



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

LEMBAGA PENGEMBANGAN DAN PEMBERDAYAAN PENDIDIK DAN TENAGA KEPENDIDIKAN

BIDANG KELAUTAN PERIKANAN TEKNOLOGI INFORMASI DAN KOMUNIKASI

(LPPPTK KPTK) GOWA

2018

RPL

REKAYASA
PERANGKAT LUNAK

Mengimplementasikan Pemrograman
Berorientasi Objek

Penulis:

Alun Sujjadah, S.Kom., MT.

Modul,
Pelatihan
Berbasis
Kompetensi



MODUL
PENGEMBANGAN KEPROFESIAN
BERKELANJUTAN BERBASIS KOMPETENSI

MENGIIMPLEMENTASIKAN
PEMROGRAMAN BERORIENTASI OBJEK
J.620100.018.02



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN R.I.
DIREKTORAT JENDERAL GURU DAN TENAGA KEPENDIDIKAN
LEMBAGA PENGEMBANGAN DAN PEMBERDAYAAN PENDIDIK DAN TENAGA KEPENDIDIKAN
BIDANG KELAUTAN, PERIKANAN,
DAN TEKNOLOGI INFORMASI DAN KOMUNIKASI
GOWA

KATA PENGANTAR

Modul pengembangan keprofesian berkelanjutan (PKB) berbasis kompetensi merupakan salah satu media pembelajaran yang dapat digunakan sebagai media transformasi pengetahuan, keterampilan dan sikap kerja kepada peserta pelatihan untuk mencapai kompetensi tertentu berdasarkan program pelatihan yang mengacu kepada Standar Kompetensi.

Modul pelatihan ini berorientasi kepada pelatihan berbasis kompetensi (Competence Based Training) diformulasikan menjadi 3 (tiga) buku, yaitu Buku Informasi, Buku Kerja dan Buku Penilaian sebagai satu kesatuan yang tidak terpisahkan dalam penggunaannya sebagai referensi dalam media pembelajaran bagi peserta pelatihan dan instruktur, agar pelaksanaan pelatihan dapat dilakukan secara efektif dan efisien. Untuk memenuhi kebutuhan pelatihan berbasis kompetensi tersebut, maka disusunlah modul pelatihan berbasis kompetensi dengan judul "Mengimplementasikan Pemrograman Berorientasi Objek".

Kami menyadari bahwa modul yang kami susun ini masih jauh dari sempurna. Oleh karena itu, kami sangat mengharapkan saran dan masukan untuk perbaikan agar tujuan dari penyusunan modul ini menjadi lebih efektif.

Demikian kami sampaikan, semoga Tuhan YME memberikan tuntunan kepada kita dalam melakukan berbagai upaya perbaikan dalam menunjang proses pelaksanaan pembelajaran di lingkungan direktorat guru dan tenaga kependidikan.

Gowa, April 2018
Kepala LPPPTK KPTK,

Prof. Dr. Irwan, M.Pd

DAFTAR ISI

KATA PENGANTAR	2
DAFTAR ISI	3
ACUAN STANDAR KOMPETENSI KERJA DAN SILABUS DIKLAT	4
A. Acuan Standar Kompetensi Kerja	4
B. Kemampuan yang Harus Dimiliki Sebelumnya	6
C. Silabus Diklat Berbasis Kompetensi	7
LAMPIRAN	11
1. BUKU INFORMASI	11
2. BUKU KERJA	11
3. BUKU PENILAIAN	11

ACUAN STANDAR KOMPETENSI KERJA DAN SILABUS DIKLAT

A. Acuan Standar Kompetensi Kerja

Materi modul pelatihan ini mengacu pada unit kompetensi terkait yang disalin dari Standar Kompetensi Kerja Sub-golongan Pemrograman dengan uraian sebagai berikut:

KODE UNIT : J.620100.018.02

JUDUL UNIT : Mengimplementasikan Pemrograman Berorientasi Objek

DESKRIPSI UNIT : Unit kompetensi ini berhubungan dengan sikap, pengetahuan, dan keterampilan yang dibutuhkan dalam membuat perangkat lunak aplikasi dalam bahasa pemrograman berorientasi objek.

ELEMEN KOMPETENSI		KRITERIA UNJUK KERJA	
1.	Membuat program berorientasi objek dengan memanfaatkan class	1.1	Program dengan menggunakan class dibuat.
		1.2	Properti class yang akan direalisasikan dalam bentuk prosedur/fungsi dibuat.
		1.3	Data didalam class dibuat mandiri.
		1.4	Hak akses dari tipe data (private, protected, public) dikelola.
2.	Menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas	2.1	Tipe data diidentifikasi.
		2.2	Sintaks program dikuasai sesuai dengan bahasa pemrogramannya.
		2.3	Control program dikuasai.
3.	Membuat program dengan konsep berbasis objek	3.1	Inheritance pada class diterapkan.
		3.2	Polymorphism pada class diterapkan.
		3.3	Overloading pada class diterapkan.
4.	Membuat program object oriented dengan interface dan paket	4.1	Interface class program dibuat.
		4.2	Paket dengan program dibuat.
5.	Mengkompilasi Program	5.1	Kesalahan dapat dikoreksi.
		5.2	Program bebas salah sintaks dihasilkan.

BATASAN VARIABEL

1. Konteks Variabel:

- 1.1 Class, object, interface dan paket merupakan istilah untuk pengaturan struktur kode pada pemrograman berorientasi objek.
- 1.2 Private, protected, public merupakan hal akses kelas pada pemrograman berorientasi objek.
- 1.3 Control program merupakan mekanisme untuk mengatur alur dan logika program dengan menggunakan pengulangan atau percabangan.
- 1.4 Inheritance, polymorphism and overloading merupakan konsep pada pemrograman berorientasi objek.

2. Peralatan dan perlengkapan

- 2.1 Peralatan
 - 2.1.1 Perangkat lunak terkait
 - 2.1.2 Algoritma program
- 2.2 Perlengkapan
 - 2.2.1 Petunjuk teknis bahasa pemrograman terkait

3. Peraturan yang diperlukan

- 3.1 Undang-Undang Nomor 11 Tahun 2008 tentang Informasi dan Transaksi Elektronik
- 3.2 Undang-Undang Nomor 14 Tahun 2008 tentang Keterbukaan Informasi Publik

4. Norma dan standar

- 4.1 Norma
 - 4.1.1 Legalitas dan etika yang terkait dengan profesi bidang teknologi informasi
- 4.2 Standar
 - 4.2.1 SNI ISO/IEC 20000-1:2009 Teknologi informasi Manajemen layanan Bagian 1: Spesifikasi
 - 4.2.2 SNI ISO/IEC 20000-2:2009 Teknologi informasi Manajemen layanan Bagian 2: Aturan Praktik
 - 4.2.3 Standar Pemrograman berorientasi objek yang ada

PANDUAN PENILAIAN

1. Konteks penilaian

- 1.1 Penilaian kompetensi dapat dilakukan dengan cara demonstrasi/ praktik, dan/atau di tempat kerja.

2. Persyaratan kompetensi

- 2.1 J.620100.004.02 : Menggunakan Struktur Data
- 2.2 J.620100.017.02 : Mengimplementasikan Pemrograman Terstruktur

3. Pengetahuan dan keterampilan yang dibutuhkan

- 3.1 Pengetahuan
 - 3.1.1 Penggunaan bahasa pemrograman yang sesuai
- 3.2 Keterampilan
 - 3.2.1 Mengoperasikan komputer

4. Sikap kerja yang diperlukan

- 4.1 Cekatan
- 4.2 Teliti

5. Aspek kritis

- 5.1 Ketepatan penggunaan polimorphy pada program yang dibuat

B. Kemampuan yang Harus Dimiliki Sebelumnya

Ada pun kemampuan yang harus dimiliki sebelumnya sebagai berikut:

- Tidak ada

C. Silabus Diklat

Judul Unit Kompetensi : Mengimplementasikan Pemrograman Berorientasi Objek

Kode Unit Kompetensi : J.620100.018.02

Deskripsi Unit Kompetensi : Unit kompetensi ini berhubungan dengan sikap, pengetahuan, dan keterampilan yang dibutuhkan dalam membuat perangkat lunak aplikasi dalam bahasa pemrograman berorientasi objek.

Perkiraan Waktu Pelatihan :

Tabel Silabus Unit Kompetensi :

Elemen Kompetensi	Kriteria Unjuk Kerja	Indikator Unjuk Kerja	Materi Diklat			Perkiraan Waktu Diklat (JP)	
			Pengetahuan (P)	Keterampilan (K)	Sikap (S)	P	K
1. Membuat program berorientasi objek dengan memanfaatkan class	1.1 Program dengan menggunakan class dibuat.	<ul style="list-style-type: none"> ✓ Mampu memahami konsep class dan object ✓ Mampu membuat program dengan menggunakan class ✓ Harus benar dan sesuai dengan kaidah bahasa pemrograman 	<ul style="list-style-type: none"> ✓ Konsep alur berfikir menggunakan Class dan Object ✓ Instance objek 	Membuat program dengan menggunakan class		15'	30'
	1.2 Properti class yang akan direalisasikan dalam bentuk prosedur/fungsi dibuat.	<ul style="list-style-type: none"> ✓ Mampu memahami konsep properti yang terdiri dari prosedur dan fungsi ✓ Mampu membuat properti class yang akan direalisasikan dalam bentuk prosedur/fungsi 	<ul style="list-style-type: none"> ✓ Konsep properti, metode dalam bentuk prosedur atau fungsi ✓ Aturan pembuatan prosedur dan fungsi ✓ Metode Setter ✓ Metode Getter ✓ Keyword this dan new 	Membuat properti class yang akan direalisasikan dalam bentuk prosedur dan fungsi		15'	60'

		✓ Harus benar dan sesuai dengan kaidah pemrograman					
	1.3 Data didalam class dibuat mandiri.	<ul style="list-style-type: none"> ✓ Dapat menjelaskan atribut class ✓ Mampu membuat data didalam class secara mandiri. ✓ Harus sesuai dengan kebutuhan permasalahan 	<ul style="list-style-type: none"> ✓ Cara mendeklarasikan data atau atribut dalam class ✓ Ruang lingkup (scope) data 	Membuat data atau atribut didalam class		10'	30'
	1.4 Hak akses dari tipe data (private, protected, public) dikelola.	<ul style="list-style-type: none"> ✓ Dapat memahami hak akses dari tipe data ✓ Mampu mengelola hak akses dari tipe data (private, protected, public) ✓ Harus sesuai dengan kebutuhan permasalahan 	Konsep tentang modifier dari tipe data (private, protected, public)	Mengelola hak akses (modifier) dari tipe data (private, protected, public)		20'	45'
2. Menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas	2.1 Tipe data diidentifikasi.	<ul style="list-style-type: none"> ✓ Mampu mengidentifikasi tipe data ✓ Harus efisien 	Jenis-jenis tipe data	Mengidentifikasi tipe data		10'	20'
	2.2 Sintaks program dikuasai sesuai dengan bahasa pemrogramannya.	<ul style="list-style-type: none"> ✓ Dapat mengidentifikasi penulisan sintaks dalam bahasa pemrograman ✓ Mampu menguasai sintaks program sesuai dengan bahasa pemrogramannya ✓ Cermat, tekun dan teliti 	<ul style="list-style-type: none"> ✓ Aturan penulisan sintaks dalam bahasa pemrograman ✓ Cara menuliskan sintaks program 	Menguasai sintaks program sesuai dengan bahasa pemrogramannya		15'	15'
	2.3 Control program dikuasai.	<ul style="list-style-type: none"> ✓ Mampu menjelaskan jenis-jenis control 	<ul style="list-style-type: none"> ✓ Jenis-jenis control program 	Menguasai control program		15'	60'

		<ul style="list-style-type: none"> program beserta kegunaannya ✓ Mampu menguasai Control program ✓ Cermat, tekun dan teliti 	<ul style="list-style-type: none"> ✓ Fungsi dari masing-masing control program 				
3. Membuat program dengan konsep berbasis objek	3.1 Inheritance pada class diterapkan.	<ul style="list-style-type: none"> ✓ Dapat mengidentifikasi kegunaan inheritance ✓ Mampu menerapkan inheritance pada class ✓ Harus berfikir analitis dan sesuai kaidah bahasa pemrograman 	<ul style="list-style-type: none"> ✓ Konsep inheritance pada class ✓ Cara penulisan 2 class atau lebih pada file yang berbeda ✓ Cara penulisan 2 class atau lebih pada file yang sama 	Menerapkan inheritance pada class		30'	60'
	3.2 Polymorphism pada class diterapkan.	<ul style="list-style-type: none"> ✓ Mampu memahami polymorphism ✓ Mampu menerapkan polymorphism pada class ✓ Harus berfikir analitis dan sesuai kaidah bahasa pemrograman 	<ul style="list-style-type: none"> ✓ Konsep Polymorphism ✓ Abstract class 	Menerapkan polymorphism pada class		15'	45'
	3.3 Overloading pada class diterapkan.	<ul style="list-style-type: none"> ✓ Mampu memahami jenis-jenis overloading ✓ Mampu menerapkan overloading pada class ✓ Harus berfikir analitis dan sesuai kaidah bahasa pemrograman 	Konsep Overloading dan Overriding Class Konstruktur	Menerapkan overloading pada class		15'	30'
4. Membuat program object oriented dengan	4.1 <i>Interface class</i> program dibuat.	<ul style="list-style-type: none"> ✓ Mampu memahami interface ✓ Mampu membuat interface class program 	<ul style="list-style-type: none"> ✓ Definisi dan cara mendeklarasikan Interface 	Membuat interface class program		10'	15'

interface dan paket		✓ Sesuai kaidah bahasa pemrograman	✓ Aturan penulisan atribut dan metode dalam interface				
	4.2 Paket dengan program dibuat.	Mampu membuat paket dengan program	<ul style="list-style-type: none"> ✓ Aturan penamaan dan penyimpanan dalam folder ✓ Aturan class package 	<ul style="list-style-type: none"> ✓ Melakukan import class ✓ Membuat paket dengan program 		15'	15'
5. Mengkompilasi Program	5.1 Kesalahan dapat dikoreksi.	<ul style="list-style-type: none"> ✓ Dapat mengidentifikasi jenis-jenis kesalahan dalam program ✓ Mampu mengoreksi kesalahan ✓ Cermat, tekun dan teliti 	<ul style="list-style-type: none"> ✓ Jenis-jenis kesalahan dalam program ✓ Pengenalan Error Debugging 	Melakukan koreksi kesalahan		15'	30'
	5.2 Program bebas salah sintaks dihasilkan.	<ul style="list-style-type: none"> ✓ Mampu menghasilkan program yang bebas dari kesalahan sintaks ✓ Cermat, tekun dan teliti 	Aturan penulisan sintaks Try Catch Finally	Menghasilkan program bebas salah sintaks		15'	30'

LAMPIRAN

1. BUKU INFORMASI
2. BUKU KERJA
3. BUKU PENILAIAN

1

LAMPIRAN BUKU INFORMASI



BUKU INFORMASI

MENGIIMPLEMENTASIKAN PEMROGRAMAN BERORIENTASI OBJEK J.620100.018.02



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN R.I.
DIREKTORAT JENDERAL GURU DAN TENAGA KEPENDIDIKAN
LEMBAGA PENGEMBANGAN DAN PEMBERDAYAAN PENDIDIK DAN TENAGA KEPENDIDIKAN
BIDANG KELAUTAN, PERIKANAN,
DAN TEKNOLOGI INFORMASI DAN KOMUNIKASI
GOWA

DAFTAR ISI

DAFTAR ISI	2
DAFTAR GAMBAR.....	4
DAFTAR LISTING	6
DAFTAR TABEL	7
BAB I PENDAHULUAN	8
A. TUJUAN UMUM	8
B. TUJUAN KHUSUS	8
BAB II Membuat program berorientasi obyek dengan memanfaatkan class ...	9
A. Pengetahuan yang Diperlukan dalam Membuat Program Berorientasi Obyek dengan Memanfaatkan Class	9
1. Konsep Class dan Object	9
2. Method	13
3. Data Variabel	14
4. Modifier	18
5. Implementasi Pembuatan Class dan Object.....	25
B. Keterampilan yang Diperlukan dalam Membuat Program Berorientasi Obyek dengan Memanfaatkan Class	28
C. Sikap Kerja yang Diperlukan dalam Membuat Program Berorientasi Obyek dengan Memanfaatkan Class.....	28
BAB III Menggunakan Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas	29
A. Pengetahuan yang Diperlukan dalam Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas	29
1. Variabel, tipe data dan konstanta	30
2. Control Program.....	
B. Keterampilan yang Diperlukan dalam Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas.....	37
C. Sikap Kerja yang Diperlukan dalam Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas.....	37
BAB IV Membuat Program Dengan Konsep Berbasis Obyek	38

A.	Pengetahuan yang Diperlukan dalam Membuat Program	
	Dengan Konsep Berbasis Obyek	38
	1. Inheritance	38
	2. Constructor.....	43
	3. Polymorphism dan Overriding	45
B.	Keterampilan yang Diperlukan dalam Membuat Program	
	Dengan Konsep Berbasis Obyek	48
C.	Sikap Kerja yang Diperlukan dalam Membuat Program	
	Dengan Konsep Berbasis Obyek	48
BAB V	Membuat Program Object Oriented Dengan Interface dan Paket	49
A.	Pengetahuan yang Diperlukan dalam Program Object	
	Oriented Dengan Interface dan Paket.....	49
	1. Interface	49
	2. Langkah Pembuatan Interface	51
	3. Membuat paket dengan program	57
B.	Keterampilan yang Diperlukan dalam Membuat Program Object	
	Oriented Dengan Interface dan Paket.....	72
C.	Sikap Kerja yang Diperlukan dalam Membuat Program Object	
	Oriented Dengan Interface dan Paket.....	73
BAB VI	Mengkompilasi Program	74
A.	Pengetahuan yang Diperlukan dalam Mengkompilasi Program	74
	1. Perbaiki Kesalahan	74
	2. Program Debugging	78
B.	Keterampilan yang Diperlukan dalam Mengkompilasi Program	84
C.	Sikap Kerja yang Diperlukan dalam Mengkompilasi Program	84
DAFTAR PUSTAKA	85
A.	Buku Referensi	85
B.	Referensi Lainnya	85
DAFTAR ALAT DAN BAHAN	86
A.	DAFTAR PERALATAN/MESIN	86
B.	DAFTAR BAHAN	86
DAFTAR PENYUSUN	87

DAFTAR GAMBAR

Gambar 2.1 Analogi class dan object	9
Gambar 2.2 Contoh diagram class	11
Gambar 2.3 Ruang lingkup	11
Gambar 2.4 Pembuatan class dan object	12
Gambar 2.5 Ilustrasi class dan objek	13
Gambar 2.6 Bagian method	13
Gambar 2.7 Sintaks penulisan method.....	14
Gambar 2.8 Member Variabel dan Local Variabel.....	16
Gambar 2.9 Tampilan IDE Netbeans.....	23
Gambar 2.10 Penentuan jenis project.....	23
Gambar 2.11 Pengisian parameter project	24
Gambar 2.12 Pembuatan class.....	25
Gambar 2.13 Pengisian Class Name.....	25
Gambar 4.1 Contoh Hierarki Class Sepeda	38
Gambar 4.2 Hasil output program	44
Gambar 4.3 Hasil Output	48
Gambar 5.1 Contoh interface	50
Gambar 5.2 Contoh class yang mengimplementasikan interface.....	50
Gambar 5.3 Tampilan IDE Netbeans.....	51
Gambar 5.4 Penentuan jenis project.....	51
Gambar 5.5 Pengisian parameter project Interface	52
Gambar 5.6 Pembuatan interface	52
Gambar 5.7 Pengisian Class Name	53
Gambar 5.8 Pembuatan Java Class.....	54
Gambar 5.9 Pengisian nama Class.....	54
Gambar 5.10 Hasil generate listing Netbeans	55
Gambar 5.11 Hasil output program.....	56
Gambar 5.12 Tampilan Dialog box Open Project	57
Gambar 5.13 Menu Clean and Build Project.....	58
Gambar 5.14 Hasil dari Pembuatan file JAR.....	58
Gambar 5.15 Setup EXE4J	59
Gambar 5.16 Persetujuan Lisensi	60
Gambar 5.17 Pemilihan direktori instalasi.....	60

Gambar 5.18 Tampilan awal EXE4J	61
Gambar 5.19 Pemilihan Tipe project	62
Gambar 5.20 Pengisian nama aplikasi dan output direktori	62
Gambar 5.21 Tipe executable	63
Gambar 5.22 Pengisian file JAR dan Main Class	64
Gambar 5.23 Konfigurasi JRE	64
Gambar 5.24 Persetujuan lisensi Inno Setup	65
Gambar 5.25 Pemilihan folder instalasi	66
Gambar 5.26 Pembuatan shortcut	66
Gambar 5.27 Instalasi Inno Setup Preprocessor	67
Gambar 5.28 Pemilihan Tipe installer	68
Gambar 5.29 Dialog pengisian parameter aplikasi	68
Gambar 5.30 Spesifikasi folder hasil instalasi	69
Gambar 5.31 Spesifikasi Aplikasi	69
Gambar 5.32 Spesifikasi shortcut aplikasi	70
Gambar 5.33 Dokumentasi Aplikasi	70
Gambar 5.34 Pemilihan Bahasa	71
Gambar 5.35 Penentuan folder hasil pembuatan installer	71
Gambar 5.36 Dialog kompilasi script	72
Gambar 6.1 Hasil output program pembagian	76
Gambar 6.2 Output hasil pembagian dengan bilangan 0	77
Gambar 6.3 Pesan kesalahan dari sistem	77
Gambar 6.4 Proses debugging	83
Gambar 6.5 Menu Debugging	83
Gambar 6.6 Tampilan debugger	84

DAFTAR LISTING

Listing 2.1 Method berupa prosedur tanpa parameter	15
Listing 2.2 Method berupa fungsi dengan parameter dan return value integer	15
Listing 2.3 Penggunaan modifier public	18
Listing 2.4 Penggunaan dan pengaksesan modifier private	19
Listing 2.5 Penggunaan modifier protected	20
Listing 2.6 Deklarasi Member Variabel	26
Listing 2.7 Pembuatan Setter Method	26
Listing 2.8 Getter Method	27
Listing 2.9 Pembuatan object didalam main method	27
Listing 3.1 Penggunaan tipe data dan variabel	31
Listing 3.2 Struktur percabangan menggunakan IF	34
Listing 3.3 Struktur percabangan menggunakan switch	35
Listing 3.4 Contoh perulangan	36
Listing 4.1 Class Sepeda	40
Listing 4.2 Class SepedaGunung	40
Listing 4.3 Class SepedaBalap	41
Listing 4.4 Class SepedaMotor	41
Listing 4.5 Class Sepeda 2 Tak	42
Listing 4.6 Class Sepeda 4 Tak	42
Listing 4.7 Super class Pesawat dengan 2 (dua) constructor method	43
Listing 4.8 Sub class PesawatKomersil	44
Listing 4.9 Super Class Guru	45
Listing 4.10 Sub Class Mapel	46
Listing 4.11 Sub class Jam Mengajar	47
Listing 4.12 Class Sekolah	47
Listing 5.1 Interface Rumah	53
Listing 5.2 Kode program class RumahSewa	55
Listing 5.3 Class RumahSewa	56
Listing 6.1 Contoh program sederhana	76
Listing 6.2 Perbaikan kesalahan program	77

DAFTAR TABEL

Tabel 2.1 Nilai atau isi dari masing-masing atribut	10
Tabel 2.2 Scope akses modifier	22

BAB I PENDAHULUAN

A. TUJUAN UMUM

Setelah mempelajari modul ini peserta diharapkan mampu membuat perangkat lunak aplikasi dalam bahasa pemrograman berorientasi objek

B. TUJUAN KHUSUS MENGIMPLEMENTASIKAN PEMROGRAMAN BERBASIS OBYEK

Adapun tujuan mempelajari unit kompetensi melalui buku informasi Mengimplementasikan Pemrograman Berorientasi Objek ini guna memfasilitasi peserta sehingga pada akhir diklat diharapkan memiliki kemampuan sebagai berikut:

1. Membuat program berorientasi objek dengan memanfaatkan class
2. Menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas
3. Membuat program dengan konsep berbasis objek
4. Membuat program object oriented dengan interface dan paket
5. Mengkompilasi program

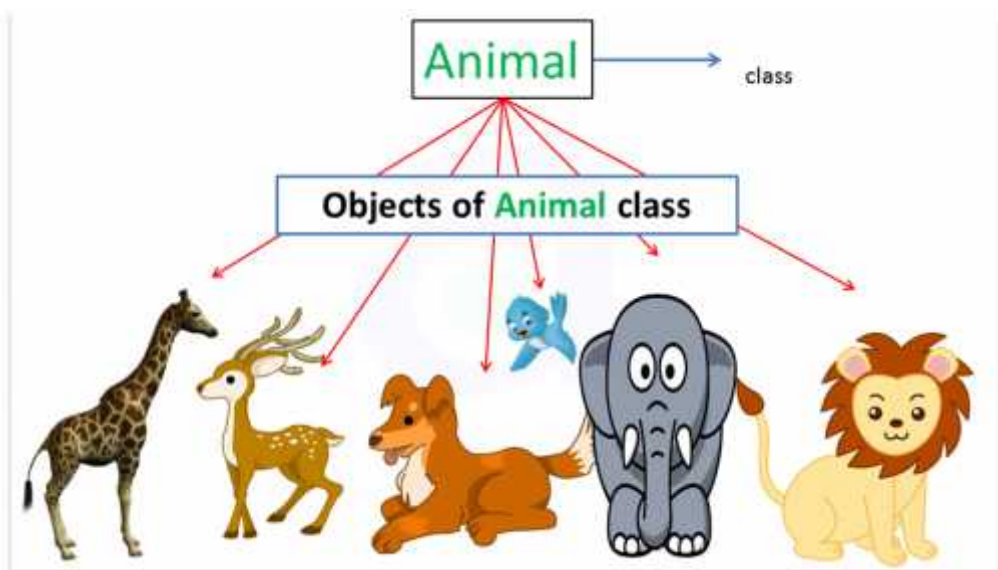
BAB II

MEMBUAT PROGRAM BERORIENTASI OBJEK DENGAN MEMANFAATKAN CLASS

A. Pengetahuan yang Diperlukan dalam Membuat Program Berorientasi Objek dengan Memanfaatkan Class

1. Konsep Class dan Object

Paradigma pemrograman berorientasi objek adalah merepresentasikan sebuah permasalahan kedalam bentuk class dan object. Seorang programmer diupayakan untuk membuat program lebih dekat dengan memodelkan cara orang berpikir seperti dunia nyata. Class dan object adalah sebuah konsep yang saling berkaitan erat dan tidak dapat dipisahkan. Class merupakan cetak biru atau kerangka dalam pembuatan program, sedangkan object adalah hasil instance atau penciptaan dari sebuah class. Class merupakan prototipe yang mendefinisikan attribute dan behavior secara umum. Saat implementasi kedalam sebuah program attribute dimodelkan sebagai variabel dan behavior dimodelkan sebagai method. Sebagai analogi untuk mempermudah memahami konsep tentang class dan object, perhatikan contoh gambar 2.1



Gambar 2.1 Analogi class dan objek

Terdapat class dengan nama `Animal`, dimana didalam sebuah class akan terdapat attribute (variabel) dan method (behavior) yang akan dibahas pada poin berikutnya. Pada sebuah class didefinisikan beberapa hal yang bersifat umum, sehingga akan dapat digunakan untuk penciptaan object dari class tersebut. Sebagai contoh didalam class `Animal` mempunyai variabel nama, umur dan jenis_hidup. Ketiga variabel tersebut digunakan karena bersifat umum, setiap `Animal` pasti mempunyai nama, umur dan jenis hidupnya di air, darat atau udara.

Selain itu didalam class terdapat beberapa method seperti `setNama`, `setUmur`, `setJenisHidup` yang berfungsi untuk memberikan nilai kepada variabel. Ketika sudah tercipta sebuah class yang merupakan kerangka atau cetak biru, maka akan dapat dibuat sebuah object seperti jerapah, kijang, anjing, burung, gajah dan singa karena objek-objek tersebut mempunyai nama, umur dan jenis hidup. Tiap-tiap object akan memiliki nilai variabel yang berbeda-beda tergantung dengan nilai yang dilewatkan pada method seperti pada contoh tabel 2.1

Tabel 2.1 Nilai atau isi dari masing-masing atribut

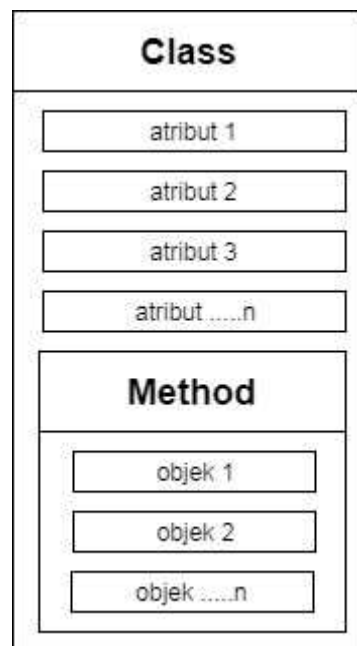
Objek	Nama	Umur	Jenis Hidup
Objek 1	Singa	10 tahun	Darat
Objek 2	Gajah	7 tahun	Darat
Objek 3	Burung	2 tahun	Udara
Objek 4	Ikan	2 tahun	Air

Pada pemrograman berorientasi objek maka setiap permasalahan akan dimodelkan dalam bentuk diagram class. Adapun contoh dari pemodelannya seperti pada gambar 2.2



Gambar 2.2 Contoh diagram class

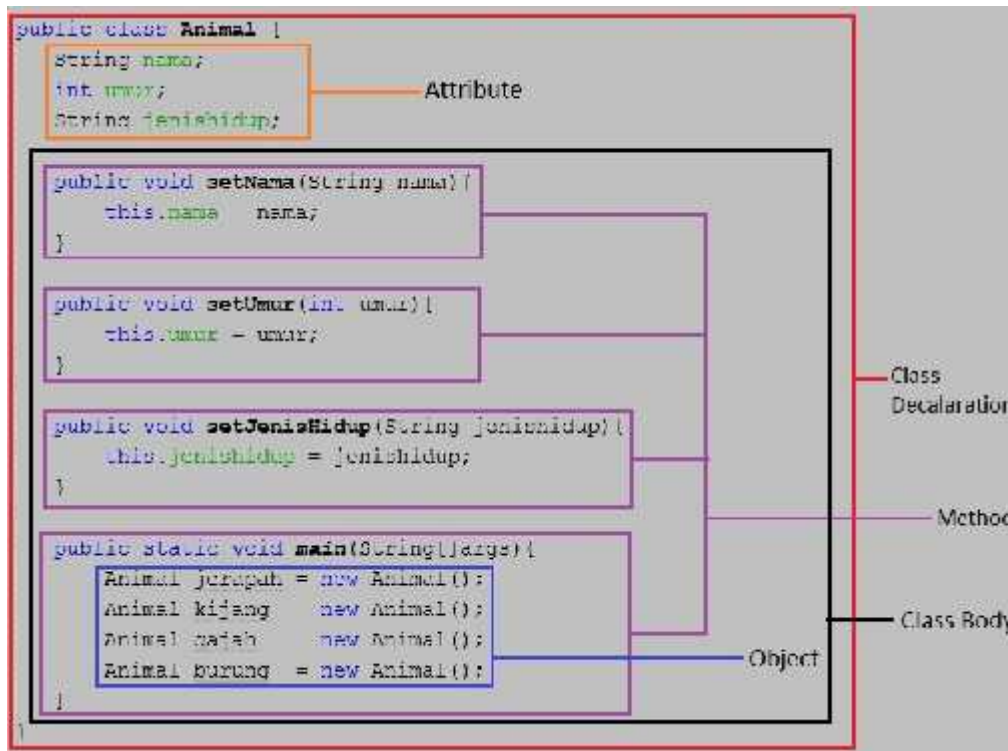
Ruang lingkup (scope) pembuatan class dan object adalah class sebagai pembungkus sebuah object dan method. Pembuatan sebuah object pada umumnya dilakukan didalam method. Untuk memudahkan pemahaman, perhatikanlah gambar 2.3



Gambar 2.3 Ruang lingkup

Deklarasi pembuatan class menggunakan kata kunci class, sedangkan penciptaan object dari sebuah class atau yang lebih dikenal dengan proses instantiate dalam sebuah program yaitu menggunakan operator new dengan sintaks yaitu `nameOfClass nameOfObject =`

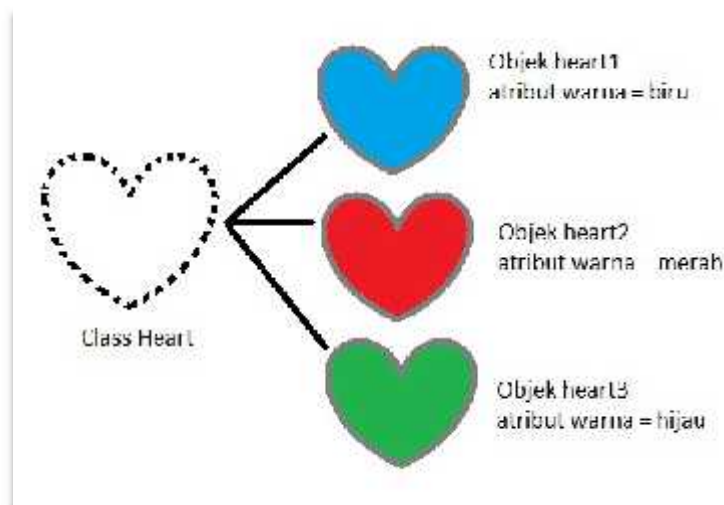
`new nameOfClass()`. Contoh penerapan pembuatan class dan object menggunakan bahasa pemrograman JAVA seperti pada gambar 2.4



Gambar 2.4 Pembuatan class dan object

Secara garis besar, susunan deklarasi class pada JAVA terdiri dari 2 bagian utama yaitu class declaration dan class body. Class declaration mendefinisikan nama class beserta beberapa variabelnya, sedangkan class body mendefinisikan method dan variabel yang ada didalamnya. Setiap penciptaan sebuah object, maka object tersebut akan memiliki semua variabel dan method yang dimiliki oleh sebuah class sesuai dengan visibilitas akses modifier seperti public, private, protected ataupun default yang akan dibahas pada poin selanjutnya. Berdasarkan contoh gambar 2.4 ketika diciptakan object jerapah dari class Animal dengan menggunakan kode program `Animal jerapah = new Animal()` maka object jerapah akan memiliki variabel nama, umur dan jenisHidup. Selain itu object jerapah juga memiliki method setNama, setUmur dan setJenisHidup. Cara mengakses sebuah method yaitu dengan sintaks `objek.namaMethod([parameter])`, misalnya `jerapah.setNama("Jerapah Asia")`, begitu juga dengan objek-objek dan method lainnya. Perhatikan

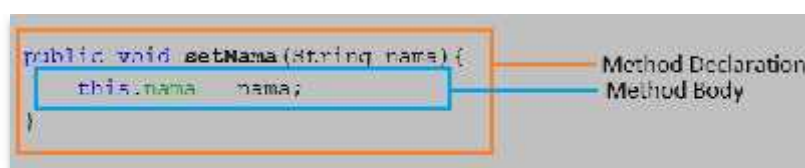
ilustrasi gambar 2.5 untuk lebih memahami konsep pemrograman berorientasi objek.



Gambar 2.5 Ilustrasi class dan objek

2. Method

Elemen penting yang termasuk dalam pembuatan class dan object adalah method. Pada umumnya method dapat berupa fungsi ataupun prosedur yang digunakan untuk melakukan output, pengisian ataupun mengambil nilai dari variabel. Perbedaan antara fungsi dan prosedur adalah pada sebuah fungsi yang digunakan dalam program akan membutuhkan return value (nilai balik) untuk digunakan pada proses selanjutnya, sedangkan pada prosedur tidak membutuhkannya. Dalam istilah lain method terbagi menjadi Setter dan Getter. Method Setter (prosedur) digunakan untuk memberikan nilai ataupun untuk menampilkan nilai dari variabel, sehingga tidak memerlukan return value, sedangkan method getter (fungsi) digunakan untuk mengambil nilai dari variabel, sehingga membutuhkan return value. Sama halnya dengan class, method terdiri dari 2 bagian untuk membuatnya yaitu method declaration dan method body seperti pada gambar 2.6



Gambar 2.6 Bagian method

Keyword `this` digunakan jika terdapat parameter yang namanya sama dengan nama variabel class. Seperti pada contoh gambar 2.4 terdapat atribut nama dengan tipe data `String` dan pada method `setNama` mempunyai parameter nama dan juga bertipe data `String`, sehingga untuk memberikan nilai variabel nama menggunakan keyword `this`, yang berarti variabel nama diberi nilai oleh parameter nama. Sintaks umum untuk pembuatan method adalah seperti pada gambar 2.7

```
[1] [Modifier] [2] returnType [3] nameOfMethod ( [4] [ dataType [5] nameOfAttribute ] ) {  
.....  
.....  
[6] return [7] returnTypeValue  
}
```

Gambar 2.7 Sintaks penulisan method

Untuk pemahaman sintaks pembuatan method yaitu bagian yang ditulis dalam kurung siku [] adalah bersifat opsional, boleh digunakan maupun tidak, tergantung permasalahan yang akan diselesaikan.

Penjelasan gambar 2.7:

1. Modifier bersifat opsional, boleh digunakan atau tidak sesuai permasalahan. Modifier adalah keyword yang digunakan untuk mengatur hak akses atau tingkat visibilitas yang dapat diterapkan pada sebuah method, class, variabel ataupun object. Modifier dapat bernilai `public`, `private`, `protected` ataupun jika tidak digunakan berarti secara otomatis menggunakan modifier default. Pembahasan modifier akan terdapat pada poin selanjutnya.
2. `returnType` dapat diisi tipe data dari return value yang dihasilkan oleh method. Jenis-jenis Tipe data tersebut adalah `String` (satu atau kumpulan dari beberapa karakter), `int` (angka bulat), `float`(angka pecahan), `boolean`(`true` atau `false`) dan lain-lain. Jika

method yang dibuat tidak menghasilkan return value, maka diisi dengan keyword void.

3. `nameOfMethod` diperlukan untuk menuliskan pengenalan yaitu nama dari method. Pemberian nama method mempunyai aturan yang sama dengan penamaan variabel diantaranya adalah
 - a. Tidak diperbolehkan menggunakan keyword dari JAVA seperti `exit`, `return`, `switch`, `main` dan lain-lain.
 - b. Diperbolehkan menggunakan angka (0-9), garis bawah (`underscore`), simbol dollar (\$).
4. `dataType` berisi tipe data yang bersifat opsional, diperlukan jika sebuah method membutuhkan parameter untuk melewati nilai variabel seperti `String`, `int`, `float`, `Boolean` dan lain-lainnya.
5. `nameOfAttribute` adalah nama dari variabel yang terdapat dalam parameter dan juga bersifat opsional jika sebuah method membutuhkan parameter.
6. `return` adalah keyword yang digunakan untuk menghasilkan nilai balik. Dibutuhkan ketika method yang dibuat tidak menggunakan `returnType void` (prosedur).
7. `returnTypeValue` adalah nilai yang akan di jadikan nilai balik.

Listing 2.1 dan 2.2 berikut ini adalah contoh untuk penulisan method yang berupa fungsi dan prosedur

```
public void cetakData() {  
  
    System.out.println("Ini adalah Method");  
    System.out.println("Method berupa prosedur tanpa parameter");  
    System.out.println("karena tidak mempunyai return value");  
}
```

Listing 2.1 Method berupa prosedur tanpa parameter

```
public int tambahDuaBilangan(int bil1, int bil2){  
  
    int hasil = bil1 + bil2;  
    return hasil;  
}
```

Listing 2.2 Method berupa fungsi dengan parameter dan return value integer

3. Data Variabel

Berdasarkan scope atau ruang lingkungannya, maka data variabel dibedakan menjadi 2 jenis yaitu:

a. Member Variabel /Class Variabel

Yaitu variabel yang dimiliki oleh class, dimana deklarasinya diletakkan didalam class, sehingga variabelnya akan dapat diakses oleh semua method dan minimal didalam class yang mendefinisikannya.

b. Local Variabel

Yaitu variabel yang dimiliki oleh method, dimana deklarasinya diletakkan didalam method, sehingga variabelnya hanya dapat diakses didalam method yang mendeklarasikannya sendiri.

Untuk lebih memperjelas pengetahuan tentang Member Variabel dan Local Variabel, perhatikanlah contoh gambar 2.8 berikut ini:

```
public class Guru {
    private String nip;
    private String nama;
    private boolean gender;
    private String alamat;
    // Member Variable/Class Variable

    public void setTitelNamaGuruRPL (String nama ){
        this.nama = nama + ", S.Kom";
    }

    public String getTitelNamaGuruRPL(){
        return nama;
    }

    void setGajiGuru(){
        int gaji3B = 3000000;
        // Local Variable
    }

    public static void main(String[] args) {
        Guru guru1 = new Guru();
        guru1.setTitelNamaGuruRPL("Helika Gigi Titimonno ");
        System.out.println(guru1.getTitelNamaGuruRPL());
        int lembur = 1000000;
        int totalgaji = gaji3B + lembur;
    }
}
```

Gambar 2.8 Member Variabel dan Local Variabel

Pada class Guru terdapat member variabel nip, nama, gender dan alamat dimana ruang lingkupnya dapat diakses disemua method, sehingga ketika variabel nama diakses oleh method setTitelNamaGuruRPL, maka kode program akan berjalan dengan benar. Sedangkan pada local variabel gaji3B yang didefinisikan didalam method setGajiGuru, maka variabel tersebut tidak dapat diakses didalam method lain, terbukti ketika diakses didalam method main muncul garis merah yang menandakan kode program dalam keadaan error.

4. Modifier

Dalam pemrograman berorientasi objek dikenal istilah modifier, yaitu sebuah keywords yang digunakan untuk mengatur hak akses sebuah variabel, method ataupun class dengan tujuan untuk menjaga integritas data ketika akan diakses oleh object lain. Sebagai analogi bentuk nyata adalah ketika kita mempunyai sebuah barang, maka ada beberapa barang yang hanya boleh kita gunakan sendiri dan tidak boleh dipinjamkan kepada orang lain misalnya sikat gigi, berarti kalau diimplementasikan kedalam bahasa pemrograman JAVA maka modifier yang digunakan adalah private. Selain itu ada barang-barang yang boleh dipinjamkan kepada orang lain misalnya buku, berarti implementasinya menggunakan modifier public. Atau barang-barang yang kita punyai hanya dapat dipinjam oleh saudara-saudara dirumah kita sendiri, maka disini menggunakan modifier protected.

Modifier terbagi menjadi 2 yaitu

a. Access Modifier

Terdapat 4 macam access modifier yaitu

- i. public yang berarti class, method maupun atribut yang mempunyai access modifier public memungkinkan untuk diakses dari manapun dan oleh kelas apapun. Access modifier public mempunyai hak akses paling luas dibanding

yang lainnya. Penggunaan modifier public seperti pada listing 2.3 berikut ini

```
public class Mobil {  
  
    public String merk;  
    public int tahun_pembuatan;  
  
    public void setDataMobil(String merk, int tahun_pembuatan) {  
        this.merk = merk;  
        this.tahun_pembuatan = tahun_pembuatan;  
    }  
  
    public String getMerk() {  
        return merk;  
    }  
  
    public int getTahun() {  
        return tahun_pembuatan;  
    }  
  
    public static void main(String[] args) {  
        Mobil mobilku = new Mobil();  
        mobilku.setDataMobil("AVANZA", 2000);  
        System.out.println("Merk Mobil : " + mobilku.getMerk());  
        System.out.println("Tahun Mobil : " + mobilku.getTahun());  
    }  
}
```

Listing 2.3 Penggunaan modifier public

- ii. private yang berarti class, method maupun variabel hanya dapat diakses secara langsung didalam class itu sendiri. Modifier private mempunyai akses tertutup, sesuai dengan konsep pemrograman berorientasi objek, setiap data harus disembunyikan. Untuk mengaksesnya diperlukan method setter dan getter. Penggunaan modifier private seperti pada listing 2.4 berikut ini

```
class Kendaraan{
    private String merk;
    private int tahun_pembuatan;

    void setDataMobil(String merk, int tahun_pembuatan){
        this.merk = merk;
        this.tahun_pembuatan = tahun_pembuatan;
    }
    String getMerk(){
        return merk;
    }
    int getTahun(){
        return tahun_pembuatan;
    }
}

public class Mobil {
    public static void main(String[] args){
        Kendaraan mobilku = new Kendaraan();
        mobilku.setDataMobil("BAJAJ", 2017);
        System.out.println("Merk Mobil : " + mobilku.getMerk());
        System.out.println("Tahun Mobil : " + mobilku.getTahun());
    }
}
```

Listing 2.4 Penggunaan dan pengaksesan modifier private

- iii. `protected` yang berarti bahwa class, method maupun variabel tersebut dapat diakses oleh kelas yang sama, package yang sama, dan kelas turunannya (subclass). Access modifier `protected` pada umumnya digunakan untuk mewariskan variabel yang ada di super class terhadap child class. Penggunaan modifier `protected` seperti pada listing 2.5 berikut ini


```
class Pelajar {
    protected String nis, nama;
    protected double nilai1, nilai2, nilai3;
    double rata;
    protected void setDataSiswa(String nis, String nama_siswa) {
        this.nis = nis;
        nama = nama_siswa;
    }
    public void setNilaiSiswa(double n1, double n2, double n3) {
        nilai1 = n1;
        nilai2 = n2;
        nilai3 = n3;
    }
    public double getNilaiRata() {
        rata = (nilai1 + nilai2 + nilai3) / 3;
        return rata;
    }
}

public class Siswa {
    public static void main(String[] args) {
        Pelajar siswa1 = new Pelajar();
        siswa1.setDataSiswa("1102020", "Andi");
        siswa1.setNilaiSiswa(90, 85, 200);
        System.out.println(siswa1.getNilaiRata());
    }
}
```

Listing 2.5 Penggunaan modifier protected

- iv. default atau disebut juga no access modifier yaitu class, method dan variabel dapat diakses disemua class manapun asal masih dalam satu package yang sama. Penulisannya tanpa menggunakan keyword.

b. Non-Access Modifier

Java menyediakan beberapa macam Non-Access Modifier, yaitu:

i. static

Static adalah salah satu jenis modifier di Java yang digunakan agar suatu atribut atau pun method dapat diakses oleh kelas atau objek tanpa harus melakukan instansiasi terhadap kelas tersebut. Method main adalah salah satu contoh method yang mempunyai modifier static.

ii. final

Final adalah salah satu modifier yang digunakan agar suatu atribut atau method bersifat final atau tidak bisa diubah nilainya. Modifier ini digunakan untuk membuat konstanta di Java.

iii. abstract

Abstract adalah modifier yang digunakan untuk membuat kelas dan method abstrak

iv. synchronized

Synchronized adalah modifier yang digunakan dalam aplikasi Java berbasis thread. Modifier ini menspesifikasikan bahwa method merupakan thread safe. Artinya bahwa hanya ada satu jalur eksekusi pada method yang menggunakan modifier jenis ini dan memaksa thread thread lain menunggu giliran.

v. native

Modifier Native digunakan untuk spesifikasi method dengan implementasi di bahasa lain, seperti C, C++.

vi. transient

Modifier ini digunakan agar suatu variabel tidak bisa di serialisasi. Serialization adalah konsep dimana sebuah objek dapat ditransfer dari suatu aplikasi ke aplikasi lainnya atau dari suatu workstation ke workstation lainnya. Konsep ini sangat diperlukan ketika membuat aplikasi client server. Salah satu tujuan serialization adalah bahwa tidak boleh ada perubahan terhadap atribut pada saat objek di transformasikan menjadi stream.

vii. volatile

viii. implements

Digunakan untuk mengimplementasikan sebuah interface.

ix. Extends

Digunakan untuk proses pewarisan class.

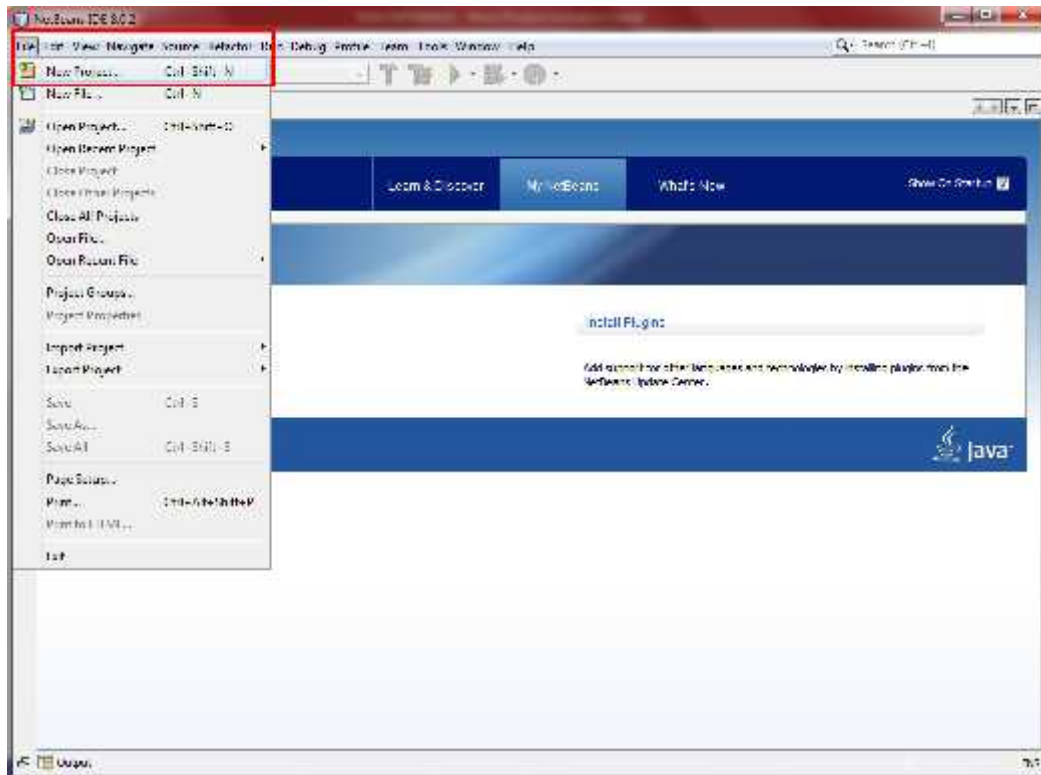
Tabel 2.2 berikut ini menyatakan scope akses modifier

Access Modifier	Same Class	Same Package	Sub Class	Other Package
public				
protected				X
default			X	X
private		X	X	X

5. Implementasi Pembuatan Class dan Object

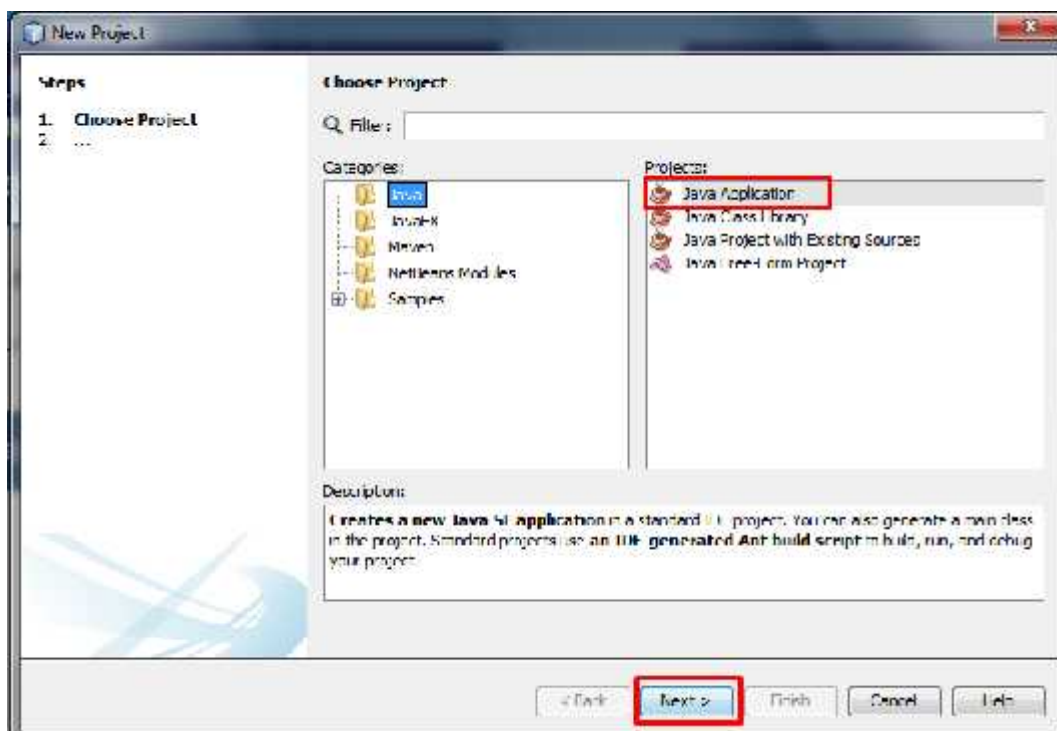
Pembuatan class dan object pada pembahasan ini adalah menggunakan bahasa pemrograman JAVA dengan menggunakan software Netbeans. Sebagai contoh kasus permasalahan adalah program untuk melakukan operasi aritmatika penjumlahan, pengurangan, perkalian dan pembagian antara 2(dua) bilangan. Langkah-langkahnya adalah sebagai berikut:

1. Bukalah editor software Netbeans yang telah terinstall di komputer. Kemudian lanjutkan dengan melakukan klik File → New Project seperti pada gambar 2.9



Gambar 2.9 Tampilan IDE Netbeans

2. Jika berhasil maka akan muncul tampilan dialog untuk menentukan jenis project seperti pada gambar 2.10, lanjutkan dengan memilih Java Application dan klik Next



Gambar 2.10 Penentuan jenis project

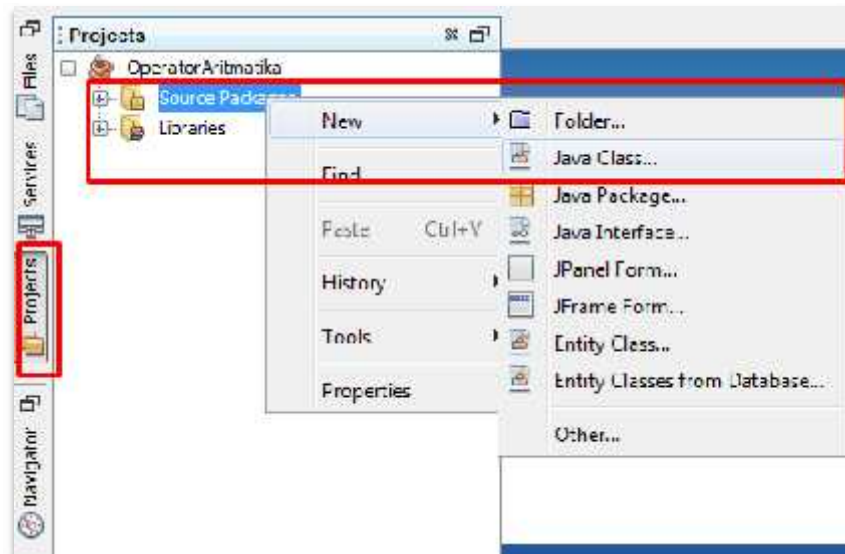
3. Isikan parameter yang dibutuhkan dalam project seperti pada gambar 2.11



Gambar 2.11 Pengisian parameter project

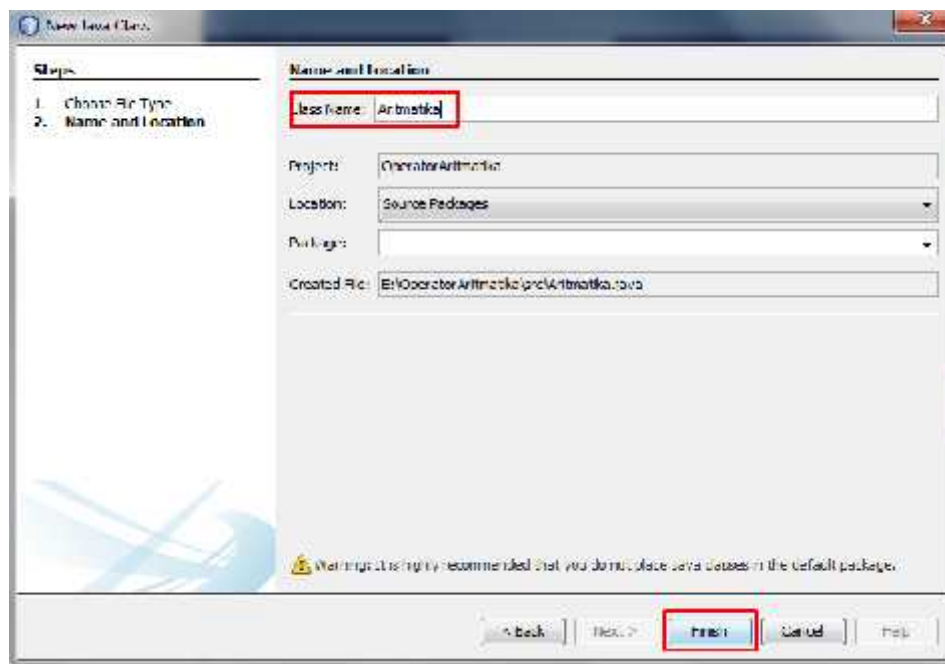
Pada contoh tersebut field Project Name diisi dengan OperatorAritmatika, kemudian tempat penyimpanan file di drive E:\ sesuaikan dengan komputer masing-masing, dan hilangkan centang pada field Create Main Class.

4. Klik pada tab projects, kemudian klik kanan pada bagian source packages dan pilih New → Java Class seperti pada gambar 2.12



Gambar 2.12 Pembuatan class

5. Isikan class name yang akan dibuat, misal nama class Aritmatika, kemudian lanjutkan dengan klik next seperti pada gambar 2.13



Gambar 2.13 Pengisian Class Name

6. Setelah berhasil membuat Java Class, maka akan otomatis terbentuk sebuah class dengan nama Aritmatika, kemudian lanjutkan dengan menambahkan Member Variabel bilangan1, bilangan2 dan hasil dengan tipe data float menggunakan access modifier private seperti pada listing 2.6

```
public class Aritmatika {  
    private float bilangan1;  
    private float bilangan2;  
    private float hasil;  
}
```

Listing 2.6 Deklarasi Member Variabel

7. Tambahkan Setter Method `setDataBilangan` dengan 2(dua) parameter `bilangan1` dan `bilangan2`, yang digunakan untuk memberikan nilai pada 2(dua) bilangan seperti pada listing 2.7

```
public class Aritmatika {  
    private float bilangan1;  
    private float bilangan2;  
    private float hasil;  
  
    public void setDataBilangan(int bilangan1, int bilangan2){  
        this.bilangan1 = bilangan1;  
        this.bilangan2 = bilangan2;  
    }  
}
```

Listing 2.7 Pembuatan Setter Method

8. Tambahkan Getter Method `getHasilJumlah`, `getHasilKurang`, `getHasilKali` dan `getHasilBagi` dengan return type value adalah float seperti pada listing 2.8

```
public class Aritmatika {
    private float bilangan1;
    private float bilangan2;
    private float hasil;

    public void setDataBilangan(int bilangan1, int bilangan2){
        this.bilangan1 = bilangan1;
        this.bilangan2 = bilangan2;
    }

    public float getHasilJumlah(){
        hasil = bilangan1+bilangan2;
        return hasil;
    }

    public float getHasilKurang(){
        hasil = bilangan1 - bilangan2;
        return hasil;
    }

    public float getHasilKali(){
        hasil = bilangan1 * bilangan2;
        return hasil;
    }

    public float getHasilBagi(){
        hasil = bilangan1 / bilangan2;
        return hasil;
    }
}
```

Listing 2.8 Getter Method

9. Tambahkan main method dimana method tersebut akan dijalankan pertama kali oleh compiler program, kemudian didalamnya tambahkan proses pembuatan object dari class Aritmatika, dan proses terakhir adalah object yang telah dibuat mengakses semua method yang terdapat dalam class, seperti pada listing 2.9

```
public static void main(String []args){
    Aritmatika arit1 = new Aritmatika();
    arit1.setDataBilangan(45,6);
    System.out.println(arit1.getHasilJumlah());
    System.out.println(arit1.getHasilKurang());
    System.out.println(arit1.getHasilKali());
    System.out.println(arit1.getHasilBagi());
}
```

Listing 2.9 Pembuatan object didalam main method

10. Jalankan program dengan menekan tombol Shift + F6

B. Keterampilan yang diperlukan dalam Membuat Program Berorientasi Objek dengan Memanfaatkan Class

1. Mengoperasikan software IDE (Integrated Development Environment) untuk bahasa pemrograman JAVA seperti Netbeans, GEL, Eclipse dan lain-lain
2. Menyusun rancangan class, method dan variabel yang digunakan dalam program.
3. Membuat method main, method setter dan getter, variabel beserta tipe data yang digunakan.
4. Menentukan access dan non access modifier
5. Melakukan proses instantiate objek dari sebuah class.
6. Memanggil method dari sebuah object atau lebih.

C. Sikap kerja yang diperlukan dalam Membuat Program Berorientasi Objek dengan Memanfaatkan Class

Harus bersikap secara:

1. Cermat dan teliti dalam menganalisis data.
2. Tekun dalam proses pemahaman sintaks.
3. Sesuai dengan kaidah-kaidah bahasa pemrograman.
4. Berpikir analitis serta evaluatif ketika melakukan trial and error.

BAB III

Menggunakan Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas

A. Pengetahuan yang diperlukan dalam Menggunakan Tipe Data dan Control Program pada Metode atau Operasi dari Suatu Kelas

1. Variabel, Tipe Data dan konstanta

Variabel merupakan container yang digunakan untuk menyimpan suatu nilai secara sementara pada sebuah program dengan tipe tertentu. Sementara berarti nilainya dapat diubah sesuai keperluan, dan ketika program dihentikan maka nilai dari variabel akan hilang. Pada dasarnya ada dua macam tipe data variabel yaitu tipe primitive dan tipe reference.

Tipe primitive meliputi:

- a. Tipe Boolean adalah tipe data logika yang hanya mempunyai 2 nilai yaitu true dan false.
- b. Tipe Numerik yang meliputi:
 - i. byte, adalah tipe data Integral 8-bit. Memiliki rentang nilai antara -2^7 sampai $2^7 - 1$ atau dari -128 sampai 127
 - ii. short, adalah tipe data Integral 16-bit. Memiliki rentang nilai antara -2^{15} sampai $2^{15} - 1$ atau dari -32768 sampai 32768.
 - iii. int, adalah tipe data Integral 32-bit. Memiliki rentang nilai antara -2^{31} sampai $2^{31} - 1$ atau dari -2,147,483,648 sampai 2,147,483,647.
 - iv. long, adalah tipe data Integral 64-bit. Memiliki rentang nilai antara -2^{63} sampai $2^{63} - 1$ atau dari -9,223,372,036,854,775,808 sampai 9,223,372,036,854,775,807.

- v. `char`, adalah tipe data Textual, yang merepresentasikan karakter unicode 16-bit. Nilai literalnya harus diapit dengan tanda petik tunggal (`'`).
- vi. `float`, adalah tipe data Floating Point 32-bit. Nilai literalnya mengandung pecahan (dipisahkan dengan tanda titik `.`)
- vii. `double`, adalah tipe data Floating Point 64-bit. Nilai literal default untuk float dan double adalah double, kecuali diberi akhiran `F`

Sedangkan tipe data variabel berupa reference terdiri atas tipe variabel data:

- a. Tipe class
- b. Tipe array
- c. Tipe interface

Untuk mendeklarasikan variabel dan tipe data menggunakan sintaks `<tipe-data> <nama variabel>`. Perhatikan contoh listing 3.1 berikut ini untuk memahami penggunaan tipe data dan variabel.

```
public class Variabel {  
  
    public static void main(String[] args) {  
        byte    nilai_byte    = 70;  
        short   nilai_short   = 500;  
        int     nilai_int     = 263456;  
        long    nilai_long    = 1200000;  
        char    nilai_char    = 'A';  
        float   nilai_float   = 190;  
        double  nilai_double  = 387;  
        boolean nilai_bool    = true;  
  
        System.out.println(nilai_byte);  
        System.out.println(nilai_short);  
        System.out.println(nilai_int);  
        System.out.println(nilai_long);  
        System.out.println(nilai_char);  
        System.out.println(nilai_float);  
        System.out.println(nilai_double);  
        System.out.println(nilai_bool);  
    }  
}
```

Listing 3.1 Penggunaan tipe data dan variabel

2. Control Program

Struktur control program terdiri dari 2(dua) jenis, yaitu struktur percabangan dan perulangan. Percabangan digunakan untuk memilih dan mengeksekusi blok kode ketika ada 2(dua) atau lebih kemungkinan kondisi. Kondisi diisikan dengan sebuah pernyataan yang melibatkan operator logika ==(sama dengan), !=(tidak sama dengan), !(negasi), >(lebih besar), >=(lebih besar sama dengan), <(lebih kecil), <=(lebih kecil sama dengan). Sedangkan perulangan digunakan untuk mengeksekusi program secara berulang-ulang.

2.1. Struktur percabangan

a. Pernyataan IF satu kondisi. Sintaks penggunaannya adalah

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

Jika kondisi bernilai true maka program akan mengeksekusi pernyataan.

b. Pernyataan IF dua kondisi. Sintaks penggunaannya adalah

```
if (kondisi){  
    pernyataan  
}  
else{  
  
}
```

Jika kondisi bernilai true maka program akan mengeksekusi pernyataan, jika bernilai false maka program akan mengeksekusi bagian blok else.

c. Pernyataan IF lebih dari 2 kondisi. Sintaks penggunaannya adalah

```
if (kondisi 1){  
    pernyataan 1  
}  
else if(kondisi 2){  
    Pernyataan 2  
}  
else{  
    Pernyataan 3  
}
```

Jika kondisi 1 bernilai true, maka program akan mengeksekusi pernyataan 1, jika bernilai false, maka program akan beralih pada bagian blok else untuk selanjutnya melakukan pengecekan pada kondisi 2. Jika kondisi 2 bernilai true maka program akan mengeksekusi pernyataan 2, jika bernilai false maka program akan mengeksekusi pernyataan 3.

d. Pernyataan SWITCH, sintaks penggunaannya adalah

```
switch(kondisi){  
    case kondisi_1 :  
        pernyataan 1;  
        break;  
    case kondisi_2 :  
        pernyataan 2;  
        break;  
    case kondisi_n :  
        pernyataan n;  
        break;  
}
```

Program akan melakukan pengecekan nilai kondisi, kemudian program akan menyesuaikan nilai tersebut termasuk dalam case sesuai kondisi masing-masing, untuk mengerjakan baris pernyataan sampai break

Contoh penggunaan struktur percabangan IF seperti pada listing 3.2

```
public class Student {
    private String nama_siswa;
    private int nilai_siswa;

    public void setDataStudent(String nama_siswa, int nilai){
        this.nama_siswa = nama_siswa;
        nilai_siswa = nilai;
    }
    public String getNilaiHuruf(){
        String nilai_huruf = null;
        if(nilai_siswa >= 80){
            nilai_huruf = "Memuaskan";
        }
        else if(nilai_siswa >=70){
            nilai_huruf = "Baik";
        }
        else if(nilai_siswa >=60){
            nilai_huruf = "Kurang";
        }

        return nilai_huruf;
    }

    public static void main(String args[]){
        Student siswa1 = new Student();
        siswa1.setDataStudent("Rudi", 90);
        System.out.println(siswa1.getNilaiHuruf());
    }
}
```

Listing 3.2 Struktur percabangan menggunakan IF

Contoh penggunaan struktur percabangan menggunakan switch seperti pada listing 3.3

```
public class Student {  
    private String nama_siswa;  
    private String gender;  
  
    public void setDataStudent(String nama_siswa, String gender){  
        this.nama_siswa = nama_siswa;  
        this.gender = gender;  
    }  
    public String getGenderInEnglish(){  
        String jk = null;  
        switch(gender){  
            case "L" :  
                jk = "MALE";  
                break;  
            case "P" :  
                jk = "FEMALE";  
                break;  
        }  
  
        return jk;  
    }  
  
    public static void main(String args[]){  
        Student siswal = new Student();  
        siswal.setDataStudent("Audi", "P");  
        System.out.println(siswal.getGenderInEnglish());  
    }  
}
```

Listing 3.3 Struktur percabangan menggunakan switch

2.2. Struktur perulangan

- a. Perulangan for, digunakan ketika permasalahan telah diketahui batas awal dan batas akhirnya. Sintaks penggunaannya adalah

```
for(<nilai awal>;<nilai akhir>;increment/decrement){  
    pernyataan  
}
```

- b. Perulangan do while, akan dikerjakan selama kondisinya memenuhi persyaratan, minimal pernyataan akan dikerjakan 1x meskipun tidak memenuhi persyaratan. Sintaksnya adalah


```
do {  
    pernyataan  
} while(kondisi);
```

- c. Perulangan While, akan dikerjakan selama kondisinya memenuhi persyaratan, minimal dikerjakan 0x. Sintaks penggunaannya adalah

```
while(kondisi){  
    pernyataan  
}
```

Contoh penggunaan perulangan seperti pada listing 3.4 berikut ini

```
public class Perulangan {  
  
    public void setPerulanganFor(){  
        for(int i=0; i<10; i++){  
            System.out.println("Nilai i adalah " + i);  
        }  
    }  
  
    public void setPerulanganDoWhile(){  
        int a = 0;  
        do{  
            System.out.println("Nilai a adalah " + a);  
            a = a + 1;  
        }while(a < 10);  
    }  
  
    public void setPerulanganWhile(){  
        int c = 0;  
        while(c<20){  
            System.out.println("Nilai c adalah " + c);  
        }  
    }  
  
    public static void main(String[] args) {  
        Perulangan pl = new Perulangan();  
        pl.setPerulanganFor();  
        pl.setPerulanganDoWhile();  
        pl.setPerulanganWhile();  
    }  
}
```

Listing 3.4 Contoh perulangan

B. Keterampilan yang diperlukan dalam Menggunakan Tipe Data dan Control Program pada Metode atau Operasi dari Suatu Kelas

- 1 Mengidentifikasi permasalahan yang akan dijadikan control program
- 2 Mengidentifikasi tipe data beserta variabel yang digunakan
- 3 Menyusun percabangan dan perulangan pada sebuah method

C. Sikap kerja yang diperlukan dalam Menggunakan Tipe Data dan Control Program pada Metode atau Operasi Suatu Kelas.

Harus bersikap secara:

1. Cermat dan teliti dalam mengidentifikasi jenis-jenis tipe data
2. Sering melakukan percobaan dan analitis terhadap hasil program.
3. Tekun dan mau berusaha untuk mencoba berkali-kali

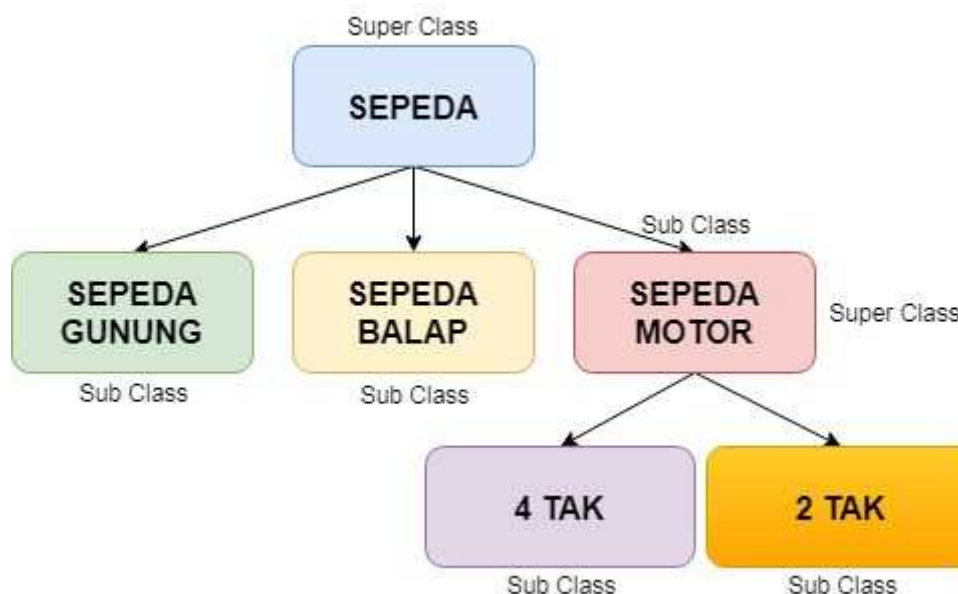
BAB IV

MEMBUAT PROGRAM DENGAN KONSEP BERBASIS OBJEK

A. Pengetahuan yang diperlukan dalam Membuat Program Dengan Konsep Berbasis Objek

1. Inheritance

Proses pewarisan merupakan pendefinisian sebuah class menggunakan referensi class lain yang telah terdefiniskan terlebih dahulu, dengan kata lain pewarisan adalah proses untuk menurunkan sifat dari semua variabel dan method yang dimiliki oleh Parent Class (Super Class) kepada Child Class (Sub Class). Pada konsep pemrograman berbasis objek dikenal istilah hierarki class, yaitu setiap Super Class menjadi parent dari Sub Class, dan setiap Sub Class menjadi Super Class bagi Sub Class lainnya. Pada contoh kasus class sepeda, perhatikan gambar 4.1 berikut ini



Gambar 4.1 Contoh Hierarki Class Sepeda

Pada gambar 4.1 tersebut dimungkinkan penciptaan class baru menggunakan referensi class Sepeda (Super Class), seperti class Sepeda Gunung, Sepeda Balap dan Sepeda Motor (Sub Class). Sub

Class akan mewarisi sifat-sifat dari atribut(variabel) dan behavior (method) dari Super Class, pewarisan itu misalnya class Sepeda memiliki atribut roda, maka Sepeda Balap, Sepeda Gunung dan Sepeda Motor akan memiliki atribut roda juga. Setiap Sub Class dapat menambahkan atribut dan method lain yang lebih spesifik sebagai bentuk perluasan dari Super Class, sehingga untuk membuat class pewarisan menggunakan keyword `extends` dengan sintaks `<[modifier]> <class> <Nama Class> extends <Nama Super Class>`.

Keuntungan dari konsep pewarisan adalah

- a. Sub Class tidak perlu menuliskan fungsi yang berulang-ulang jika bentuk fungsinya mempunyai kesamaan dengan class lain, cukup dengan menggunakan method yang terdapat pada Super Class sehingga program akan lebih efisien.
- b. Struktur program akan lebih tertata rapi, sehingga mudah untuk dilakukan perawatan. Misal jika terdapat penambahan atau perubahan fitur program, maka cukup dilakukan modifikasi pada Super Class.

Listing 4.1, 4.2, 4.3, 4.4, 4.5 dan 4.6 berikut ini adalah implementasi dari konsep pewarisan class Sepeda, Sepeda Gunung, Sepeda Balap dan Sepeda Motor dengan menggunakan file yang terpisah satu sama lain.

```
/*  
Nama File : Sepeda.java  
Fungsinya sebagai Super Class  
*/  
public class Sepeda {  
    protected String jenis_roda;  
    protected String merk;  
    protected int tahun_pembuatan;  
  
    public void setDataSepeda(String jroda,String merk, int tahun){  
        jenis_roda = jroda;  
        this.merk = merk;  
        tahun_pembuatan = tahun;  
    }  
  
    public String getJenisRoda(){  
        return jenis_roda;  
    }  
  
    public String getMerk(){  
        return merk;  
    }  
  
    public int getTahunBuat(){  
        return tahun_pembuatan;  
    }  
}
```

Listing 4.1 Class Sepeda

```
/*  
Nama File : SepedaGunung.java  
Sub Class dari Class Sepeda  
*/  
public class SepedaGunung extends Sepeda{  
  
    void cetakKeterangan(){  
        System.out.println("Ini Adalah Sepeda Gunung");  
    }  
  
    public static void main(String []args){  
        SepedaGunung s1 = new SepedaGunung();  
        s1.cetakKeterangan();  
        s1.setDataSepeda("Roda Medium","Polygon", 2015);  
        System.out.println("Jenis Roda : " + s1.getJenisRoda());  
        System.out.println("Merk Sepeda : " + s1.getMerk());  
        System.out.println("Tahun Pembuatan : " + s1.getTahunBuat());  
    }  
}
```

Listing 4.2 Class SepedaGunung

```
/*  
Nama File : SepedaBalap.java  
Sub Class dari Class Sepeda  
*/  
public class SepedaBalap extends Sepeda{  
    void cetakKeterangan(){  
        System.out.println("Ini Adalah Sepeda Balap");  
    }  
    public static void main(String []args){  
        SepedaBalap s1 = new SepedaBalap();  
        s1.cetakKeterangan();  
        s1.setDataSepeda("Roda Soft", "BMX", 2000);  
        System.out.println("Jenis Roda : " + s1.getJenisRoda());  
        System.out.println("Merk Sepeda : " + s1.getMerk());  
        System.out.println("Tahun Pembuatan : " + s1.getTahunBuat());  
    }  
}
```

Listing 4.3 Class SepedaBalap

```
/*  
Nama File : SepedaMotor.java  
Sub Class dari Class Sepeda  
*/  
public class SepedaMotor extends Sepeda{  
    protected String mesin;  
  
    public void setMesin(String mesin){  
        this.mesin = mesin;  
    }  
    public String getMesin(){  
        return mesin;  
    }  
    void cetakKeterangan(){  
        System.out.println("Ini Adalah Sepeda Motor");  
    }  
    public static void main(String []args){  
        SepedaMotor s1 = new SepedaMotor();  
        s1.cetakKeterangan();  
        s1.setDataSepeda("Roda Hard", "Honda", 2014);  
        System.out.println("Jenis Roda : " + s1.getJenisRoda());  
        System.out.println("Merk Sepeda : " + s1.getMerk());  
        System.out.println("Tahun Pembuatan : " + s1.getTahunBuat());  
    }  
}
```

Listing 4.4 Class SepedaMotor

```
/*  
Nama File : Sepeda2Tak.java  
Sub Class dari Class SepedaMotor  
*/  
public class Sepeda2Tak extends SepedaMotor{  
  
    @Override  
    void cetakKeterangan(){  
        System.out.println("Ini Adalah Sepeda Motor 2 Tak");  
    }  
    public static void main(String[] args){  
        Sepeda2Tak s2tak = new Sepeda2Tak();  
        s2tak.cetakKeterangan();  
        s2tak.setMesin("150 CC");  
        s2tak.setDataSepeda("Roda Hard", "Honda Beat", 2013);  
        System.out.println("Kapasitan Mesin : " + s2tak.getMesin());  
        System.out.println("Jenis Roda : " + s2tak.getJenisRoda());  
        System.out.println("Merk Sepeda : " + s2tak.getMerk());  
        System.out.println("Tahun Pembuatan : " + s2tak.getTahunBuat());  
    }  
}
```

Listing 4.5 Class Sepeda2Tak

```
/*  
Nama File : Sepeda4Tak.java  
Sub Class dari Class SepedaMotor  
*/  
public class Sepeda4Tak extends SepedaMotor{  
  
    @Override  
    void cetakKeterangan(){  
        System.out.println("Ini Adalah Sepeda Motor 4 Tak");  
    }  
    public static void main(String[] args){  
        Sepeda4Tak s4tak = new Sepeda4Tak();  
        s4tak.cetakKeterangan();  
        s4tak.setMesin("500 CC");  
        s4tak.setDataSepeda("Roda Hard", "Yamaha Vixion", 2013);  
        System.out.println("Kapasitan Mesin : " + s4tak.getMesin());  
        System.out.println("Jenis Roda : " + s4tak.getJenisRoda());  
        System.out.println("Merk Sepeda : " + s4tak.getMerk());  
        System.out.println("Tahun Pembuatan : " + s4tak.getTahunBuat());  
    }  
}
```

Listing 4.6 Class Sepeda4Tak

2. Constructor

Constructor adalah sebuah method yang digunakan untuk memberikan nilai default atau inialisasi nilai pada sebuah objek ketika diciptakan. Dalam sebuah class diperbolehkan terdapat satu ataupun lebih Constructor. Sebagai ilustrasi, misalnya pembuatan objek enemy pada sebuah game, maka objek tersebut akan memiliki nilai default kekuatannya, kemampuannya seperti apa, kemudian seiring berjalannya waktu pada game tersebut kekuatan enemy dapat bertambah ataupun berkurang. Constructor hanya dipanggil satu kali yaitu saat penciptaan sebuah objek. Pada sub class yang memanggil constructor dari super class nya dapat menggunakan keyword super, artinya sub class memanggil constructor dari super class nya. Ciri-ciri sebuah method yaitu nama method nya sama dengan nama class, tidak memiliki return value dan tidak menggunakan keyword void. Listing 4.7 berikut ini adalah contoh implementasi constructor method.

```
/**
 * Nama File : Pesawat.java
 * @author javajawara
 */
public class Pesawat {

    String jenis_pesawat;
    String nama_airlines;

    /* Constructor Method tanpa parameter */
    public Pesawat(){

    }
    public Pesawat(String jenis_pesawat, String nama_airlines){
        this.jenis_pesawat = jenis_pesawat;
        this.nama_airlines = nama_airlines;
    }

    public void infoPesawat(){
        System.out.println("Jenis Pesawat : " + this.jenis_pesawat);
        System.out.println("Nama Air Line : " + this.nama_airlines);
    }
}
```

Listing 4.7 Super class Pesawat dengan 2 (dua) constructor method

Pada class Pesawat terdapat 2(dua) buah constructor method, yaitu tanpa parameter dan menggunakan parameter. Adapun pemanggilan constructor tergantung pada objek pemanggilnya, seperti pada contoh listing 4.8 berikut ini

```
public class PesawatKomersil extends Pesawat{  
  
    /*Constructor tanpa parameter yang memanggil constructor tanpa parameter super classnya */  
    public PesawatKomersil(){  
        super();  
    }  
  
    /*Constructor yang memanggil constructor dengan parameter super classnya */  
    public PesawatKomersil(String jenis, String nama){  
        super(jenis, nama);  
    }  
  
    public static void main(String [] args){  
        PesawatKomersil pesawatku = new PesawatKomersil();  
        PesawatKomersil pesawatmu = new PesawatKomersil("BOEING", "GARUDA AIRLINES");  
        System.out.println("PESAWATKU");  
        pesawatku.infoPesawat();  
        System.out.println("PESAWATMU");  
        pesawatmu.infoPesawat();  
    }  
}
```

Listing 4.8 Sub class PesawatKomersil

Jika kode program tersebut dijalankan maka akan tampil keluaran seperti pada gambar 4.2 berikut ini

```
PESAWATKU  
Jenis Pesawat : null  
Nama Air Line : null  
PESAWATMU  
Jenis Pesawat : BOEING  
Nama Air Line : GARUDA AIRLINES
```

Gambar 4.2 Hasil output program

Hasil jenis pesawat dan nama air line dari objek pesawatku adalah null dikarenakan objek pesawatku memanggil constructor tanpa parameter yang tidak mempunyai implementasi program, sehingga tidak ada nilai yang diberikan pada constructor method. Sedangkan pada pesawatmu memanggil constructor dengan parameter, sehingga nilai BOEING dan GARUDA AIRLINES diberikan pada masing-masing parameter.

3. Polymorphism dan Overriding

Secara harfiah arti dari polymorphism adalah mempunyai banyak bentuk. Polymorphism adalah sebuah konsep yang sangat penting dari pemrograman berorientasi objek. Pada bahasa pemrograman JAVA konsep ini dikenali dengan penggunaan lebih dari satu method yang memiliki nama yang sama. Sebagai contoh misalkan seorang guru mempunyai kewajiban untuk melakukan pengajaran, akan tetapi tiap-tiap guru tersebut mempunyai perbedaan matakuliah apa yang diajarkan, begitu juga dengan jumlah jam mata pelajaran, sehingga dapat diartikan bahwa satu method mengajar mempunyai banyak bentuk dalam implementasinya. Polymorphism berkaitan erat dengan proses pewarisan (inheritance), karena proses pembentukannya melalui super class dan sub class. Penggunaan method dengan nama yang sama dapat diimplementasikan dengan method overriding. Overriding method adalah fungsi yang digunakan untuk menimpa implementasi dari program dengan menggunakan nama method yang sama. Berikut ini adalah contoh listing untuk lebih memahami penggunaan polymorphism dan overriding.

```
/**
 * Nama file : Guru.java
 * @author javajawara
 */
public class Guru {

    protected String nama_guru;

    public Guru(String nama_guru){
        this.nama_guru = nama_guru;
    }

    public void infoGuru(){
        System.out.println("Saya adalah seorang Guru");
    }

}
```

Listing 4.9 Super Class Guru

Class Guru merupakan super class yang akan di extends pada class lainnya sehingga properti dan method yang terdapat didalam class Guru dapat digunakan pada class lain yang menjadi turunan dari class Guru. Berikutnya adalah menambahkan kode pada class Mapel seperti pada listing 4.10 berikut ini

```
/**
 *
 * @author jawaaware
 */
public class Mapel extends Guru {

    protected String nama_mapel;

    public Mapel(String nama_mapel, String nama_dosen){
        super(nama_dosen);
        this.nama_mapel = nama_mapel;
    }

    public void infoGuru(){
        System.out.println("Nama Guru : " + super.nama_guru);
        System.out.println("Mengajar Mata Pelajaran : " + this.nama_mapel);
    }
}
```

Listing 4.10 Sub Class Mapel

Pada class Mapel diatas terlihat bahwa properti yang terdapat pada super class (Guru) yaitu nama_guru ikut dikonstruksi bersamaan dengan properti dari class Mapel, selain itu terdapat pula penggunaan overriding method, karena terdapat method yang sama dengan yang ada pada super class yaitu infoGuru, perbedaannya method pada class Mapel memiliki output yang akan ditampilkan ketika program dieksekusi. Selanjutnya adalah pembuatan class JamMengajar, seperti pada listing 4.11 berikut ini

```
/**
 *
 * @author javajawara
 */
public class JamMengajar extends Guru {

    protected int jml_jam;

    public JamMengajar(int jml_jam, String nama_dosen){
        super(nama_dosen);
        this.jml_jam = jml_jam;
    }

    public void infoGuru(){
        System.out.println("Nama Guru : " + super.nama_guru);
        System.out.println("Jumlah Jam Mengajar : " + this.jml_jam + " jam");
    }
}
```

Listing 4.11 Sub class JamMengajar

Pada sub class JamMengajar, memiliki penjelasan yang sama dengan class Mapel. Hanya saja pada output menampilkan jumlah jam mengajar. Kemudian diakhir adalah pembuatan class Sekolah dimana terdapat method main yang merupakan fungsi untuk penciptaan beberapa objek seperti pada listing 4.12 berikut ini

```
/**
 *
 * @author javajawara
 */
public class Sekolah {

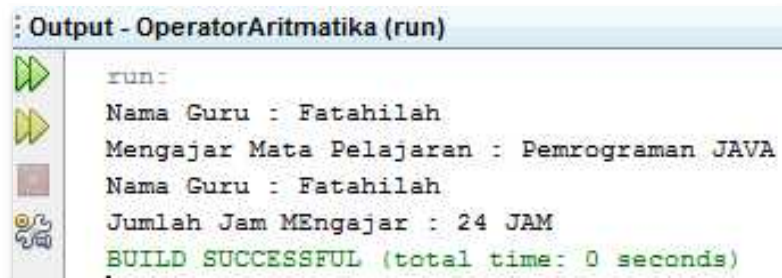
    public static void main(String[] args) {
        // TODO code application logic here
        Guru guruku = null;

        Mapel mapel = new Mapel("Pemrograman JAVA", "Fatahilah");
        JamMengajar jml_jam = new JamMengajar(24, "Fatahilah");
        guruku = mapel;
        guruku.infoGuru();

        guruku = jml_jam;
        guruku.infoGuru();
    }
}
```

Listing 4.12 Class Sekolah

Gambar 4.3 adalah hasil output ketika program dijalankan



```
run:
Nama Guru : Fatahilah
Mengajar Mata Pelajaran : Pemrograman JAVA
Nama Guru : Fatahilah
Jumlah Jam Mengajar : 24 JAM
BUILD SUCCESSFUL (total time: 0 seconds)
```

Gambar 4.3 Hasil Output

B. Keterampilan yang diperlukan dalam Membuat Program Dengan Konsep Berbasis Objek

1. Mengidentifikasi penggunaan Inheritance
2. Menulis class 2 (dua) atau lebih pada file yang berbeda.
3. Mengorganisasikan Super class dan Sub class.
4. Mengimplementasikan polymorphism
5. Mengimplementasikan overrrding method dan overloading method.
6. Menggunakan keyword super

C. Sikap kerja yang diperlukan dalam Membuat Program Dengan Konsep Berbasis Objek

Harus bersikap secara:

1. Cermat dan teliti dalam mengorganisasikan super class dan sub class
2. Sering melakukan uji coba kode program.
3. Sesuai dengan kaidah penulisan sintaks bahasa pemrograman JAVA.
4. Berpikir analitis serta evaluatif waktu melakukan uji coba kode program.

BAB V

MEMBUAT PROGRAM OBJECT ORIENTED DENGAN INTERFACE DAN PAKET

A. Pengetahuan yang diperlukan dalam Membuat Program Dengan Konsep Berbasis Objek

1. Interface

Interface digunakan untuk membuat kesamaan pada nama method ataupun variabel. Misal pada dunia nyata sebuah Mobil mempunyai cara untuk mengubah porsenelling, akan tetapi antara mobil manual dan mobil automatic akan berbeda cara penggunaannya. Interface terlihat seperti class, tapi bukan class, karena keyword yang digunakan adalah Interface bukan class. Interface merupakan kumpulan deklarasi method tanpa body program. Selain itu didalam interface diperbolehkan mendeklarasikan variabel dan harus diberi nilai awal sehingga mempunyai bentuk final (tidak dapat diubah nilainya).

Interface diperlukan karena seperti dalam bahasa pemrograman C/C++ terdapat konsep multiple inheritance, artinya satu sub class atau child class dapat memiliki banyak super class/parent class. Jadi pada pemrograman C/C++ diijinkan sebuah class anak mempunyai banyak class orang tua. Pada bahasa pemrograman JAVA hal tersebut tidak diperbolehkan karena bertentangan dengan konsep dunia nyata bahwa seorang anak pasti hanya mempunyai satu orang tua.

Untuk mengatasi permasalahan tersebut maka digunakan konsep interface. Sintaks penggunaan interface adalah `<[modifier]> interface <nama interface>`. Untuk class yang mengimplementasikan interface diharuskan mengimplementasikan semua method yang terdapat dalam interface, jadi tidak diperbolehkan hanya mengimplementasikan sebagian saja. Sintak untuk melakukan implementasi interface adalah menggunakan keyword implements yaitu `<[modifier]> class <nama class> [extends <super class>] implements <nama interface>`. Jika

mengimplementasikan banyak interface, maka tiap-tiap interface dipisahkan dengan tanda koma(.). Gambar 5.1 dan 5.2 berikut adalah contoh interface dan class yang mengimplementasikannya.

```
/**
 *
 * @author java_maven
 */
public interface Rumah {
    String jenis_rumah = "Rumah Adat";
    int jumlah_lantai = 0;
    int tipe_rumah = 36;
    public void setRumah(String jenis_rumah, int jumlah_lantai);
    public void setTipeRumah(int tipe_rumah);
    public void infoRumah();
}
```

Deklarasi method tanpa body program

Deklarasi variabel dengan nilai awal

Gambar 5.1 Contoh interface

```
public class RumahSewa implements Rumah {
    int jml_pintu;
    int jml_jendela;
    int tipe_rumah;

    public void setRumah(int jml_pintu, int jml_jendela) {
        this.jml_pintu = jml_pintu;
        this.jml_jendela = jml_jendela;
    }

    public void setTipeRumah(int tipe_rumah) {
        this.tipe_rumah = tipe_rumah;
    }

    public void infoRumah() {
        System.out.println("Jenis Rumah : " + RumahSewa.jenis_rumah);
        System.out.println("Jumlah Lantai : " + RumahSewa.jumlah_lantai);
        System.out.println("Tipe Rumah : " + this.tipe_rumah);
        System.out.println("Jumlah Pintu : " + this.jml_pintu);
        System.out.println("Jumlah Jendela : " + this.jml_jendela);
    }

    public static void main(String []arga) {
        RumahSewa rs = new RumahSewa();
        rs.setRumah(1, 2);
        rs.setTipeRumah(45);
        rs.infoRumah();
    }
}
```

class RumahSewa adalah implementasi dari interface Rumah

Variabel Class/Member

Mengakses variabel interface

Penciptaan objek RumahSewa

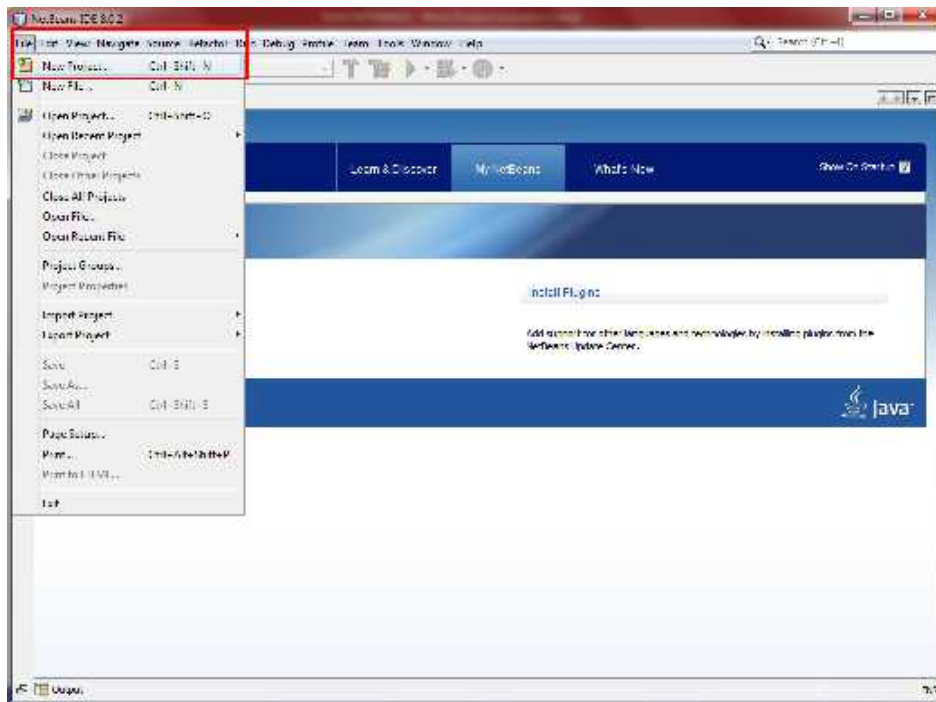
Pemanggilan method RumahSewa

Implementasi Method dari interface Rumah

Gambar 5.2 Contoh class yang mengimplementasikan interface

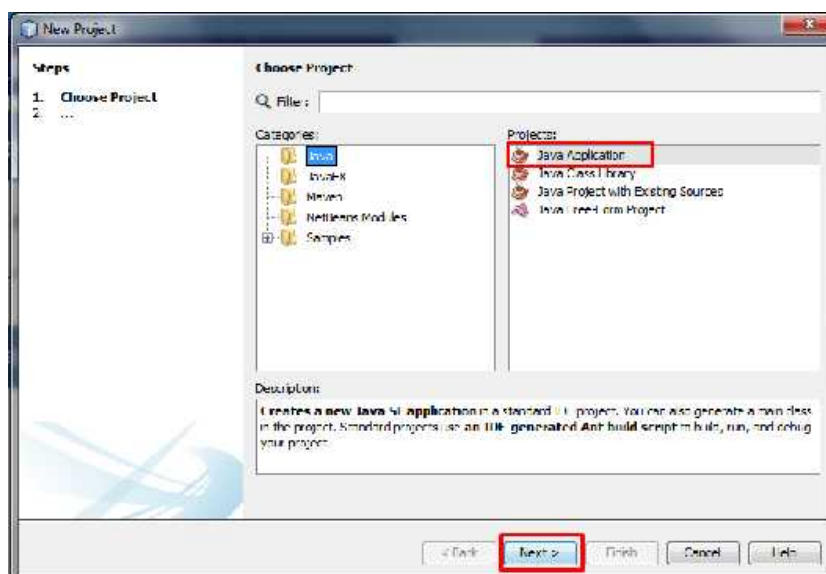
2. Langkah Pembuatan Interface

1. Bukalah editor software Netbeans yang telah terinstall di komputer.
Kemudian lanjutkan dengan melakukan klik File → New Project seperti pada gambar 5.3



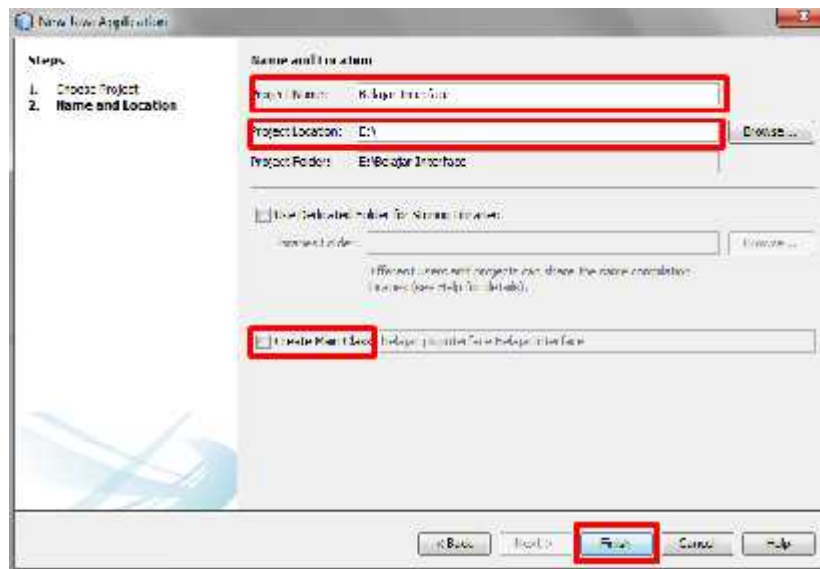
Gambar 5.3 Tampilan IDE Netbeans

2. Jika berhasil maka akan muncul tampilan dialog untuk menentukan jenis project seperti pada gambar 5.4, lanjutkan dengan memilih Java Application dan klik Next



Gambar 5.4 Penentuan jenis project

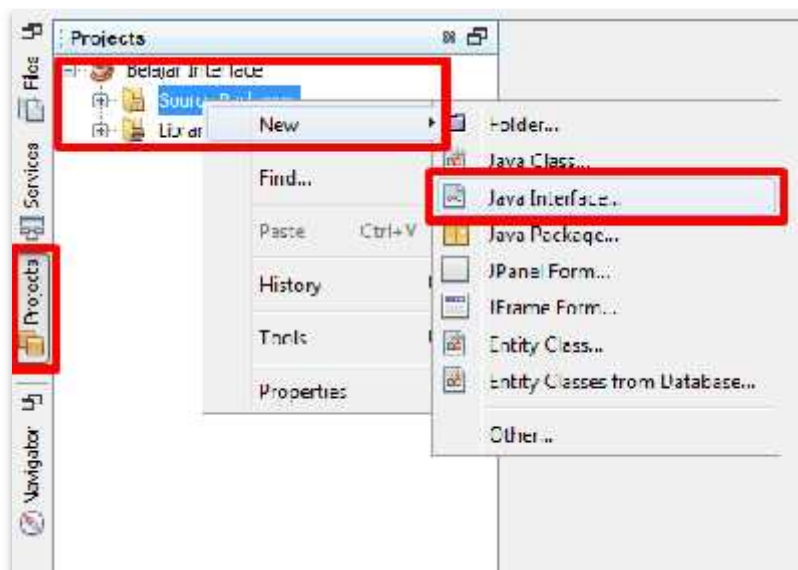
3. Isikan parameter yang dibutuhkan dalam project seperti pada gambar 5.5



Gambar 5.5 Pengisian parameter project Interface

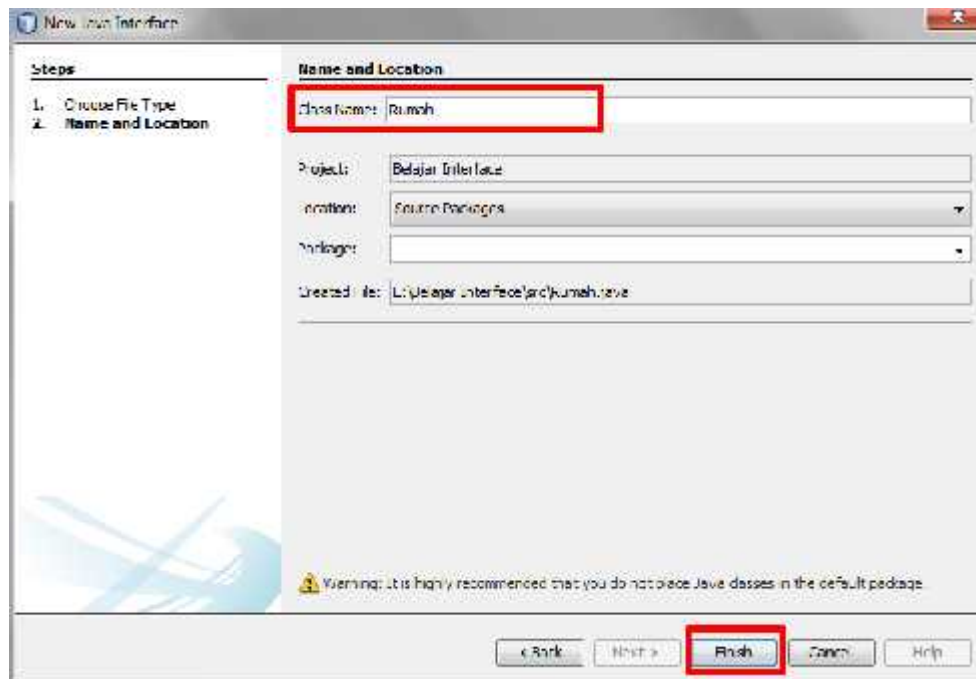
Pada contoh tersebut field Project Name diisi dengan Belajar Interface, kemudian tempat penyimpanan file di drive E:\ disesuaikan dengan komputer masing-masing, dan hilangkan centang pada field Create Main Class.

4. Klik pada tab projects, kemudian klik kanan pada bagian source packages dan pilih New → Java Interface seperti pada gambar 5.6



Gambar 5.6 Pembuatan interface

- Isikan class name yang akan dibuat, misal nama class Rumah, kemudian lanjutkan dengan klik next seperti pada gambar 5.7



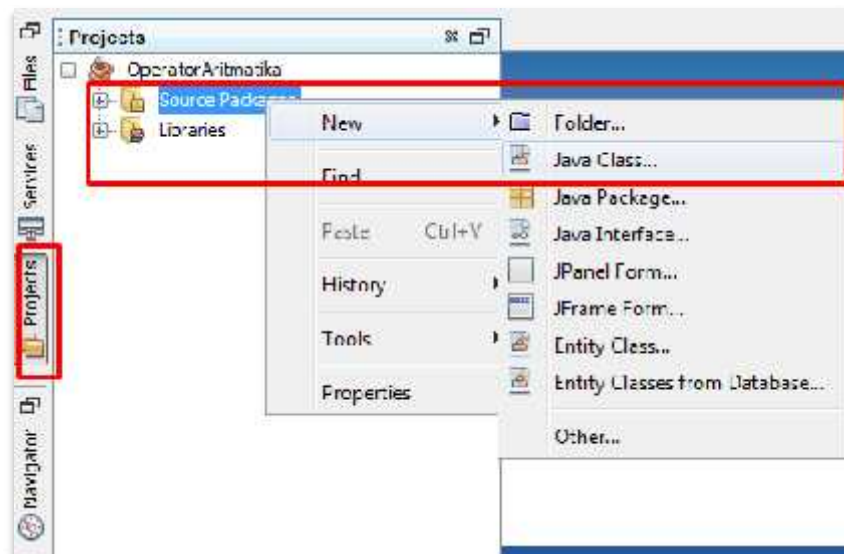
Gambar 5.7 Pengisian Class Name

- Isikan file Rumah.java dengan listing program seperti pada listing 5.1 berikut ini

```
public interface Rumah {  
    String jenis_rumah = "Rumah Adat";  
    int jumlah_lantai =2;  
    public void setRumah(int jml_pintu,int jml_jendela);  
    public void setTypeRumah(int tipe_rumah);  
    public void infoRumah();  
}
```

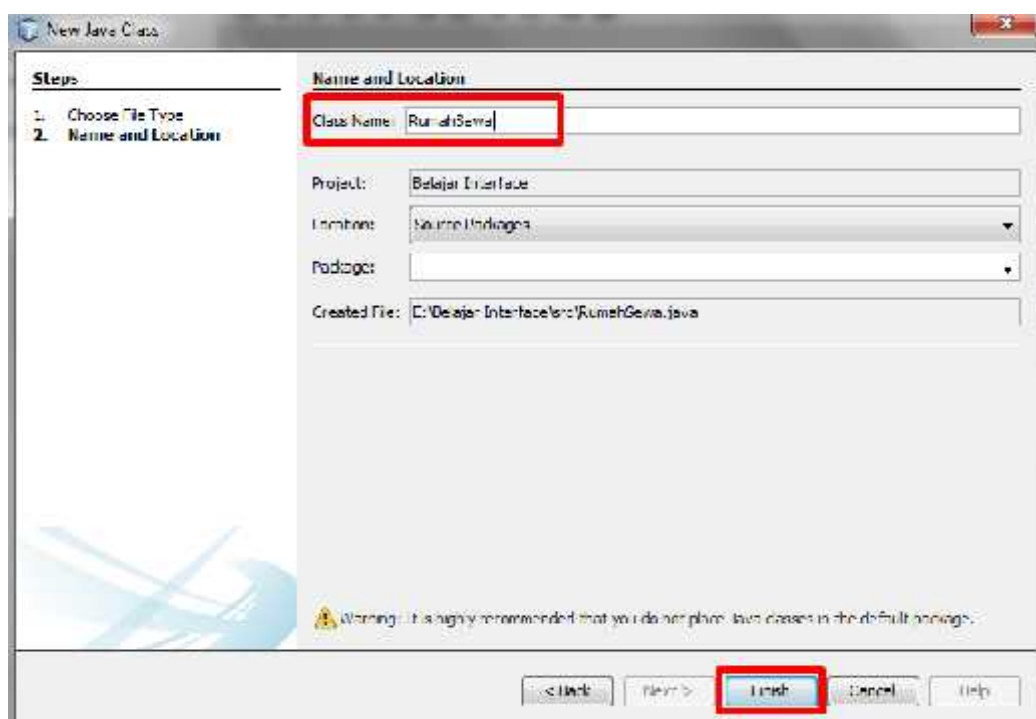
Listing 5.1 Interface Rumah

- Klik pada tab projects, kemudian klik kanan pada bagian source packages dan pilih New → Java Class seperti pada gambar 5.8



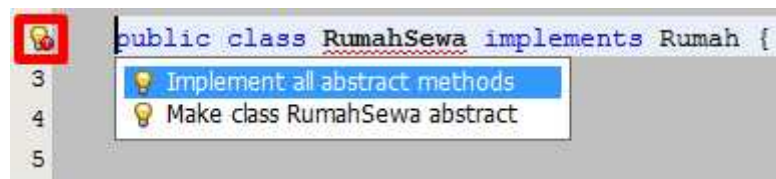
Gambar 5.8 Pembuatan Java Class

8. Isikan nama class dengan RumahSewa seperti pada gambar 5.9 berikut ini

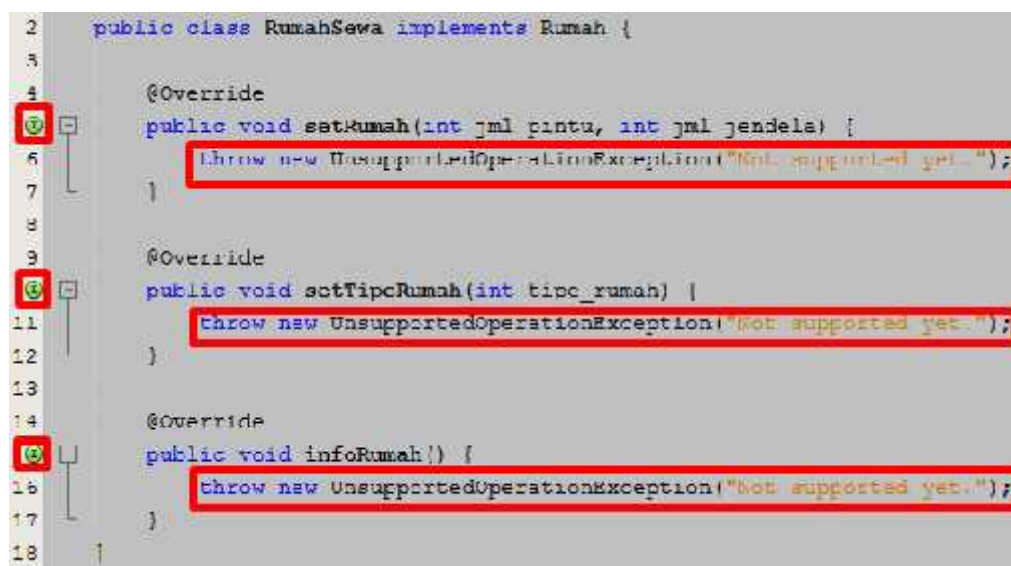


Gambar 5.9 Pengisian nama class

9. Isikan file RumahSewa.java seperti pada listing 5.2 berikut ini, hasil dari kode program tersebut akan error dikarenakan class tersebut belum mengimplementasikan semua method yang dimiliki oleh interface Rumah. Solusinya adalah melakukan klik pada tanda icon lampu di sebelah kiri editor, dan pilih menu **Implement all abstract methods**, sehingga akan muncul listing program yang di generate oleh Netbeans seperti pada gambar 5.10



Listing 5.2 Kode program class RumahSewa



Gambar 5.10 Hasil generate listing Netbeans

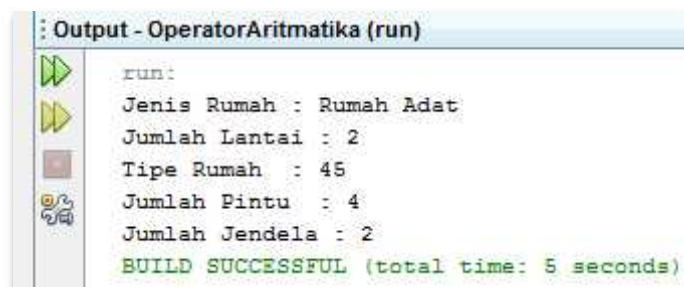
Tanda icon berwarna hijau di sebelah kiri editor adalah menandakan bahwa method tersebut melakukan override dari interface Rumah. Hapus listing program "throw new UnsupportedOperationException("Not supported yet.");"

10. Kemudian tambahkan kode program seperti pada listing 5.3 berikut ini

```
public class RumahSewa implements Rumah {  
  
    int jml_pintu;  
    int jml_jendela;  
    int tipe_rumah;  
    public void setRumah(int jml_pintu, int jml_jendela) {  
        this.jml_pintu = jml_pintu;  
        this.jml_jendela = jml_jendela;  
    }  
  
    public void setTypeRumah(int tipe_rumah) {  
        this.tipe_rumah = tipe_rumah;  
    }  
  
    public void infoRumah() {  
        System.out.println("Jenis rumah : " + RumahSewa.jenis_rumah);  
        System.out.println("Jumlah Lantai : " + RumahSewa.jumlah_lantai);  
        System.out.println("Tipe Rumah : " + this.tipe_rumah);  
        System.out.println("Jumlah Pintu : " + this.jml_pintu);  
        System.out.println("Jumlah Jendela : " + this.jml_jendela);  
    }  
  
    public static void main(String []args){  
        RumahSewa rs = new RumahSewa();  
        rs.setRumah(4,2);  
        rs.setTypeRumah(45);  
        rs.infoRumah();  
    }  
}
```

Listing 5.3 Class RumahSewa

11. Jalankan program dengan menekan tombol Shift + F6, dan hasil output nya adalah seperti gambar 5.11 berikut ini

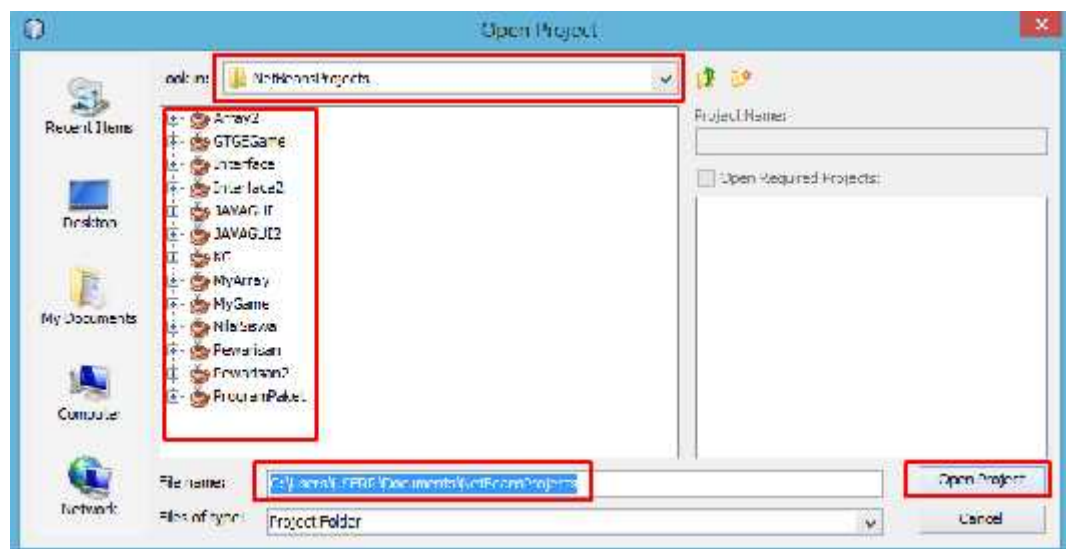


Gambar 5.11 Hasil output program

3. Membuat Paket dengan Program

Sebuah program yang telah siap pakai, maka harus di distribusikan dalam bentuk paket. Fungsi dari pembuatan paket ini adalah agar program tersebut dapat dijalankan di semua komputer, tanpa melalui editor pemrograman seperti Netbeans. Bentuk distribusinya berupa paket installer dimana fungsinya adalah melakukan registrasi setiap library yang digunakan dalam program seperti pada program menggunakan JAVA, maka setiap komputer yang menjalankannya memerlukan JRE (Java Runtime Environment) . Terdapat beberapa tahapan dalam pembuatan paket yaitu

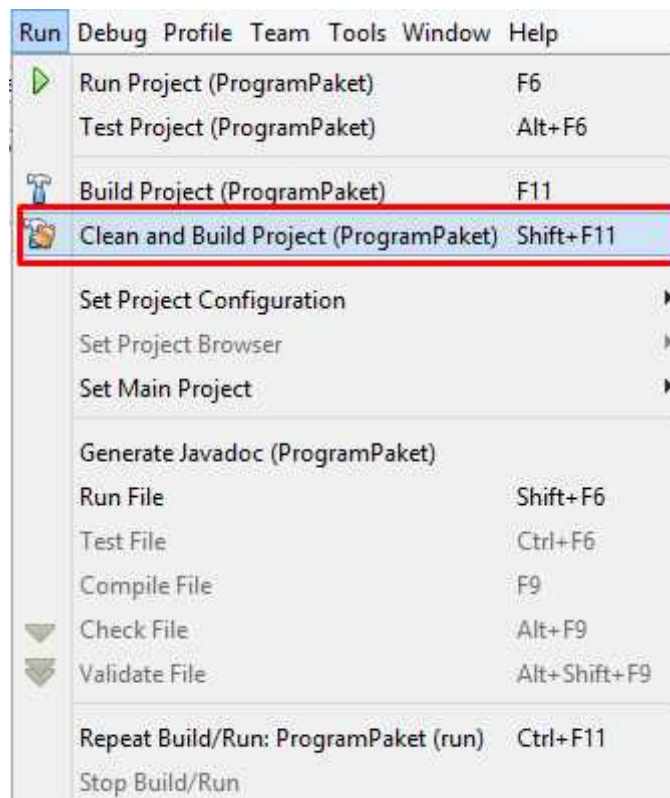
1. Membuat aplikasi berbentuk JAR (Java Archive) kusus untuk aplikasi yang dibuat menggunakan Bahasa pemrograman JAVA. Cara untuk membuatnya adalah jalankan editor Netbeans, kemudian pilih menu File → Open Project maka akan tampil dialog box dan kemudian pilih project program yang akan dijadikan JAR menggunakan, seperti pada gambar 5.12 berikut ini



Gambar 5.12 Tampilan Dialog box Open Project

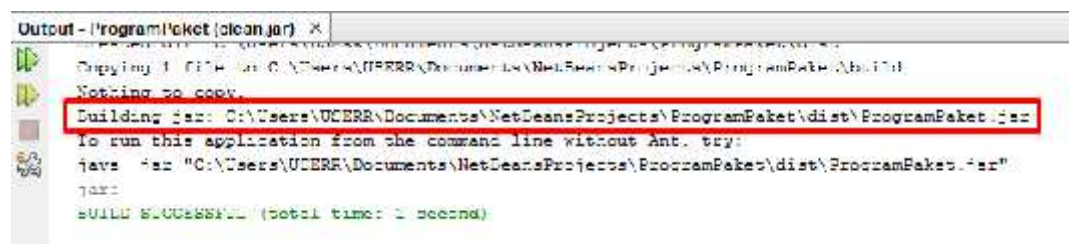
Pilihlah lokasi folder penyimpanan project yang telah dibuat. Jika belum mempunyai project yang akan dibuat, dapat mengunduh contoh program pada alamat url <http://bit.ly/2EQIhc0>,

lakukan pengekstrakan file tersebut menggunakan program archiever seperti Win Rar atau Winzip, kemudian buka project melalui netbeans. Proses selanjutnya adalah membuat file JAR dengan memilih menu Run → Clean and Build Project seperti pada gambar 5.13 berikut ini



Gambar 5.13 Menu Clean and Build Project

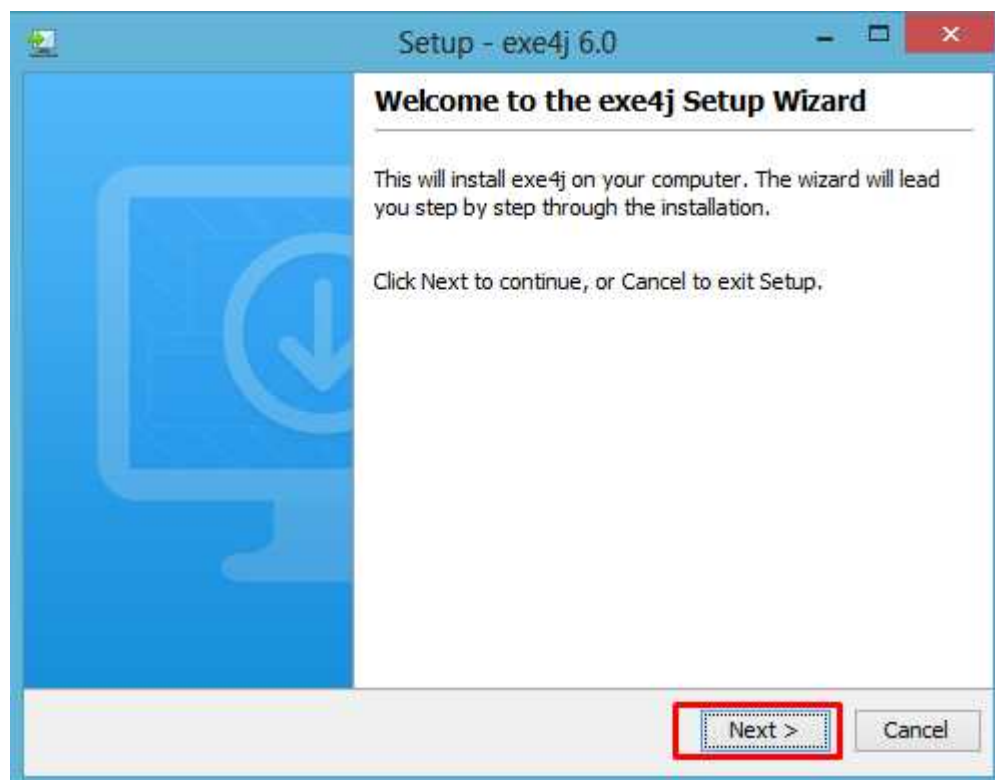
Jika proses berhasil, maka perhatikan pada tampilan output disisi bawah editor terdapat keterangan bahwa file JAR telah sukses dibuat, beserta letak lokasi, pada umumnya file tersebut akan berada pada folder dist, seperti pada gambar 5.14 berikut ini



Gambar 5.14 Hasil dari Pembuatan file JAR

2. Mengkonversi file JAR (Java Archive) menjadi file exe (executable)

Pada proses ini, dibutuhkan software tambahan yang mungkin sangat banyak di internet, akan tetapi pada modul ini menggunakan produk EXE4J yang dapat anda unduh di alamat url <http://bit.ly/2GpCqq5> . Lakukan pengunduhan dan sesuaikan dengan sistem operasi pada komputer masing-masing, kemudian jalankan software EXE4J dengan melakukan double klik pada file yang telah diunduh, maka akan tampil seperti pada gambar 5.15 berikut ini dan klik Next



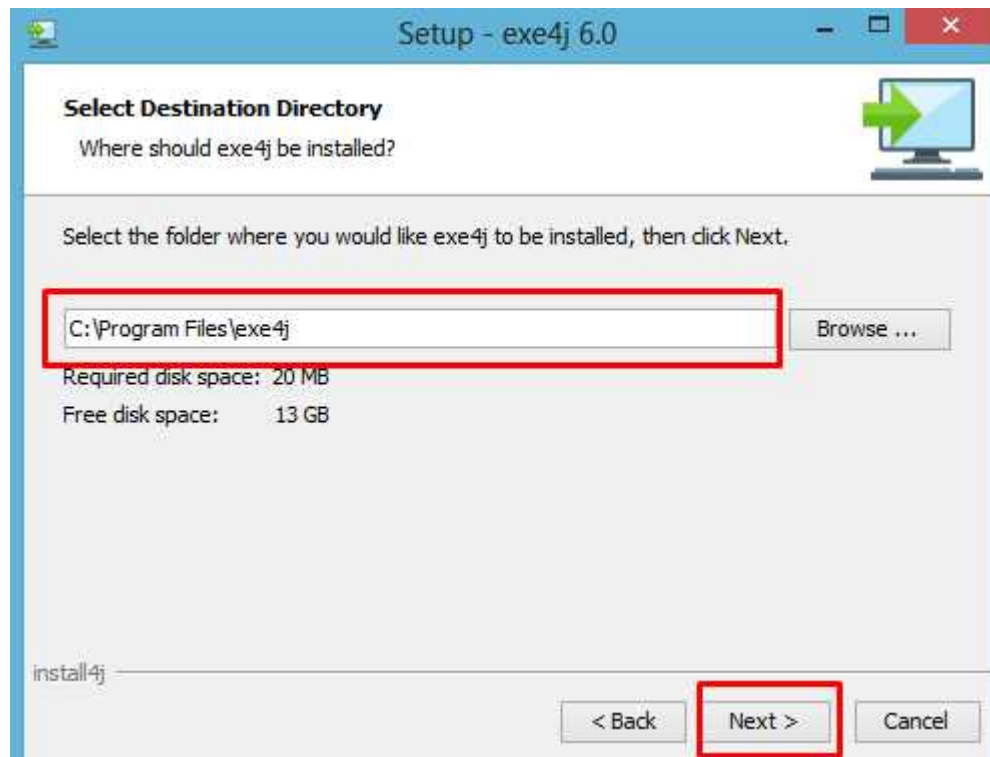
Gambar 5.15 Setup EXE4J

Proses berikutnya adalah tampil dialog konfirmasi persetujuan untuk menyatakan bahwa pengguna telah menyetujui aturan lisensi yang telah ditetapkan oleh pembuat software EXE4J. Centang pilihan "I accept the agreement" seperti pada gambar 5.16



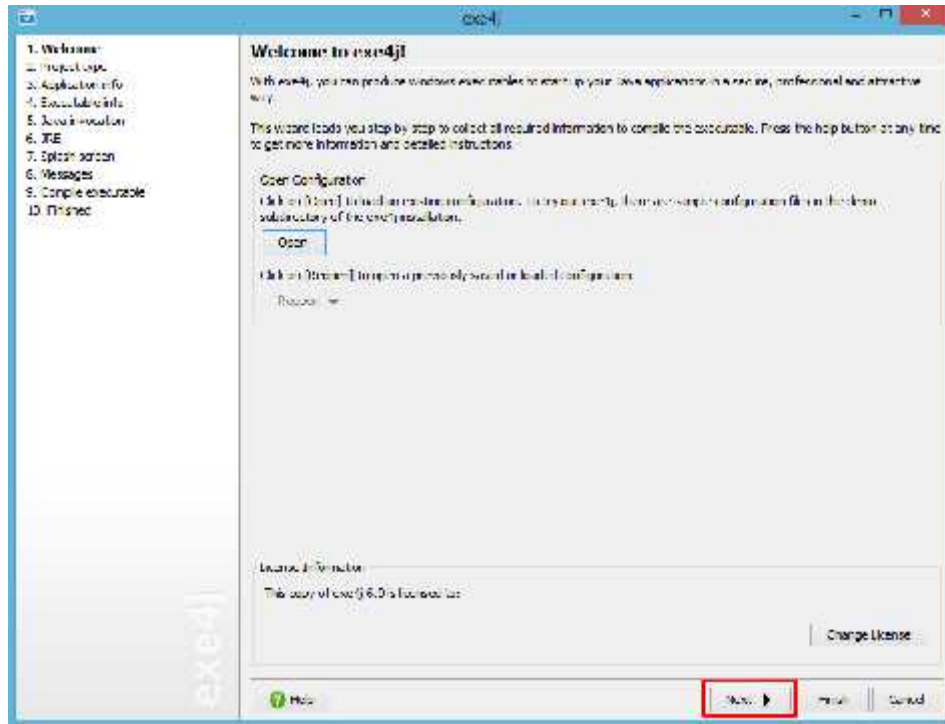
Gambar 5.16 Persetujuan Lisensi

Kemudian tampil pilihan lokasi folder untuk hasil instalasi, kemudian pilih tombol Next seperti pada gambar 5.17



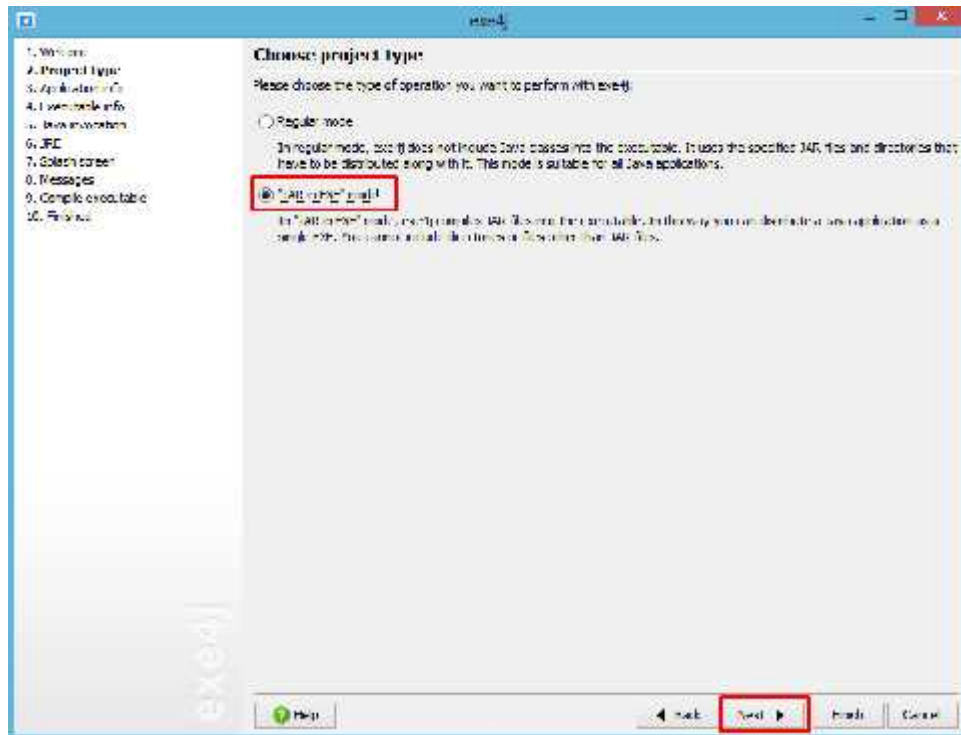
Gambar 5.17 Pemilihan direktori instalasi

Tunggu sejenak untuk menyelesaikan proses instalasi, setelah selesai lakukan klik Run dan akan tampil seperti pada gambar 5.18 berikut ini



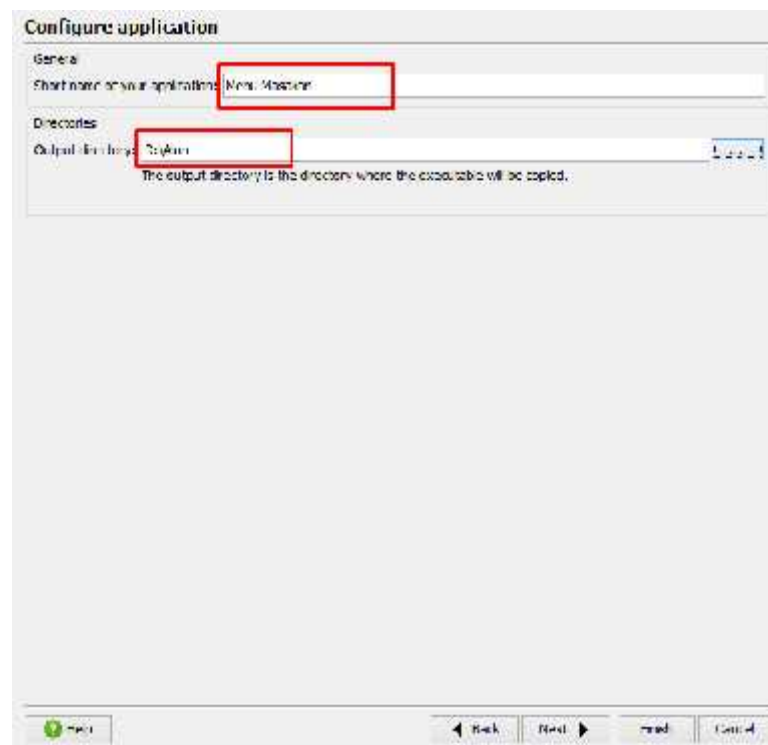
Gambar 5.18 Tampilan awal EXE4J

Kemudian klik tombol Next, maka akan tampil seperti pada gambar 5.19



Gambar 5.19 Pemilihan Tipe project

Pilihlah opsi JAR in EXE mode, dan lanjutkan dengan melakukan klik pada tombol Next dan akan tampil seperti pada gambar 5.20 berikut ini



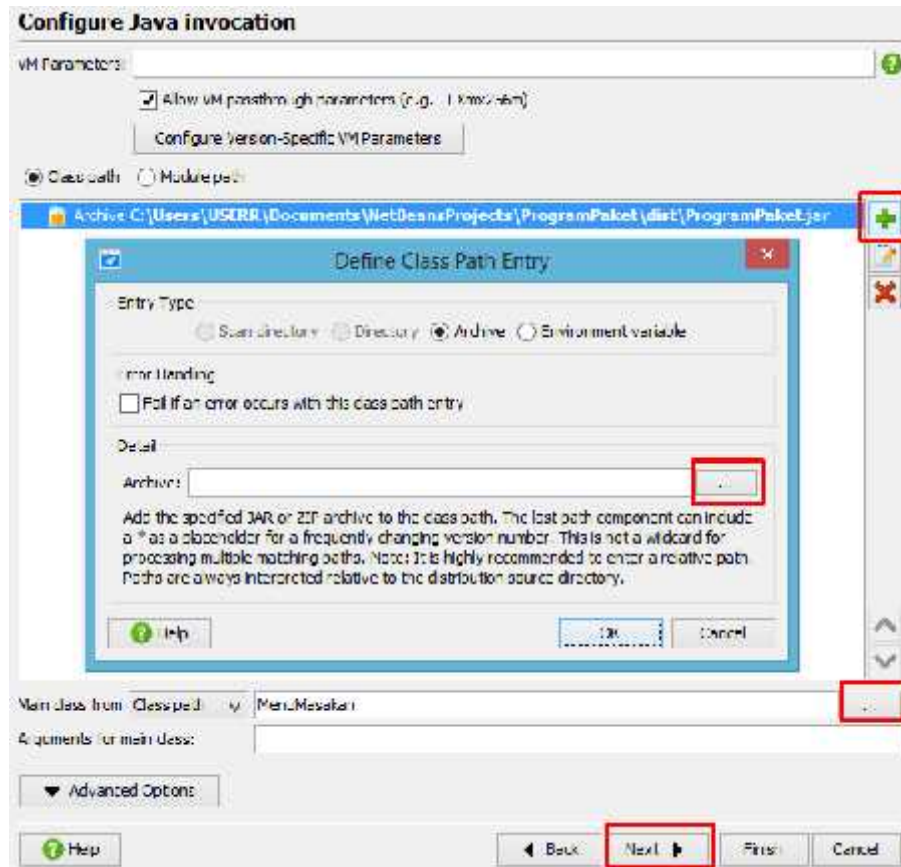
Gambar 5.20 Pengisian nama aplikasi dan output direktori

Isikan nama dari aplikasi yang akan dibuat beserta lokasi direktori hasil pembuatan installer, kemudian lanjutkan menekan tombol Next dan akan tampil seperti gambar 5.21 berikut ini



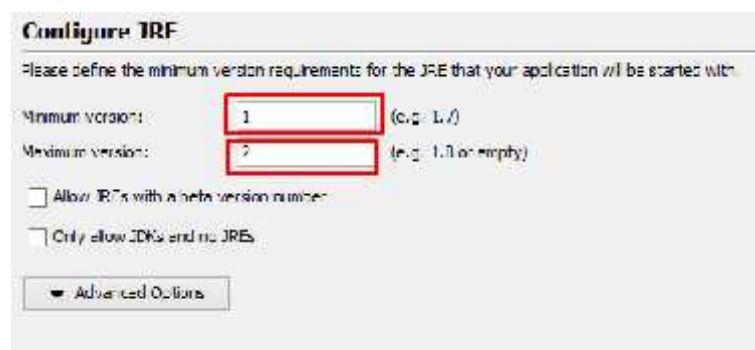
Gambar 5.21 Tipe executable

Proses berikutnya adalah pemilihan file JAR yang telah dibuat pada tahapan sebelumnya dengan cara melakukan klik pada tombol +, kemudian isikan lokasi dari file JAR dan isikan main method untuk setting Class path, lanjutkan dengan klik pada tombol Next seperti pada gambar 5.22 berikut ini



Gambar 5.22 Pengisian file JAR dan Main Class

Tahapan berikutnya adalah untuk konfigurasi JRE, isikan nilai minimum = 1 dan maximum =2 dan lanjutkan dengan klik pada tombol Next seperti pada gambar 5.23 berikut ini

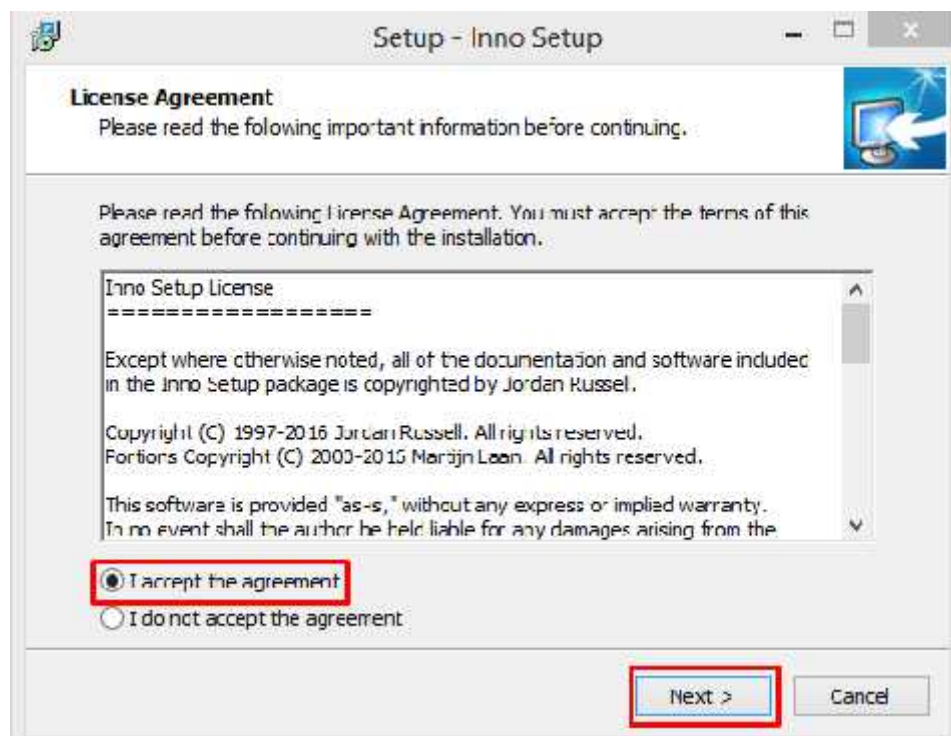


Gambar 5.23 Konfigurasi JRE

Selanjutnya adalah hanya melakukan klik Next jika tanpa gambar splash screen, tetapi jika akan menggunakan isikan file gambar yang akan digunakan, kemudian klik Next sekali lagi dan proses akan berjalan membuat file executable.

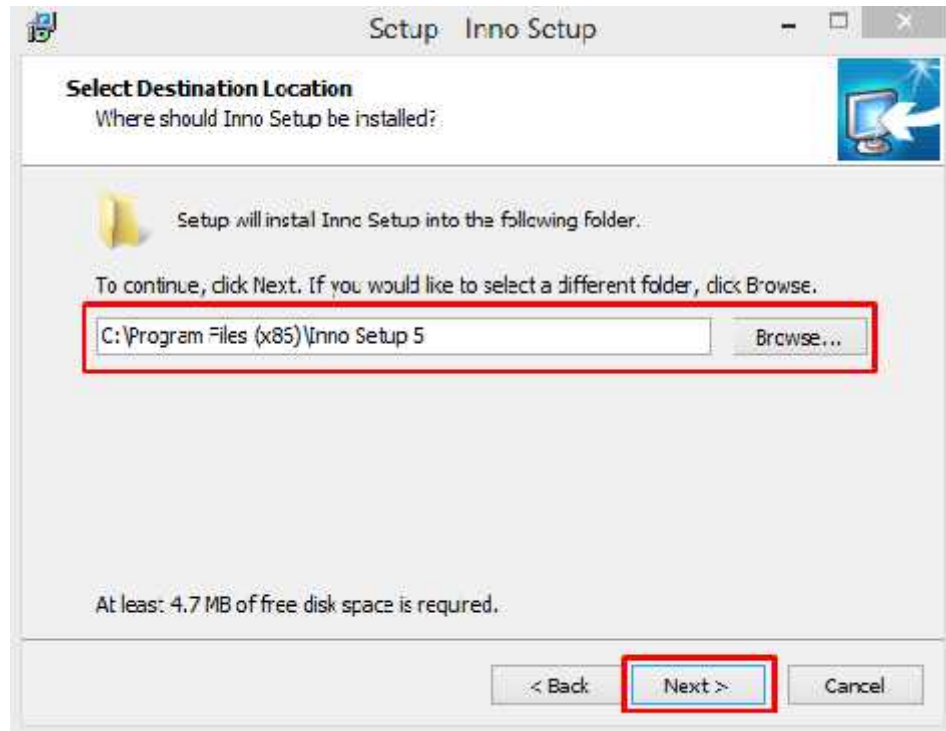
3. Pembuatan installer menggunakan program

Pada tahapan ini menggunakan software Inno Setup, yang dapat diunduh pada alamat url <http://www.jrsoftware.org/isdl.php> . Setelah berhasil melakukan pengunduhan maka jalankan file tersebut maka akan tampil persetujuan lisensi, pilih I accept the agreement dan klik tombol Next seperti pada gambar 5.24 berikut ini



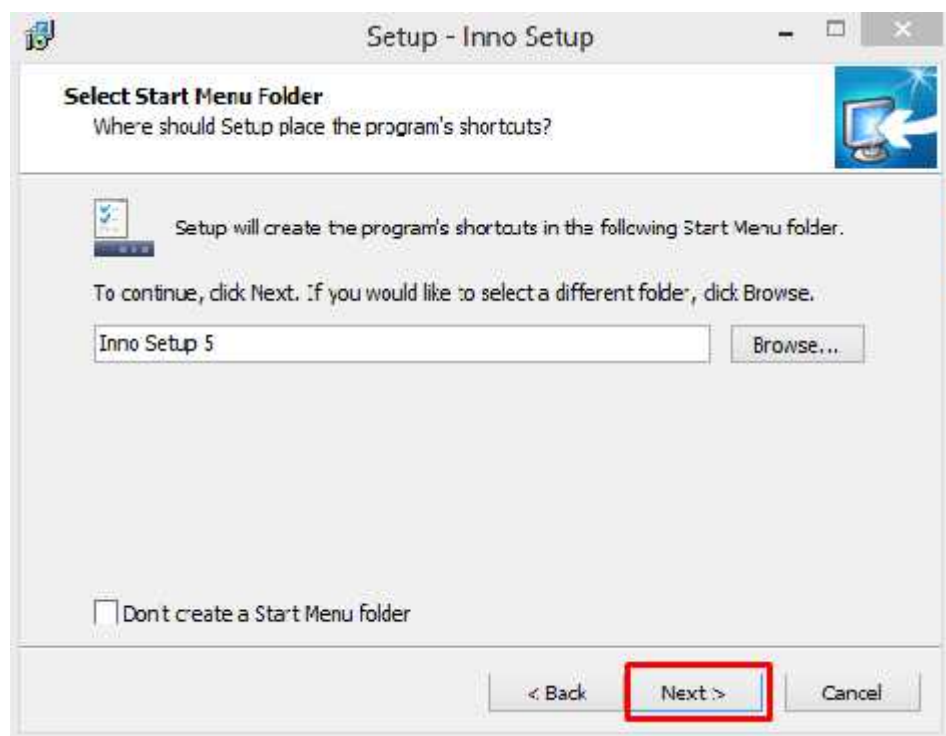
Gambar 5.24 Persetujuan lisensi Inno Setup

Proses selanjutnya adalah menentukan letak direktori hasil instalasi Inno Setup dan lanjutkan klik tombol Next seperti pada gambar 5.25 berikut ini



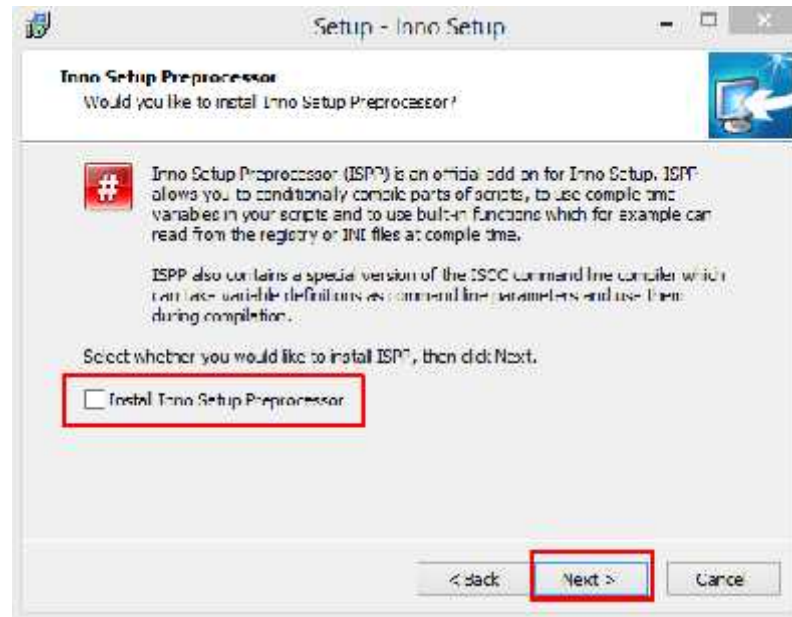
Gambar 5.25 Pemilihan folder instalasi

Proses berikutnya adalah pembuatan shortcut untuk ditampilkan pada start menu folder, pada tahapan ini biarkan semua diatur secara default, lanjutkan dengan klik tombol Next seperti pada gambar 5.26 berikut ini



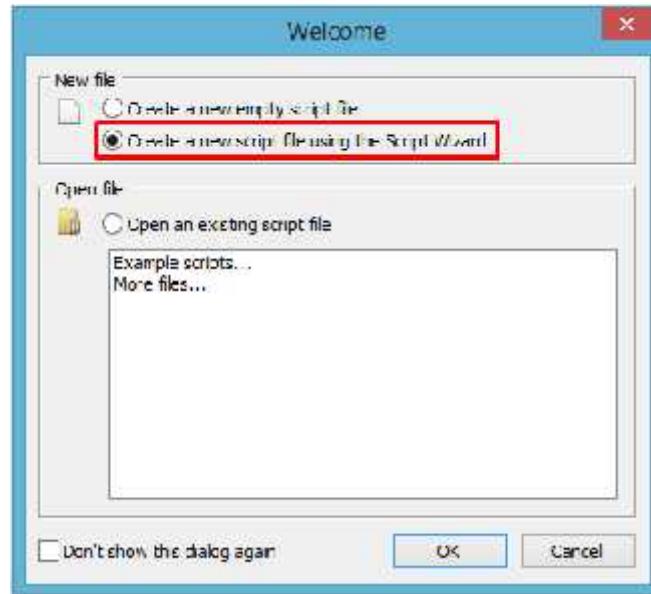
Gambar 5.26 Pembuatan shortcut

Langkah selanjutnya adalah melakukan uncheck pada pilihan Install Inno Setup Preprocessor dan klik tombol Next, artinya saat ini belum diperlukan untuk menambahkan plugin tambahan seperti pada gambar 5.27 berikut ini



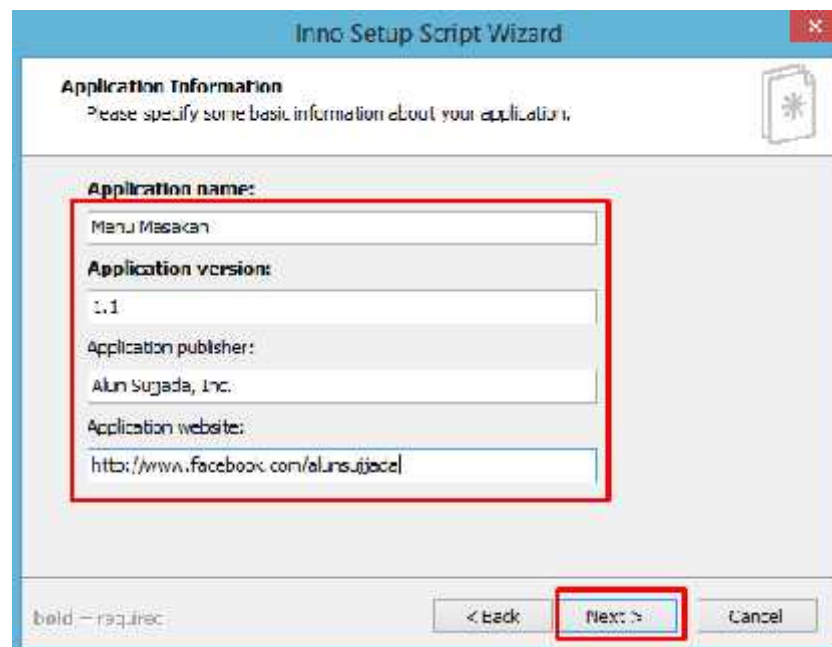
Gambar 5.27 Instalasi Inno Setup Preprocessor

Kemudian lewatkan bagian Additional task dan langsung melakukan klik tombol Next, kemudian aplikasi Inno Setup akan siap untuk di install dan lakukan klik tombol install. Setelah semua proses instalasi selesai, jalankan program Inno Setup maka akan tampil program seperti pada gambar 5.28 dan pilih opsi Create a new script file using the script wizard



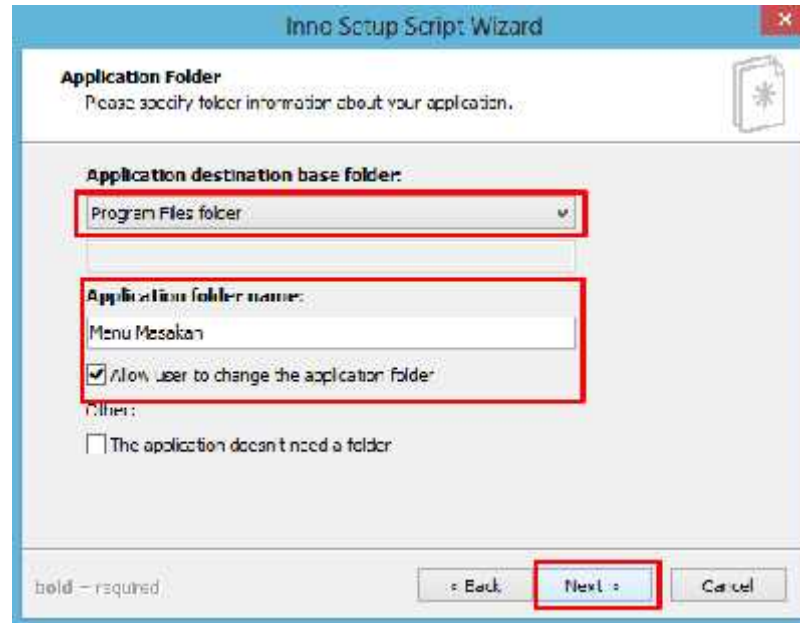
Gambar 5.28 Pemilihan Tipe installer

Setelah melakukan klik pada tombol OK maka akan muncul dialog wizard, pada tahapan ini cukup dengan klik tombol Next dan akan muncul dialog pengisian parameter seperti nama aplikasi, versi, penerbit dan alamat website seperti pada gambar 5.29 berikut ini



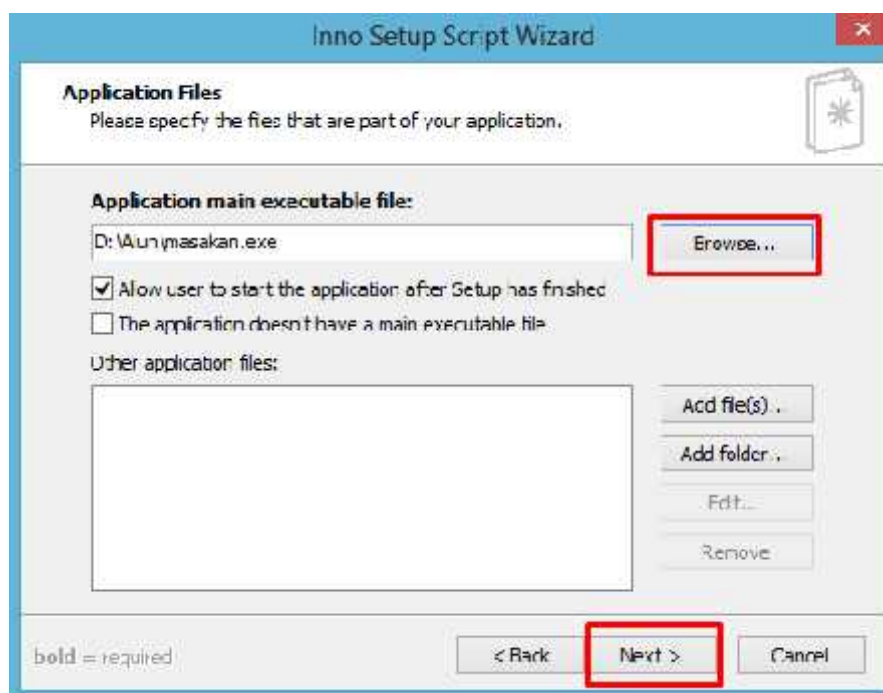
Gambar 5.29 Dialog pengisian parameter aplikasi

Setelah melakukan klik tombol Next, proses selanjutnya adalah pemilihan spesifikasi folder hasil instalasi file executable seperti pada gambar 5.30 berikut ini



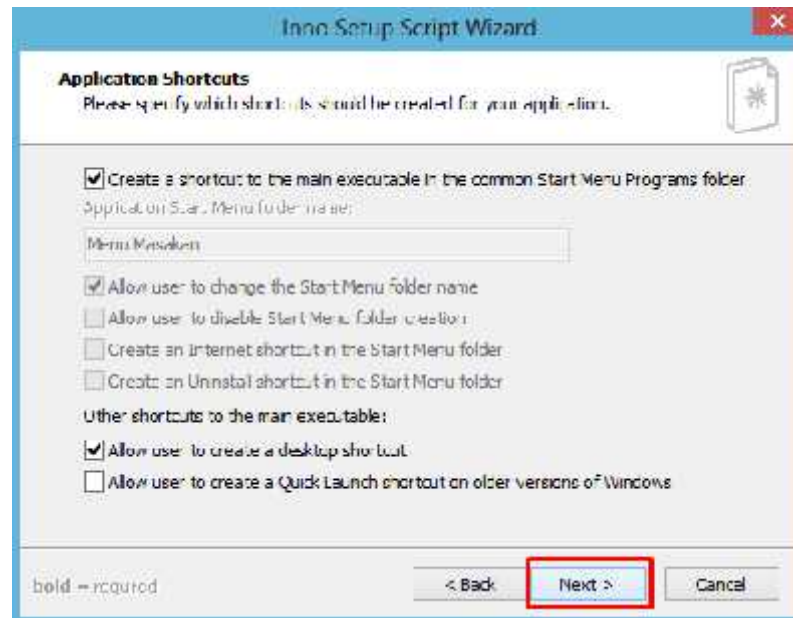
Gambar 5.30 Spesifikasi folder hasil instalasi

Tahapan selanjutnya adalah pemilihan file executable hasil dari konversi JAR yang telah dibuat pada tahapan sebelumnya dengan melakukan klik pada tombol Browse dan lanjutkan dengan klik tombol Next seperti pada gambar 5.31 berikut ini



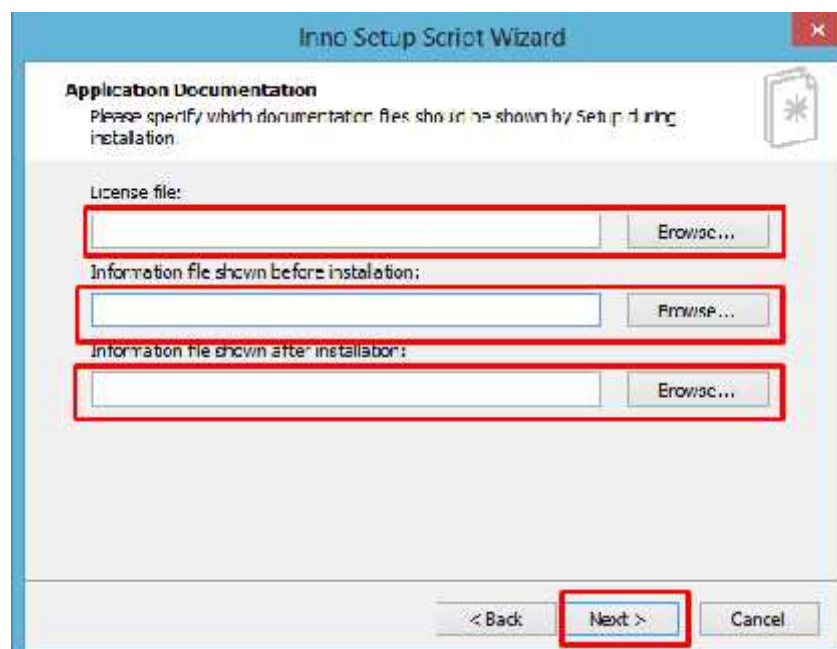
Gambar 5.31 Spesifikasi Aplikasi

Proses selanjutnya adalah melakukan klik pada dialog pemilihan shortcut dan klik tombol Next seperti pada gambar 5.32 berikut ini



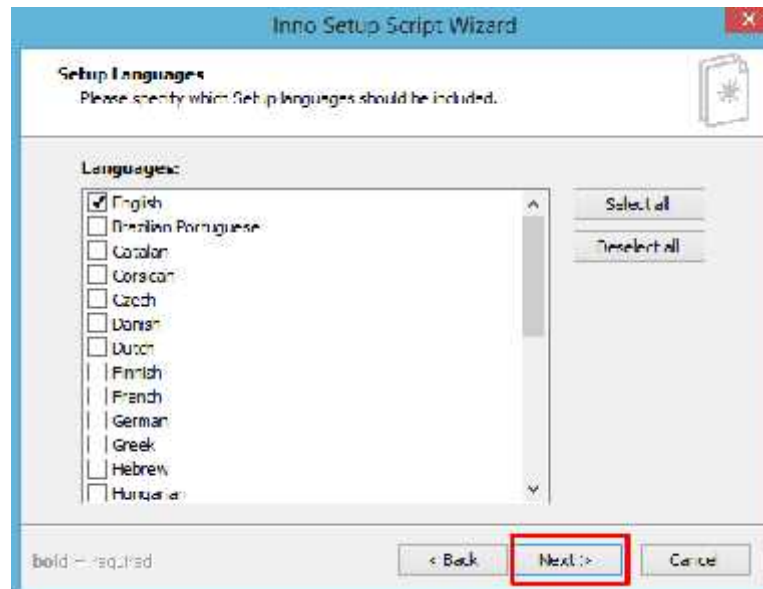
Gambar 5.32 Spesifikasi shortcut aplikasi

Tahapan berikutnya adalah pengisian dokumentasi aplikasi jika mempunyai lisensi, informasi file sebelum instalasi dan sesudah instalasi, untuk tahapan saat ini dikosongkan terlebih dahulu dan lanjutkan dengan klik tombol Next seperti pada gambar 5.33 berikut ini



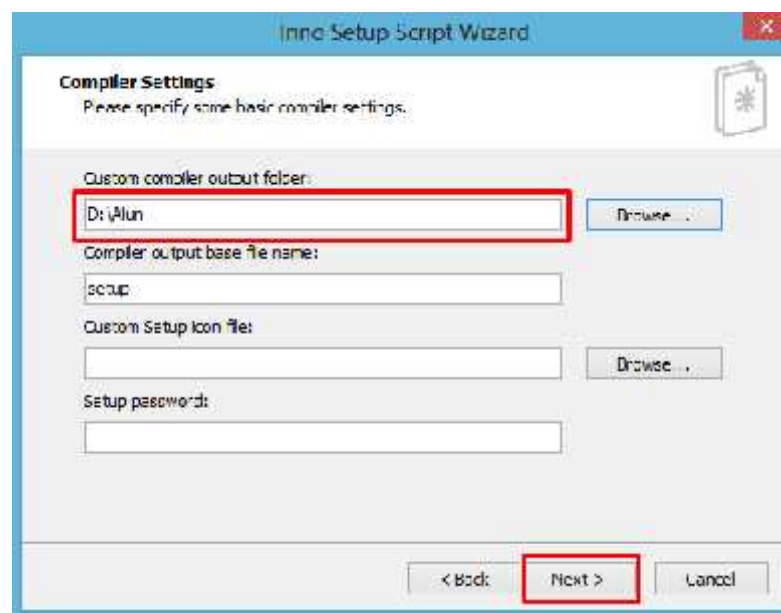
Gambar 5.33 Dokumentasi Aplikasi

Proses berikutnya adalah pemilihan Bahasa yang dipakai pada saat proses instalasi, sebagai bahasa default adalah menggunakan Bahasa Inggris, dan lanjutkan dengan klik tombol Next seperti pada gambar 5.34 berikut ini



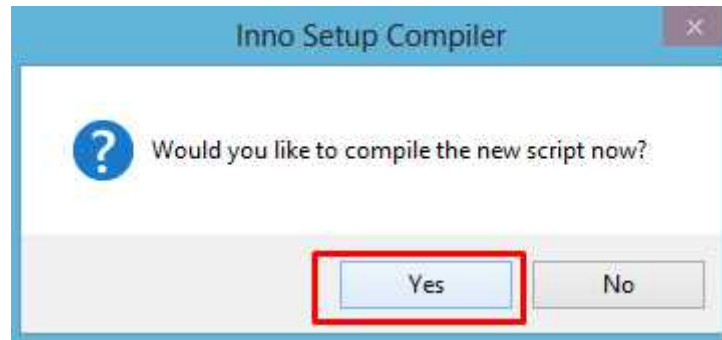
Gambar 5.34 Pemilihan Bahasa

Tahapan selanjutnya adalah menentukan letak hasil dari pembuatan installer dan lanjutkan dengan klik tombol Next seperti pada gambar 5.35 berikut ini



Gambar 5.35 Penentuan folder hasil pembuatan installer

Kemudian lanjutkan dengan klik tombol Yes untuk melakukan proses kompilasi script untuk pembuatan file installer seperti pada gambar 5.36 berikut ini



Gambar 5.36 Dialog kompilasi script

Jika proses berhasil maka pada folder output yang telah diisikan akan terbuat file secara otomatis file setup.exe. Double klik file tersebut untuk melakukan instalasi program.

B. Keterampilan yang diperlukan dalam Membuat Program Object Oriented Dengan Interface Dan Paket

1. Mengidentifikasi penggunaan Inheritance
2. Menulis class 2 (dua) atau lebih pada file yang berbeda.
3. Mengorganisasikan Super class dan Sub class.
4. Mengimplementasikan polymorphism
5. Mengimplementasikan overriding method dan overloading method.
6. Menggunakan keyword super
7. Mengoperasikan software IDE (Integrated Development Environment) untuk bahasa pemrograman JAVA seperti Netbeans, GEL, Eclipse dan lain-lain
8. Melakukan instalasi dan menjalankan software EXE4J.
9. Melakukan instalasi dan menjalankan software Inno Setup.

C. Sikap kerja yang diperlukan dalam Membuat Program Object Oriented Dengan Interface Dan Paket

Harus bersikap secara:

1. Cermat dan teliti dalam mengorganisasikan super class dan sub class
2. Sering melakukan uji coba kode program.
3. Sesuai dengan kaidah penulisan sintaks bahasa pemrograman JAVA.
4. Berpikir analitis serta evaluatif waktu melakukan uji coba kode program.

BAB VI

MENGGOMPILASI PROGRAM

A. Pengetahuan yang Diperlukan dalam Mengkompilasi Program

1. Perbaiki Kesalahan

Sebuah program yang muncul pesan kesalahan atau error adalah hal yang wajar pada dunia pemrograman. Semakin kompleks dan besar skala program maka akan semakin banyak kemungkinan terjadinya error. Hampir tidak ada satupun program yang tidak memiliki error, semua dikarenakan keterbatasan manusia sebagai pembuatnya. Melihat kenyataan tersebut yang dapat dilakukan oleh seorang programmer adalah berusaha untuk mencari kesalahan-kesalahan tersebut baik kesalahan besar ataupun kecil untuk diperbaiki terus menerus, melakukan update program dengan tujuan untuk meminimalisir sebuah error pada program. Bayangkan jika kesalahan menuliskan hasil output 1000 menjadi 100, sudah pasti hal tersebut akan mengganggu kinerja dari penggunanya. Untuk mengenali kesalahan-kesalahan tersebut maka setidaknya seorang programmer mengetahui jenis-jenis kesalahan program. Secara garis besar error pada program dibagi menjadi 3 (tiga) yaitu

a. Syntax Error

Kesalahan yang paling sering ditemukan pada saat membuat program adalah kesalahan sintaks atau Syntax Error, dimana perintah atau statement yang diketikkan menyalahi aturan pengkodean yang dimiliki oleh bahasa pemrograman yang digunakan.

Sebuah bahasa pemrograman memiliki aturan pengkodean tersendiri yang harus dipatuhi, sebagai contoh pada bahasa pemrograman JAVA, setiap statement diwajibkan diakhiri dengan tanda titik koma (;). Jika tidak dituliskan, maka program akan menampilkan pesan Syntax Error pada saat dijalankan. Setiap

bahasa pemrograman memiliki keyword, yaitu perintah-perintah baku yang digunakan. Sebagai contoh, keyword yang umum adalah kondisi if, perulangan for atau while, penulisan fungsi dan lambang aritmatika seperti modulus, pangkat, dan lain-lain. Kesalahan penulisan keyword juga merupakan Syntax Error.

Kesalahan penulisan parameter pada sebuah function atau procedure juga termasuk dalam kategori Syntax Error, misalnya jika function yang digunakan memerlukan parameter sementara programmer lupa menuliskan parameter tersebut. Syntax Error merupakan jenis kesalahan yang paling sering ditemui, tetapi juga pada umumnya paling mudah untuk ditanggulangi. Syntax Error cukup mudah diketahui karena setiap bahasa pemrograman yang digunakan akan menampilkan pesan kesalahan dan menunjukkan baris kesalahan dengan tepat. Pada beberapa bahasa pemrograman, disediakan fasilitas Auto Syntax Check, dimana akan muncul sebuah pesan peringatan ketika mengetikkan sintaks yang salah.

b. Run-time Error

Jenis kesalahan Run-time Error terjadi ketika kode program melakukan sesuatu yang tidak dimungkinkan. Contohnya pada saat sebuah aplikasi mencoba mengakses file yang tidak terdapat dalam media penyimpanan, atau terjadi kesalahan alokasi ruang memori. Terkadang Run-time Error terjadi karena berbagai aspek dan tidak selalu karena kesalahan pemrograman, sebagai contoh jika programmer sengaja menghapus beberapa file penting yang digunakan oleh suatu aplikasi, maka terdapat kemungkinan akan terjadi Run-time Error pada saat aplikasi tersebut dijalankan. Run-time Error menyebabkan sebuah program akan langsung berhenti. Walaupun demikian, pencegahan semaksimal mungkin dengan memberikan validasi dan pesan yang user-friendly saat terjadi kesalahan pada aplikasi, akan sangat membantu untuk

mengetahui mengapa aplikasi tidak berjalan dengan semestinya. Salah satu cara mengatasi Run-time error dalam bahasa pemrograman JAVA yaitu menggunakan blok kode try catch dan finally, jadi jika terdapat kesalahan maka akan muncul dialog box sesuai dengan apa yang akan diinformasikan mengenai kesalahan tersebut. Contoh penggunaanya seperti pada listing program 6.1 berikut ini

```
import javax.swing.*;
public class KoreksiKesalahan {

    public static void main(String [] args) {

        String bilangan1 = JOptionPane.showInputDialog("Tekan Bilang Bilangan 1 :");
        String bilangan2 = JOptionPane.showInputDialog("Tekan Bilang Bilangan 2 :");

        int hasil = Integer.parseInt(bilangan1) / Integer.parseInt(bilangan2);
        JOptionPane.showMessageDialog(
            null, "Hasil * = " + bilangan1 + " dibagi dengan " + bilangan2 + " = " + hasil);
    }
}
```

Listing 6.1 Contoh program sederhana

Contoh program sederhana pada listing 6.1 adalah berfungsi untuk menghitung pembagian 2(dua) bilangan. Program tersebut tidak mengalami kesalahan sintaks, karena terbukti bahwa program tersebut dapat dikompilasi dan dijalankan, selain itu juga tidak terdapat kesalahan logika (logical error), karena ketika program diisikan bilangan1 dengan nilai 60 dan bilangan2 dengan nilai 5 maka hasil output yang tampil adalah 12 seperti pada gambar 6.1 berikut ini



Gambar 6.1 Hasil output program pembagian

Akan tetapi program tersebut mengalami Run-time Error ketika nilai pada bilangan2 diisi dengan angka 0(nol), seperti pada gambar 6.2, karena dalam operasi aritmatika tidak diijinkan melakukan pembagian dengan 0(nol)

```
exception in thread "main" java.lang.ArithmeticException: / by zero
|   at KoreksiKesalahan.main(KoreksiKesalahan.java:10)
C:\Users\USER\AppData\Local\NetBeans\Cache\8.2\workspace-snippets\run.xml:53: Java returned: 1
BUILD FAILED (total time: seconds)
```

Gambar 6.2 Output hasil pembagian dengan bilangan 0

Untuk mengatasi hal tersebut seorang programmer perlu memperbaiki Runtime Error dengan menggunakan blok try catch dan finally seperti pada listing 6.2 berikut ini

```
import javax.swing.*;
public class KoreksiKesalahan {
    public static void main(String [] args) {
        int hasil=0;
        String bilangan1 = JOptionPane.showInputDialog("Masukan Bilangan 1");
        String bilangan2 = JOptionPane.showInputDialog("Masukan Bilangan 2");

        try{
            hasil = Integer.parseInt(bilangan1) / Integer.parseInt(bilangan2);
        }
        catch(ArithmeticException e){
            JOptionPane.showMessageDialog(null, e.getMessage());
        }
        finally{
            JOptionPane.showMessageDialog(
                null, "Hasil " + bilangan1 + " dibagi dengan " + bilangan2 + " = " + hasil);
        }
    }
}
```

Listing 6.2 Perbaikan kesalahan program

Program tersebut ketika dijalankan dan bilangan2 diisi dengan nilai 0(nol) maka akan muncul pesan kesalahan “/ by zero” seperti pada gambar 6.3 berikut ini



Gambar 6.3 Pesan kesalahan dari sistem

Ataupun dapat digantikan pesan kesalahan sesuai apa yang diinginkan dengan menuliskan program pada bagian catch.

c. Logical Error

Logical Error merupakan jenis kesalahan yang cukup sulit untuk ditemukan penyebabnya. Karena aplikasi yang mengandung Logical Error berjalan tanpa pesan kesalahan, tetapi mengeluarkan hasil yang tidak diharapkan, misalnya jika aplikasi program menghasilkan perhitungan yang salah. Logical Error baru dapat diketahui setelah dilakukan testing dan meninjau hasilnya. Logical Error dapat diperbaiki dengan memeriksa alur program dan nilai variabel yang dihasilkan.

2. Program Debugging

Sebuah error pada aplikasi disebut dengan istilah bug, atau dalam Bahasa Inggris berarti kutu atau binatang kecil. Konon istilah bug muncul karena ditemukannya binatang kecil yang menyebabkan kerusakan pada sebuah komputer tabung pada tahun 1945. Bug aplikasi terdapat pada kode program, yang dapat mengganggu kenyamanan pengguna aplikasi. Pada tingkat tertentu, keberadaan bug cukup memberikan efek yang besar.

Mungkin belum melupakan saat dimana orang ramai membicarakan "Y2K Bug" atau bug tahun 2000, atau munculnya istilah "Blue screen of Death" pada sistem operasi Windows, atau "Kernel Panic" pada sistem operasi Linux, semua contoh tersebut menunjukkan sebuah bug serius dapat mengakibatkan dampak negatif yang cukup besar.

Proses mencari penyebab bug disebut dengan debugging, yang juga merupakan tugas programmer setelah menerima laporan bug. Walaupun demikian, jangan menjadikan pekerjaan sebagai pencari bug, untuk itu hanya ada satu cara, minimalkan bug pada aplikasi yang dibuat. Apa yang harus dilakukan untuk menghindari jenis-jenis kesalahan yang

telah disebutkan di atas. Bisa jadi tidak ada program yang sempurna, tetapi selalu ada cara untuk mengatasi atau menghindari kesalahan semaksimal mungkin.

a. Selalu Deklarasikan Variabel

Syntax Error, bahkan Logical Error, mungkin terjadi jika terdapat penulisan variabel yang salah. Sebaiknya mendeklarasikan variabel yang digunakan walaupun bisa jadi bahasa pemrograman yang digunakan mengizinkan untuk tidak melakukan deklarasi variabel. Visual Basic merupakan salah satu bahasa pemrograman yang mengizinkan penggunaan variabel tanpa deklarasi, walaupun demikian disarankan tetap menggunakan deklarasi variabel. Hal tersebut akan memperkecil kesalahan penulisan variabel. Masih dengan contoh Visual Basic yang dapat menambahkan perintah Option Explicit pada program untuk mencegah kesalahan tulis pada variabel, jika terdapat variabel yang belum dideklarasikan, maka Visual Basic akan menampilkan pesan kesalahan. Sebaiknya memiliki suatu skema standard untuk pemberian nama variabel dan konsisten dengan penggunaannya. Contohnya berikan nama variabel diawali dengan huruf s jika bertipe data string, misalnya sResult, sTemp, dan lain-lain. Pada Visual Basic maupun beberapa bahasa pemrograman lain yang berorientasi object, kita dapat mendeklarasikan variabel dengan tipe data object. Terdapat berbagai jenis macam object yang dikenal, dan sebaiknya menuliskannya dengan lengkap object yang dimaksud. Misalnya object ListBox, Label, dan lain-lain.

b. Gunakan Variabel Lokal

Sangat disarankan agar selalu menggunakan variabel lokal. Salah satu manfaatnya adalah jika terjadi kesalahan program (terutama Logical Error), maka penyebab kesalahan dan solusinya akan lebih mudah ditemukan. Hal ini dikarenakan variabel lokal memiliki ruang lingkup penggunaan yang lebih kecil dibandingkan variabel global, yang dapat diakses oleh procedure yang mana

saja. Penggunaan variabel global, sering menimbulkan kerancuan dan memperbesar kemungkinan terjadinya kesalahan tanpa disadari.

c. Kenali Jenis Bug

Bug yang timbul pada sebuah aplikasi memiliki karakteristik. Karena itu selalu baca dan perhatikan baik-baik pesan kesalahan yang timbul. Beberapa jenis bug berdasarkan karakteristiknya adalah sebagai berikut:

i. Divide By Zero.

Jika pada sebuah pembagian, pembagi bernilai 0, maka program akan terhenti dan mengalami error.

ii. Infinite Loop.

Pengertian loop adalah perulangan, yang sering digunakan dalam teknik pemrograman. Penggunaan loop yang salah dapat mengakibatkan program menjalankan sebuah procedure tanpa akhir.

iii. Arithmetic overflow or Underflow.

Overflow terjadi saat sebuah perhitungan menghasilkan nilai yang lebih besar daripada nilai yang dapat ditampung oleh media/variabel penyimpanan. Sementara underflow merupakan kebalikannya. Pada perhitungan aritmatik, hal ini sering ditemukan dan menjadi masalah.

iv. Exceeding Array Bounds.

Array merupakan variabel berdimensi yang memiliki indeks. Saat program mengakses indeks diluar array yang ditentukan, maka akan mengakibatkan error.

v. Access Violation.

Hal yang terjadi saat sebuah proses mencoba melewati batas yang diijinkan oleh sistem. Misalnya menulis sebuah nilai pada alamat memory, segmen, atau media yang diproteksi.

vi. Memory Leak.

Penggunaan memory yang tidak diinginkan, dapat terjadi karena program gagal melepaskan memory yang sudah tidak digunakan.

vii. Stack Overflow or Underflow.

Stack merupakan struktur data dengan prinsip LIFO (Last In First Out), pada program dapat mengimplementasikan logika stack untuk suatu tujuan. Tetapi jika stack melebihi atau dibawah nilai yang diijinkan oleh program, maka akan timbul kesalahan Stack Overflow/Underflow.

viii. Buffer Overflow.

Buffer merupakan tempat penyimpanan sementara dalam teknik pemrograman. Buffer Overflow terjadi jika menyimpan terlalu banyak data yang tidak dapat ditampung oleh buffer yang disediakan.

ix. Deadlock.

Merupakan suatu kondisi dimana dua atau lebih proses saling menunggu satu sama lain untuk menyelesaikan prosesnya, dan tidak satupun dari proses tersebut yang selesai. Problem deadlock sering ditemukan pada multiprocessing.

x. Off By One Error.

Merupakan istilah untuk menggambarkan perulangan yang terlalu banyak atau terlalu sedikit. Misalnya perulangan yang dikehendaki adalah lima kali, tetapi kenyataan yang terjadi aplikasi mengulang proses tersebut sebanyak empat kali atau enam kali. Kesalahan ini pada umumnya terjadi karena kesalahan logika penulisan kode pada proses perulangan.

xi. Berikan Komentar

Jangan kuatir kode program dipenuhi oleh komentar. Karena akan lebih mudah bagi programmer untuk mempelajari lagi kode-kode program yang pernah dibuat dengan membaca komentar. Dengan mengerti kode program dengan baik, maka akan menjadi lebih mudah jika pada suatu saat

terdapat Logical Error yang membutuhkan analisa ulang kode program.

xii. Gunakan Log File

Informasi proses yang dijalankan aplikasi dan tersimpan pada sebuah log file (dapat berupa file text atau table) dapat menjadi informasi yang sangat berguna untuk menganalisa bug yang mungkin terjadi. Terutama informasi yang menjelaskan apa yang terjadi sebelum, selama, dan sesudah bug terjadi. Untuk menghindari log file yang terlalu besar, yaitu dapat memisahkan log file terbagi menjadi log untuk komponen-komponen utama pada aplikasi. Jangan lupa untuk selalu mencatat waktu (timestamp) pada setiap record, dan pastikan dapat menghapus atau melakukan backup pada log file secara periodik.

xiii. Validasi

Tidak semua orang mematuhi aturan yang diterapkan pada aplikasi, karena itu programmer harus melakukan validasi untuk data yang dimasukkan oleh pengguna. Misalnya pada suatu form pendaftaran, sebaiknya melakukan validasi untuk input yang tidak boleh kosong (mandatory/required fields), melakukan pembatasan karakter, dan validasi huruf/angka yang diperlukan.

xiv. Mengenal Environment

Saat mengetikkan kode program, menjalankannya, atau melakukan debug pada program, maka terdapat environment yang berbeda-beda. Terdapat 3 environment yang umum dikenal, yaitu:

a. Design Time.

Aplikasi yang dikerjakan dilakukan pada saat design time.

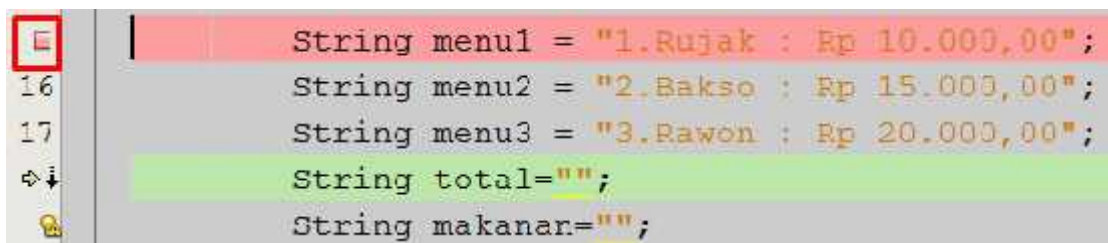
b. Run Time.

Saat menjalankan aplikasi.

c. Break Mode.

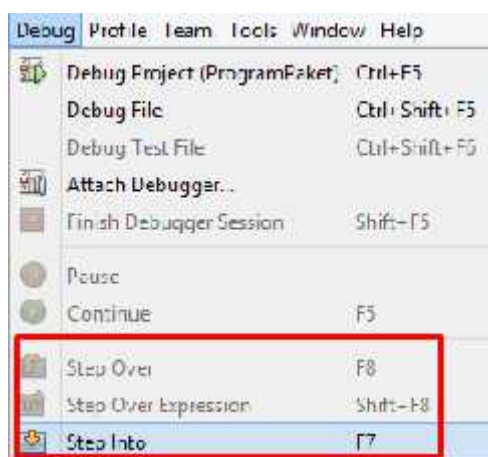
Environment saat melakukan proses debug atau melihat kode program saat program tersebut dijalankan, dapat melihat alur program dan perubahan nilai pada variabel, sehingga dapat menelusuri kesalahan yang terjadi. Break Mode terletak diantara Design Time dan Run Time.

Untuk melakukan debugging pemrograman JAVA menggunakan editor Netbeans yaitu dengan melakukan double klik pada baris kode yang dicurigai terdapat kesalahan, sehingga akan muncul tanda kotak berwarna merah, seperti pada gambar 6.4 berikut ini



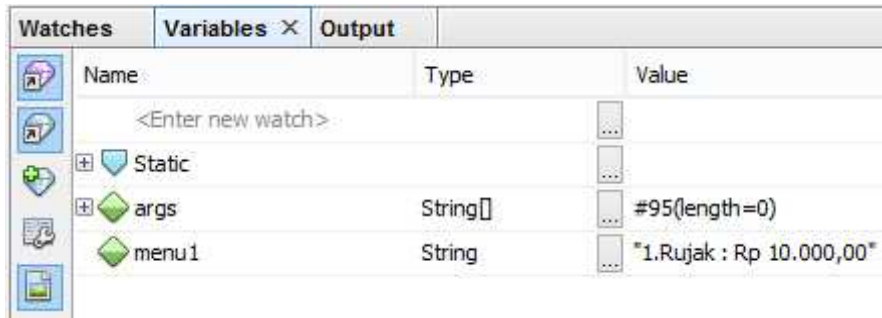
Gambar 6.4 Proses debugging

Kemudian pilih menu Debug → Step Into/Step Over atau dengan menggunakan shortcut F7 atau F8 seperti pada gambar 6.5 berikut ini



Gambar 6.5 Menu Debugging

Kemudian perhatikan tampilan output program dibagian bawah editor, dari tampilan tersebut dapat mengetahui perubahan nilai variabel, method, stack, watch dan lain-lain untuk membantu proses debugging seperti pada gambar 6.6 berikut ini



Watches	Variables X	Output	
	Name	Type	Value
	<Enter new watch>		...
+	Static		...
+	args	String[]	#95(length=0)
	menu1	String	"1.Rujak : Rp 10.000,00"

Gambar 6.6 Tampilan debugger

B. Keterampilan yang diperlukan dalam Mengkompilasi Program

1. Identifikasi jenis-jenis kesalahan pada program.
2. Mengoperasikan software IDE (Integrated Development Environment) untuk bahasa pemrograman JAVA seperti Netbeans, GEL, Eclipse dan lain-lain
3. Melakukan instalasi dan menjalankan software EXE4J.
4. Melakukan instalasi dan menjalankan software Inno Setup.

C. Sikap kerja yang diperlukan dalam Mengkompilasi Program

Harus bersikap secara:

1. Cermat dan teliti dalam menganalisis kode program.
2. Tekun dalam proses pemahaman sintaks.
3. Sesuai dengan kaidah-kaidah bahasa pemrograman.
4. Berpikir analitis serta evaluatif ketika melakukan trial and error.
5. Giat dalam mencari bugs dan memperbaikinya.

DAFTAR PUSTAKA

A. Buku Referensi

- a. Eck, David dan Smith, Introduction to Programming Using Java, Geneva New York, Seventh Edition Version 7.0 August 2014
- b. Downey and Mayfield, Think Java ,Green Tea Press, Needham USA, 2016
- c. Wicaksono, Seri Penuntun praktis Dasar-dasar Pemrograman Java 2, Penerbit PT Elex Media Komputindo Kelompok Gramedia JAKARTA, 2002

B. Referensi Lainnya

- a. Netbeans Documentaion, alamat url : <https://netbeans.org/kb/>, diakses pada tanggal 18 Februari 2018

DAFTAR ALAT DAN BAHAN

A. Daftar Peralatan/Mesin

No.	Nama Peralatan/Mesin	Keterangan
1.	Laptop, infocus, laserpointer	Untuk di ruang teori
2.	Laptop	Untuk setiap peserta
3.	Software JDK (Java Development Kit) 1.8	Untuk setiap peserta
4.	IDE Netbeans 7.0	Untuk setiap peserta
5.	Software EXE4J	Untuk setiap peserta
6.	Software Inno Setup	Untuk setiap peserta

B. Daftar Bahan

No.	Nama Bahan	Keterangan
1.	Kertas A4	Setiap peserta
2.	Alat Tulis	Setiap peserta

DAFTAR PENYUSUN

No.	Nama	Profesi
1.	Alun Sujjada, S.Kom.,M.T	<ol style="list-style-type: none">1. Dosen STT Atlas Nusantara Malang Prodi Teknik Informatika2. Asesor LSP STT Atlas Nusantara Malang3. Anggota APTIKOM (Asosiasi Perguruan Tinggi Komputer)



LAMPIRAN BUKU KERJA



BUKU KERJA

MENGIIMPLEMENTASIKAN PEMROGRAMAN BERORIENTASI OBJEK J.620100.018.02



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN R.I.
DIREKTORAT JENDERAL GURU DAN TENAGA KEPENDIDIKAN
LEMBAGA PENGEMBANGAN DAN PEMBERDAYAAN PENDIDIK DAN TENAGA
KEPENDIDIKAN BIDANG KELAUTAN, PERIKANAN,
DAN TEKNOLOGI INFORMASI DAN KOMUNIKASI
GOWA

PENJELASAN UMUM

Pengembangan Keprofesian Berkelanjutan berbasis kompetensi mengharuskan proses pelatihan memenuhi unit kompetensi secara utuh yang terdiri atas pengetahuan, keterampilan, dan sikap kerja. Dalam buku informasi Mengimplementasikan Pemrograman Berorientasi Objek telah disampaikan informasi apa saja yang diperlukan sebagai pengetahuan yang harus dimiliki untuk melakukan praktik/keterampilan terhadap unit kompetensi tersebut. Setelah memperoleh pengetahuan dilanjutkan dengan latihan-latihan guna mengaplikasikan pengetahuan yang telah dimiliki tersebut. Untuk itu diperlukan buku kerja Mengimplementasikan Pemrograman Berorientasi Objek ini sebagai media praktik dan sekaligus mengaplikasikan sikap kerja yang telah ditetapkan karena sikap kerja melekat pada keterampilan. Adapun tujuan dibuatnya buku kerja ini adalah:

1. Prinsip pelatihan berbasis kompetensi dapat dilakukan sesuai dengan konsep yang telah digariskan, yaitu pelatihan ditempuh elemen kompetensi per elemen kompetensi, baik secara teori maupun praktik;
2. Prinsip praktik dapat dilakukan setelah dinyatakan kompeten teorinya dapat dilakukan secara jelas dan tegas;
3. Pengukuran unjuk kerja dapat dilakukan dengan jelas dan pasti.

Ruang lingkup buku kerja ini meliputi pengerjaan tugas-tugas teori dan praktik per elemen kompetensi dan kriteria unjuk kerja berdasarkan SKKNI Subgolongan Pemrograman. Ruang lingkup buku kerja ini meliputi pengerjaan tugas-tugas teori dan praktik per elemen kompetensi dan kriteria unjuk kerja berdasarkan SKKNI Subgolongan Pemrograman.

DAFTAR ISI

PENJELASAN UMUM	2
DAFTAR ISI	3
BAB I TUGAS TEORI DAN PRAKTIK	4
A. Membuat Program Berorientasi Objek dengan Memanfaatkan Class ...	4
1. Tugas Teori I	4
2. Tugas Praktik I	8
B. Menggunakan Tipe Data dan Control Program Pada Metode Atau Operasi dari Suatu Class	11
1. Tugas Teori II	11
2. Tugas Praktik II	14
C. Membuat Program Dengan Konsep Berbasis Objek	17
1. Tugas Teori III	17
2. Tugas Praktik III	20
D. Membuat Program Object Oriented Dengan Interface Dan Paket	23
1. Tugas Teori IV	23
2. Tugas Praktik IV	25
E. Mengkompilasi Program.....	28
1. Tugas Teori V	28
2. Tugas Praktik V.....	30
BAB II CEK LIST TUGAS	32

BAB I

TUGAS TEORI DAN PRAKTIK

A. MEMBUAT PROGRAM BERORIENTASI OBJEK DENGAN MEMANFAATKAN CLASS

1. Tugas Teori I

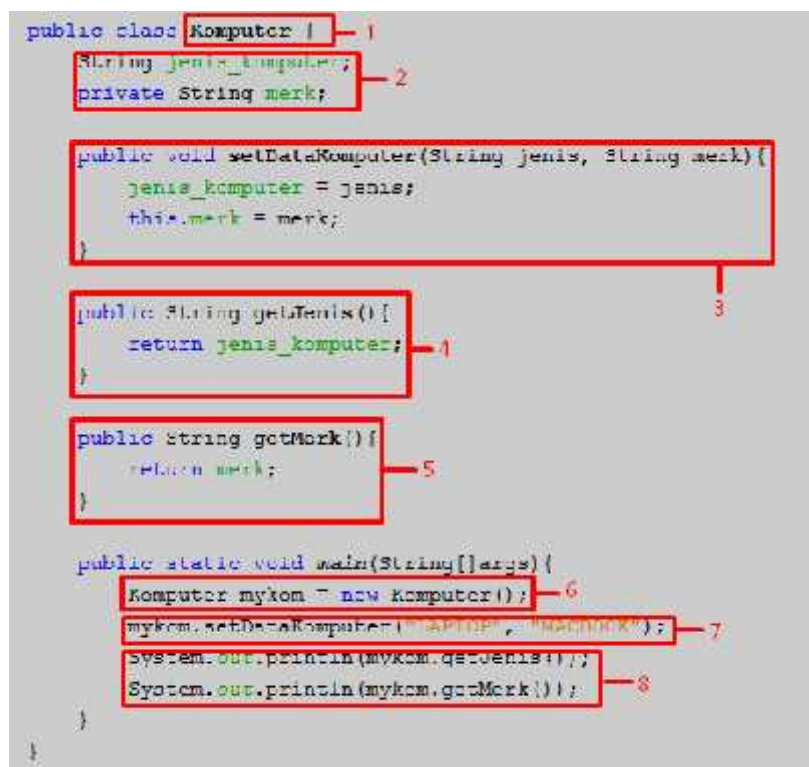
Perintah : Jawablah soal di bawah ini

Waktu Penyelesaian : 60 menit

Soal :

1. Jelaskan apa yang dimaksud dengan class dan object !
2. Sebutkan ada berapa jenis method dan jelaskan!
3. Berdasarkan gambar berikut ini, jelaskan masing-masing bagian sesuai dengan nomor yang ada!

```
public class Komputer {  
    String jenis_komputer;  
    private String merk;  
  
    public void setDataKomputer(String jenis, String merk){  
        jenis_komputer = jenis;  
        this.merk = merk;  
    }  
  
    public String getJenis(){  
        return jenis_komputer;  
    }  
  
    public String getMerk(){  
        return merk;  
    }  
  
    public static void main(String[] args){  
        Komputer mykom = new Komputer();  
        mykom.setDataKomputer("Laptop", "Mitsubisi");  
        System.out.println(mykom.getJenis());  
        System.out.println(mykom.getMerk());  
    }  
}
```



4. Tuliskan perbaikan pada kode program berikut ini, agar kode program tersebut dapat dicompile sehingga program akan berjalan dengan benar!

```
2 public class HandPhone {
3     String jenis_hp;
4     int tahun_pembuatan;
5
6     String setDataHP(String jenis_hp, int tahun_pembuatan){
7         jenis_hp = jenis_hp;
8         tahun_pembuatan = tahun_pembuatan;
9     }
10
11    String getJenisHP(){
12
13    }
14
15    String getTahunPembuatan(){
16
17    }
18
19
20    public static main void(String args){}
21        HandPhone hp = new HandPhone();
22        hp.setDataHP(jenis_hp, tahun_pembuatan);
23        hp.getJenisHP();
24        hp.getTahunPembuatan();
25    }
```

5. Sebutkan dan jelaskan ada berapa macam-macam jenis variabel !

Jawaban:

1. Class merupakan cetak biru (blue print) yang didalamnya terdapat atribut (variabel) dan behavior (method). Object adalah hasil instance dari class, sehingga setiap object yang diciptakan dari class akan memiliki semua atribut dan method yang dimiliki oleh class sesuai dengan hak akses modifier.
2. Method terbagi menjadi 2(dua) jenis yaitu getter method dan setter method. Getter digunakan untuk mengambil nilai variabel sehingga membutuhkan nilai balik (return value) dan tidak membutuhkan parameter, sedangkan setter digunakan untuk memberikan nilai pada variabel sehingga membutuhkan parameter, menggunakan keyword void dan tidak membutuhkan nilai balik (return value).
3. No. 1 adalah Nama dari class program
No. 2 adalah Member variabel / Class Member

No. 3 adalah Setter method dengan 2(dua) parameter.

No. 4 adalah Getter method dengan nilai balik berupa String

No. 5 adalah Getter method dengan nilai balik berupa String

No. 6 adalah proses pembuatan objek mykom dari class Komputer

No. 7 adalah proses pemanggilan setter method

No. 8 adalah proses pemanggilan getter method untuk dicetak pada layer monitor.

4.

```
public class HandPhone {
    String jenis_hp;
    int tahun_pembuatan;

    public void setDataHP(String jenis_hp, int tahun_pembuatan){
        this.jenis_hp = jenis_hp;
        this.tahun_pembuatan = tahun_pembuatan;
    }

    String getJenisHP(){
        return jenis_hp;
    }

    int getTahunPembuatan(){
        return getTahunPembuatan();
    }

    public static void main(String args[]){
        HandPhone hp = new HandPhone();
        hp.setDataHP("SAMSUNG", 2018);
        hp.getJenisHP();
        hp.getTahunPembuatan();
    }
}
```

5. Variabel terbagi menjadi 2 (dua) yaitu class variabel / member variabel. Class variabel adalah dimiliki oleh class, dimana deklarasinya diletakkan didalam class, sehingga variabelnya akan dapat diakses oleh semua method dan minimal didalam class yang mendefinisikannya. Sedangkan local variabel adalah dimiliki oleh method, dimana deklarasinya diletakkan didalam method, sehingga variabelnya hanya dapat diakses didalam method yang mendeklarasikannya sendiri.

Lembar Evaluasi Tugas Teori Membuat Program Berorientasi Objek Dengan Memanfaatkan Class

Semua kesalahan harus diperbaiki terlebih dahulu sebelum ditandatangani.

No.	Benar	Salah
1.		
2.		
3.		
4.		
5.		

Apakah semua pertanyaan Tugas Teori Membuat Program Berorientasi Objek Dengan Memanfaatkan Class dijawab dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

2. Tugas Praktik I

a. Elemen Kompetensi : Membuat Program Berorientasi Objek dengan Memanfaatkan Class

b. Waktu Penyelesaian : 60 menit

c. Capaian Unjuk Kerja :

Setelah menyelesaikan tugas Membuat Program Berorientasi Objek dengan Memanfaatkan class, maka peserta mampu:

- 1) Membuat program dengan menggunakan class
- 2) Membuat fungsi dan prosedur sebagai properti dari class
- 3) Mengelola hak akses dari tipe data

d. Daftar Alat/Mesin dan Bahan :

NO	NAMA BARANG	SPESIFIKASI	KETERANGAN
A.	ALAT		
1.	Komputer/Laptop	Standar processor Intel Pentim I3	Setiap peserta
2.	Kertas dan alat tulis	Standar	
B.	BAHAN		
1.	Library JAVA JDK 1.7	Versi 1.6 atau di atasnya	Setiap peserta
2.	Netbeans IDE	Versi 6.8 atau di atasnya	

e. Indikator Unjuk Kerja (IUK):

- 1) Mampu membuat program dengan menggunakan class
- 2) Mampu membuat fungsi dan prosedur sebagai property dari class
- 3) Mampu mengelola hak akses dari tipe data

f. Keselamatan dan Kesehatan Kerja

Keselamatan dan kesehatan kerja yang perlu dilakukan pada waktu melakukan praktik kerja ini adalah:

- 1) Memposisikan sikap badan tegak, mengatur jarak pandang terhadap layar monitor
- 2) Waktu menggunakan komputer, printer, dan alat lainnya mengikuti petunjuknya masing-masing yang sudah ditetapkan.

g. Standar Kinerja

- 1) Dikerjakan selesai tepat waktu, waktu yang digunakan tidak lebih dari yang ditetapkan.

2) Toleransi kesalahan 5% dari hasil yang harus dicapai, tetapi bukan pada kesalahan kegiatan kritis.

h. Tugas

Abstraksi Tugas Praktik I

Terdapat sebuah class dengan nama "Baju", rancanglah bentuk diagram class untuk memodelkan kedalam pemrograman berorientasi objek. Kemudian berdasarkan rancangan tersebut buatlah implementasi programnya menggunakan Netbeans!

i. Instruksi Kerja

Setelah membaca abstraksi nomor h selanjutnya ikuti instruksi kerja sebagai berikut:

- 1) Siapkan peralatan tulis dan kertas.
- 2) Tentukan atribut(variabel) yang akan digunakan.
- 3) Tentukan method yang akan digunakan.
- 4) Gambar dalam bentuk diagram class.
- 5) Jalankan Netbeans editor
- 6) Implementasikan diagram class kedalam bahasa pemrograman JAVA.

j. Daftar Cek Unjuk Kerja Tugas I

NO	DAFTAR TUGAS/INSTRUKSI	POIN YANG DICEK	PENCAPAIAN		PENILAIAN	
			YA	TIDAK	K	BK
1.	Siapkan peralatan tulis dan kertas.	Pemahaman fungsi alat tulis dan kertas				
2.	Tentukan atribut(variabel) yang akan digunakan.	Kesesuaian dengan permasalahan, kesesuaian penulisan sintaks				
3.	Tentukan method yang akan digunakan.	Kesesuaian dengan permasalahan, kesesuaian penulisan sintaks				
4.	Gambar dalam bentuk diagram class.	Penempatan class, atribut dan method				
5.	Jalankan Netbeans editor	Cara menjalankan program				
6.	Implementasikan diagram class kedalam bahasa pemrograman JAVA.	Pembuatan class, method, atribut dan pemanggilan method efisien dan sesuai permasalahan				

Apakah semua instruksi kerja tugas praktik Membuat Program Berorientasi Objek dengan Memanfaatkan Class dilaksanakan dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

B. MENGGUNAKAN TIPE DATA DAN CONTROL PROGRAM PADA METODE ATAU OPERASI DARI SUATU KELAS

1. Tugas Teori II

Perintah : Jawablah soal dibawah ini

Waktu Penyelesaian : 60 menit

Soal :

1. Sebutkan dan jelaskan beberapa macam pembagian tipe data!
2. Berikanlah contoh penggunaan masing-masing tipe data dan variabel!
3. Sebutkan control program yang digunakan dalam menyelesaikan permasalahan program!
4. Terdapat sebuah program yang mampu melakukan pengecekan sebuah bilangan termasuk dalam kategori bilangan genap atau ganjil, tuliskanlah sintaks progam yang digunakan!
5. Andi disuruh ibunya untuk belanja ke pasar selama seminggu karena mendapat pesanan masakan. Hari ke 1 total belanjanya sebesar Rp 200.000, hari ke 2 Rp 175.000, hari ke 3 Rp 215.000, hari ke 4 Rp 200.000, hari ke 5 Rp 300.000, hari ke 6 Rp Rp 275.000 dan hari ke 7 Rp 225.000, tuliskanlah sintaks program untuk menghitung semua total keseluruhan biaya yang dikeluarkan untuk belanja!

Jawaban:

1. Tipe data terbagi menjadi 2 yaitu, tipe data primitive dan tipe data reference. Tipe data primitive terdiri dari (boolean, byte, short, int, long, char, float dan double), sedangkan tipe data reference terdiri dari (tipe class, tipe array, tipe interface).
2.

```
String nama = "Fahmi";  
boolean gender = true;  
byte jml_absensi = 5;  
short nilai_UAS = 80;  
float total_belanja = 900000.00  
int harga = 70000;
```
3. Control program terdiri dari 2 (dua) jenis yaitu seleksi/kondisi

percabangan dan perulangan. Kondisi percabangan menggunakan sintaks if dan switch, sedangkan perulangan menggunakan sintaks for, do while dan while

4.

```
public class GanjilGenap {  
  
    public static void main(String [] args){  
        int bilangan = 13;  
  
        /*Menggunkan if */  
        if(bilangan % 2 == 0){  
            System.out.println(bilangan + " adalah bilangan genap");  
        }  
        else{  
            System.out.println(bilangan + " adalah bilangan ganjil");  
        }  
  
        /*Menggunkan switch */  
        int hasilmodulus = bilangan % 2;  
        switch(hasilmodulus){  
            case 0:  
                System.out.println(bilangan + " adalah bilangan genap");  
                break;  
            case 1:  
                System.out.println(bilangan + " adalah bilangan ganjil");  
                break;  
        }  
    }  
}
```

5.

```
public class HitungBelanja {  
    public static void main(String[] args){  
        int belanja[] = {200000, 175000, 215000, 200000, 300000, 275000, 225000};  
        int total = 0;  
        for(int i=0; i<7; i++){  
            total =total + belanja[i];  
        }  
        System.out.println("Total Belanja adalah sebesar : " + total);  
    }  
}
```

Lembar Evaluasi Tugas Teori Menggunakan Tipe Data Dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas

Semua kesalahan harus diperbaiki terlebih dahulu sebelum ditandatangani.

No.	Benar	Salah
1.		
2.		
3.		
4.		
5.		

Apakah semua pertanyaan Tugas Teori Menggunakan Tipe Data Dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas dijawab dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

2. Tugas Praktik II

a. Elemen Kompetensi : Menggunakan Tipe Data Dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas

b. Waktu Penyelesaian : 60 menit

c. Capaian Unjuk Kerja :

Setelah menyelesaikan tugas Menggunakan Tipe Data dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas, maka peserta mampu:

- 1) Mengidentifikasi tipe data.
- 2) Menguasai sintaks program sesuai bahasa pemrogramannya.
- 3) Menguasai control program

d. Daftar Alat/Mesin dan Bahan :

NO	NAMA BARANG	SPESIFIKASI	KETERANGAN
A.	ALAT		
1.	Komputer/Laptop	Standar processor Intel Pentim I3	Setiap peserta
2.	Kertas dan alat tulis	Standar	
B.	BAHAN		
1.	Library JAVA JDK 1.7	Versi 1.6 atau di atasnya	Setiap peserta
2.	Netbeans IDE	Versi 6.8 atau di atasnya	

e. Indikator Unjuk Kerja (IUK):

- i. Mampu mengidentifikasi tipe data.
- ii. Mampu menguasai sintaks program sesuai bahasa pemrogramannya
- iii. Mampu menguasai control program

f. Keselamatan dan Kesehatan Kerja

Keselamatan dan kesehatan kerja yang perlu dilakukan pada waktu melakukan praktik kerja ini adalah:

- i. Memposisikan sikap badan tegak, mengatur jarak pandang terhadap layar monitor
- ii. Waktu menggunakan komputer, printer, dan alat lainnya mengikuti petunjuknya masing-masing yang sudah ditetapkan.

g. Standar Kinerja

- i. Dikerjakan selesai tepat waktu, waktu yang digunakan tidak lebih dari yang ditetapkan.

ii. Toleransi kesalahan 5% dari hasil yang harus dicapai, tetapi bukan pada kesalahan kegiatan kritis.

h. Tugas

Abstraksi Tugas Praktik I

Buatlah program menggunakan bahasa JAVA yang berfungsi untuk melakukan pengecekan sebuah bilangan dengan batas awal dan akhir termasuk bilangan prima atau bukan prima menggunakan method!

Contoh Input :

Isikan bilangan awal : 1

Isikan bilangan akhir : 10

Contoh output :

1 adalah bilangan bukan prima
2 adalah bilangan prima
3 adalah bilangan prima
4 adalah bilangan bukan prima
5 adalah bilangan prima
6 adalah bilangan bukan prima
7 adalah bilangan prima
8 adalah bilangan bukan prima
9 adalah bilangan bukan prima
10 adalah bilangan bukan prima

i. Instruksi Kerja

Setelah membaca abstraksi nomor h selanjutnya ikuti instruksi kerja sebagai berikut:

- 1) Siapkan peralatan komputer atau laptop masing-masing
- 2) Jalankan software editor Netbeans
- 3) Buatlah sebuah class dengan nama yang sesuai permasalahan
- 4) Buatlah method dengan nama yang sesuai permasalahan
- 5) Implementasikan program menggunakan kondisi percabangan dan perulangan

j. Daftar Cek Unjuk Kerja Tugas I

NO	DAFTAR TUGAS/INSTRUKSI	POIN YANG DICEK	PENCAPAIAN		PENILAIAN	
			YA	TIDAK	K	BK
1.	Siapkan peralatan komputer atau laptop masing-masing	Kesesuaian dengan langkah-langkah keselamatan kerja				
2.	Jalankan software editor Netbeans	Kemampuan mengoperasikan				
3.	Buatlah sebuah class	Ketepatan pembuatan dan nama class yang sesuai permasalahan				
4.	Buatlah method	Ketepatan pembuatan dan nama method yang sesuai permasalahan				
5.	Implementasikan program menggunakan kondisi percabangan dan perulangan	Ketepatan program sesuai permasalahan dan efisiensi				

Apakah semua instruksi kerja tugas praktik Menggunakan Tipe Data Dan Control Program Pada Metode Atau Operasi Dari Suatu Kelas dilaksanakan dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

C. MEMBUAT PROGRAM DENGAN KONSEP BERBASIS OBJEK

1. Tugas Teori III

Perintah : Jawablah soal dibawah ini

Waktu Penyelesaian : 60 menit

Soal :

1. Jelaskan apa yang dimaksud dengan super class dan sub class !
2. Buatlah analogi tentang konsep pewarisan dalam dunia nyata !
3. Apakah yang dimaksud dengan constructor method ? Jelaskan kegunaanya!
4. Jelaskan konsep tentang polymorphism dan overriding!
5. Tuliskan kode program sederhana menggunakan sintaks pemrograman JAVA yang mengimplementasikan polymorphism dan overriding !

Jawaban:

1. Super class adalah class yang mempunyai sifat variabel dan method secara umum agar sifat tersebut dapat diwariskan kepada sub class lain, sehingga struktur file dapat terorganisasi dengan baik dan mempermudah pembuatan program. Sub class adalah class yang mewarisi variabel dan method dari super class nya, sehingga kode program akan lebih efisien karena tidak ada pemanggilan method yang berulang-ulang. Sub class dapat memperluas (extends) method nya sendiri.
2. Super class :
Sekolah
Sub class :
 - SekolahFormal
Sub class :
 - SekolahSD
 - SekolahSMP
 - SekolahSMK
 - SekolahNonFormal
3. Constructor method adalah method yang namanya sama dengan

nama class, tidak mempunyai nilai balik dan tidak menggunakan keyword void. Method ini hanya dipanggil sekali ketika penciptaan sebuah objek. Fungsinya adalah untuk memberikan nilai awal atau inisialisasi terhadap variabel.

4. Polymorphism adalah sebuah konsep yang berkaitan erat dengan pewarisan, yaitu sebuah method dengan nama yang sama, akan tetapi mempunyai banyak bentuk implementasi programnya. Proses implementasinya menggunakan overriding yang dilakukan pada sub class.

```
5. class Ayah{
    String jenis_rambut;
    String warna_rambut;

    public Ayah(String jenis_rambut, String warna_rambut){
        this.jenis_rambut = jenis_rambut;
        this.warna_rambut = warna_rambut;
    }

    public String getWarnaRambut(){
        return warna_rambut;
    }
}

public class Anak extends Ayah{

    public Anak(){
        super("IKAL", "HITAM");
    }

    public String getWarnaRambut(){
        return "KUNING";
    }

    public static void main(String []args){
        Anak anak1 = new Anak();
        anak1.getWarnaRambut();
    }
}
```

Lembar Evaluasi Tugas Teori Membuat Program Dengan Konsep Berbasis Objek

Semua kesalahan harus diperbaiki terlebih dahulu sebelum ditandatangani.

No.	Benar	Salah
1.		
2.		
3.		
4.		
5.		

Apakah semua pertanyaan Tugas Teori Membuat Program Dengan Konsep Berbasis Objek dijawab dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

2. Tugas Praktek III

a. Elemen Kompetensi : Membuat Program Dengan Konsep Berbasis Objek

b. Waktu Penyelesaian : 60 menit

c. Capaian Unjuk Kerja :

Setelah menyelesaikan tugas Membuat Program Dengan Konsep Berbasis Objek, maka peserta mampu

1) Menerapkan inheritance pada class

2) Menerapkan polymorphism pada class

3) Menerapkan overloading pada class

d. Daftar Alat/Mesin dan Bahan :

NO	NAMA BARANG	SPESIFIKASI	KETERANGAN
A.	ALAT		
1.	Komputer/Laptop	Standar processor Intel Pentim I3	Setiap peserta
2.	Kertas dan alat tulis	Standar	
B.	BAHAN		
1.	Library JAVA JDK 1.7	Versi 1.6 atau di atasnya	Setiap peserta
2.	Netbeans IDE	Versi 6.8 atau di atasnya	

e. Indikator Unjuk Kerja (IUK):

1) Mampu menerapkan inheritance pada class

2) Mampu menerapkan polymorphism pada class

3) Mampu menerapkan overloading pada class

f. Keselamatan dan Kesehatan Kerja

Keselamatan dan kesehatan kerja yang perlu dilakukan pada waktu melakukan praktik kerja ini adalah:

1) Memosisikan sikap badan tegak, mengatur jarak pandang terhadap layar monitor

2) Waktu menggunakan komputer, printer, dan alat lainnya mengikuti petunjuknya masing-masing yang sudah ditetapkan.

g. Standar Kinerja

1) Dikerjakan selesai tepat waktu, waktu yang digunakan tidak lebih dari yang ditetapkan.

2) Toleransi kesalahan 5% dari hasil yang harus dicapai, tetapi bukan pada kesalahan kegiatan kritis.

h. Tugas

Abstraksi Tugas Praktik I

Buatlah program dengan mengimplementasikan konsep polymorphism tentang segala kegiatan yang berhubungan dengan sekolah anda!

i. Instruksi Kerja

- 1) Tentukan permasalahan yang akan diselesaikan
- 2) Buatlah super class.
- 3) Buatlah variabel dan method yang akan digunakan pada class.
- 4) Buatlah sub class yang mewarisi sifat dari sub class.
- 5) Buatlah variabel, method yang mempunyai kesamaan nama dengan method pada super class, tapi implementasi programnya yang berbeda.
- 6) Buatlah fungsi main pada sub class yang didalamnya terdapat pembuatan objek

j. Daftar Cek Unjuk Kerja Tugas I

NO	DAFTAR TUGAS/INSTRUKSI	POIN YANG DICEK	PENCAPAIAN		PENILAIAN	
			YA	TIDAK	K	BK
1.	Tentukan permasalahan yang akan diselesaikan	Ketepatan pengambilan contoh permasalahan				
2.	Buatlah super class	Kesesuaian dengan sintaks				
3.	Buatlah variabel dan method yang akan digunakan pada class.	Jenis variabel dan Penggunaan method				
4.	Buatlah sub class yang mewarisi sifat dari sub class.	Keywords untuk melakukan pewarisan				
5.	Buatlah variabel, method yang mempunyai kesamaan nama dengan method pada super class, tapi implementasi programnya yang berbeda.	Overriding				
6.	Buatlah fungsi main pada sub class yang didalamnya terdapat pembuatan objek	Pembuatan objek dalam method main				

Apakah semua instruksi kerja tugas praktik Membuat Program Dengan Konsep Berbasis Objek dilaksanakan dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

D. MEMBUAT PROGRAM OBJECT ORIENTED DENGAN INTERFACE DAN PAKET

1. Tugas Teori IV

Perintah : Jawablah soal dibawah ini

Waktu Penyelesain : 60 menit

Soal :

1. Jelaskan definisi interface
2. Sebutkan aturan-aturan apa saja yang digunakan dalam membuat sebuah interface !
3. Jelaskan fungsi dari pembuatan paket program!
4. Sebutkan langkah-langkah dalam pembuatan paket program untuk aplikasi berbasis JAVA!

Jawaban:

1. Interface adalah sebuah kumpulan deklarasi variabel dengan nilai awal dan method tanpa implementasi program. Interface digunakan untuk mengatasi konsep multiple inheritance pada bahasa pemrograman C++.
2.
 - a. Deklarasi method tanpa implementasi program.
 - b. Deklarasi variabel dengan nilai awal.
 - c. Class yang melakukan implementasi dari interface, harus mengimplementasikan semua method yang dimiliki oleh interface
3. Pembuatan paket program digunakan untuk mendistribusikan program yang telah siap pakai agar dapat dijalankan di komputer/laptop lain tanpa membuka kode program. Dengan melakukan paket program, maka akan terbentuk file installer yang berupa kumpulan kode program beserta library yang digunakan dipaket menjadi satu file kesatuan yang utuh.
4.
 - a. Mengkompilasi program menjadi file JAR
 - b. Melakukan konversi file JAR menjadi file executable.
 - c. Membuat paket installer dengan program.

Lembar Evaluasi Tugas Teori Membuat Program Object Oriented Dengan Interface Dan Paket

Semua kesalahan harus diperbaiki terlebih dahulu sebelum ditandatangani.

No.	Benar	Salah
1.		
2.		
3.		
4.		
5.		

Apakah semua pertanyaan Tugas Teori Membuat Program Object Oriented Dengan Interface Dan Paket dijawab dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

2. Tugas Praktek IV

a. Elemen Kompetensi : Membuat Program Object Oriented Dengan Interface Dan Paket.

b. Waktu Penyelesaian : 60 menit

c. Capaian Unjuk Kerja :

Setelah menyelesaikan tugas Membuat Program Object Oriented Dengan Interface Dan Paket, maka peserta mampu:

- 1) Membuat program interface class
- 2) Membuat program dengan paket

d. Daftar Alat/Mesin dan Bahan:

NO	NAMA BARANG	SPESIFIKASI	KETERANGAN
A.	ALAT		
1.	Komputer/Laptop	Standar processor Intel Pentim I3	Setiap peserta
2.	Kertas dan alat tulis	Standar	
B.	BAHAN		
1.	Library JAVA JDK 1.7	Versi 1.6 atau di atasnya	Setiap peserta
2.	Netbeans IDE	Versi 6.8 atau di atasnya	

e. Indikator Unjuk Kerja (IUK):

- i. Mampu membuat program interface class.
- ii. Mampu membuat paket dengan program.

f. Keselamatan dan Kesehatan Kerja

Keselamatan dan kesehatan kerja yang perlu dilakukan pada waktu melakukan praktik kerja ini adalah:

- i. Memposisikan sikap badan tegak, mengatur jarak pandang terhadap layar monitor
- ii. Waktu menggunakan komputer, printer, dan alat lainnya mengikuti petunjuknya masing-masing yang sudah ditetapkan.

g. Standar Kinerja

- i. Dikerjakan selesai tepat waktu, waktu yang digunakan tidak lebih dari yang ditetapkan.
- ii. Toleransi kesalahan 5% dari hasil yang harus dicapai, tetapi bukan pada kesalahan kegiatan kritis.

h. Tugas

Abstraksi Tugas I

Buatlah program yang memiliki interface Penerbang yang didalamnya terdapat 3 method dan class Superman yang mengimplementasikannya!

i. Instruksi Kerja

Setelah membaca abstraksi nomor h selanjutnya ikuti instruksi kerja sebagai berikut:

- 1) Buatlah interface Penerbang
- 2) Buatlah deklarasi variabel dan 3 method pada interface
- 3) Buatlah class Superman
- 4) Implementasikan interface Penerbang.

j. Daftar Cek Unjuk Kerja Tugas I

NO	DAFTAR TUGAS/INSTRUKSI	POIN YANG DICEK	PENCAPAIAN		PENILAIAN	
			YA	TIDAK	K	BK
1.	Buatlah interface Penerbang	Cara pembuatan interface Penerbang dan kesesuaiannya.				
2.	Buatlah deklarasi variabel dan 3 method pada interface	Kesesuaian dengan permasalahan				
3.	Buatlah class Superman	Ketepatan pembuatan class Superman				
4.	Implementasikan interface Penerbang	Ketepatan implementasi dari interface				

Apakah semua instruksi kerja tugas praktik Membuat Program Object Oriented Dengan Interface Dan Paket dilaksanakan dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

E. MENGGOMPILASI PROGRAM

1. Tugas Teori V

Perintah : Jawablah soal dibawah ini

Waktu Penyelesaian : 30 menit

Soal :

1. Sebutkan dan jelaskan jenis-jenis kesalahan program!
2. Sebutkan cara-cara untuk meminimalisir kesalahan!
3. Tuliskan sintaks penggunaan blok kode try catch dan finally!

Jawaban:

1. a. Syntax Error

Kesalahan yang paling sering ditemukan pada saat membuat program adalah kesalahan sintaks atau Syntax Error, dimana perintah atau statement yang diketikkan menyalahi aturan pengkodean yang dimiliki oleh bahasa pemrograman yang digunakan

b. Logical Error

Error yang disebabkan karena kesalahan logika saat membuat program

c. Run-time Error

Error yang terjadi ketika kode program melakukan sesuatu yang tidak dimungkinkan

2. Selalu mendeklarasikan variabel, menggunakan variabel lokal, mengenali jenis bug,

3. Try{

 Pernyataan program yang akan dieksekusi

}

Catch(<Tipe eksepsi> <nama variabel>){

 Pesan error yang akan ditampilkan

}

Finally{

 Program yang akan selalu dieksekusi

}

Lembar Evaluasi Tugas Teori Mengkompilasi Program

Semua kesalahan harus diperbaiki terlebih dahulu sebelum ditandatangani.

No.	Benar	Salah
1.		
2.		
3.		
4.		
5.		

Apakah semua pertanyaan Tugas Teori Mengkompilasi Program dijawab dengan benar dengan waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:

2. Tugas Praktek V

a. Elemen Kompetensi : Mengkompilasi Program

b. Waktu Penyelesaian : 60 menit

c. Capaian Unjuk Kerja :

Setelah menyelesaikan tugas Mengkompilasi Program, maka peserta mampu

1) Mengoreksi Kesalahan

2) Menghasilkan program yang bebas dari kesalahan syntax

d. Daftar Alat/Mesin dan Bahan

NO	NAMA BARANG	SPESIFIKASI	KETERANGAN
A.	ALAT		
1.	Komputer/Laptop	Standar processor Intel Pentim I3	Setiap peserta
2.	Kertas dan alat tulis	Standar	
B.	BAHAN		
1.	Library JAVA JDK 1.7	Versi 1.6 atau di atasnya	Setiap peserta
2.	Netbeans IDE	Versi 6.8 atau di atasnya	

e. Indikator Unjuk Kerja (IUK)

1) Mampu mengoreksi kesalahan

2) Mampu menghasilkan program yang bebas dari kesalahan sintaks

f. Keselamatan dan Kesehatan Kerja

Keselamatan dan kesehatan kerja yang perlu dilakukan pada waktu melakukan praktik kerja ini adalah:

1) Memosisikan sikap badan tegak, mengatur jarak pandang terhadap layar monitor

2) Waktu menggunakan komputer, printer, dan alat lainnya mengikuti petunjuknya masing-masing yang sudah ditetapkan.

g. Standar Kinerja

1) Dikerjakan selesai tepat waktu, waktu yang digunakan tidak lebih dari yang ditetapkan.

2) Toleransi kesalahan 5% dari hasil yang harus dicapai, tetapi bukan pada kesalahan kegiatan kritis.

h. Tugas

Abstraksi Tugas Praktik I

Lakukan debugging pada kode program yang telah anda buat pada bab sebelumnya, dan perbaiki program tersebut jika terjadi kesalahan

i. Instruksi Kerja

- 1) Buka project program bab sebelumnya.
- 2) Koreksi kesalahan
- 3) Lakukan debugging

j. Daftar Cek Unjuk Kerja Tugas I

NO	DAFTAR TUGAS/INSTRUKSI	POIN YANG DICEK	PENCAPAIAN		PENILAIAN	
			YA	TIDAK	K	BK
1.	Buka project program bab sebelumnya	Cara membuka kembali project				
2.	Koreksi kesalahan	Cara mengkoreksi kesalahan				
3.	Lakukan debugging	Cara melakukan debugging				

BAB II
 CEK LIS TUGAS

NO	TUGAS UNJUK KERJA	PENILAIAN		TANGGAL
		K	BK	
1.	Membuat program berorientasi objek dengan memanfaatkan class			
2.	Menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas			
3.	Membuat program dengan konsep berbasis objek			
4.	Membuat program object oriented dengan interface dan paket			
5.	Mengkompilasi Program			

Apakah semua tugas unjuk kerja Mengimplementasikan Pemrograman Berorientasi Objek telah dilaksanakan dengan benar dan dalam waktu yang telah ditentukan?

YA

TIDAK

	NAMA	TANDA TANGAN
PESERTA
PENILAI

Catatan Penilai:



LAMPIRAN BUKU PENILAIAN



BUKU PENILAIAN

MENGIIMPLEMENTASIKAN PEMROGRAMAN BERORIENTASI OBJEK J.620100.018.02



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN R.I.
DIREKTORAT JENDERAL GURU DAN TENAGA KEPENDIDIKAN
LEMBAGA PENGEMBANGAN DAN PEMBERDAYAAN PENDIDIK DAN TENAGA KEPENDIDIKAN
BIDANG KELAUTAN, PERIKANAN,
DAN TEKNOLOGI INFORMASI DAN KOMUNIKASI
GOWA

PENJELASAN UMUM

Buku penilaian untuk unit kompetensi Mengimplementasikan Pemrograman Berorientasi Objek dibuat sebagai konsekuensi logis dalam pelatihan berbasis kompetensi yang telah menempuh tahapan penerimaan pengetahuan, keterampilan, dan sikap kerja melalui buku informasi dan buku kerja. Setelah latihan-latihan (exercise) dilakukan berdasarkan buku kerja maka untuk mengetahui sejauh mana kompetensi yang dimilikinya perlu dilakukan uji komprehensif secara utuh per unit kompetensi dan materi uji komprehensif itu ada dalam buku penilaian ini.

Adapun tujuan dibuatnya buku penilaian ini, yaitu untuk menguji kompetensi peserta pelatihan setelah selesai menempuh buku informasi dan buku kerja secara komprehensif dan berdasarkan hasil uji inilah peserta akan dinyatakan kompeten atau belum kompeten terhadap unit kompetensi Mengimplementasikan Pemrograman Berorientasi Objek. Metoda Penilaian yang dilakukan meliputi penilaian dengan opsi sebagai berikut:

1. Metoda Penilaian Pengetahuan

- a. Tes Tertulis

Untuk menilai pengetahuan yang telah disampaikan selama proses pelatihan terlebih dahulu dilakukan tes tertulis melalui pemberian materi tes dalam bentuk tertulis yang dijawab secara tertulis juga. Untuk menilai pengetahuan dalam proses pelatihan materi tes disampaikan lebih dominan dalam bentuk obyektif tes, dalam hal ini jawaban singkat, menjodohkan, benar-salah, dan pilihan ganda. Tes essay bisa diberikan selama tes essay tersebut tes essay tertutup, tidak essay terbuka, hal ini dimaksudkan untuk mengurangi faktor subyektif penilai.

- b. Tes Wawancara

Tes wawancara dilakukan untuk menggali atau memastikan hasil tes tertulis sejauh itu diperlukan. Tes wawancara ini dilakukan secara perseorangan antara penilai dengan peserta uji/peserta pelatihan. Penilai sebaiknya lebih dari satu orang.

2. Metoda Penilaian Keterampilan

a. Tes Simulasi

Tes simulasi ini digunakan untuk menilai keterampilan dengan menggunakan media bukan yang sebenarnya, misalnya menggunakan tempat kerja tiruan (bukan tempat kerja yang sebenarnya), obyek pekerjaan disediakan atau hasil rekayasa sendiri, bukan obyek kerja yang sebenarnya.

b. Aktivitas Praktik

Penilaian dilakukan secara sebenarnya, di tempat kerja sebenarnya dengan menggunakan obyek kerja sebenarnya.

3. Metoda Penilaian Sikap Kerja

a. Observasi

Untuk melakukan penilaian sikap kerja digunakan metoda observasi terstruktur, artinya pengamatan yang dilakukan menggunakan lembar penilaian yang sudah disiapkan sehingga pengamatan yang dilakukan mengikuti petunjuk penilaian yang dituntut oleh lembar penilaian tersebut. Pengamatan dilakukan pada waktu peserta uji/peserta pelatihan melakukan keterampilan kompetensi yang dinilai karena sikap kerja melekat pada keterampilan tersebut.

DAFTAR ISI

PENJELASAN UMUM	2
DAFTAR ISI	4
BAB I PENILAIAN TEORI	5
A. Lembar Penilaian Teori	5
B. Ceklis Penilaian Teori.....	11
BAB II PENILAIAN PRAKTIK	12
A. Lembar Penilaian Praktik	12
B. Ceklis Aktivitas Praktik	13
BAB III CEKLIS PENILAIAN SIKAP KERJA	14
A. Penilaian Sikap Kerja	14
LAMPIRAN	15
Lampiran 1. Kunci Jawaban	16

BAB I PENILAIAN TEORI

A. Lembar Penilaian Teori

Unit Kompetensi : Mengimplementasikan Pemrograman
Berorientasi Objek

Diklat :

Waktu : 60 menit

PETUNJUK UMUM

1. Jawablah materi tes ini pada lembar jawaban/kertas yang sudah disediakan.
2. Modul terkait dengan unit kompetensi agar disimpan.
3. Bacalah materi tes secara cermat dan teliti.

Isian

Jawablah pertanyaan dibawah ini dengan singkat dan tepat!

1. Disebut dengan istilah apakah Instance dari sebuah class ?
2. Sebutkan pembagian method secara garis besar !
3. Dikenal dengan istilah apakah method yang digunakan untuk memberikan nilai variabel melalui parameter dikenal?
4. Dikenal dengan istilah apakah method yang digunakan untuk mengambil nilai variabel?
5. Keyword apakah yang digunakan untuk mengakses varibel class, jika mempunyai kesamaan nama dengan variabel parameter?
6. Tipe data apakah yang digunakan untuk menyimpan nilai:
 - a. True atau false
 - b. Bilangan bulat
 - c. Bilangan pecahan

7. Perhatikan gambar berikut ini

```
public class DayOfWeek {
    static int hari =4;
    static int waktu = 2;
    public static void main(String[] args){
        if(hari--1){
            if(waktu--1){
                System.out.print("SEKIN PAGI" );
            }
            else if(waktu==2){
                System.out.print("SENIN SIANG" );
            }
            else{
                System.out.print("SENIN MALAM" );
            }
        }
        else{
            if(waktu--1){
                System.out.print("SELASA PAGI" );
            }
            else if(waktu==2){
                System.out.print("SELASA SIANG" );
            }
            else{
                System.out.print("SELASA MALAM" );
            }
        }
    }
}
```

Berdasarkan listing program tersebut, output apakah yang akan tampil pada layar monitor?

8. Berdasarkan listing program berikut ini, ditambahkan kode apakah agar program dapat berjalan dan menampilkan negara produsen handphone?

```
public class MarkHP {

    static String merk="SAMSUNG";
    public static void main(String []args){

        switch(....){
            case "SAMSUNG" :
                System.out.println("Negara Korea");
                break;
            case "APPLE" :
                System.out.println("Negara Amerika");
                break;
            case "XIAOMI" :
                System.out.println("Negara China");
                break;
        }
    }
}
```

9. Output apakah yang akan ditampilkan pada listing program berikut ini

```
public class Perulangan {  
  
    public static void main(String []args){  
        int a = 1;  
  
        for (int i = 1; i<8;i++){  
            a = a * i;  
            System.out.println(a);  
        }  
    }  
}
```

10 Perhatikan 2 listing program berikut ini

```
public class Guru {  
  
    protected String nama_guru;  
  
    public Guru(String nama_guru) {  
        this.nama_guru = nama_guru;  
    }  
  
    public void infoGuru(){  
        System.out.println("Saya adalah seorang Guru");  
    }  
}
```

```
/**  
 *  
 * Author :ovajowano  
 */  
public class Mapel extends Guru {  
  
    protected String nama_mapel;  
  
    public Mapel(String nama_mapel, String nama_dosen){  
        super(nama_dosen);  
        this.nama_mapel = nama_mapel;  
    }  
  
    public void infoGuru(){  
        System.out.println("Nama Guru : " + super.nama_guru);  
        System.out.println("Mengejar Mata Pelajaran : " + this.nama_mapel);  
    }  
}
```

Disebut dengan istilah apakah method infoGuru() pada class Guru dan class Mapel?

Benar-Salah

Nyatakan pernyataan di bawah ini benar atau salah dengan cara menulis huruf B jika Benar dan huruf S jika Salah.

- | | | |
|---|---|--|
| B | S | 1. Super class dikenal juga dengan istilah child class |
| B | S | 2. Secara hierarki super class berada diatas sub class |
| B | S | 3. Atribut dapat diartikan sebagai variabel dan method dapat diartikan sebagai objek. |
| B | S | 4. Getter method membutuhkan return value |
| B | S | 5. <code>Buku bukuku = new Buku("Pelajaran");</code> adalah contoh kode program yang digunakan untuk memanggil sebuah objek |
| B | S | 6. Modifier yang digunakan agar sebuah method hanya dapat digunakan pada class nya sendiri dan sub class turunannya adalah private |
| B | S | 7. Modifier non access final adalah berfungsi method atau variabel tidak dapat diubah nilainya. |
| B | S | 8. Proses pewarisan atau inheritance adalah dengan menggunakan kata kunci implements |
| B | S | 9. Jika terdapat method dimana nama yang digunakan sama dengan nama class, maka method dikenal sebagai getter method |
| B | S | 10. Pembuatan kode program JAVA menjadi file JAR disebut dengan istilah pembuatan file installer atau distribution. |

Pilihan Ganda

Jawablah pertanyaan/pernyataan di bawah ini dengan cara memilih pilihan jawaban yang tepat dan menuliskan huruf A/B/C/D yang sesuai dengan pilihan tersebut.

- Manakah diantara pernyataan berikut ini yang menyatakan aturan pembuatan constructor method?
 - Tidak mempunyai return value
 - Nama method sama dengan nama class
 - Tidak diperbolehkan menggunakan void
 - Semua benar
- Sebuah class yang hanya memperbolehkan penulisan deklarasasi method tanpa implementasi program adalah
 - Interface
 - Constructor
 - Overloading
 - Object
- Terdapat deklarasi variabel String nilai =20; maka variabel tersebut mempunyai hak akses
 - Public
 - Private
 - Protected
 - Default

4. Berapakah hasil output dari potongan kode program berikut ini

```
public class Perulangan {  
  
    public static void main(String []args){  
        int nilai_awal = 1;  
  
        for (int i = 1; i<10;i++){  
            nilai_awal = nilai_awal + 10;  
        }  
        System.out.println(nilai_awal);  
    }  
}
```

- a. 91
b. 100
c. 101
d. 90
5. Berapakah hasil output dari potongan kode program berikut ini

```
public class Perulangan {  
  
    public static void main(String []args){  
        int nilai =100;  
        do{  
            System.out.println(nilai);  
            nilai++;  
        }while(nilai <100);  
    }  
}
```

- a. 0
b. 100
c. Null
d. Error
6. Berapakah hasil output dari potongan kode program berikut ini

```
public class Perulangan {  
  
    public static void main(String []args){  
        int nilai =0;  
        while(nilai < 100){  
            System.out.println(nilai);  
        }  
    }  
}
```

- a. 100
b. Error program
c. Null
d. 0

7. Berikut ini adalah jenis-jenis kesalahan yang terdapat pada program, kecuali
 - a. Sintaks Error
 - b. Logic Error
 - c. Class Error
 - d. Run-time Error
8. Manakah pernyataan berikut ini yang merupakan cara tepat untuk meminimalisir kesalahan pada pembuatan program?
 - a. Memberikan komentar pada setiap method
 - b. Membuat nama method yang sama dengan nama class
 - c. Selalu mendeklarasikan variabel
 - d. Membuat varibel dengan keyword public
9. Sebuah jenis error yang terjadi ketika terdapat kesalahan pada hasil output adalah
 - a. Sintaks Error
 - b. Logic Error
 - c. Class Error
 - d. Run-time Error
10. Berikut ini adalah contoh software yang digunakan untuk membuat paket file installer
 - a. Inno setup
 - b. Netbeans
 - c. EXE4J
 - d. JDK

Essay

Jawablah pertanyaan-pertanyaan di bawah ini dengan jelas dan benar!

1. Tentukan class beserta method dan variabelnya, jika program yang akan dibuat berfungsi untuk menghitung jumlah total pembelian di supermarket!
2. Tuliskan fungsi perulangan yang digunakan untuk menghitung nilai factorial!
misal:
 - a. input =4 , output = $4 \times 3 \times 2 \times 1 = 24$
 - b. input =3 , output = $3 \times 2 \times 1 = 6$

A. Ceklis Penilaian Teori

NO. KUK	NO. SOAL	KUNCI JAWABAN	JAWABAN PESERTA	PENILAIAN		KETERANGAN
				K	BK	
	Isian					
	1.	Objek				
	2.	Setter dan Getter				
	3.	Setter				
	4.	Getter				
	5.	this				
	6.	a. Boolean b. int, long, short c. float, double				
	7.	SELASA SIANG				
	8.	merk				
	9.	1 2 6 24 120 720 5040				
	10.	Overriding				
	B-S					
	1.	S				
	2.	B				
	3.	S				
	4.	B				
	5.	S				
	6.	S				
	7.	B				
	8.	S				
	9.	S				
	10.	S				
	PG					
	1.	D				
	2.	A				
	3.	D				
	4.	A				
	5.	B				
	6.	B				
	7.	C				
	8.	C				
	9.	B				
	10.	A				
	Essay					
	1.	Terlampir				
	2.	Terlampir				

BAB II

PENILAIAN PRAKTIK

A. Lembar Penilaian Praktik

Tugas Unjuk Kerja Mengimplementasikan Pemrograman Berbasis Objek

1. Waktu : 120 menit
2. Alat : laptop/PC, alat tulis, Kertas
3. Bahan : IDE Netbeans 6.9 keatas
4. Indikator Unjuk Kerja
 - a. Mampu membuat program berorientasi objek dengan memanfaatkan class
 - b. Mampu menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas.
 - c. Mampu membuat program dengan konsep berbasis objek.
 - d. Mampu membuat program object oriented dengan interface dan paket
 - e. Mampu mengkompilasi program.

5. Standar Kinerja

- a. Selesai dikerjakan tidak melebihi waktu yang telah ditetapkan.
- b. Toleransi kesalahan 5% (lima persen), tetapi tidak pada aspek kritis.

6. Instruksi Kerja

Abstraksi tugas:

Buatlah sebuah program berorientasi objek yang digunakan untuk peminjaman buku pada perpustakaan, kemudian buatlah paket distribusi berupa file installer.

Untuk menyelesaikan tugas ini, ikuti instruksi selanjutnya di bawah ini.

- a. Tentukan class, variabel dan method yang akan digunakan.
- b. Gambarlah kerangka kerjanya dalam bentuk diagram class.
- c. Buatlah class yang digunakan dalam sebuah program
- d. Buatlah setter dan getter method.
- e. Masukkan kode program pada method yaitu kondisi percabangan dan perulangan.
- f. Buatlah sebuah interface dan inheritance.
- g. Implementasikan polymorphism dengan menggunakan overriding.
- h. Buatlah distribusi file installer.

B. Ceklis Aktivitas Praktik

Kode Unit Kompetensi : J.620100.018.02

Judul Unit Kompetensi : Mengimplementasikan Pemrograman Berorientasi Objek

Nama Peserta/Asesi :

INDIKATOR UNJUK KERJA	TUGAS	HAL-HAL YANG DIAMATI	PENILAIAN	
			K	BK
1. Mampu membuat program berorientasi objek dengan memanfaatkan class	1.1 Tentukan class, variabel dan method. 1.2 Buatlah class yang digunakan dalam sebuah program 1.3 Buatlah setter dan getter method.) Penggunaan nama class, variabel dan method.) Kesesuaian dengan permasalahan.		
2. Mampu Menggunakan tipe data dan control program pada metode atau operasi dari suatu kelas.	Masukkan kode program pada method yaitu kondisi percabangan dan perulangan.) Kode program kondisi percabangan) Kode program perulangan) Kesesuaian dengan sintaks		
3. Mampu membuat program dengan konsep berbasis objek.	3.1 Buatlah sebuah interface dan inheritance 3.2 Implementasikan polymorphism dengan menggunakan overriding) Kode program inheritance) Kode program polymorphism) Kode program overriding		
4. Mampu membuat program object oriented dengan interface dan paket	4.1 Buatlah sebuah interface dan inheritance	Kode program interface		
5. Mengkompilasi program	Buatlah distribusi file installer.	Hasil file installer		

Catatan :

.....

.....

.....

.....

Tanda Tangan Peserta Pelatihan :

Tanda Tangan Instruktur :

BAB III
PENILAIAN SIKAP KERJA

CEKLIS PENILAIAN SIKAP KERJA				
Mengimplementasikan Pemrograman Berorientasi Objek				
INDIKATOR UNJUK KERJA	NO. KUK	K	BK	KETERANGAN
1. Harus benar dan sesuai dengan kaidah bahasa pemrograman	1.1			
2. Harus sesuai dengan kebutuhan permasalahan	1.2			
3. Harus sesuai dengan kebutuhan permasalahan	1.3			
4. Harus efisien	2.1			
5. Cermat, tekun dan teliti	2.2			
6. Cermat, tekun dan teliti	2.3			
7. Harus berfikir analitis dan sesuai kaidah Bahasa pemrograman	3.1			
8. Harus berfikir analitis dan sesuai kaidah Bahasa pemrograman	3.2			
9. Harus berfikir analitis dan sesuai kaidah Bahasa pemrograman	3.3			
10. Sesuai kaidah Bahasa pemrograman	4.1			
11. Cermat, tekun dan teliti	5.1			
12. Cermat, tekun dan teliti	5.2			

Catatan:

.....

.....

.....

.....

.....

.....

Tanda Tangan Peserta :

Tanda Tangan Instruktur :

LAMPIRAN-LAMPIRAN

LAMPIRAN 1

Kunci Jawaban Penilaian Teori

NO. KUK	NO. SOAL	KUNCI JAWABAN
	Isian	
	1.	Objek
	2.	Setter dan Getter
	3.	Setter
	4.	Getter
	5.	this
	6.	a. Boolean b. int, long, short c. float, double
	7.	SELASA SIANG
	8.	merk
	9.	1 2 6 24 120 720 5040
	10.	Overriding
	B-S	
	1.	S
	2.	B
	3.	S
	4.	B
	5.	S
	6.	S
	7.	B
	8.	S
	9.	S
	10.	S
	PG	
	1.	D
	2.	A
	3.	D
	4.	A
	5.	B
	6.	B
	7.	C
	8.	C
	9.	B
	10.	A
	Essay	
	1.	Terlampir
	2.	Terlampir

Jawaban Soal Essay

1.

```
public class SuperMarket {  
  
    String nama_barang[]={"PENSIL", "BOKU", "PENGHAPUS"};  
    int harga_barang[]={1000, 2000, 3000};  
    int total_belanja;  
    int i=0;  
  
    public void setBelanja(String belanjaan, int jumlah){  
  
        for(int i=0; i<2; i++){  
            if(belanjaan.equals(nama_barang[i])){  
                total_belanja = total_belanja + (harga_barang[i]*jumlah);  
            }  
        }  
    }  
  
    public int getTotal(){  
        return total_belanja;  
    }  
  
    public static void main(String[] args) {  
        SuperMarket sm = new SuperMarket();  
        sm.setBelanja("PENSIL", 5);  
        sm.setBelanja("BOKU", 5);  
        System.out.println(sm.getTotal());  
    }  
}
```

2.

```
import javax.swing.*;  
public class Faktorial {  
  
    public static void main(String[] args){  
        int f = Integer.parseInt(JOptionPane.showInputDialog(null, "INPUT NILAI FAKTORIAL"));  
        int fak=1;  
        String ket = null;  
        for(int i=1; i<=f; i++){  
            fak = fak * i;  
            ket = ket + i + " x ";  
            System.out.print(i + "x");  
        }  
        System.out.print("\n");  
        System.out.println(fak);  
    }  
}
```



kptk.or.id



[instagram.com/lp3tk](https://www.instagram.com/lp3tk)



[facebook.com/lp3tk](https://www.facebook.com/lp3tk)



twitter.com/lp3tk



[youtube.com/lp3tk](https://www.youtube.com/lp3tk)