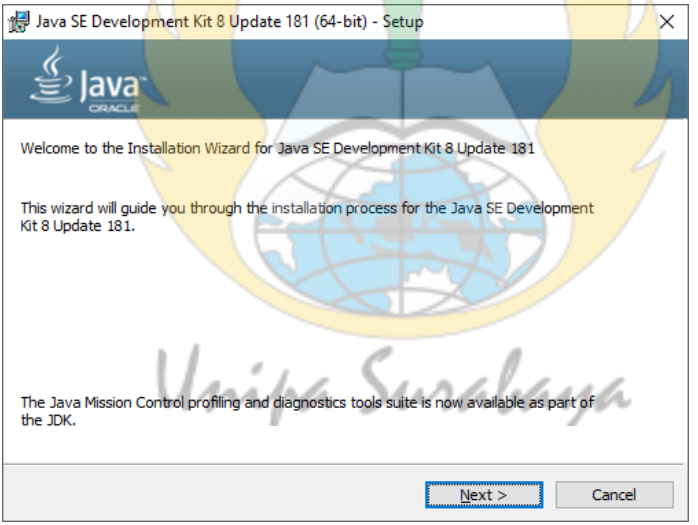
Eclipse juga salah satu IDE yang gratis dan didukung oleh banyak komunitas. Eclipse juga multi-platform sama seperti Netbeans dan mendukung banyak aplikasi pemrograman seperti PHP, C++, Cobol, Phyton, Perl dan lain lain. Eclipse saat ini juga menjadi salah satu IDE favorit dikarenakan gratis dan open source yang berarti setiap orang boleh melihat kode pemrograman perangkat lunak ini. Eclipse bias didapatkan di halaman http://www.eclipse.org/downloads/ .

c. Intelij IDEA

Intelij IDEA adalah IDE yang dibuat oleh Jetbrains. Intelij IDEA terdapat 2 versi yaitu versi berbayar dan gratis. Untuk versi gratis memiliki fitur yang terbatas hanya bias digunakan untuk membuat program Android dan aplikasi Desktop. Jika ingin mencoba Intelij IDEA bias di unduh di https://www.jetbrains.com/idea/download/#section=linux.

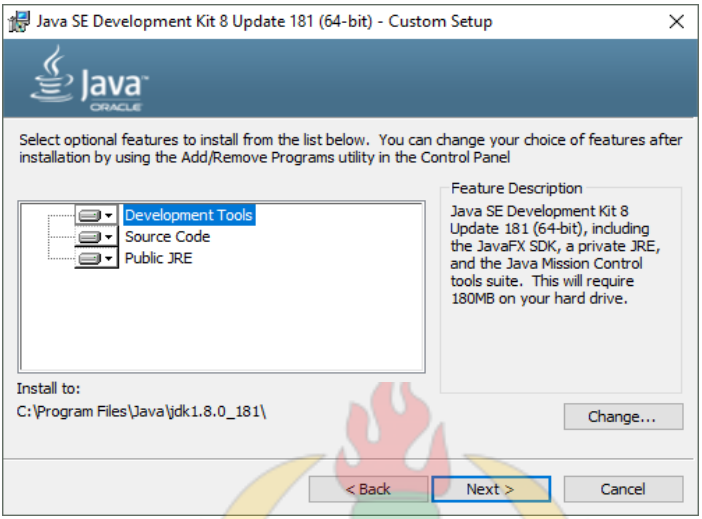
Instalasi Java

Langkah pertama sebelum memasang Java ke sistem adalah mengunduh file instalasi Java di: http://www.oracle.com/technetwork/java/javase/downloads/jdk10-downloads-4416644.html. Saat buku ini dibuat, versi Java sudah mencapai Java SE Development Kit versi 8. Sebelum mengunduh file instalasi, Kamu harus menyetujui persetujuan lisensi Oracle dengan memilih opsi Accept License Agreement setelah itu file dapat diunduh. Setelah file selesai di unduh, klik dua kali pada file tersebut.



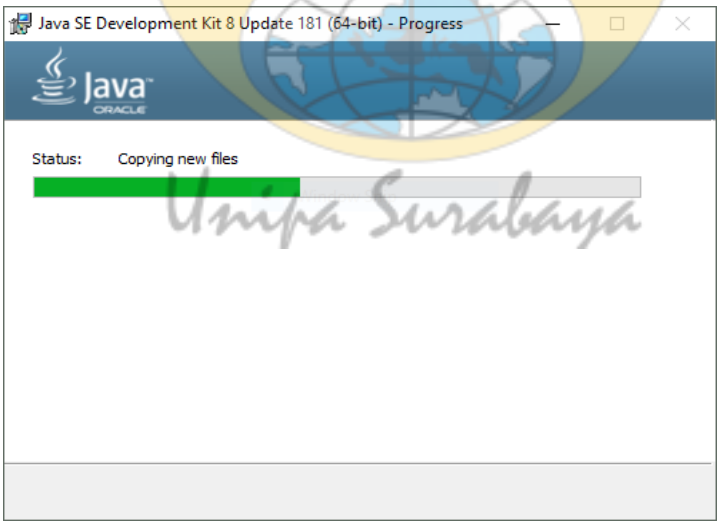
Gambar 1

Tekan Next untuk untuk melanjutkan.



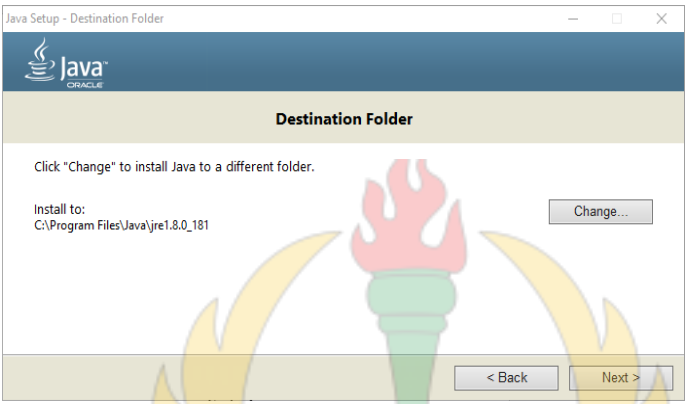
Gambar 2

Tahap berikutnya adalah menentukan lokasi di mana kita akan meletakan sistem Java. Saya sarankan untuk mengikuti *default* saja yaitu di lokasi program file. Klik Next.



Gambar 3

Java mulai memasang JDK ke sistem komputer, tunggu sampai proses ini selesai. Jika proses memasang JDK selesai, proses selanjutnya adalah memasang Java Runtime Environment atau disingkat JRE.

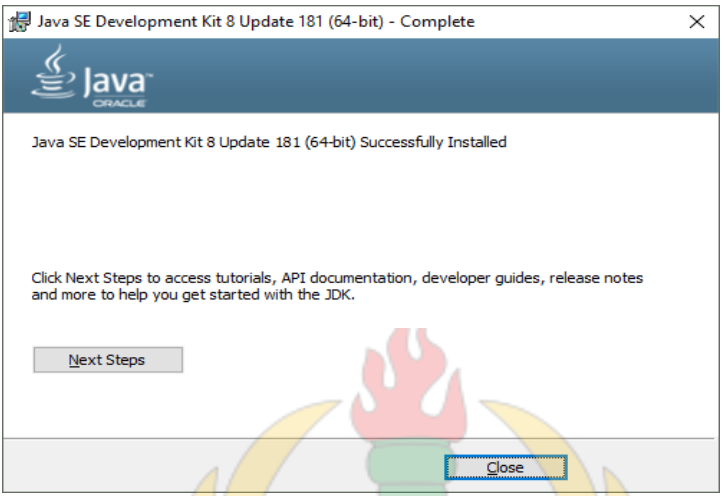


Gambar 4

Sama seperti langkah sebelumnya, tahap ini menentukan lokasi untuk memasang JRE, Saya sarankan ikuti default saja. Tekan tombol Next.

Gambar 5

Tahap berikutnya Java memulai untuk memasang JRE ke sistem komputer, tunggu sampai selesai.

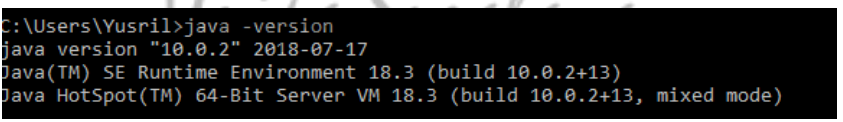


Gambar 6

Tekan Close untuk mennyelesaikan proses instalasi Java. Untuk memastikan apakah Java sudah terpasang dengan benar ke sistem, buka command prompt kemudian ketikan perintah :

```
Java -version
```

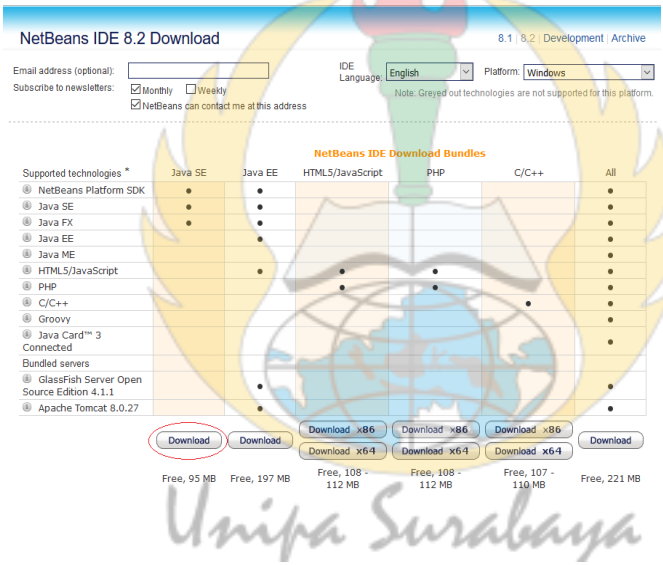
Output dari perintah di atas adalah:



Gambar 7

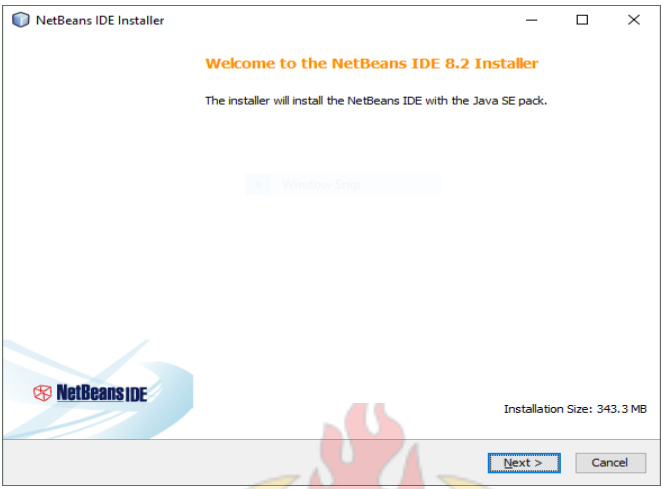
Instalasi Netbeans

Pada saat buku ini dibuat, versi Netbeans sudah mencapai versi 8.2. File installer dapat di unduh di halaman https://netbeans.org/downloads/. Saat link tersebut dibuka, terdapat 6 pilihan file yang berbeda – beda ukuranya, Saya sarankan untuk memilih Java SE karena pada buku ini sebagian besar membahas Java SE. Fitur – fitur lainya dapat diaktifkan nanti.



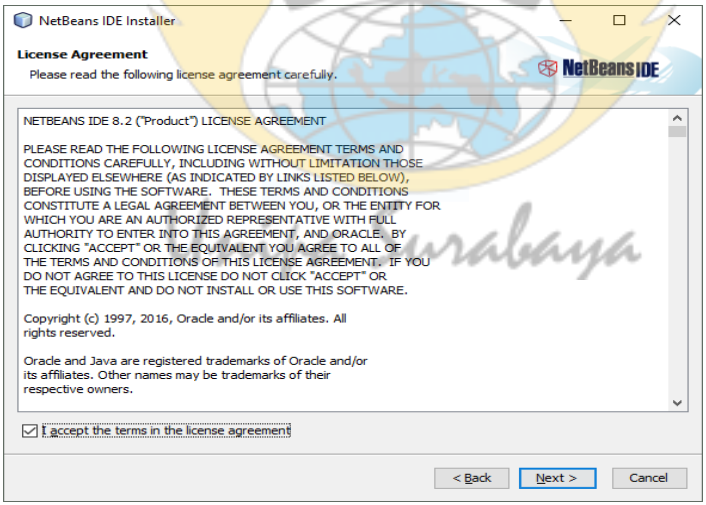
Gambar 8

Setelah file installer selesai diunduh, jalankan file tersebut dan muncul jendela **Welcome IDE Installer**, klik Next.



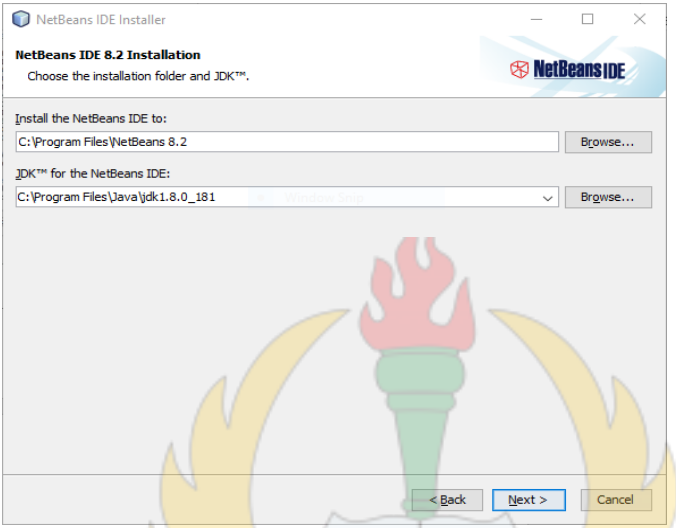
Gambar 9

Langkah selanjutnya, kamu harus menyetujui perarturan lisensi dengan beri tanda centang pada opsi **I accept the terms in the license agreement**.



Gambar 10

Langkah selanjutnya menentukan lokasi Aplikasi Netbeans dan menentukan letak JDK yang baru diinstall pada sub bab 3 sebelumnya. Disarankan biarkan default saja. Tekan tombol Next.



Gambar 11

Langkah selanjutnya, hilangkan tanda centang di opsi **Check for Updates** agar saat proses memasang, Netbeans tidak mencari Update dimana ini membutuhkan koneksi internet dan waktu lebih lama. Tekan tombol Install.

Unipa Surabaya

Bab 2 Algoritma

Pengertian Algoritma

Algoritma adalah sebuah urutan atau langkah – langkah logis untuk mempermudah menyelesaikan suatu masalah. Algoritma berasal dari nama Abu Ja'far Muhammad Ibn Musa Al-Khwarizmi (770 - 840). Di dalam literatur barat beliau dikenal dengan sebutan Algorizm. Contoh sederhana Algoritma dalam kehidupan sehari – hari adalah memasak nasi. Langkah – langkah memasak nasi adalah sebagai berikut :

a. Ambil beras secukupnya
b. Cuci beras sampai bersih
c. Masukan beras ke Rice Cooker
d. Tunggu beras matang.

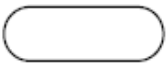
Di dalam dunia komputer membuat Algoritma sangat penting dan diperlukan untuk menyelesaikan berbagai masalah pemrograman terutama yang berhubungan dengan perhitungan. Algoritma tentu saja bisa berbeda dari setiap orang dan hal tersebut adalah wajar selama hasilnya sama atau sesuai tujuan dari memecahkan masalah. Dengan Algoritma kita bisa lebih mudah mengkonversi masalah dari dunia nyata ke Bahasa pemrograman. Cara membuat algoritma bisa dalam bentuk tulisan atau gambar. Jika menggunakan tulisan kita dapat membuat Pseudocode atau dengan Bahasa tertentu. Pseudocode adalah kode yang mirip dengan Bahasa pemrograman sebenarnya. Jika kita membuatnya

dengan gambar maka secara umum kita bisa gunakan flowchart. Berikut ini adalah symbol – symbol flowchart :

a. Simbol input (masukan)

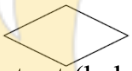
b. Simbol proses

c. Simbol mulai dan selesai



d. Simbol garis alur

e. Simbol percabangan



f. Simbol output (keluaran)

1. Cara membuat Algoritma

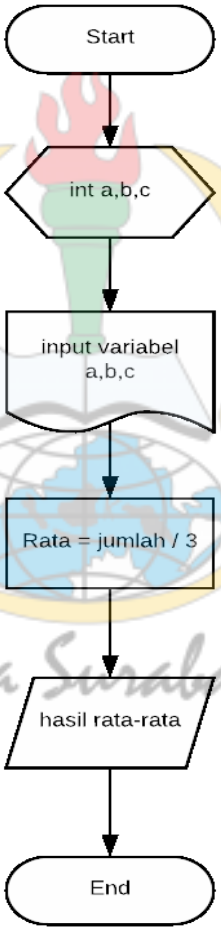
Berikut ini adalah cara – cara untuk membuat Algoritma menghitung rata – rata dari tiga buah bilangan :

a. Algoritma Tulisan (bahasa) :
   - Baca bilangan a, b dan c.
   - Jumlahkan ketiga bilangan tersebut.
   - Bagi jumlah bilangan dengan 3
   - Tulis hasilnya.

b. Algoritma Pseudocode :
   - Int a,b,c,hasil;

- a = 5 , b = 5 , c = 5
- Hasil = (a + b + c) / 3
- Tampilkan Hasil

c. Flowchart



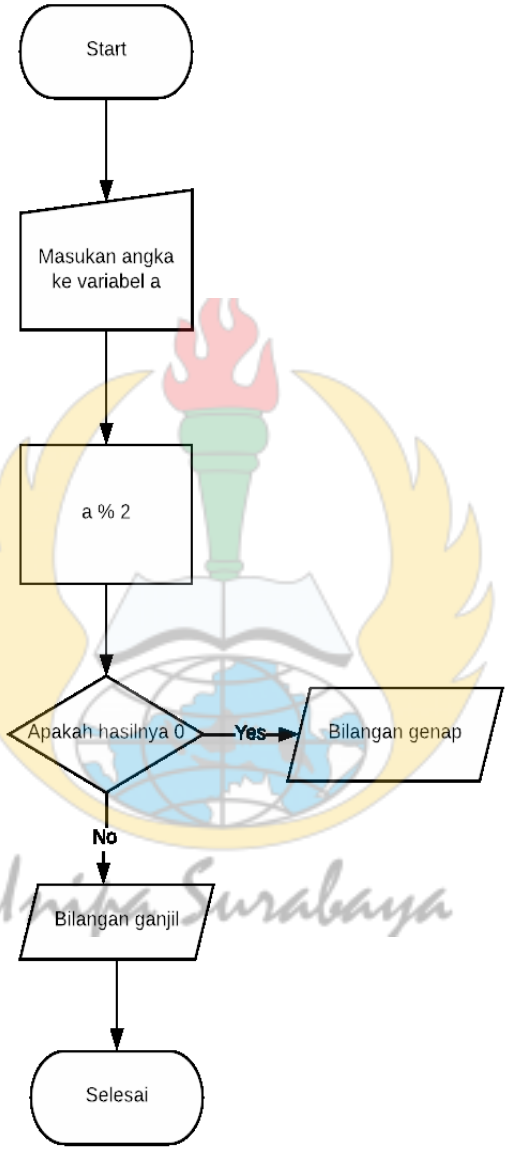
Gambar 14

Dari contoh di atas kita sudah belajar membuat sebuah algoritma dari tulisan sampai flowchart satu arah. Bagaimana jika sebuah Algoritma membutuhkan percabangan ?. Percabangan bisa terjadi jika program memiliki lebih dari satu alur. Contohnya algoritma menentukan ganjil atau genap :

a. Algoritma tulisan (bahasa) :
    - Masukan inputan berupa angka.
    - Apakah inputan angka habis dibagi 2 ? Jika iya maka genap jika tidak maka ganjil.

b. Algoritma Pseudocode :
    - Int a,hasil.
    - a = 10
    - hasil = a % 2
    - if hasil = 0 then genap else if hasil = 1 then ganjil.
    - Cetak variable hasil.

c. Flowchart



Gambar 15

Dalam membuat sebuah algoritma hal yang perlu dicatat dan diingat adalah setiap perintah harus mempunyai satu tafsiran dan langkah – langkahnya harus jelas. Dari bab ini kita sudah belajar membuat sebuah algoritma untuk menyelesaikan masalah. Konsep pemrograman sebenarnya bukan membuat sebuah kode program tapi membuat algoritma. Setelah kita bisa membuat algoritma dengan baik baru lakukan konversi ke Bahasa pemrograman.

2. Latihan

Buatlah algoritma untuk menyelesaikan masalah berikut :

a. Mencari nilai terbesar dari 2 bilangan.
b. Menghitung luas persegi panjang.

BAB 3 Program Java Pertama

Mengenal Netbeans

Pada sub bab sebelumnya kita sudah mencoba untuk memasang Netbeans di komputer, selanjutnya pada sub bab ini kita akan mengenal sedikit mengenai Netbeans. Saya tidak akan membahas semua fitur Netbeans karena akan membutuhkan waktu yang lama namun Saya akan membahas fitur yang paling sering dipakai.

a. Jendela IDE Netbeans

Tampilan pertama kali saat Netbeans di buka adalah seperti gambar di bawah ini.

Gambar 16

Keterangan dari gambar tersebut adalah sebagai berikut :

Tabel 1

Jendela	Keterangan
Learn & Discover	Jendela yang berisi kumpulan proyek demo.

Jendela	Keterangan
My Netbeans	Jendela yang berisi informasi proyek terakhir yang kamu buat atau dibuka. Selain itu, di dalam jendela ini kamu juga bisa menambahkan fitur dengan menekan tombol install plugin.
What's New	Jendela ini berisi informasi dari internet mengenai teknologi Java itu sendiri. Selain itu, jendela ini juga menyuguhkan tutorial seputar Java yang diambil dari internet.
Projects	Tab Projects berisi kumpulan proyek – proyek yang sedang kamu buka
Files	Tab Files sama seperti Tab Projects. Tab Files membuka proyek dalam bentuk tampilan direktori
Service	Tab Service berisi kumpulan service dari database atau server web. Tab ini berguna untuk administrasi server database atau server tomcat

Jendela	Keterangan
Output	Jendela ini berguna untuk menampilkan log dari project yang sedang kamu jalankan. Melihat pesan error dan informasi – informasi lainya.
Search Result	Jendela ini berguna untuk melihat file – file yang ditampilkan sesuai pencarian.
Versioning Output	Jendela ini beguna untuk melihat log dari versioning seperti Git atau Subversion.

b. Shortcut dan Navigasi

Di dalam Netbeans terdapat banyak jendela. Tidak semua jendela terbuka pada saat yang sama. Namun demikian pada saat bersamaan dapat terbuka lebih dari satu jendela. Untuk menampilkan atau pindah antar jendela, Anda dapat memakai klik mouse. Jika ingin memakai keyboard, tersedia tombol-tombol shortcut yang dapat dipakai untuk navigasi.

Tabel 2

Shortcut Key	Aksi
Alt-Shift-2	Membuka jendela debugger Watches.
Alt-Shift-3	Membuka jendela debugger Call Stack.
Alt-Shift-4	Membuka jendela debugger Classes.

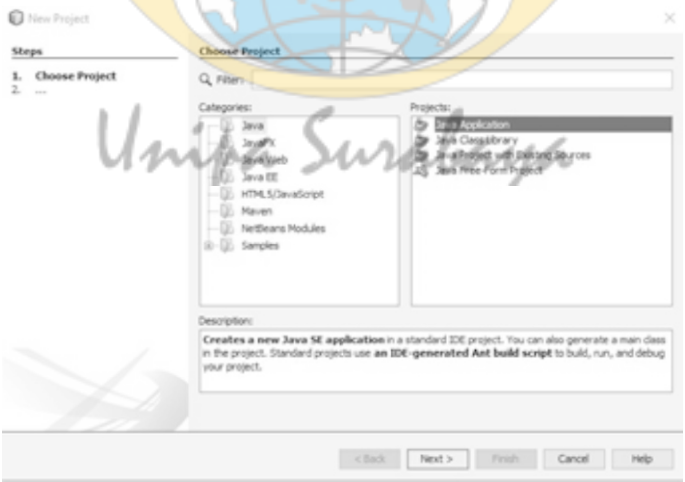
Shortcut Key	Aksi
Alt-Shift-5	Membuka jendela debugger Breakpoints.
Alt-Shift-6	Membuka jendela debugger Sessions.
Alt-Shift-7	Membuka jendela debgger Threads.
Alt-Shift-8	Membuka jendela debugger Sources.
Ctrl-Tab	Memilih salah satu jendela yang sedang dibuka dalam Source Editor, dengan urutan berdasarkan jendela yang terakhir dibuka.
Shift-Escape	Melakukan maximize terhadap source editor atau terhadap jendela yang sedang dibuka. Untuk mengembalikan keadaan seperti semula tekan shortcut ini lagi.
Ctrl-F4	Menutup tab yang sedang aktif di dalam jendela yang aktif. Jika jendela tidak memiliki tab, keseluruhan jendela akan ditutup.
Ctrl-Shift-F4	Menutup semua dokumen yang sedang terbuka di source editor.
Shift-F4	Membuka dialog Documents. Dialog ini dapat dipakai untuk berpindah,

Shortcut Key	Aksi
	menyimpan atau menutup dokumen atau kelompok dokumen yang sedang terbuka dalam Source Editor.
Alt-right	Menampilkan tab berikutnya dari jendela yang aktif.
Alt-left	Menampilkan tab sebelumnya dari jendela yang aktif.

c. Membuat Proyek Baru

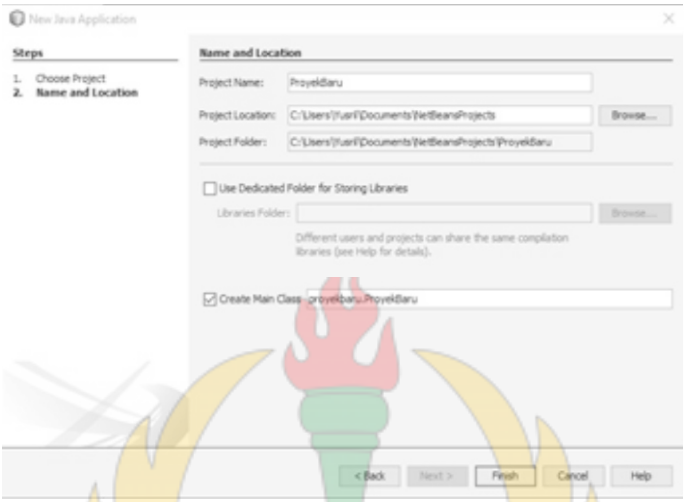
Ketika membuat sebuah proyek akan banyak sekali langkah-langkahnya, Mulai dari membuat proyek baru sampai melakukan deployment. Berikut ini cara membuat proyek di Netbeans :

- Dari jendela Netbeans klik File -> New Project atau tekan tombol shortcut **Ctrl-Shift-N**. Kemudian akan muncul jedela seperti berikut. Pilih **Java Application** kemudian tekan tombol **Next**.



Gambar 17

- Beri nama **ProyekBaru** sebagai contoh. Kemudian tekan tombol **Finish**. Secara default proyek akan disimpan di folder **NetBeansProjects**



Gambar 18

- Jendela Source Editor akan terbuka seperti gambar dibawah ini.

Gambar 19

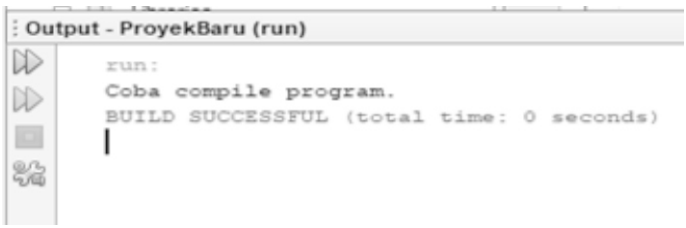
d. Menjalankan Program Dan Membaca Pesan Log

Setelah Kamu menulis beberapa baris kode, tentu saja Kamu harus mencoba menjalankan baris kode tersebut untuk mengetahui hasilnya error atau tidak. Berikut ini cara compile program atau menjalankan program di Netbeans :

- Cara pertama di dalam Source editor klik kanan kemudian pilih **Run File**.
- Cara kedua pilih Proyek **ProyekBaru** di tab **Projects** kemudian klik kanan dan pilih **Run**.
- Cara ketiga klik tombol **Run Project** yang letaknya di navigasi bar.

Gambar 20

- Cara keempat yang paling mudah menggunakan shortcut **Shift-F4**.
- Jika Program berhasil di compile dan dijalankan oleh Netbeans maka akan keluar output seperti gambar di bawah ini.



Gambar 21

- Di dalam Output juga akan menunjukan letak error jika program gagal di compile atau gagal di jalankan. Berikut ini contohnya.

Gambar 22

Mengenal Class

Class adalah bagian utama dari pemrograman Java dan Class merupakan cetak biru pemrograman Java untuk menciptakan sebuah Objek. Di dalam class terdapat satu Constructor, beberapa variabel dan beberapa prosedur. Bentuk umum dari sebuah Class adalah.

public class Anak extends Induk implements Interface {//masukan kode di sini.}

Public adalah Modifier. Sebuah modifier bisa berbentuk default, public, protected, private, final. Extend adalah kata kunci untuk menandakan sebuah turunan dari induk Class dan Implements adalah kata kunci untuk implement prosedur dari Class Interface. Di bawah ini adalah keterangan dari masing - masing bagian Class:

a. Modifier berfungsi untuk mengatur hubungan dengan Class lain. Memberikan dampak pada Class, Method dan variabel. Macam - macam bentuk akses modifier dan penjelasan fungsinya:
   - Default
     Anggota Class dengan default access dapat dilihat Class lain yang sepaket. Tidak ada kata kunci khusus untuk mendeklarasikan modifier Default.
   - Public
     Modifier Public membuat variabel dan method dapat diakses

oleh siapapun baik di dalam atau di luar Class. Hal ini berarti anggota Public akan tampak dan dapat diakses oleh sembarang objek lain.

- Protected

  Modifier Protected membuat anggota Class hanya dapat diakses oleh method di Class itu dan Subclass dari turunan Class itu. Ini berarti anggota Protected hanya terbatas pada Class dan Subclassnya.

- Private

  Modifier Private membuat anggota hanya dapat diakses oleh Class itu sendiri. Modifier Private paling terbatas fungsinya.

- Final

  Modifier Final artinya tidak bisa dimodifikasi atau diubah nilainya. Harus diisi saat dideklarasikan. Setiap anggota yang diberi modifier ini tidak dapat diubah - ubah.

- Static

  Modifier Static artinya variabel diasosiasikan dengan Class dan dipakai Bersama objek-objek kelas itu. Variabel Static disebut variabel Class bukan pada satu objek. Serupa dengan data statis, kita juga dapat membuat method yang hanya bertindak pada Class, jadi hanya dapat mengakses variabel Static saja.

b. Penamaan Class

   Aturan untuk memberi nama Class adalah nama Class harus dimulai dengan huruf besar. Nama Class juga harus sama seperti nama filenya, contoh jika nama Class Hallo maka penamaan file harus Hallo.java.

c. Extend Kelas Induk

   Extend adalah pewarisan yang digunakan untuk mewariskan sifat-sifat dan method dari Parent Class atau Kelas Induk yang dipanggil oleh Child Class atau Kelas Anak.

d. Implement Interface
   Kelas tersebut menggunakan method dari Class Interface.

Mengenal Main Class

Main Class adalah adalah sebuah Class yang akan dijalankan pertama kali saat program dieksekusi. Sebuah Main Class memiliki satu method yang menggunakan fungsi Main. Bentuk umum method main adalah :

public static void main (String[] args) {//Sebuah perintah}

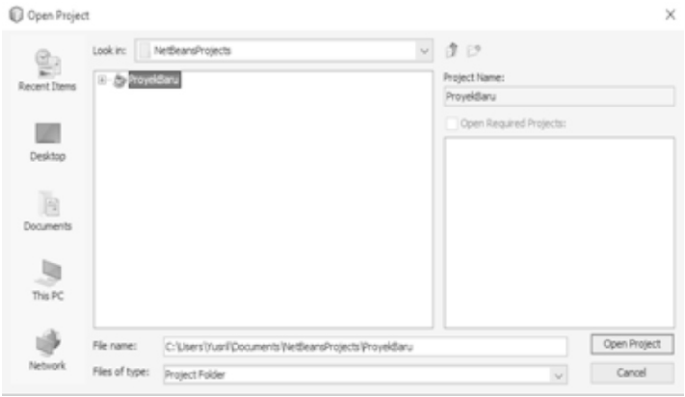
Fungsi Main harus ditetapkan sebagai berikut :

a. Public berarti method dapat dipanggil dari manapun di dalam atau di luar Class
b. Static berarti sama untuk seluruh instan dari Class.
c. Void berarti berbentuk method yang tidak mengembalikan nilai apapun ketika dieksekusi.
   Argumen args[ ] adalah array objek string.

Mengenal Blok Program

Blok program adalah tempat dimana kita menaruh sebuah printah program untuk dieksekusi. Blok program di dalam Java berbentuk { } dan kita dapat menaruh perintah di dalam tanda tersebut.

1. Halo Dunia (Hello World)
   Pada bab ini kita akan mencoba membuat sebuah program sederhana yaitu menampilkan kalimat Halo Dunia. Langkah-langkahnya adalah sebagai berikut :
   - Buka Proyek ProyekBaru yang telah kita buat pada sub bab sebelumnya. Pilih File -> Open Project atau dengan menekan shortcut Ctrl-Shift-O.



Gambar 23

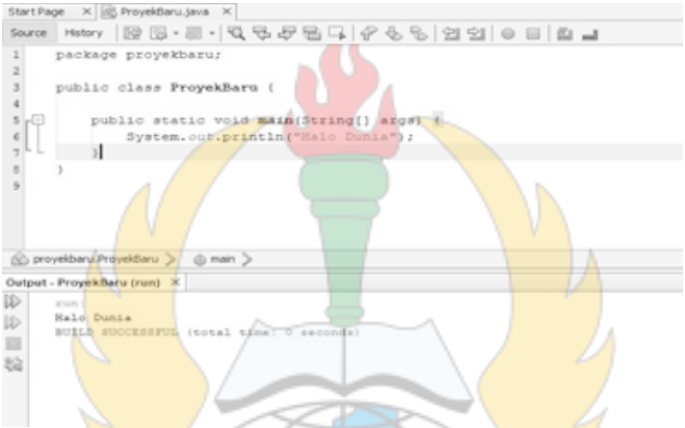
- Pilih dan buka main **Class ProjectBaru.java.**

Gambar 24

- Langkah selanjutnya adalah ketikan kode program seperti di bawah ini :

```
package proyekbaru;

public class ProyekBaru {public static void main(String[]
args) { System.out.println("Halo Dunia");}}
```

Ekspresi di atas memanggil fungsi **println()** di objek **out** di kelas **System** yang akan mencetak kalimat **Halo Dunia** di dalam tab output atau konsol. Hal yang perlu diingat di dalam kode Java adalah tanda **semicolon ;**. Tanda ini menandakan akhir dari sebuah baris program. Coba compile dan jalankan main class, jika tidak error maka hasilnya akan seperti di bawah ini.



Gambar 25

Java menyediakan dua perintah untuk menampilkan data ke layar.

- Perintah System.out.println(“Halo Dunia”) akan mencetak kalimat Halo Dunia dan data berikutnya akan dicetak pada baris selanjutnya.
- Perintah System.out.print(“Halo Dunia”) akan mencetak kalimat Halo Dunia dan data berikutnya akan dicetak setelah data sebelumnya.

Jika data yang ingin dicetak lebih dari satu maka gunakan karakter +. Berikut ini contoh output.

Penulisan	Hasil
System.out.println("Halo"+" "+"Dunia");	Halo Dunia
System.out.println(2+" "+5);	2 5
System.out.println(2+5);	7
System.out.println("Angka"+" "+5);	Angka 5
System.out.println("2 + 5"+" = "+2+5);	2 + 5 = 25
System.out.println("2 + 5"+" = "+(2+5));	2 + 5 = 7

2. Membaca Inputan User

   Di dalam Java terdapat 2 cara untuk menerima nilai masukan dari User, antara lain :

   a. Class Scanner

      Scanner adalah Class java dari **Java.util.Scanner** yang digunakan untuk mengambil inputan dari User. Berikut di bawah ini contoh menggunakan Class Scanner :

```
public static void main(String[]args) {
    String namaDepan = "";
    String namaBelakang = "";
    Scanner input = new Scanner(System.in);
    System.out.println("Nama Depan : ");
    //membaca inputan user
    namaDepan = input.next();
    System.out.println("Nama Belakang : ");
    //membaca inputan user
    namaBelakang = input.next();
    System.out.println("Nama Saya adalah :");
    System.out.println(namaDepan + " " + namaBelakang);
    }
```

Pada saat di jalankan, program di atas akan memberikan output seperti di bawah ini:

```
Nama Depan :
Yusril
Nama Belakang :
Arief
Nama Saya adalah :
Yusril Arief
BUILD SUCCESSFUL (total time: 8 seconds)
```

Gambar 26

b. Class BuffeReader

Class ini terdapat di paket **java.io** dan bisa kita gunakan untuk membaca inputan User, berikut ini contohnya:

```
public static void main(String[] args)

        throws IOException {
    String namaDepan = "";
    String namaBelakang = "";


    //buat objek dari inputStream
    InputStreamReader ireader =
            new  InputStreamReader(System.in);

    //buat objek bufferreader
    BufferedReader breader =
            new BufferedReader(ireader);


    System.out.println("Nama Depan : ");

    //membaca inputan user
    namaDepan =
    breader.readLine();
    System.out.println("Nama Belakang : ");

    //membaca inputan user
    namaBelakang =
    breader.readLine();
    System.out.println("Nama Saya adalah :");
    System.out.println(namaDepan + " " +
    namaBelakang);
}
```

Output dari potongan kode diatas adalah sebagai berikut:

```
Nama Depan :
Yusril
Nama Belakang :
Arief
Nama Saya adalah :
Yusril Arief
BUILD SUCCESSFUL (total time: 8 seconds)
```

Gambar 27

3. Class JoptionPane

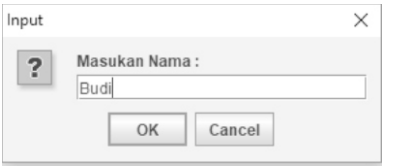
JoptionPane adalah sebuah Class yang menyediakan sarana menerima dan menampilkan data berbasis grafis. Model tampilan akan mengikuti tampilan yang disediakan oleh Sistem Operasi. JoptionPane terdapat di dalam Class **javax.swing**. Berikut beberapa contoh JOptionPane yang dapat kita gunakan :

c. Method ShowInputDialog()

Method ini digunakan untuk menerima inputan User dan Method ini selalu mengembalikan nilai bertipe String. Jika ingin mengembalikan nilai bilangan bulat atau pecahan, ShowInputDialog() harus dikonversi dahulu.

```
public static void main(String[]
        args) { String nama =
                JOptionPane.showInputDialog("Masukan Nama
        :");
        System.out.println("Nama : "+nama);
        }
```

Output dari potongan kode di atas adalah:



Gambar 28

Pesan yang dihasilkan adalah Nama : Budi seperti gambar di bawah ini:

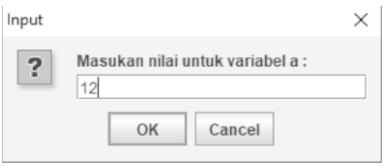
```
run:
Nama : Budi
BUILD SUCCESSFUL (total time: 3 minutes 8 seconds)
```

Gambar 29

Seperti yang sudah dijelaskan di atas, jika ingin menghasilkan bilangan bulat harus di konversi dahulu. Berikut di bawah ini contoh kodenya:

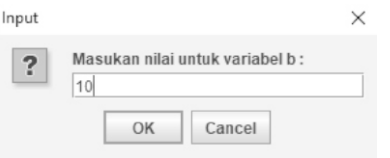
```
public static void
      main(String[]
      args) { String a =
            JOptionPane.showInputDialog("Masukan nilai
      untuk variabel a : ");


      String b =

            JOptionPane.showInputDialog("Masukan nilai
      untuk variabel b : ");


      System.out.println("Hasil :" +

            (Integer.parseInt(a) *
            Integer.parseInt(b)));
}
```


Hasil output dari potongan kode di atas adalah:



Gambar 30

Setelah memasukan angka untuk variabel a, selanjutnya memasukan angka untuk variabel b seperti gambar di bawah ini:



Gambar 31

Kita akan mengkalikan kedua variabel tersebut dan menghasilkan nilai 120.

```
run:
Hasil :120
BUILD SUCCESSFUL (total time: 7 minutes 7 seconds)
```

Gambar 32

d. Method ShowMessageDialog()

Method ini digunakan untuk menampilkan pesan dalam bentuk grafik.

```
public static void main(String[] args) {
        String nama = "Budi";
        String jurusan = "Sistem Informasi";

        JOptionPane.showMessageDialog(null, "Halo " + nama
                + ",jurusan Kamu " + jurusan);
    }
```

Output dari potongan kode di atas adalah:

Gambar 33

Parameter pertama dari method ShowMessageDialog() adalah nama untuk komponen induk. Di dalam contoh, kita isikan null karena memang tidak ada komponen induk yang memiliki form tersebut. Tidak hanya itu, kita juga dapat mengatur judul yang muncul pada form message dan kita juga dapat mengubah icon. Java sudah menyediakan beberapa tipe pesan sesuai tujuan tertentu:

a. JOptionPane.INFORMATION_MESSAGE

Tipe pesan ini dapat kita gunakan untuk menampilkan sebuah informasi untuk User. Di bawah ini contoh potongan kodenya:

```
public static void main(String[] args) {
      JOptionPane.showMessageDialog(
          null, "Sebuah Informasi.","Informasi",
          JOptionPane.INFORMATION_MESSAGE);
 }
```

Output dari potongan kode di atas adalah:

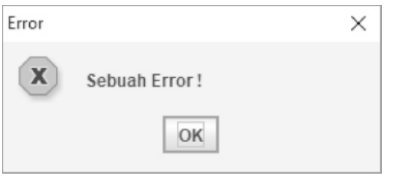
Gambar 34

b. JOptionPane.ERROR_MESSAGE

Tipe pesan ini tepat digunakan untuk menunjukan pesan error ke User. Berikut potongan kodenya:

```
public static void main(String[] args) {
        JOptionPane.showMessageDialog(
                null, "Sebuah Error !","Error",
                JOptionPane.ERROR_MESSAGE);
    }
```

Output dari potongan kode di atas adalah:



Gambar 35

c. JOptionPane.WARNING_MESSAGE

Tipe pesan ini tepat jika digunakan untuk menampilkan peringatan ke User. Berikut contoh potogan kodenya:

```
public static void main(String[] args) {
        JOptionPane.showMessageDialog(
            null, "Sebuah Peringatan.","Peringatan",
            JOptionPane.WARNING_MESSAGE);
        }
```

Output dari potongan kode di atas adalah:

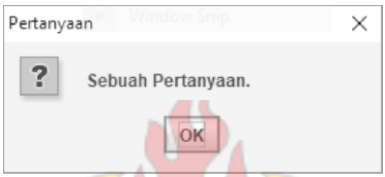
Gambar 36

d. JOptionPane.QUESTION_MESSAGE

Tipe ini digunakan untuk menampilkan sebuah pertanyaan kepada User. Pertanyaan tersebut biasanya digunakan untuk meyakinkan User, contoh pertanyaanya Apakah Anda yakin ingin menghapus data ?. Berikut di bawah ini potongan kodenya:

```
public static void main(String[] args) {
        JOptionPane.showMessageDialog(
                null, "Sebuah Pertanyaan.", "Pertanyaan",
                JOptionPane.QUESTION_MESSAGE);
}
```

Output dari potongan kode di atas adalah:



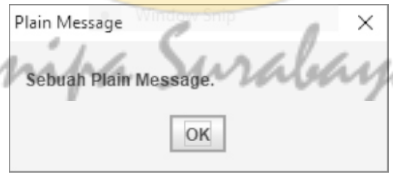
Gambar 37

e. JOptionPane.PLAIN_MESSAGE

Tipe ini hanya menampilkan kalimat saja dan menghilangkan icon. Berikut potongan kodenya:

```
JOptionPane.showMessageDialog(
      null, "Sebuah Plain Message.","Plain Message",
      JOptionPane.PLAIN_MESSAGE);
}
```

Output dari potongan kode di atas adalah:



Gambar 38

f. Method ShowConfirmDialog()

Method ini kita gunakan jika ingin melakukan konfirmasi kepada User. Method ini mengembalikan nilai int, jika mengembalikan 0 maka Yes. Jika mengembalikan 1 maka No

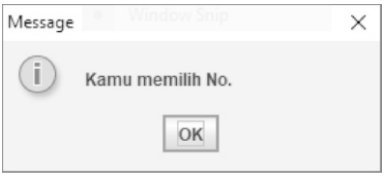
dan jika mengembalikan 2 maka Cancel. Masing – masing pilihan bisa kita letakan perintah selanjutnya, misalnya pertanyaan untuk menghapus data, jika user memilih Yes maka data akan di hapus jika User memilih No maka data tidak jadi dihapus. Untuk lebih jelas lihat potongan kode di bawah ini:

```
public static void main(String[] args) { int pertanyaan =
        JOptionPane.showConfirmDialog(null, "Pilih Yes atau No
                ?", "Pertanyaan",JOptionPane.YES_NO_OPTION);

    if(JOptionPane.YES_OPTION == pertanyaan){
    JOptionPane.showMessageDialog(null, "Kamu memilih Yes.");
    }else if(JOptionPane.NO_OPTION == pertanyaan){
    JOptionPane.showMessageDialog(null, "Kamu memilih No.");
    }}
```

Variabel pertanyaan bertipe int berguna untuk menampung nilai pengembalian dari pilihan User. Jika User memilih Yes Option maka keluar output seperti di bawah ini:

Gambar 39

Apabila sebaliknya, User memilih No maka keluar Output sepeti di bawah ini:



Gambar 40

Unipa Surabaya

BAB 4 Tipe Data dan Variabel

Variabel

Setiap data yang akan, sedang dan selesai diproses oleh komputer harus disimpan di dalam memori. Agar data di memori ini dapat dimanipulasi seperti ditambah, diubah atau dihapus maka lokasi tempat penyimpanan data ini harus diberi nama. Konsep ini yang disebut Variabel. Dalam membuat variabel dan menyimpan ke memori harus dideklarasikan dahulu. Kita harus menentukan tipe data untuk sebuah variabel. Sebuah variable bisa diisi saat program pertama kali dijalankan, interaksi dari Pengguna atau dari parameter dan sebuah variabel hanya dapat menyimpan satu data jika ada data lainya masuk maka data sebelumnya akan digantikan dengan data yang baru.

Tipe Data

Secara umum ada tiga jenis tipe data yang dikenal komputer, antara lain:

- Numerik adalah tipe data bebentuk bilangan bulat atau pecahan.
- Karakter adalah tipe data yang berbentuk karakter tunggal.
- Logika tipe data yang berbentuk benar (true) atau salah (false).

Java dibagi menjadi dua jenis tipe data yaitu tipe data primitif dan tipe data objek.

- Tipe Data Primitif

Tipe data yang diadopsi dari tipe data klasik. Tipe ini diadopsi dari berbagai Bahasa seperti C++ dan Pascal. Contoh tipe data primitif adalah sebagai berikut :

a. Integer

Integer adalah tipe data berbentuk bilangan bulat. Tipe data ini isinya berupa nilai positif atau negatif.

Tipe	Bit	Range
Byte	8	-128 s/d 127
Short	16	-32768 s/d 32767
Int	32	-2147483648 s/d 2147483647
Long	64	-9223372036854775808 s/d 9223372036854775807

Keterangan :

Byte

Byte adalah tipe data yang biasa digunakan bila berhubungan dengan data stream dari suatu file seperti proses baca atau tulis ke file Excel.

Short

Short adalah tipe data yang umumnya diaplikasikan pada Komputer 16-bit.

Int

Int adalah tipe data yang paling sering dipakai di dalam Java. Sering dipakai dalam pengulangan, array dan lain-lain.

Long

Long adalah tipe data yang sering dipakai apabila panjangnya sudah melebihi tipe data Int.

b. Floating Point.

Foating Point adalah tipe data yang sering dipakai apabila panjangnya sudah melebihi tipe data Int atau pecahan. Tipe data ini terbagi mejadi dua bagian yaitu Float dan Double.

c. Char

Char adalah tipe data berupa karakter yang menggunakan *Unicode* untuk merepresentasikan karakter.

d. Boolean

Tipe data Boolean adalah tipe data yang hanya memiliki dua kemungkinan yaitu true atau false. Tipe data ini ditandai dengan kata kunci Boolean.

- Tipe Data Objek

Tipe data ini juga disebut sebagai tipe data referensi. Berikut di bawah ini macam-macam tipe data referensi.

a. Class

Dalam dunia pemrograman suatu Class juga dapat disebut sebagai tipe data sederhana karena saat kita membuat sebuah Class maka kita juga membuat *blue print* atau cetak biru dari sebuah objek.

b. Array

Tipe data ini memiliki kemampuan untuk menyimpan sebuah variabel ke dalam data list. Sebuah array bisa berupa satu dimensi atau dua dimensi.

c. Interface

Interface adalah sebuah Class yang berisi method tanpa ada detail implementasinya. Untuk mengisi detail implementasinya harus melalui Class yang mengimplementasikanya.

Cara Membuat Variabel

Di dalam Java, sebuah variabel dideklarasikan dengan kode sebagai berikut :

Tipe_data nama_variabel = nilai_awal;

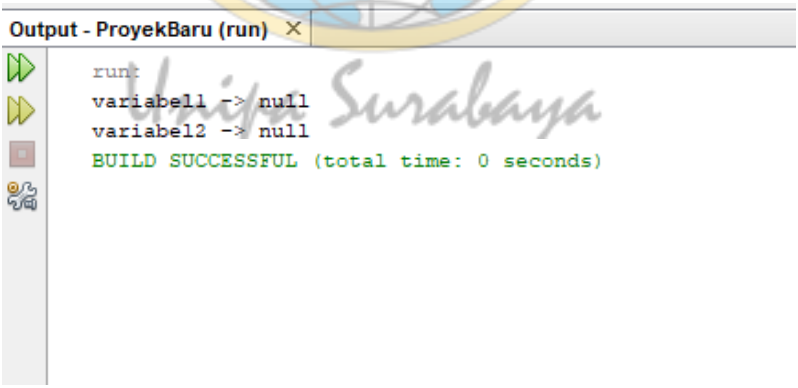
Keterangan :

- Tipe_data merupakan deklarasi tipe data seperti yang jelaskan sebelumnya
- Nama_variabel adalah nama dari deklarasi variabel. Aturan pemberian nama huruf awal harus menggunakan huruf kecil.
- Nilai_awal bisa berupa data langsung, ekspresi atau fungsi.

Berikut di bawah ini contoh membuat variabel di Java :

```
package proyekbaru;

public class DeklarasiVariabel {

    public static String variabel1;
    public static String variabel2;

    public static void main(String[] args) {
        System.out.println("variabel1 -> " + variabel1);
        System.out.println("variabel2 -> " + variabel2);
    }

}
```

Jika saat mendeklarasikan variabel tidak diberi isi maka saat program di eksekusi akan mengisi dengan nilai default, contohnya null jika tipe data string atau 0 jika tipe data Integer. Hasil output dari potongan kode di atas adalah :

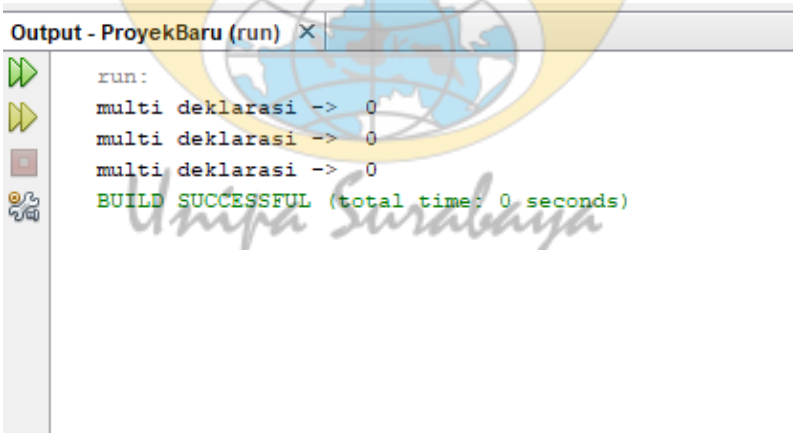


Gambar 41

Di dalam Java juga dapat mendeklarasikan variabel lebih dari satu secara bersamaan, dengan syarat tipe data harus sejenis. Berikut ini contohnya :

```
package proyekbaru;

public class DeklarasiVariabelMulti {

    public static int var1, var2, var3;

    public     static     void     main(String[]     args)     {
        System.out.println("multi deklarasi -> " + " " + var1);
        System.out.println("multi deklarasi -> " + " " + var2);
        System.out.println("multi deklarasi -> " + " " + var3);
    }

}
```

Hasil dari potongan kode di atas adalah :



Gambar 42

Cara mengisi sebuah variabel adalah sebagai berikut :

```
package proyekbaru;


public class DeklarasiVariabel {


    public   static   String   variabel1;

    public static String variabel2;


    public      static      void      main(String[]      args)      {

        System.out.println("variabel1    ->    "    +    variabel1);

        System.out.println("variabel2    ->    "    +    variabel2);

        System.out.println("setelah diisi :");

        variabel1    =    "Hello";

        variabel2 = "Word";

        System.out.println("variabel1    ->    "    +    variabel1);

        System.out.println("variabel2 -> " + variabel2);

    }


}
```

Output dari potongan kode di atas adalah :

```
run:
variabel1 -> null
variabel2 -> null
setelah diisi :
variabel1 -> Hello
variabel2 -> Word
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 43

Konversi Data

Konversi data sering dibutuhkan ketika memproses data bertipe tertentu sebagai data bertipe lain, misalnya String harus dikonversi ke tipe data Integer. Proses ini bisa dilakukan degan dua acara, antara lain :

- Konversi Konvensional

  Koversi Konvensional menggunakan class yang sudah disediakan untuk tipe data tertentu yang akan diproses. Sebagai contoh tipe data method Integer.parseint(String) digunakan untuk mengkonversi data String menjadi Integer. Berikut beberapa method Java yang dapat digunakan untuk konversi data.

  a. String ke Integer

     Jika ingin mengkonversi tipe data String ke Integer gunakan method Integer.parseint(String).

  b. String ke Double

     Jika ingin mengkonversi tipe data String ke Double gunakan method Double.parsedouble(String).

  c. String ke Short

     Jika ingin mengkonversi tipe data String ke Double gunakan method Short.parsedouble(String).

Berikut di bawah ini contoh kode dan output dari tiga metod konversi yang dijelaskan di atas:

```
public class Konversi1 {


    public static void main(String[] args) {
        String var1 = "1234567899";
        String var2 = "3.14";
        String var3 = "1";

        int bulat = Integer.parseInt(var1);
        double pecahan = Double.parseDouble(var2);
        short bulat2 = Short.parseShort(var3);

        System.out.println("variabel var1 " + var1);
        System.out.println("variabel var2 " + var2);
        System.out.println("variabel var3 " + var3);
        System.out.println("---");
        System.out.println("konversi bilangan bulat " + bulat);
        System.out.println("konversi bilangan pecahan " + pecahan);
        System.out.println("konversi bilangan bulat2 " + bulat2);
    }
}
```

Di bawah ini output dari potongan kode di atas :

```
variabel var1 1234567899
variabel var2 3.14
variabel var3 1
---
konversi bilangan bulat 1234567899
konversi bilangan pecahan 3.14
konversi bilangan bulat2 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 44

- Tipe Casting

  Konversi data mengunakan cara ini akan menyebabkan suatu data akan mengalami suatu perubahan tipe data ketika diproses. Berikut di bawah ini contoh Casting :

```
public class Konversi2 {


    public static void main(String[] args) {
        int var1 = 5;
        int var2 = 3;
        System.out.println("Hasil : "+ var1 / var2);
    }
}
```

Hasil dari potongan kode di atas adalah :

```
Hasil : 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 45

Hasil dari potongan kode di atas adalah kurang tepat karena hasil yang seharusnya adalah pecahan. Hal ini terjadi karena isi dari variabel var1 dan var2 adalah bilangan bulat maka ketika proses hasilnya menjadi bilangan bulat juga. Untuk menghasilkan pecahan kita harus konversi terlebih dahulu, berikut di bawah ini potongan kode untuk melakukan Casting :

```
public class Konversi2 {

    public  static  void  main(String[]
    args) { int var1 = 5;
    int var2 = 3;

    System.out.println("Hasil  :  "+  (double)var1  /
    var2);
    }
}
```

Di bawah ini output dari potongan kode di atas :

```
Hasil : 1.6666666666666667
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 46

Unipa Surabaya

BAB 5 Operator di Java

Operator Bilangan Bulat

Terdapat 3 macam operator yang dapat dilakukan oleh tipe data Integer, antara lain :

- Unary : berlaku untuk 1 angka Integer.
- Binary : berlaku pada pasangan bilangan Integer
- Relasional : berlaku pada bilangan Integer tapi mengembalikan hasil Boolean berupa nilai true atau false.

Operator Bilangan Bulat Unary

Contoh operator ini adalah :

Deskripsi	Operator
Increment	+ +
Decrement	- -
Negasi	-

Operator increment (+ +) dan decrement (- -) digunakan untuk menambah dan mengurangi 1 angka dari sebuah variable Integer. Contohnya penggunaan increment seperti di bawah ini :

```
public static void main(String[] args) {
    Integer sebelum = 20;
    Integer sesudah = 0;


    System.out.println("Sebelum Increment : " + sebelum);
    sesudah = ++sebelum;
    System.out.println("Sesudah Increment : " + sesudah);
}
```

Di dalam contoh di atas variabel **sebelum** diisi 20 saat program pertama kali dijalankan, kemudian kita increment dengan tanda + + dan ditampung ke variabel **sesudah** dan otomatis variabel **sebelum** akan ditambahkan 1 angka. Output dari potongan kode di atas adalah :

```
run:
Sebelum Increment : 20
Sesudah Increment : 21
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 47

Selanjutnya di bawah ini contoh penggunaan decrement :

```
public static void main(String[] args) {

        Integer sebelum = 9;

        Integer sesudah = 0;


        System.out.println("Sebelum Decrement : " + sebelum);

        sesudah = --sebelum;

        System.out.println("Sesudah Decrement : " + sesudah);

    }
```


Dari potongan kode di atas, variabel **sebelum** diisi angka 9 kemudian kita decrement menggunakan tanda - - dan ditampung di variabel **sesudah**. Output dari potongan kode di atas adalah :

```
run:
Sebelum Decrement : 9
Sesudah Decrement : 8
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 48

Operator Negasi (-) digunakan untuk mengubah nilai bilangan bulat. Contohnya seperti potongan kode di bawah ini :

```
public static void main(String[] args) {
        int a = 2;
        int b = 0;

        System.out.println("Sebelum negasi : " + a);
        b = -a;
        System.out.println("Sebelum negasi : " + b);
    }
```

Hasil output dari potongan kode di atas adalah :

```
run:
Sebelum negasi : 2
Sebelum negasi : -2
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 49

Operator Bilangan Bulat Binary

Berikut di bawah ini beberapa contoh bilangan bulat binary :

Deskripsi	Operator
Penambahan	+
Pengurangan	-
Perkalian	*
Pembagian	/
Sisa bagi	%

Operator penjumlahan, pengurangan, perkalian, pembagian adalah operator aritmatika. Berikut ini contoh penggunaan operator di pada table 6 :

```
public static void main(String[] args) {
        //Penambaha
        n  int  a1  =
        10; int a2 =
        2;
        System.out.println("Penambahan 10 + 2 : " + (a1 + a2));
        //Penguranga
        n int b1 =
        10; int b2 =
        2;
        System.out.println("Pengurangan 10 - 2 : " + (b1 - b2));

        //Perkalian
        int c1 = 16;
        int c2 = 2;
        System.out.println("Perkalian 16 * 2 : " + (c1 * c2));

        //Pembagian
        int d1 = 16;
        int d2 = 2;
        System.out.println("Pembagian 16 / 2 : " + (d1 / d2));

        //Sisa Bagi
        int e1 = 16;
        int e2 = 2;
```

```
        System.out.println("Sisa bagi 16 dan 2 : " + (e1 %e2));
    }
```

Hasil potongan kode di atas seperti di bawah ini :

```
run:
Penambahan 10 + 2 : 12
Pengurangan 10 - 2 : 8
Perkalian 16 * 2 : 32
Pembagian 16 / 2 : 8
Sisa bagi 16 dan 2 : 0
BUILD SUCCESSFUL (total time: 1 second)
```

Gambar 50

Operator Bilangan Bulat Relasional

Di Bawah ini contoh operator bilangan bulat Relasional :

Deskripsi	Operator
Kurang dari	<
Lebih besar dari	>
Lebih kecil atau sama dengan	<=
Lebih besar dari atau sama dengan	>=
Sama dengan	= =
Tidak sama dengan	!=

Operator-operator ini digunakan untuk melakukan pembandingan antara dua bilangan bulat. Di bawah ini contoh operator **kurang dari** :

```
public static void main(String[] args) {
        int a = 10;
        int b = 15;

        System.out.println("Variabel a : " + a);
        System.out.println("Variabel b : " + b);
        System.out.println("Apakah 10 kurang dari 15 " + (a < b));
        System.out.println("Apakah 10 kurang dari 15 " + (b < a));
        ystem.out.println("Apakah 10 kurang dari 10 " + (b < b));
```

```
}
```

Hasil dari potongan kode di atas adalah :

```
Variabel a : 10
Variabel b : 15
Apakah 10 kurang dari 15 true
Apakah 15 kurang dari 10 false
Apakah 15 kurang dari 15 false
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 51

Dari output program di atas nilai 10 kurang dari 15 akan menghasilkan true yang artinya nilai 10 memang kurang dari nilai 15. Selanjutnya adalah operator **lebih besar dari**, di bawah ini contoh kodenya :

```
public static void main(String[] args) {
        int a = 2;
        int b = 3;
        System.out.println("Variabel a : " + a);
        System.out.println("Variabel b : " + b);
        System.out.println("Apakah 2 lebih besar dari 3 " + (a > b));
        System.out.println("Apakah 3 lebih besar dari 2 " + (b < a));
        System.out.println("Apakah 3 lebih besar dari 3 " + (b < b));
    }
```

Sama seperti sebelumnya, jika nilai 3 lebih besar dari 2 maka akan mengembalikan nilai true, di bawah ini output dari potongan kode di atas :

```
Variabel a : 2
Variabel b : 3
Apakah 2 lebih besar dari 3 false
Apakah 3 lebih besar dari 2 false
Apakah 3 lebih besar dari 3 false
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 52

Selanjutnya operator **lebih kecil atau sama dengan**. Di bawah ini contohnya :

```
public static void main(String[] args) {
        int a = 3;
        int b = 3;
        System.out.println("Variabel a : " + a);
        System.out.println("Variabel b : " + b);
        System.out.println("Apakah 3 lebih kecil atau sama dengan 3 "
        + (a <= b));
        a = 9;
        System.out.println("Variabel a diubah : " + a);
        System.out.println("Apakah 9 lebih kecil atau sama dengan 3 "
        + (a <= b));
    }
```

Output dari potongan kode di atas :

```
Variabel a : 3
Variabel b : 3
Apakah 3 lebih kecil atau sama dengan 3 true
Variabel a diubah : 9
Apakah 9 lebih kecil atau sama dengan 3 false
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 53

Operator selanjutnya adalah **lebih besar dari atau sama dengan**. Berikut di bawah ini contoh kodenya :

```
public static void main(String[] args)
        { int a = 3;
        int b = 3;
        System.out.println("Variabel a : " + a);
        System.out.println("Variabel b : " + b);
        System.out.println("Apakah 3 lebih besar atau sama dengan 3 "
        + (a >= b));
        a = 2;
        System.out.println("Variabel a diubah : " + a);
        System.out.println("Apakah 2 lebih besar atau sama dengan 3 "
        + (a >= b));
    }
```

Output dari potongan kode di atas adalah:

```
Variabel a : 3
Variabel b : 3
Apakah 3 lebih besar atau sama dengan 3 true
Variabel a diubah : 2
Apakah 2 lebih besar atau sama dengan 3 false
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 54

Selanjutnya adalah operator **sama dengan** dan operator **tidak sama dengan**. Di bawah ini contoh potongan kodenya:

```
public static void main(String[] args)
        { int a = 3;
        int b = 3;
```

```
        System.out.println("Variabel a : " + a);
        System.out.println("Variabel b : " + b);
        System.out.println("Apakah 3 sama dengan 3 " + (a ==
        b)); a = 2;
        System.out.println("Apakah 2 sama dengan 3 " + (a == b));
        System.out.println("Apakah 2 tidak sama dengan 3 " + (a !=
        b));
    }
```

Output dari potongan kode di atas adalah:

```
Variabel a : 3
Variabel b : 3
Apakah 3 sama dengan 3 true
Apakah 2 sama dengan 3 false
Apakah 2 tidak sama dengan 3 true
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 55

Operator Boolean

Daftar Operator Boolean :

Deskripsi	Operator
Logika dan	&&
Logika atau	||

Logika dan (&&) akan menghasilkan nilai true jika kedua sisi benar, sebaliknya bernilai false jika salah satu sisi nilainya salah. Di bawah ini contoh penggunaan operator && :

```
public static void main(String[] args) {
        System.out.println("true and true : " + (true && true));
        System.out.println("true and false : " + (true && false));
        System.out.println("false and false : " + (false && false));
        System.out.println("false and true : " + (false && true));
    }
```

Output dari potongan kode di atas:

```
true and true : true
true and false : false
false and false : false
false and true : false
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 56

Logika atau (||) akan menghasilkan nilai true jika salah satu sisi terdapat nilai true. Contoh penggunakan operator || sepeti di bawah ini:

```
System.out.println("true or true : " + (true || true));
System.out.println("true or false : " + (true || false));
System.out.println("false or false : " + (false || false));
System.out.println("false or true : " + (false || true));
```

Output dari potongan kode di atas adalah:

```
true or true : true
true or false : true
false or false : false
false or true : true
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 57

BAB 6 Percabangan

Percabangan atau Branching

Percabangan atau Banching adalah suatu alur dari program yang memiliki cabang alur lain. Percabangan sangat sering digunakan dalam membuat program. Alurnya adalah program akan melakukan pengecekan dahulu terhadap suatu kondisi kemudian jika benar atau true maka perintah akan di eksekusi. Jenis percabangan ada dua yaitu **If – Else if** dan **Switch Case**.

- If Else if

  Program akan melakukan pengecekan di kondisi if yang pertama, kemudian jika memenuhi maka baris kode di dalam **if** akan di eksekusi. Sebaliknya, jika tidak memenuhi maka program akan mengecek di kondisi **else if**. Jika memenuhi maka baris kode di dalam **else if** akan dieksekusi. Dibawah ini contoh penggunaan **else if** untuk menentukan bilangan ganjil atau genap:

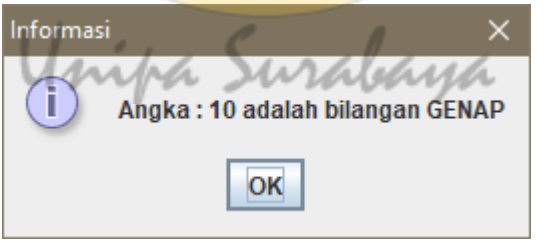
```
public static void main(String[] args) {

       String temp = JOptionPane.showInputDialog(null,
       "Masukan Angka", "Menentukan Ganjil Genap",
       JOptionPane.QUESTION_MESSAGE);
       int input =
       Integer.parseInt(temp);
       System.out.println((input %
       2)); if ((input % 2) == 0) {
          JOptionPane.showMessageDialog(null, "Angka : "
          + input + " "

                + "adalah bilangan GENAP", "Informasi",
             JOptionPane.INFORMATION_MESSAGE);
       } else if ((input % 2) == 1) {
          JOptionPane.showMessageDialog(null, "Angka : "
          + input + " "
```

```
                + "adalah bilangan GANJIL", "Informasi",
            JOptionPane.INFORMATION_MESSAGE);
    }

}
```

Ketika program dijalankan, program akan menampilkan GUI dengan perintah **Masukan Angka**.

Gambar 58

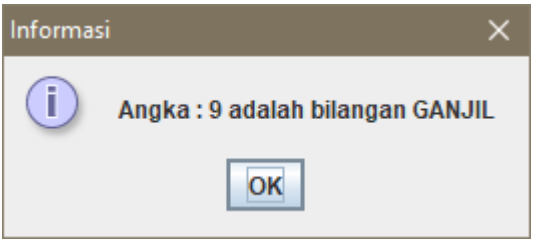
Setelah User memasukan angka, Program akan melakukan pengecekan menggunakan kondisi **IF** yang pertama. Jika bilangan habis dibagi dua maka tampilkan informasi bilangan tersebut adalah bilangan genap.



Gambar 59

Apabila tidak habis dibagi dua, maka pengecekan **ELSE IF** akan

dijalankan. Contoh, jika user memasukan angka sembilan, maka output akan keluar seperti di bawah ini :



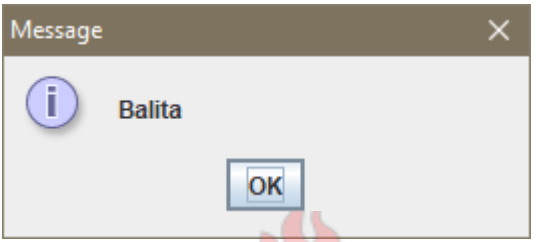
Gambar 60

- If Else

  Percabangan tipe ini sama seperti percabangan **If Else If** namun yang membedakan jika kondisi pertama dan seterusnya tidak pernah memenuhi, maka kondisi else yang paling terakhir dieksekusi. Di bawah ini contoh potongan kodenya:

```
public static void main(String[] args) {
        String umur = JOptionPane.showInputDialog(null, "Kamu umur berapa ?");

        if (Integer.parseInt(umur) <= 5) {
            JOptionPane.showMessageDialog(null,  "Balita");
        } else if (Integer.parseInt(umur) >= 6 && Integer.parseInt(umur) <= 11)
            {  JOptionPane.showMessageDialog(null, "Anak - anak");
        } else if (Integer.parseInt(umur) >= 12 && Integer.parseInt(umur) <= 25)
            {  JOptionPane.showMessageDialog(null,  "Remaja");
        } else {
            JOptionPane.showMessageDialog(null,  "Dewasa");
        }
    }
```

Jika User memasukan angka antara 0 sampai 5 maka akan menampilkan pesan Balita. Outputnya seperti gambar di bawah ini:

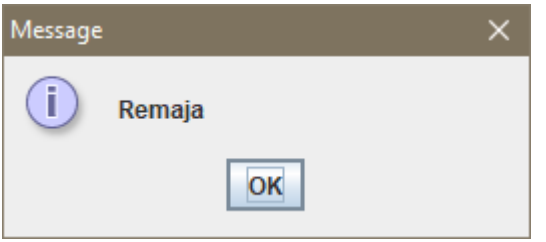


Gambar 61

Jika User memasukan angka antara 6 sampai 11 maka program akan menampilkan pesan Anak – anak.

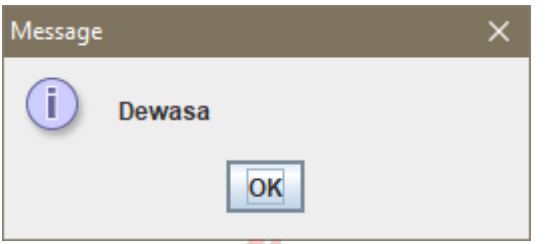
Gambar 62

Jika User memasukan angka antara 12 sampai 25 maka program akan menampilkan pesan Remaja.



Gambar 63

Jika User memasukan angka selain angka di atas maka program akan masuk **ELSE** dan menampilkan pesan **Dewasa**.



Gambar 64

- Switch Case

Di bawah ini contoh menggunakan **switch case**:

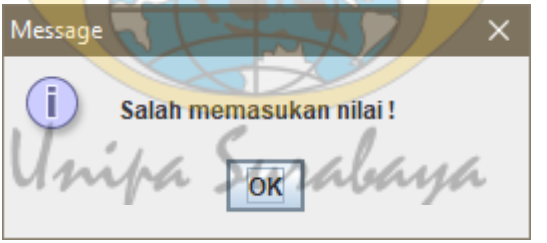
```
public static void main(String[] args) {

        String nilai = JOptionPane.showInputDialog
        (null, "Masukan nilai :");
        switch (nilai.toUpperCase()) {
            case "A":
                JOptionPane.showMessageDialog(null, "Sangat baik.");
                break;
            case "B":

                JOptionPane.showMessageDialog(null, "Baik.");
                break;
            case "C":

                JOptionPane.showMessageDialog(null, "Cukup.");
            case "D":
                JOptionPane.showMessageDialog(null, "Jelek.");
                break;
            default :

                JOptionPane.showMessageDialog(null,
                 "Salah memasukan nilai !");
        }
```

Saat User memasukan inputan, program akan melakukan pengecekan satu per satu sampai sebuah kondisi memenuhi. Kata

kunci **break** digunakan agar saat kondisi sudah memenuhi, program akan keluar. Jika tidak diberi kata kunci **break** program akan melanjutkan pengecekan berikutnya dan hal ini kurang tepat. Kata kunci default fungsinya sama seperti **ELSE**, jika tidak ada yang memenuhi maka **default** dijalankan. Dibawah ini contoh output jika User memasukan nilai C:

Gambar 65

Jika User memasukan huruf selain yang ada di kondisi, misalnya E maka program akan masuk ke default dan mengeluarkan informasi **Salah memasukan nilai**.



Gambar 6

BAB 7 Array

Pengenalan Array

Array adalah sebuah variabel yang dapat menyimpan banyak data. Array menggunakan indeks untuk menandai saat dialokasikan ke memori dan indeks Array selalu dimulai dari 0.

Gambar 67

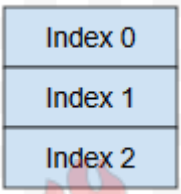
Gambar di atas menjelaskan bagaimana sebuah Array menyimpan isinya ke dalam memori, nama Budi berada di indeks 0, Andi berada di indeks 1 dan Dwi berada di indeks 2. Cara membuat Array di java adalah sebagai berikut:

```
public static void main(String[] args) {
        //Cara 1
        String[] nama1;

        //Cara 2 String
        nama2[];

        //Cara3
        String[] nama3 = new String[2];
}
```

Array memiliki 2 macam yaitu Array 1 dimensi dan Array 2 dimensi.

1. Array Satu Dimensi

Array ini memilik satu dimensi, konsepnya seperti di bawah ini:



Gambar 68

Di bawah ini potongan kodenya:

```
public static void main(String[] args) {

        String nama[] = new String[3];

        nama[0] = "Budi";

        nama[1] = "Andi";

        nama[2] = "Dwi";


        System.out.println("Index 0 : "+nama[0]);

        System.out.println("Index 1 : "+nama[1]);

        System.out.println("Index 2 : "+nama[2]);

}
```

Output dari potongan kode di atas adalah:

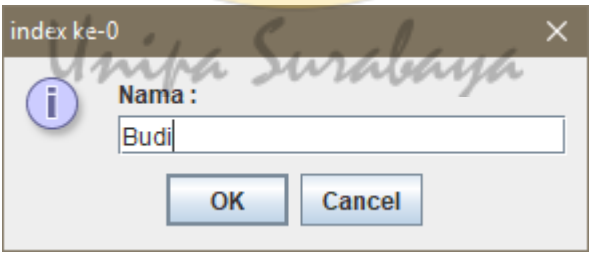
```
Index 0 : Budi
Index 1 : Andi
Index 2 : Dwi
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 69

Jika kita ingin mengambil nilai dari index tertentu, kita harus tentukan index mana yang ingin di ambil. Perhatikan potongan kode di atas **nama[2]** artinya kita ambil isi dari index 2. Contoh potongan kode di atas mengisi variabel Array masih dari dalam, bagaimana jika mengisi Array dari masukan User? Di bawah ini contoh potongan kodenya:

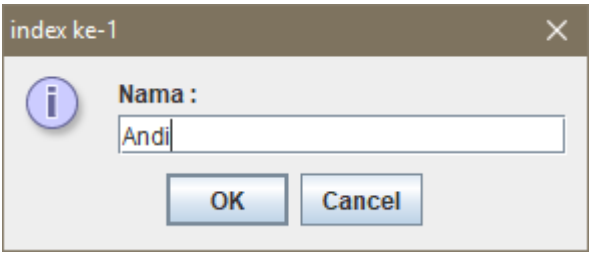
```
public static void main(String[] args)

        { String[] nama = new

        String[3];

        nama[0] = JOptionPane.showInputDialog(null,"Nama :","index ke-0
        ",JOptionPane.INFORMATION_MESSAGE);

        nama[1] = JOptionPane.showInputDialog(null,"Nama :","index ke-1
        ",JOptionPane.INFORMATION_MESSAGE);

        nama[2] = JOptionPane.showInputDialog(null,"Nama :","index ke-2
        ",JOptionPane.INFORMATION_MESSAGE);



        JOptionPane.showMessageDialog(null, nama[0]+", "+nama[1]+",
        "+nama[2]);

}
```

Saat program dijalankan, outputnya seperti di bawah ini:



Gambar 70

Tampilan pertama, User akan diminta memasukan index ke-0.

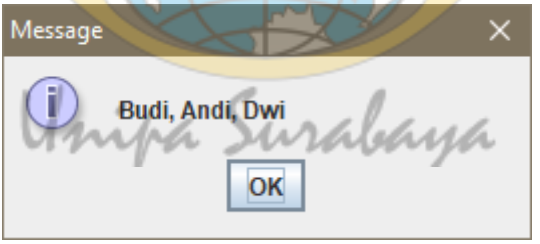


Gambar 71

Tampilan kedua, User diminta untuk memasukan index-1.

Gambar 72

Tampilan ketiga, User diminta untuk memasukan index-2 dan ini yang terakhir. Setelah semua index 0 sampai 2 diisi maka program akan menampilkan isi Array seperti gambar di bawah ini:

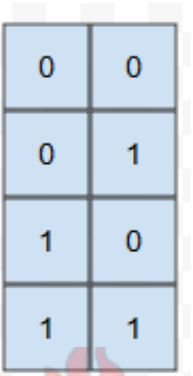


Gambar 73

Array Multi Dimensi

Array Multi Dimensi adalah Array yang memiliki Array di dalamnya, lebih singkatnya Array di dalam Array. Konsepnya

seperti gambar di bawah ini.



Gambar 74

Keterangan gambar di atas adalah index kiri menunjukan baris dan index kanan menunjukan kolom. Jadi artinya index 0 memiliki index 0 dan 1, begitu juga index 1 memiliki index 0 dan 1. Contoh potongan kodenya seperti di bawah ini:

```
public static void main(String[] args) {
        String umur[][] = new String[2][2];
        umur[0][0] = "Budi";
        umur[0][1] = "20";
        umur[1][0] = "Andi";
        umur[1][1] = "45";

        System.out.println("Nama : "+umur[0][0]+" umur : "+umur[0][1]);
        System.out.println("Nama : "+umur[1][0]+" umur : "+umur[1][1]);
}
```

Output dari potongan kode di atas adalah sebagai berikut:

```
run:
Nama : Budi umur : 20
Nama : Andi umur : 45
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 75

Membuat array multi dimensi sama seperti membuat array biasa menggunakan kata kunci new, lihat potongan kode:

```
String umur[][] = new String[2][2];
```

Artinya, kita membuat array bertipe string berisi masing – masing 2 index array.

BAB 8 Pengulangan

Pengenalan Pengulangan (Looping)

Pengulangan disebut juga Looping adalah suatu alur program yang mengulang – ulang suatu proses. Contohnya, jika kita ingin menampilkan sebuah pesan **Halo Dunia** sebanyak 10 kali, tentu logikanya kita akan mencetak kalimat tersebut menggunakan **System.out.Print** sebanyak 10 kali. Cara ini benar namun tidak efisien. Bagaimana jika kita ingin mengulang 1000 kali ? tentu sangat merepotkan. Kita bisa membuatnya singkat dengan menggunakan looping.

1. Pengulangan For

Pengulangan **for** digunakan jika mengetahui jumlah pengulangan yang diinginkan. Contoh potongan kodenya seperti di bawah ini:

```
public static void main(String[] args) {
        for (int a = 0; a < 10; a++) {
            System.out.println("Halo Dunia");
        }
}
```


Variabel a digunakan untuk menampung jumlah pengulangan, jika melihat contoh di atas variabel a dimulai dari index 0 diulang sampai index mencapai 9, lihat kode a < 9. Sedangkan kode a++ digunakan untuk increment atau menambahkan 1 angka setiap pengulangan. Output dari potongan kode di atas adalah:

```
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
Halo Dunia
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 76

Saya akan berikan contoh lagi potongan kode menggunakan For.

```
public static void main(String[] args) {
        for (int a = 2; a < 12; a += 2) {
            System.out.println("Bilanga Genap : " + a);
        }
    }
```

Potongan kode di atas digunakan untuk mencetak bilangan genap. Logikanya, kita mengulang kurang dari 12 kali dan setiap pengulangan akan di tambahkan 2. Outputnya seperti gambar di bawah ini:

```
Bilangan Genap : 2
Bilangan Genap : 4
Bilangan Genap : 6
Bilangan Genap : 8
Bilangan Genap : 10
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 77

For Loop Bersarang

For Loop bersarang adalah pengulangan di dalam pengulangan menggunakan kata kunci for. Contohnya seperti potongan kode di

bawah ini:

```
public static void main(String[] args) {
        int baris = 5;

        for (int i = 0; i <= baris; i++) {

            for (int j = 0; j < i; j++) {
                System.out.print("* ");
            }

            System.out.println("");
        }
    }
```

Output dari potongan kode di atas adalah membuat segitiga siku siku:

```
*
* *
* * *
* * * *
* * * * *
BUILD SUCCESSFUL (total time: 0 seconds)
```

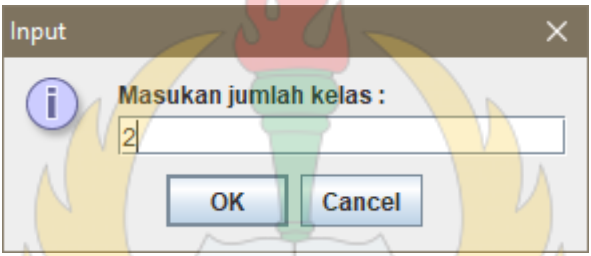
Gambar 78

Looping yang pertama digunakan untuk membuat baris dan looping yang kedua digunakan untuk mencetak tanda bintang. Tanda bintang akan dibuat sebanyak 5 baris. Selanjutnya, coba kita lihat contoh kode penggunaan looping yang lebih kompleks di bawah ini.

```
public static void main(String[] args) {
        int jumlahKelas =
        Integer.parseInt(JOptionPane.showInputDialog(null,
        "Masukan jumlah kelas : ",
        "Input",
        JOptionPane.INFORMATION_MESSAGE));
        String isiKelas[][] = new
        String[jumlahKelas][];

        for (int kelas = 0; kelas < jumlahKelas; kelas++) {
            String namaKelas = JOptionPane.showInputDialog(null, "Nama
            Kelas :", "Masukan Nama Kelas",
            JOptionPane.INFORMATION_MESSAGE);
            int jumlahKursi =
            Integer.parseInt(JOptionPane.showInputDialog(null, "Jumlah
            murid masing - masing kelas : ",
            "Input",
            JOptionPane.INFORMATION_MESSAGE));
            isiKelas[kelas] = new
            String[jumlahKursi];
            System.out.println("Kelas " + namaKelas
            + " : "); for (int kursi = 0; kursi <
            jumlahKursi; kursi++) {
                String namaSiswa = JOptionPane.showInputDialog(null,
                "Nama Siswa", "Kelas " + namaKelas,
                JOptionPane.INFORMATION_MESSAGE); isiKelas[kelas][kursi]
                = namaSiswa;
                System.out.println("Nama Siswa " +
                isiKelas[kelas][kursi]);
            }
```

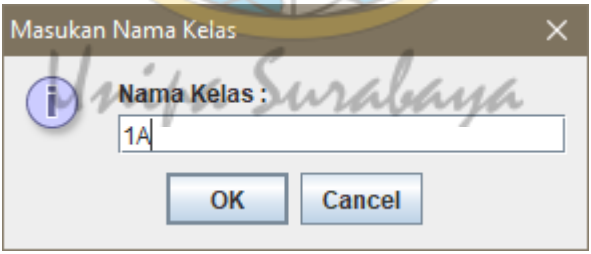
```
            System.out.println("----------------------------------------
            ");
        }

    }
```

Potongan kode di atas mengkombinasikan pengulangan dan array multi dimensi. Tujuan akhirnya adalah menampilkan nama kelas dan nama siswa di masing – masing kelas. Saat program di jalankan, output yang keluar seperti di bawah ini:



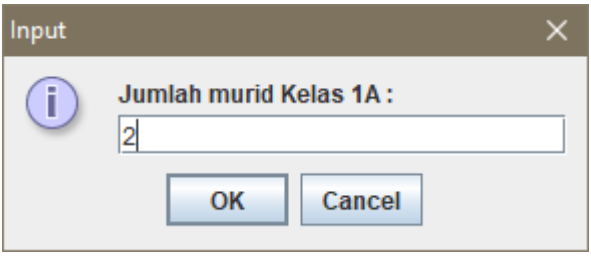
Gambar 79

Jika User memasukan angka dua maka program akan membuat array dengan isi 2 index. Output selanjutnya adalah:



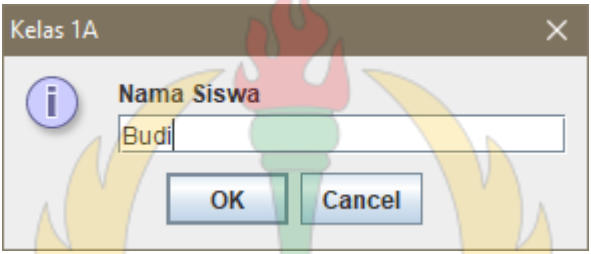
Gambar 80

Index ke-0 diisi dengan nama **Kelas 1A**. Output selanjutnya adalah:



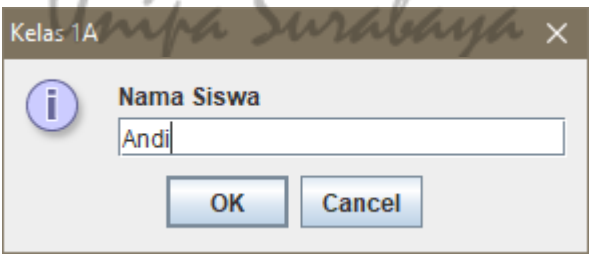
Gambar 81

User harus memasukan jumlah kursi yang ada di Kelas 1A. Output selanjutnya seperti di bawah ini.



Gambar 82

Selanjutnya User memasukan nama Siswa. Program akan mengulang sebanyak jumlah Kelas yang dimasukan tadi. Jika User sudah memasukan dan menakan tombol ok maka output selanjutnya seperti di bawah ini:



Gambar 83

Masih sama seperti output sebelumnya. Saat User menekan tombol OK, pengulangan untuk kelas 1A selesai dan program akan kembali ke perintah memasukan nama kelas seperti gambar diatas. Kemudian program melanjutkan seperti gambar diatas untuk memasukan jumlah Kelas dan seterusnya. Output akhir dari progam ini seperti adalah sebagai berikut:

```
Kelas 1A :
Nama Siswa Budi
Nama Siswa Andi
---------------------------------------
Kelas 1B :
Nama Siswa Dwi
Nama Siswa Budi
Nama Siswa Arif
Nama Siswa Fadli
Nama Siswa Agus
---------------------------------------
BUILD SUCCESSFUL (total time: 19 minutes 40 seconds)
```

Gambar 84

While

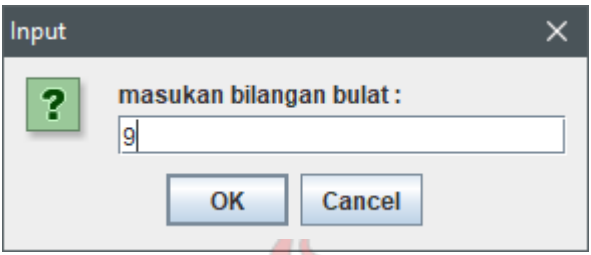
Pengulangan **While** adalah pengulangan yang mempunyai kondisi, jika kondisi tersebut mengembalikan nilai **true** maka program akan mengulang terus. Tapi jika mengembalikan nilai **false** maka program akan berhenti. Contoh penggunaanya seperti potongan kode di bawah ini:

```
int arr[] = new int[4];
boolean hasil = true;
arr[0] = 2;
arr[1] = 5;
arr[2] = 7;
```

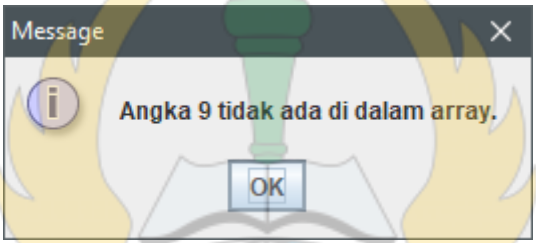
```
        arr[3] = 1;
        while (hasil == true) {
            String input = JOptionPane.showInputDialog(null,
            "masukan bilangan bulat :");
            int cekInputan = Integer.parseInt(input);
            for (int a = 0; a < 4; a++) {
                if (cekInputan == arr[a]) {
                    hasil = false;
                    break;
                } else {
                    hasil = true;
                }
            }
            if (hasil == true) {
                JOptionPane.showMessageDialog(null, "Angka " + cekInputan
                        + " tidak ada di dalam array.");
            } else {
                JOptionPane.showMessageDialog(null, "Angka " + cekInputan
                        + " ada di dalam array.");
            }
        }
    }
```

Potongan kode diatas melakukan pengecekan terhadap inputan User, apakah ada di dalam array atau tidak. Jika ada di dalam variabel array, maka berhenti mengulang tetapi jika tidak ada di dalam variabel array maka program akan mengulang terus sampai kondisi variabel a kurang dari 4 terpenuhi. Kata kunci break digunakan agar program berhenti mengulang jika inputan User

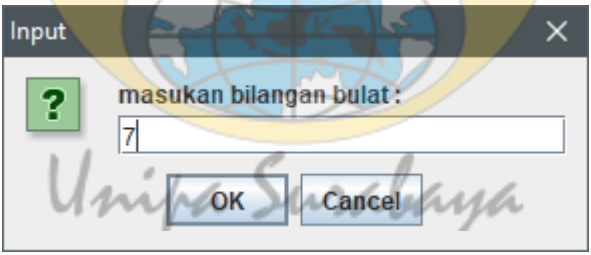
ditemukan di variabel array. Jika isi variabel **hasil** sudah mejadi **false** maka program akan berhenti mengulang dari pengulangan **while**. Output potongan kode di atas adalah sebagai berikut:



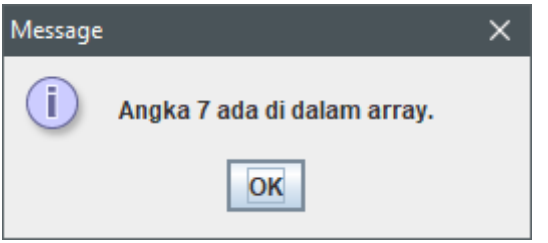
Gambar 85



Gambar 86



Gambar 87



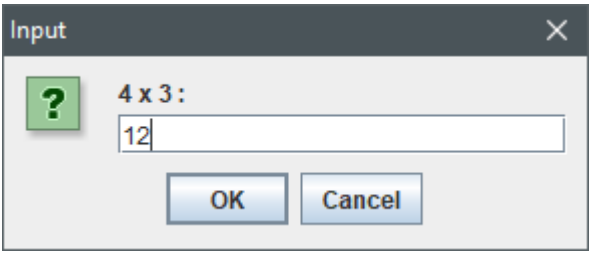
Gambar 88

Do While

Pengulangan ini sama seperti pengulangan **While** namun yang membedakan adalah pengulangan ini melakukan perintah terlebih dahulu kemudian melakukan pengecekan pengulangannya. Contohnya seperti potongan kode di bawah ini:

```
public static void main(String[] args) {
        String input = "";
        int cekbil = 0;
        do{
            input = JOptionPane.showInputDialog(null, "4 x 3 :");
            cekbil = Integer.parseInt(input);
        }while(cekbil != 12);
}
```

Jika User memasukan angka selain 12, program akan selalu mengulang. Output dari potongan kode di atas adalah:



Gambar 89

Unipa Surabaya

BAB 9 Mengenal Class

Seperti yang dijelaskan pada bab sebelumnya Class adalah bagian utama dari pemrograman Java dan Class merupakan cetak biru pemrograman Java untuk menciptakan sebuah Objek. Sebuah Class memiliki atribut dan prosedur atau function. Di bawah ini penjelasan dari atribut, prosedur atau function.

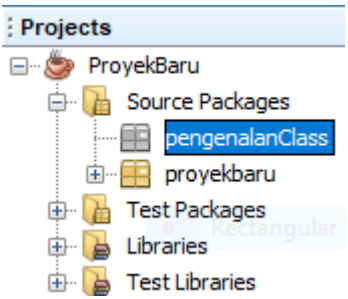
Class

Di dalam membuat program menggunakan Java, memandang class sebagai objek di dunia nyata sangat sering di lakukan. Dalam membuat objek kita harus memikirkan kata benda yang merupakan kata khusus. Sebagai contoh Manusia, hewan atau kendaraan.

Gambar 90

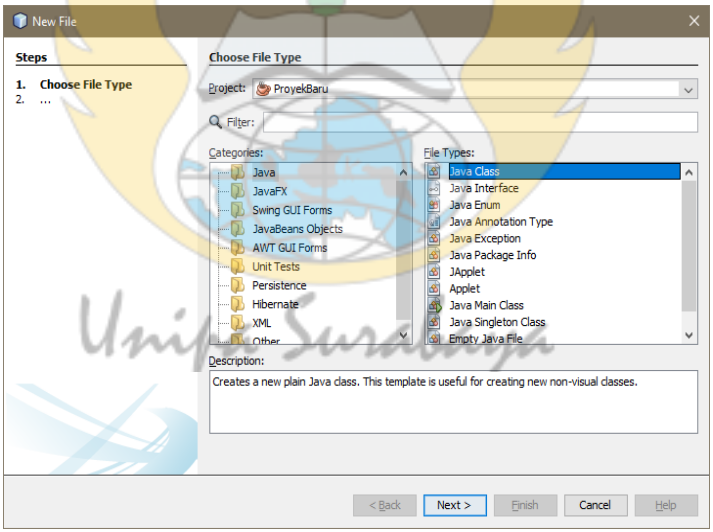
Di dalam Netbeans membuat Class caranya sangat mudah, mari kita langsung praktekan menggunakan **ProyekBaru**:

a. Buat paket baru dengan nama **pengenalanClass**



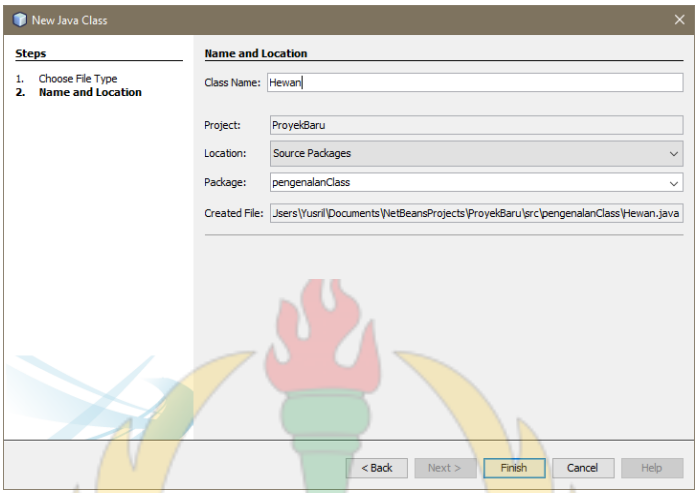
Gambar 91

b. Pilih **File** kemudian pilih **New File**. Pilih kategori **Java** dan tipe file **Java Class**. Klik Next.



Gambar 92

c. Beri nama Class dengan nama **Hewan**. Pastikan package berada di paket **pengenalanClass**. Tekan Finish.



Gambar 93

d. Netbeans akan membuat Class Hewan seperti di bawah ini:

```
package pengenalanClass;


public class Hewan {


}
```

Setelah kita membuat Class hewan, mari kita coba membuat objek baru dari kelas tersebut, contohnya kita buat objek Anjing. Berikut di bawah ini langkah – langkahnya:

a. Buat main class baru dengan nama **MainApp**, letakan di paket **pengenalanClass**.

b. Di dalam main Class **MainApp** ketikan **Hewan anjing = new Hewan()** seperti contoh potongan kode di bawah ini:

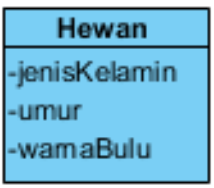
```
public class MainApp {

    public static void main(String[] args) {
        Hewan anjing = new Hewan();
    }

}
```

membuat objek dari class harus menggunakan kata kunci **new** dengan diikuti memanggil constructor Hewan(). Constructor adalah method yang pertama kali dijalankan saat objek diciptakan. Jika tidak ada constructor yang disediakan maka akan dibuatkan constructor default tanpa parameter dan tanpa body. Berikut di bawah ini contohnya:

```
package pengenalanClass;


public class Hewan {

    //constructor default
    public Hewan() {
    }

}
```

Atribut Class

Setiap Class memiliki atribut, atribut adalah kumpulan variabel di dalam Class atau bisa berupa sifat – sifat dari Class.



Gambar 94

Setiap kita membuat objek dari suatu Class, atribut akan selalu ikut. Contoh seperti di bawah ini, kita buat atribut di Class Hewan yang isi atributnya seperti gambar 94:

```
package pengenalanClass;


public class Hewan {

    public String jenisKelamin;
    public int umur;
    public String warnaBulu;


    //constructor default
    public Hewan() {
    }


    //constructor dengan parameter

    public Hewan(String jenisKelamin, int umur, String warnaBulu) {
        this.jenisKelamin = jenisKelamin;
        this.umur = umur;
        this.warnaBulu = warnaBulu;
    }

}
```

Saat kita memanggil kata kunci new dan objek Anjing dibuat, maka objek anjing memiliki atribut jenisKelamin, umur dan warnaBulu. Perhatikan potongan kode di bawah ini:

```
public static void main(String[] args)
        {  Hewan anjing = new Hewan();
        anjing.jenisKelamin = "Jantan";
        anjing.umur = 5;
        anjing.warnaBulu = "Cokelat";

        System.out.println("Anjing " + anjing.jenisKelamin + " berumur "

                + anjing.umur + " tahun berbulu " + anjing.warnaBulu);

}
```

Hasil dari potongan kode di atas akan mencetak kalimat **Anjing Jantan berumur 5 tahun berbulu Cokelat**.

Prosedur atau Function

Prosedur atau Function bisa dilihat sebagai perilaku sebuah Class, contohnya berlari, melompat, menggigit dan lain lain. Perbedaan antara Prosedur dan Function adalah:

a. Function

Function adalah bagian dari program yang jika dipanggil akan menjalankan perintah dan mengembalikan hasil. Jika hasil dikembalikan dalam bentuk String maka function harus mengembalikan bentuk String, bila mengembalikan dalam bentuk int maka function juga harus dalam bentu int. Jenis Function harus sesuai nilai yang akan dikembalikan saat menjalankan perintah di dalamnnya. Contohnya seperti potongan kode di bawah ini, Class Hewan diberi Function **melompat()**:

```
public String melompat() {

        String perilaku = "Anjing melompati pagar.";
        return perilaku;
}
```


Cara menggunakannya seperti gambar di bawah ini:

```
public static void main(String[] args)
        {  Hewan anjing = new Hewan();
        anjing.jenisKelamin = "Jantan";
        anjing.umur = 5;
        anjing.warnaBulu = "Cokelat";
        System.out.println("Anjing " + anjing.jenisKelamin + " berumur "

                + anjing.umur + " tahun berbulu " + anjing.warnaBulu);
```

```
        //cara memanggi prosedur
        System.out.println(anjing.melompat());
    }
```

Hasil dari potongan kode di atas adalah:

```
Anjing Jantan berumur 5 tahun berbulu Cokelat
Anjing melompati pagar.
```

Gambar 95

Sebuah Function juga dapat menerima parameter, contohnya seperti di bawah ini:

```
public String melompat(String parameter) {

        String perilaku = "Anjing melompati " + parameter + ".";
        return perilaku;
}
```

Cara memanggil Function dengan parameter adalah seperti di bawah ini:

```
public static void main(String[] args)
        {  Hewan anjing = new Hewan();
        anjing.jenisKelamin = "Jantan";
        anjing.umur = 5;
        anjing.warnaBulu = "Cokelat";

        System.out.println("Anjing " + anjing.jenisKelamin + " berumur "

                + anjing.umur + " tahun berbulu " + anjing.warnaBulu);


        //cara memanggi prosedur
        System.out.println(anjing.melompat("Kursi"));
    }
```

Saat Function melompat dipanggil, parameter diisi dengan kata **Kursi**. Hasil dari potongan kode di atas adalah:

```
Anjing Jantan berumur 5 tahun berbulu Cokelat
Anjing melompati Kursi.
```

Gambar 96

Prosedur

Prosedur adalah bagian dari program yang jika dipanggil akan menjalankan perintah sesuai yang dideskripsikan tapi tidak mengembalikan perintah. Contoh prosedur di Class Hewan seperti potongan kode di bawah ini:

```
public void memakan(){

        System.out.println("Anjing sedang memakan daging.");
}
```

Contoh penggunaan prosedur adalah seperti di bawah ini:

```
public static void main(String[] args)
        { Hewan anjing = new Hewan();
        anjing.jenisKelamin = "Jantan";
        anjing.umur = 5;
        anjing.warnaBulu = "Cokelat";


        System.out.println("Anjing " + anjing.jenisKelamin + " berumur "

                + anjing.umur + " tahun berbulu " + anjing.warnaBulu);


        //cara memanggi prosedur
        anjing.memakan();
}
```

Hasil dari perintah di atas adalah:

```
Anjing Jantan berumur 5 tahun berbulu Cokelat
Anjing sedang memakan daging.
```

Gambar 97

Sama seperti Function, prosedur juga dapat menerima parameter, contohnya seperti potongan kode di bawah ini:

```
public void memakan(String parameter) {
    System.out.println("Anjing sedang memakan
    "+parameter+".");
}
```

Cara menggunakan prosedur tersebut seperti potongan kode di bawah ini:

```
public static void main(String[]
        args) {  Hewan anjing = new
        Hewan();
        anjing.jenisKelamin =
        "Jantan";  anjing.umur = 5;
        anjing.warnaBulu =
        "Cokelat";

        System.out.println("Anjing " + anjing.jenisKelamin + " berumur
        "
                + anjing.umur + " tahun berbulu " + anjing.warnaBulu);

        //cara memanggi prosedur
        anjing.memakan("Daging");
    }
```

Bila di jalankan, hasilnya sama seperti gambar diatas.

Unipa Surabaya

Data Diri Penulis

Nama	:	Dwi Hastuti
Tempat dan Tanggal Lahir	:	Banjarbaru, 5 April 1989
		Merupakan tenaga pengajar di Universitas PGRI Adi Buana Surabaya. Mendapat gelar S1 dari jurusan Sistem Informasi di Universitas Pembangunan Nasional 'Veteran' Jawa Timur. Mendapat gelar S2 dari jurusan Sistem Informasi di Institut Teknologi Bandung.
Nama	:	Yusril Arief
Tempat dan Tanggal Lahir	:	Surabaya, 10 November 1989
		Bekerja sebagai *staff Research and Development* di perusahaan swasta berskala internasional yang bergerak dibidang teknologi informasi, khususnya mengenai *Enterprise Resource Planning* (ERP). Mendapatkan gelar S1 dari jurusan Sistem Informasi di Universitas Pembangunan Nasional 'Veteran' Jawa Timur.

Unipa Surabaya

DASAR-DASAR PEMOGRAMAN JAVA UNTUK PEMULA MENGGUNAKAN NETBEANS

Buku ini dapat dimanfaatkan untuk pemula dalam memelajari dasar- dasar pemrograman Java. Masing-masing materi disajikan secara sistematis disertai contoh agar lebih mudah dipahami dan dipelajari. Tak lupa penulis mengucapkan terima kasih yang sebesarnya kepada pihak-pihak yang telah ikut membantu dan memberi masukan sehingga buku ini siap untuk dimanfaatkan oleh masyarakat. Saran yang membangun, sangat penulis harapkan demi perbaikan buku untuk edisi selanjutnya agar semakin lebih baik. Salam literasi

unipress

ISBN: 978-623-416-011-6