

SISTEM INFORMASI PERPUSTAKAAN MENGGUNAKAN BAHASA PEMROGRAMAN JAVA

Diajukan untuk Memenuhi Tugas Project

Mata Kuliah Pemrograman Berorientasi Objek

Dosen Pengampu : **Alun Sujjada, S.Kom, M.T**



Disusun oleh :

Ai Dina Agustin	(20210040065)
Restu Bumi Ryan Ramadhan	(20210040006)
Tegar Pratama	(20210040036)
Willy Renaldi Naibaho	(20210040004)
Yulva Cintakandida	(20210040079)

**PROGRAM STUDI TEKNIK INFORMATIKA
UNIVERSITAS NUSA PUTRA
2023**

KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Allah SWT yang telah memberikan Rahmat dan Karunia -Nya juga yang telah melimpahkan Hidayah, dan Inayah-Nya, sehingga penulis dapat menyelesaikan penulisan makalah ini dengan judul “**SISTEM INFORMASI PERPUSTAKAAN MENGGUNAKAN BAHASA PEMROGRAMAN JAVA**”.

Penulisan makalah ini diajukan untuk memenuhi salah project Mata kuliah Pemrograman Berorientasi Objek Prodi Teknik Informatika.

Kami menyadari bahwa masih banyak kekurangan yang mendasar pada makalah ini. Oleh karena itu kami mengharapkan pembaca untuk memberikan saran serta kritik yang dapat membangun kami. Kritik konstruktif dari pembaca sangat kami harapkan untuk penyempurnaan makalah selanjutnya. Akhir kata semoga makalah ini dapat memberikan manfaat bagi kita sekalian.

Sukabumi, 17 Januari 2023

Penulis

BAB I

PENDAHULUAN

1. Latar belakang

Perpustakaan merupakan salah satu sumber informasi yang mempunyai peranan penting dalam bidang pengelolaan dan penyebaran informasi. Pada era globalisasi saat ini, teknologi informasi dan komunikasi semakin canggih dan cepat. Oleh karena itu, perpustakaan harus dapat memanfaatkan teknologi informasi dan komunikasi dengan akurat, relevan dan tepat waktu. Sehingga peranan perpustakaan terhadap masyarakat dapat meningkat.

Sumber:eprints.ums.ac.id

Untuk meningkatkan efisiensi dan efektivitas dalam pengelolaan perpustakaan. Dengan menggunakan sistem informasi ini, proses peminjaman, pengembalian, dan pencarian buku dapat dilakukan dengan lebih cepat dan mudah. Selain itu, sistem ini juga dapat digunakan untuk mengelola data anggota dan statistik perpustakaan secara real-time. Pemilihan bahasa Java sebagai bahasa pemrograman untuk sistem ini dikarenakan kemampuannya dalam pengembangan aplikasi jaringan (network-based) dan platform independent.

2. Rumusan Masalah

- Bagaimana mengimplementasikan sistem informasi perpustakaan menggunakan bahasa pemrograman Java yang efisien dan efektif dalam mengelola data peminjaman, pengembalian, dan pencarian buku?
- Bagaimana mengelola data anggota dan statistik perpustakaan secara real-time dengan menggunakan sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java?
- Bagaimana menerapkan keamanan dan akses kontrol dalam sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java?
- Bagaimana meningkatkan interaksi antara anggota dan sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java?

3. Tujuan

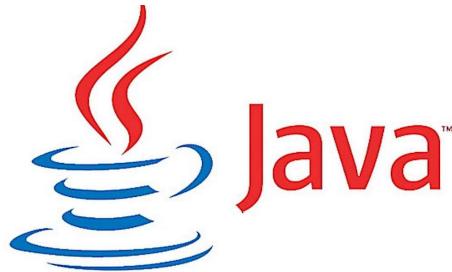
- Tujuan Makalah ini dibuat untuk memenuhi tugas akhir mata kuliah Pemrograman Berorientasi Objek
- Menjelaskan proses dan metode yang digunakan dalam implementasi sistem informasi perpustakaan menggunakan bahasa pemrograman Java.
- Menunjukkan bagaimana sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java dapat meningkatkan efisiensi dan efektivitas dalam pengelolaan perpustakaan.

- Memberikan contoh aplikasi sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java dan menjelaskan bagaimana aplikasi tersebut dapat digunakan.
- Menganalisis hasil dari sistem informasi perpustakaan yang dibuat menggunakan bahasa pemrograman Java dan memberikan rekomendasi untuk perbaikan dan pengembangan lebih lanjut.

BAB II **STUDI PUSTAKA**

2.1 Bahasa Pemrograman Java

Java adalah bahasa pemrograman yang dapat digunakan untuk mengembangkan aplikasi dan perangkat lunak untuk berbagai perangkat dan sistem operasi. Ini adalah bahasa yang di-compile (dikompilasi) ke dalam bytecode yang dapat dijalankan di Java Virtual Machine (JVM). Java dikembangkan oleh James Gosling saat masih bergabung di Sun Microsystems (sekarang Oracle) dan dirilis pada tahun 1995. Java menawarkan fitur seperti keamanan, portabilitas, dan dukungan untuk pemrograman berorientasi objek. Banyak aplikasi desktop, web, dan mobile dikembangkan menggunakan Java.



2.2 MySQL

MySQL adalah sebuah sistem manajemen basis data (DBMS) yang menggunakan bahasa query SQL (Structured Query Language). MySQL dikembangkan, didistribusikan, dan didukung oleh MySQL AB, sebuah perusahaan yang sekarang dimiliki oleh Oracle Corporation. MySQL dapat digunakan untuk mengelola berbagai jenis data, seperti data transaksi, data log, data website, dan lainnya. MySQL juga merupakan salah satu DBMS open-source yang paling populer digunakan pada saat ini.

MySQL menyediakan banyak fitur yang memungkinkan untuk:

- Membuat, mengubah, dan menghapus database dan tabel
- Menambahkan, mengubah, dan menghapus data yang disimpan dalam tabel
- Menjalankan query dan perintah SQL untuk mengambil data dari tabel
- Mengelola akses dan hak pengguna pada data
- Membuat dan mengeksekusi stored procedure dan trigger
- Membuat dan mengelola indeks untuk meningkatkan kinerja query
- Mendukung replikasi dan pemeliharaan basis data
- Integrasi dengan berbagai bahasa pemrograman seperti PHP, Java, Python, dll.

MySQL juga dapat digunakan untuk mengelola data pada berbagai aplikasi seperti website, aplikasi mobile, aplikasi desktop, dan lainnya.

MySQL bekerja dengan cara menyimpan data dalam tabel yang terdapat dalam database. Setiap database dapat berisi banyak tabel yang masing-masing dapat menyimpan berbagai jenis data. MySQL menggunakan Structured Query Language (SQL) untuk mengakses dan mengelola data dalam tabel.

Proses kerja MySQL dapat dibagi menjadi beberapa tahap :

- Koneksi ke MySQL server: Aplikasi atau klien harus terlebih dahulu terkoneksi ke MySQL server untuk dapat mengakses atau mengelola data dalam MySQL.
- Mengirimkan perintah SQL: Setelah terkoneksi, aplikasi atau klien dapat mengirimkan perintah SQL ke MySQL server melalui interface yang disediakan.
- Parsing perintah SQL: MySQL server akan menerima perintah SQL dan mengeksekusi perintah tersebut dengan mengecek sintaks dan semantik dari perintah tersebut.
- Mengelola data: MySQL server akan mengelola data sesuai dengan perintah SQL yang diterima, seperti menambah, mengubah, atau menghapus data dalam tabel.
- Mengirimkan hasil: MySQL server akan mengirimkan hasil dari perintah SQL ke aplikasi atau klien yang mengirimkan perintah tersebut.
- MySQL juga menyediakan fitur replikasi untuk menyalin data dari satu server ke server lainnya dan fitur pemeliharaan untuk menjaga performa dan integritas data.

2.3 Pemrograman Berorientasi Objek

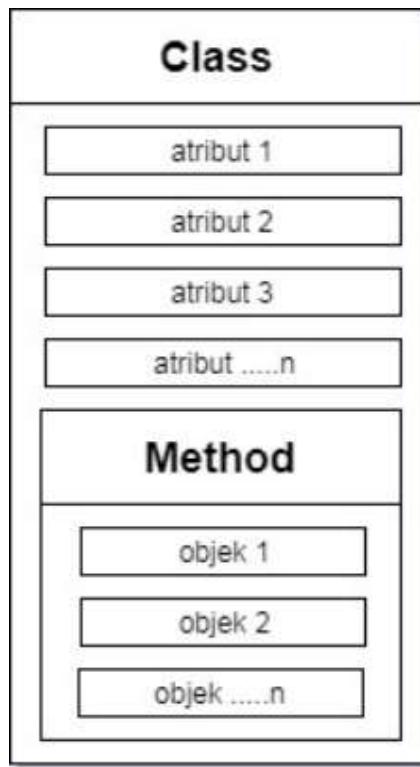
PBO adalah singkatan dari "Pemrograman Berorientasi Objek". Pemrograman berorientasi objek adalah paradigma pemrograman yang menekankan pada objek sebagai dasar pemrograman. Dalam pemrograman berorientasi objek, segala sesuatu dalam program dibentuk sebagai objek yang memiliki atribut (properti) dan perilaku (method). Objek-objek ini saling berinteraksi satu sama lain untuk menyelesaikan tugas-tugas dalam program.

Beberapa konsep dasar dalam PBO adalah kelas, objek, atribut, dan method. Kelas adalah blueprint atau desain dari suatu objek, yang menentukan atribut dan method yang dimilikinya.

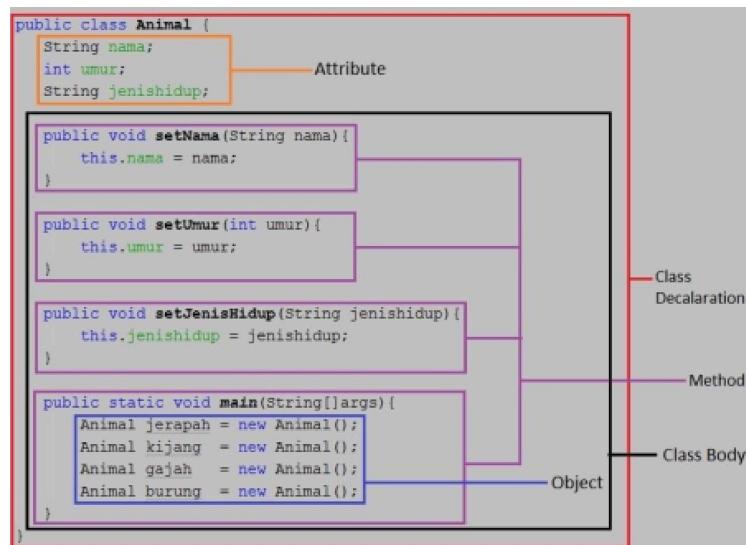
1. Kelas dan Objek

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 atau yang lebih dikenal dikalangan programmer adalah *function*.

Ruang lingkup (scope) pembuatan class dan object adalah class sebagai pembungkus sebuah object dan method. Pembuatan sebuah obyek pada umumnya dilakukan didalam method.



Deklarasi pembuatan class menggunakan kata kunci **class**, sedangkan penciptaan obyek 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 berikut :



Secara garis besar, susunan deklarasi class pada JAVA terdiri dari 2(dua) 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 obyek, maka obyek 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.

2. Atribut

Atribut adalah variabel yang digunakan untuk menyimpan data dalam sebuah objek. Atribut dalam Java didefinisikan dengan menggunakan tipe data seperti int, float, boolean, dan lainnya. Beberapa atribut yang umum digunakan dalam Java antara lain :

- private : digunakan untuk menandakan bahwa atribut hanya dapat diakses oleh kelas yang mengandung atribut tersebut.
- protected : digunakan untuk menandakan bahwa atribut dapat diakses oleh kelas yang mengandung atribut tersebut dan kelas turunannya.
- public : digunakan untuk menandakan bahwa atribut dapat diakses oleh kelas manapun.
- final : digunakan untuk menandakan bahwa atribut hanya dapat diisi sekali saja.
- static : digunakan untuk menandakan bahwa atribut adalah atribut kelas dan tidak berubah-ubah sesuai dengan objek yang dibuat.

Beberapa atribut juga dapat diberikan nilai default saat didefinisikan, seperti :

- int age = 0;
- boolean isMarried = false;
- String name = "";

Beberapa atribut juga dapat diberikan nilai saat pembuatan objek, seperti :

- Student student1 = new Student("John", 22, true);

Atribut dalam Java dapat diakses dan diubah melalui method getter dan setter yang didefinisikan dalam kelas.

3. Method

Elemen penting yang termasuk dalam pembuatan class dan object adalah method. Pada umumnya method dapat berupa fungsi ataupun prosedur yang digunakan untuk melakukan input, output 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 (Mutator) dan Getter (Accessor). 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.

2.4 Polimorfisme

Secara harfiah arti dari polymorphism adalah mempunyai banyak bentuk. Polimorfisme adalah sebuah konsep yang sangat penting dari pemrograman berorientasi objek. Pada bahasa pemrograman JAVA konsep ini dikenal dengan penggunaan lebih dari satu method yang memiliki nama yang sama.

Polymorphism berkaitan erat dengan proses pewarisan (inheritance), karena proses pembentukannya melalui superclass dan subclass. Polimorfisme pada Java terdiri dari 2 macam:

1. Static Polymorphism (Polimorfisme statis)
2. Dynamic Polymorphism (Polimorfisme dinamis)

Perbedaan keduanya adalah terletak pada bagaimana cara membuatnya. Polimorfisme statis dibuat dengan menggunakan **method overloading**, sedangkan polimorfisme dinamis menggunakan **method overriding**.

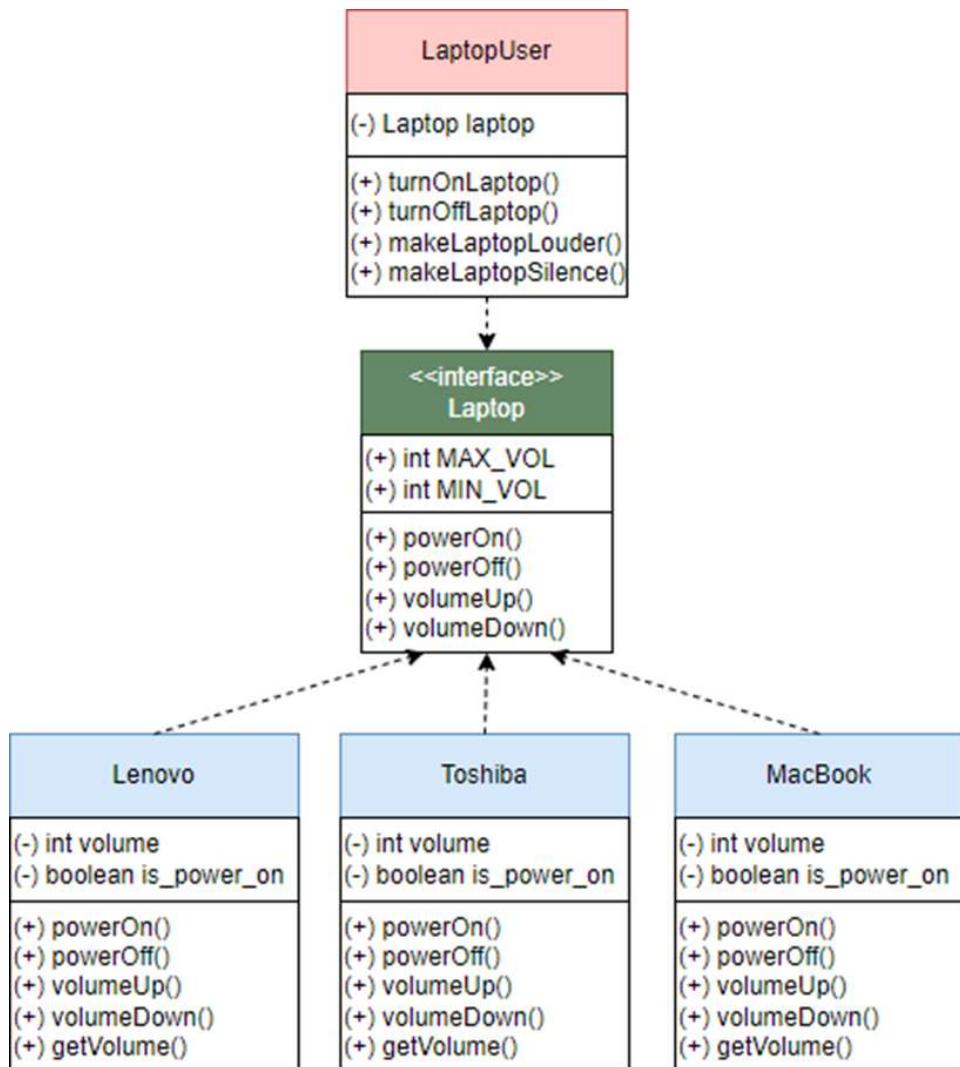
2.5 Interface

Interface adalah sebuah cara dari JAVA untuk mengakomodir konsep *multiple inheritance* yang dianut oleh bahasa pemrograman lain seperti C, C++ dan lain sebagainya. Interface adalah *blueprint* dari sebuah *class* yang berupa *method* kosong atau nama *method* tanpa implementasi program. Implementasi program dituliskan pada *class* yang menggunakan *interface*. Berdasarkan definisi *interface* adalah antarmuka, seperti yang sudah sering terdengar misalnya GUI (*Graphical User Interface*). *Interface* pada JAVA lebih tepatnya diartikan sebagai “penghubung” antara sesuatu yang masih bersifat abstrak dengan sesuatu yang telah memiliki nilai nyata. Perhatikan gambar 6.15 berikut ini:



Gambar 6.15 Ilustrasi Interface

Laptop pada gambar 6.15 mempunyai tombol-tombol keyboard. Tombol itulah yang dimaksud dengan *interface*, sedangkan *user* tidak akan tahu seberapa kompleks dan abstrak tampilan sistem yang ada pada layar monitor. Saat tombol *keyboard* tersebut ditekan, maka hasil yang didapatkan dari masing-masing merk akan berbeda. Misal ketika ditekan tombol F1 pada laptop merk A akan menonaktifkan suara (*volume*) sedangkan pada laptop merk B akan menambah tingkat kecerahan. Hal ini bergantung pada implementasi yang diberikan pada tiap-tiap tombol tersebut. Beda merk akan beda pula cara memperlakukannya. Sebagai contoh digunakan 2 objek yaitu Laptop dan User sebagai pengguna laptop seperti contoh *class diagram* pada gambar 6.16



Gambar 6.16 Class Diagram Laptop

Kode program 6.10, 6.11, 6.12, 6.13 berikut ini adalah implementasi dari class diagram

```
1. public interface Laptop {  
2.     int MAX_VOL=100;  
3.     int MIN_VOL=0;  
4.  
5.     void powerOn();  
6.     void powerOff();  
7.     void volumeUp();  
8.     void volumeDown();  
9.  
10. }
```

Kode 6.10 Interface Laptop

```
1. public class LaptopUser {  
2.     private Laptop laptop;  
3.  
4.     public LaptopUser(Laptop laptop) {  
5.         this.laptop = laptop;  
6.     }  
7.  
8.     void turnOnLaptop(){  
9.         this.laptop.powerOn();  
10.    }  
11.  
12.    void turnOffLaptop(){  
13.        this.laptop.powerOff();  
14.    }  
15.  
16.    void makeLaptopLouder(){  
17.        this.laptop.volumeUp();  
18.    }  
19.  
20.    void makeLaptopSilence(){  
21.        this.laptop.volumeDown();  
22.    }  
23.  
24. }
```

Kode 6.11 Class LaptopUser

Kemudian dibuat implementasi dari interface Laptop yaitu class Lenovo seperti pada kode 6.12 berikut ini:

```
1. public class Lenovo implements Laptop {
2.
3.     private int volume;
4.     boolean is_power_on;
5.
6.     public Lenovo(){
7.         this.volume = 50;
8.     }
9.
10.    @Override
11.    public void powerOn() {
12.        is_power_on = true;
13.        System.out.println("Laptop is On");
14.        System.out.println("Lenovo ThinkPad");
15.    }
16.
17.    @Override
18.    public void powerOff() {
19.        is_power_on = false;
20.        System.out.println("Shutdown is process ...");
21.    }
22.
23.    @Override
24.    public void volumeUp() {
25.        if(is_power_on){
26.            if(this.volume==MAX_VOL){
27.                System.out.println("Volume is Full ");
28.            }
29.            else{
30.                this.volume += 10;
31.                System.out.println("Volume is :" + this.getVolume());
32.            }
33.        }
34.    }
35.
36.
37.    @Override
38.    public void volumeDown() {
39.        if(is_power_on){
40.            if(this.volume==MIN_VOL){
41.                System.out.println("Volume is 0% ");
42.            }
43.            else{
44.                this.volume -= 10;
45.                System.out.println("Volume is :" + this.getVolume());
46.            }
47.        }
48.    }
49.
50.    public int getVolume(){
51.        return this.volume;
52.    }
53.}
```

Kode 6.22 Class Lenovo

Kemudian dibuat class Main untuk mengimplementasikan pembuatan objek seperti pada kode 6.13 berikut ini:

```
1. import java.util.Scanner;
2.
3. public class Main {
4.
5.     public static void main(String[] args) {
6.         Laptop thinkpad = new Lenovo();
7.         LaptopUser andri = new LaptopUser(thinkpad);
8.
9.         andri.turnOnLaptop();
10.        andri.makeLaptopLouder();
11.        andri.makeLaptopLouder();
12.        andri.makeLaptopSilence();
13.        andri.turnOffLaptop();
14.    }
15. }
```

Kode 6.33 Class Main

2.6 Abstract

Salah satu contoh yang mewakili konsep *abstract* yaitu class “Kendaraan”. Hal tersebut terjadi karena class “Kendaraan” belum dapat menentukan secara spesifik benda yang mewakilinya. Kendaraan bisa saja adalah sebuah motor, pesawat atau mobil dan lain sebagainya. Tetapi semuanya itu termasuk dalam kategori “Kendaraan”. Setiap “Kendaraan” pasti bisa berjalan, namun ketika disebut “Kendaraan Berjalan” akan muncul pertanyaan “Bagaimana cara jalannya?”, “Terbang atau jalan di darat atau laut?”. Tetapi jika disebut “Mobil berjalan”, maka setiap orang pasti sudah tau maksudnya. Mobil adalah benda yang konkret sedangkan “Kendaraan” adalah benda yang masih abstrak.

Class Abstract

Class abstrak adalah class yang masih dalam bentuk abstract, karena bentuknya masih abstrak, maka class tersebut tidak bisa dibuat langsung menjadi objek. Sebuah class agar dapat disebut class abstract setidaknya memiliki satu atau lebih method abstract. Method abstract adalah method yang tidak memiliki implementasi atau tidak ada bentuk konkritisnya. Gambar 6.13 berikut ini adalah contoh dari method abstract dan bukan abstract.

```
//Contoh class Abstract
void setData();

//Contoh bukan class Abstract
void setData(){
    this.name = name;
}
```

Gambar 6.13 Contoh method abstract dan non abstract

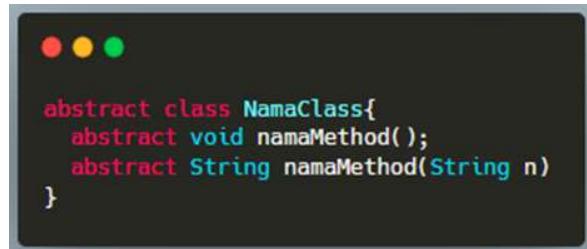
Penggunaan Class Abstract

Class abstract memang belum dapat digunakan secara langsung, sehingga untuk menggunakannya maka harus dibuat terlebih dahulu bentuk konkritisnya. Cara untuk membuat class abstract menjadi bentuk konkret adalah dengan mengimplementasikan method yang ada didalamnya. Hal tersebut dapat dilakukan dengan menggunakan cara pewarisan (*inheritance*). Class Abstract dibuat sebagai parent class dari class yang lain. Child class membuat bentuk konkret dari class abstract.

Mengapa dibutuhkan sebuah class abstract, padahal hal tersebut dapat dilakukan menggunakan class biasa saja?. Pernyataan tersebut adalah benar, akan tetapi pada sebuah kondisi tertentu parent class tidak ingin dibuat sebagai *object*, karena methodnya belum jelas akan diimplementasikan seperti apa? maka sebaiknya *class* tersebut dibuat menjadi *abstract*.

Cara Membuat Class Abstract

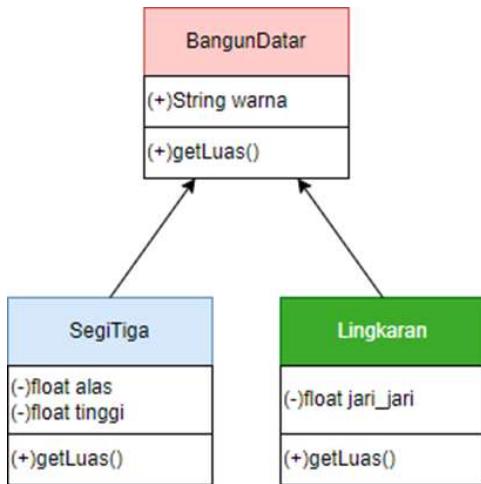
Cara membuat class abstract adalah menambahkan kata kunci *abstract* pada class dan method. Gambar 6.14 berikut ini adalah contoh dari class abstract:



```
abstract class NamaClass{
    abstract void namaMethod();
    abstract String namaMethod(String n)
}
```

Gambar 6.14 Membuat class abstract

Berikut ini adalah contoh program dari class abstract



Gambar 6.15 Diagram Class

Class BangunDatar adalah class abstrak, karena mempunyai method abstrak getLuas(). Jika akan membuat objek dengan class BangunDatar dan memanggil method getLuas(), maka class BangunDatar akan bingung. BangunDatar atau bentuk bangun datar yang mau dihitung luasnya seperti apa? bagaimana rumusnya? oleh karena itu, class ini dijadikan sebagai abstrak. Sedangkan seperti yang diketahui secara umum bahwa setiap bangun datar pasti mempunyai luas. Kode 6.6 berikut ini adalah contoh implementasi penyelesaian masalah bangun datar.

```

1. public abstract class BangunDatar {
2.     String warna;
3.
4.     String getWarna() {
5.         return warna;
6.     }
7.
8.     void setWarna(String warna) {
9.         this.warna = warna;
10.    }
11.
12.    abstract float getLuas();
13. }
```

Kode 6.6 Class BangunDatar

Kemudian dibuat class SegiTiga dan Lingkaran seperti kode program 6.7 dan 6.8 berikut ini:

```
1. public class SegiTiga extends BangunDatar {
2.
3.     private float alas;
4.     private float tinggi;
5.
6.     public SegiTiga(float alas, float tinggi) {
7.         this.alas = alas;
8.         this.tinggi = tinggi;
9.     }
10.
11.    @Override
12.    float getLuas() {
13.        return (float)0.5 * alas * tinggi;
14.    }
15.}
```

Kode 6.7 Class SegiTiga

```
1. public class Lingkaran extends BangunDatar {
2.
3.     float jari_jari;
4.
5.     public Lingkaran(float jari_jari) {
6.         this.jari_jari = jari_jari;
7.     }
8.
9.    @Override
10.    float getLuas() {
11.        return (float) Math.PI * jari_jari * jari_jari;
12.    }
13.}
```

Kode 6.8 Class Lingkaran

Proses terakhir adalah membuat class Main untuk mengimplementasikan pembuatan objek dengan class yang telah dibuat sebelumnya.

```
1. public class Main {
2.
3.     public static void main(String[] args) {
4.
5.         BangunDatar segitiga = new SegiTiga(12, 20);
6.         BangunDatar lingkaran = new Lingkaran(60);
7.
8.         System.out.println("Luas dari bangun datar segitiga : " + segitiga.getLuas());
9.         System.out.println("Luas dari bangun datar lingkaran : " + lingkaran.getLuas());
10.    }
11.}
```

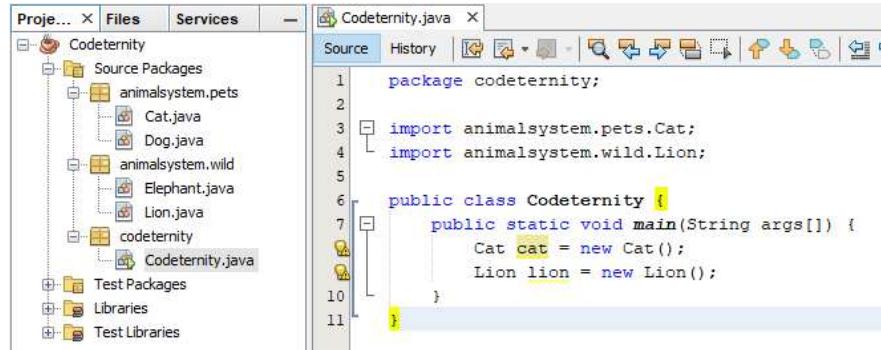
Kode 6.9 Class Main

Pada class Main terdiri dari objek yaitu objek segitiga dan lingkaran. Kedua objek ini memiliki tipe yang sama yaitu BangunDatar, tetapi dibuat dari class yang berbeda.

2.7 Package

Package adalah kumpulan dari kelas, interface, dan paket lainnya yang digunakan untuk mengelompokkan kode-kode yang berhubungan dan membuat organisasi kode lebih baik. Package juga digunakan untuk mengelola akses pada kode dan menghindari konflik nama dengan kode dari pihak lain. Package juga memungkinkan untuk mengakses kelas dari package lain dengan menggunakan kata kunci import.

Fungsi Package digunakan untuk mengelompokkan file class yang terkait dan menyimpannya dalam direktori yang sama. Dengan mengelompokkan class berdasarkan jenis, fungsinya, atau alasan lainnya, membuat program lebih mudah dikelola dan dipahami. Directive package yang dituliskan pada setiap class akan mengacu pada direktori yang sesuai dengan package tersebut, sehingga memudahkan dalam pencarian dan pengaksesan class tersebut. Package juga memungkinkan untuk menghindari konflik nama class dan mengatur hak akses pada class tersebut.



Contoh struktur proyek Java di Netbeans menggunakan package

- Built-in Package

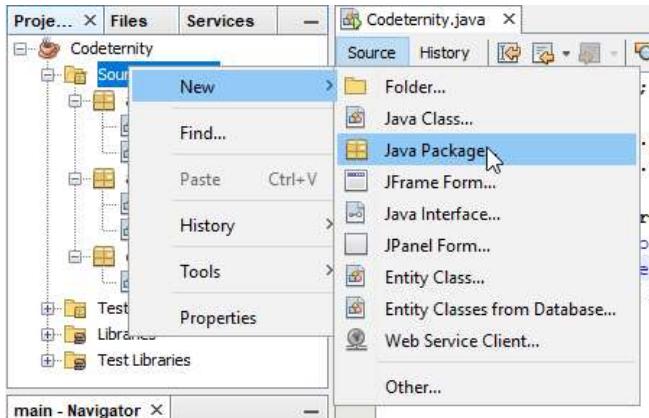
Built-in package adalah package bawaan yang telah disediakan oleh Java. Beberapa contoh package built-in yang disediakan Java :

- java.lang: package ini menyediakan class-class dasar yang digunakan dalam setiap program Java, seperti class String, Math, dan lainnya.
- java.util: package ini menyediakan class-class yang digunakan untuk membuat aplikasi yang lebih kompleks, seperti class ArrayList, HashMap, dan lainnya.
- java.io: package ini menyediakan class-class yang digunakan untuk melakukan operasi input dan output, seperti class File, BufferedReader, dan lainnya.
- java.net: package ini menyediakan class-class yang digunakan untuk melakukan koneksi jaringan, seperti class URL, Socket, dan lainnya.
- java.awt: package ini menyediakan class-class yang digunakan untuk membuat aplikasi grafis, seperti class Button, Frame, dan lainnya.
- java.sql: package ini menyediakan class-class yang digunakan untuk melakukan operasi database, seperti class Connection, Statement, dan lainnya.

Ini hanyalah beberapa contoh dari banyak package built-in yang disediakan oleh Java dan digunakan untuk membuat aplikasi yang lebih kompleks dan canggih.

- User Defined Package

Cara termudah membuat package pada IDE netbeans adalah dengan klik kanan source package > New > Java Package.



Membuat New Package pada IDE Netbeans

Saat kelas dibuat pada package maka IDE akan otomatis mendeklarasikan package dengan kata kunci *package*.

```
//package declaration
package animalsystem;

public class Animal {
```

- Sub - Package

Sub-package adalah package yang dibuat di dalam package lain. Tanda titik (.) digunakan untuk memisahkan antara package induk dan anak. Misalnya, jika kita memiliki package induk yang bernama "com" dan package anak yang bernama "example", nama package lengkapnya adalah "com.example". Sub-package digunakan untuk mengatur kelas dan interface yang terkait, serta untuk menghindari konflik nama antar package.

- Import Package

Untuk mengimport package pada java, kita dapat menggunakan perintah "import" sebelum deklarasi class yang akan digunakan. Contohnya:

```
package codernity;

import java.util.ArrayList;

class Main {
    public static void main(String[] args) {
        ArrayList<Integer> numbers = new ArrayList<Integer>();
        // code lain
    }
}
```

Penjelasan program : Dalam contoh di atas, package "java.util" telah diimport dan digunakan untuk membuat objek ArrayList. Kita juga dapat mengimport lebih dari satu package dengan memisahkan setiap perintah "import" dengan tanda titik koma. Contohnya:

```
package codernity;

import java.util.ArrayList;
import java.io.File;

class Main {
    public static void main(String[] args) {
        ArrayList<Integer> numbers = new ArrayList<Integer>();
        File file = new File("example.txt");
        // code lain
    }
}
```

Selain itu, kita juga dapat mengimport seluruh package yang ada dalam suatu package dengan menggunakan wildcard (*). Contohnya:

```
package codernity;

import java.util.*;

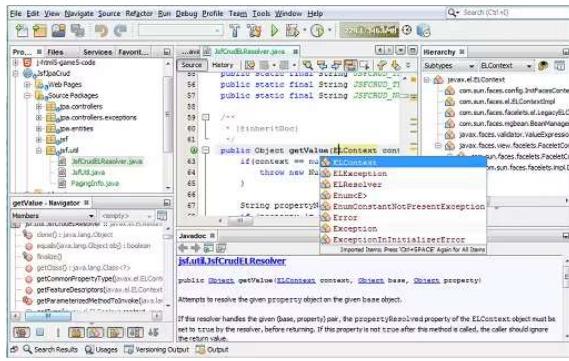
class Main {
    public static void main(String[] args) {
        ArrayList<Integer> numbers = new ArrayList<Integer>();
        // code lain
    }
}
```

Perlu diingat bahwa menggunakan wildcard akan membuat program menjadi lebih berat dan memperlambat kinerja karena semua class dalam package tersebut akan di import. Sebaiknya hanya mengimpor package yang dibutuhkan saja.

sumber : [Pengertian Package Pada Java - Codernity](#)

2.8 Netbeans

Netbeans merupakan sebuah aplikasi *Integrated Development Environment* (IDE) berbasis Java dari *Sun Microsystems* yang berjalan di atas swing. Swing disini adalah sebuah teknologi Java untuk pengembangan aplikasi desktop yang dapat berjalan di berbagai macam platform seperti windows, linux, Mac OS X dan juga Solaris. IDE ini dikembangkan oleh Oracle dan dapat digunakan untuk membuat aplikasi berbasis Java SE, JavaFX, Java Web, dan Java EE.



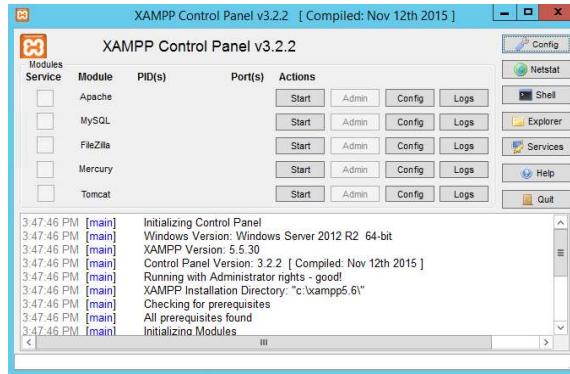
NetBeans menyediakan berbagai fitur yang memudahkan developer dalam menulis kode, seperti code completion, debugging, dan refactoring. IDE ini juga dilengkapi dengan tools seperti profiler, GUI builder, dan version control. NetBeans juga dilengkapi dengan fitur drag and drop yang memungkinkan developer untuk membuat aplikasi dengan cepat dan mudah. Fitur ini juga memungkinkan developer untuk mengubah tata letak aplikasi dengan mudah tanpa harus menulis kode.

Netbeans ini memiliki beberapa fitur adapun Fitur fitur yang terdapat dalam netbeans antara lain:

1. Smart Code Completion: yaitu berfungsi untuk mengusulkan nama variabel dari suatu tipe, melengkapi keyword dan juga mengusulkan tipe parameter dari sebuah method.
2. Bookmarking: fitur yang satu ini digunakan untuk menandai baris yang suatu saat ingin kita modifikasi.
3. Go to commands: fitur ini digunakan untuk jump ke deklarasi variabel, source code atau file yang terdapat pada project yang sama.
4. Code generator: Apabila kita menggunakan fitur ini kita akan bisa meng-generate constructor, setter and getter method dan yang lainnya.
5. Error stripe: fitur ini akan menandai baris yang eror dengan memberi highlight red.
6. Debugging: NetBeans menyediakan fitur debugging yang memungkinkan developer untuk menemukan dan memperbaiki kesalahan dalam kode. Fitur ini termasuk breakpoint, step-by-step debugging, dan watch variables.
7. GUI Builder: NetBeans menyediakan fitur GUI builder yang memungkinkan developer untuk membuat interface grafis tanpa harus menulis kode. Fitur ini termasuk drag and drop, properties editor, dan preview.
8. Refactoring: NetBeans menyediakan fitur refactoring yang memungkinkan developer untuk mengubah struktur kode tanpa merusak fungsi dari kode tersebut. Fitur ini termasuk rename, move, and extract method.

sumber : [Pengertian Netbeans - Fitur-Fitur dan Sejarah Netbeans \(materibelajar.co.id\)](http://materibelajar.co.id)

2.9 Xampp



XAMPP adalah sebuah software yang menyediakan lingkungan pengembangan web yang lengkap dan mudah digunakan. XAMPP merupakan singkatan dari "X" (untuk sistem operasi apa saja), Apache, MySQL, PHP, dan Perl. XAMPP adalah software gratisan dan open source, serta dapat diinstall di berbagai platform, seperti Windows, Linux, maupun OS X.

XAMPP menyediakan instalasi cepat dan mudah untuk Apache (web server), MySQL (database server), PHP (bahasa pemrograman), dan Perl (bahasa skrip), sehingga developer dapat dengan mudah membuat aplikasi web tanpa harus mengkonfigurasi setiap komponen secara terpisah.

XAMPP juga memungkinkan developer untuk menjalankan aplikasi web di komputer local, sehingga mereka dapat melakukan pengujian dan debugging aplikasi tanpa harus mengunggahnya ke server web.

sumber : [Apa itu XAMPP? Sejarah, Fungsi, dan Fitur-fitur XAMPP \(niagahoster.co.id\)](http://niagahoster.co.id)

BAB III

METODOLOGI

Metode Pembuatan Program

Dalam proses pelaksanaan dan pengembangan aplikasi yang dibuat, metode yang digunakan adalah metode prototype. Metode Prototipe adalah proses iterative (berulang) dalam pengembangan aplikasinya, dimana kebutuhan dapat diubah ke dalam sistem yang bekerja secara terus menerus dievaluasi dan diperbaiki sehingga aplikasi tersebut berjalan dengan semestinya sesuai dengan diharapkan oleh pengguna. Berikut tahapan-tahapan metode yang digunakan dalam pembuatan alat

GAMBAR

Tahapan-tahapan dalam metode Prototipe adalah sebagai berikut:

1. Identifikasi Kebutuhan Dalam tahap ini pengguna dan pengembangan Bersama-sama melakukan identifikasi format keseluruhan sistem yang akan dibuat, mengidentifikasi semua kebutuhan dan garis besar sistem yang akan dibuat.
2. Studi Literatur Studi literatur yang dilakukan yaitu dengan melakukan pencarian terhadap berbagai sumber tertulis, baik berupa buku-buku, arsip, majalah, artikel, dan jurnal, atau dokumen dokumen yang relevan dengan permasalahan yang dikaji. Sehingga informasi yang didapat dari studi kepustakaan ini dijadikan rujukan untuk memperkuat argumentasi dari identifikasi kebutuhan.
3. Membuat Prototipe Pada tahap ini penulis mulai membangun prototype yaitu aplikasi (software) yaitu membuat program/pengkodingan.
4. Menguji program Setelah program selesai dibuatkan prototipenya, selanjutnya dilakukan pengujian prototipe oleh pengguna (user) dan pengguna dapat memberikan kritik dan saran.
5. Perbaikan program Pada tahap ini inovator melakukan modifikasi dan evaluasi sesuai dengan hasil pengujian program atau hasil masukan dari pengguna. Evaluasi dan perbaikan dilakukan terus menerus hingga aplikasi siap pakai.

BAB IV

HASIL DAN DISKUSI

Identifikasi Program

Program ini dibuat menggunakan Bahasa pemrograman Java. Program ini dibuat guna untuk mendata peminjaman dan pengembalian buku yang ada pada perpustakaan kemudian data tersebut dapat membantu pekerjaan untuk para pustakawan atau petugas yang berjaga dalam menangani peminjaman buku dan pengembalian buku.

Penggunaan program ini hanya dikelola oleh petugas. Ada 5 menu yang tersedia dalam program ini diantaranya yaitu : menu Data Buku, menu Data Petugas, menu Data Peminjaman, menu Data Pengembalian. Dalam menu - menu tersebut terdapat fungsi yaitu : Create, Read, Search, Update dan Delete. Untuk menu yang disediakan untuk anggota perpustakaan yaitu ada 3 menu di antaranya :

Rancangan Program

Algoritma

Algoritma adalah langkah-langkah yang disusun secara terstruktur dan terurut untuk menjawab suatu persoalan dengan menggunakan penyajian deskriptif, flowchart atau pseudocode.

Algoritma Deskriptif dan Flowchart

1. Start
2. Deklarasi Variable :

Tampilan Program

Tampilan program Menu

Tampilan data Buku

Tampilan data Petugas

Tampilan data Peminjaman

Tampilan data Pengembalian

Pengujian Program

BAB V

PENUTUP

Kesimpulan

Sistem informasi perpustakaan menggunakan bahasa pemrograman Java dapat digunakan untuk mengelola data buku, anggota, peminjaman, dan pengembalian dengan lebih efisien dan mudah. Sistem ini juga dapat memberikan informasi yang cepat dan akurat tentang ketersediaan buku, riwayat peminjaman, dan lainnya. Dengan menggunakan Java, sistem ini dapat dijalankan pada berbagai platform dan dapat dikembangkan dengan fitur-fitur yang lebih canggih. Namun, sistem ini memerlukan tenaga kerja yang memahami bahasa pemrograman Java untuk pengembangan dan pemeliharaannya.

Saran