MODUL PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

Oleh:
Arie Gunawan, S.Kom., MMSI

Fakultas Teknologi Komunikasi dan Informatika
Program Studi Sistem Informasi
Universitas Nasional
2022

MODUL 1

METHOD

Method adalah kumpulan program yang mempunyai nama. Program harus dibungkus dalam method. Dengan method kita bisa memanggil kumpulan program hanya dengan memanggil nama methodnya, pekerjaan jadi lebih singkat dan tidak boros menuliskan program, program menjadi lebih terstruktur, praktis, dan efisien.

Bentuk umum:

```
<nama_method>( <parameter/argument>);
// menggunakan tanda kurung setelah nama method itu kuncinya
// parameter/argument bersifat opsional, tergantung kebutuhan
```

Contoh:

```
cetak();
//method bernama cetak
```

Ada 2 jenis method:

1. **Method yang tidak mengembalikan data** (diberi tipe void)
   Contoh: static void cetak();
   //method static bernama cetak yang tidak mengembalikan nilai.
2. **Method yang bisa mengembalilkan nilai** menggunakan statement return dan tipe data
   Contoh: static int cetak(int b);
   return b;
   //method bernama cetak yang mengembalikan nilai integer
   //mempunyai 1 parameter/argument yaitu b dan bertipe integer
   //return adalah keyword untuk mengembalikan nilai

Catatan:

method yang dipanggil dari dalam method static harus static.
method tidak static hanya bisa dipanggil melalui method tidak statik dan mekanisme instansiasi.
Jika memanggil method dalam satu kelas method harus static.
method static hanya bisa memanggil method yang static.

Parameter/argument adalah suatu nilai yang dapat diubah-ubah dari luar untuk menentukan hasil

mengirim >> parameter/argument aktual (yang di main).
menerima >> parameter/argument formal.

Contoh method tidak mengembalikan nilai (tanpa parameter/argument) dengan nama file Contoh1.java

```
1 public class Contoh1 {
2     static void cetak(){
3         System.out.print("I Love ");
4     }
5     public static void main(String[] args) {
6         cetak();
7         System.out.println("JAVA");
8     }
9 }
```

Output: I Love JAVA

Contoh method tidak mengembalikan nilai (dengan parameter/argument) dengan nama file Contoh2.java

```
1  class Contoh2 {
2     static void cetak(int a){
3       System.out.println("Nilai x: "+a);
4     }
5     public static void main(String[] args) {
6       int x;
7       for (x=0; x<3; x++){
8         cetak(x);
9       }
10      System.out.println("Nilai x terakhir: "+x);
11    }
12 }
```

Output:
Nilai x: 0
Nilai x: 1
Nilai x: 2
Nilai x terakhir: 3

Catatan:
• static void cetak(int a). int a adalah parameter/argument formal.
• cetak(x). x adalah parameter/argument aktual atau parameter/argument yang dikirim.

Contoh method mengembalikan nilai dengan nama file Contoh3.java

```
1  class Contoh3 {
2    static int kuadrat(int bil){
3      return bil*bil;
4     }
5     public static void main(String[] args){
6       int x=5;
7       System.out.println(x +" kuadrat adalah "+ kuadrat(x));
8     }
9  }
```

Output: 5 kuadrat adalah 25

Contoh method menggunakan 2 parameter/argument dengan nama file Contoh4.java

```
1  class Contoh4 {
2    static int penjumlahan(int bil_1, int bil_2){
3      return bil_1 + bil_2;
4     }
5     public static void main(String[] args){
6       int x1 = 2;
7       int x2 = 3;
8       System.out.println(x1+" + "+x2+" = "+ penjumlahan(x1,x2) );
9     }
10 }
```

Output: 2 + 3 = 5

MODUL 2

PENGENALAN *CLASS*, *OBJECT* & ENKAPSULASI

A. *Class*

Class merupakan cetak biru (blue print) dari objek atau dengan kata lain sebuah Class menggambarkan ciri-ciri objek secara umum. Sebagai contoh Suzuki Smash, Yamaha VegaR, Honda SupraFit, dan Kawasaki KazeR merupakan objek dari Class sepeda motor. Suzuki Smash dan objek lainnya juga memiliki kesamaan atribut (merk, tipe, berat, kapasitas bensin, tipe mesin, warna, harga) dan method untuk mengakses data pada atributnya (misal fungsi untuk menginputkan data merk, tipe, berat, dsb serta fungsi untuk mencetak data merk, tipe, berat, dsb).

Contoh :

```
1  public class SepedaMotor {
2     private String merk;
3     private long harga;
4
5     public void setMerk(String merkMotor) {
6        merk = merkMotor;
7     }
8
9     public String getMerk(){
10       return merk;
11    }
12
13    public long Harga(long hargaMotor) {
14       return harga = hargaMotor;
15    }
16 }
```

B. *Object*

Objek (*Object*) merupakan segala sesuatu yang ada di dunia ini, yaitu manusia, hewan, tumbuhan, rumah, kendaraan, dan lain sebagainya. Contoh-contoh objek yang telah disebutkan diatas merupakan contoh objek nyata pada kehidupan kita.

Pada pemrograman berorientasi objek, kita akan belajar bagaimana membawa konsep objek dalam kehidupan nyata menjadi objek dalam dunia pemrograman. Setiap objek dalam dunia nyata pasti memiliki 2 elemen penyusunnya, yaitu keadaan (*state*) dan perilaku/sifat (*behaviour*). Sebagai contoh, sepeda memiliki keadaan yaitu warna, merk, jumlah roda, ukuran roda. Dan perilaku/sifat sepeda adalah berjalan, berhenti, belok, menambah kecepatan, mengerem.

Pada saat objek diterjemahkan ke dalam konsep PBO, maka elemen penyusunnya juga terdiri atas 2 bagian, yaitu :

Atribut, merupakan ciri-ciri yang melekat pada suatu objek (*state*).

Method, merupakan fungsi-fungsi yang digunakan untuk memanipulasi nilai-nilai pada atribut atau untuk melakukan hal-hal yang dapat dilakukan suatu objek (*behaviour*).

Objek dalam konsep PBO memiliki keadaan dan perilaku yang sama seperti halnya objek di dunia nyata, karena objek dalam konsep PBO merupakan representasi objek dari dunia nyata.

Objek dalam PBO merepresentasikan keadaan melalui variabel-variabel (Atribut), sedangkan perilakunya direpresentasikan dengan method (yang merupakan suatu fungsi yang berhubungan dengan perilaku objek tersebut maupun berhubungan dengan atribut dari objek tersebut). Objek yang memiliki kesamaan atribut dan method dapat dikelompokkan menjadi sebuah Class. Dan objek-objek yang dibuat dari suatu class itulah yang disebut dengan Instant of class.

Untuk menginstansi (membuat) objek dari class, gunakan operator *new*.

Sintaks membuat objek dari suatu class :

```
namaClass namaObjek = new namaClass()
```

Class utama dari program :

```
1  public class Main {
2      public static void main (String []args){
3          SepedaMotor motor = new SepedaMotor();
4          motor.setMerk("Suzuki");
5          System.out.println("Motor ini bermerk " +motor.getMerk());
6          System.out.println("Motor ini berharga " +motor.Harga(11000000));
7      }
8  }
```

Perhatikan class Main diatas !

Nama objek dari class SepedaMotor adalah motor.

Silahkan dicoba untuk melihat hasilnya !

C. Enkapsulasi

Enkapsulasi (*encapsulation*) merupakan cara untuk melindungi property (atribut) / method tertentu dari sebuah kelas agar tidak sembarangan diakses dan dimodifikasi oleh suatu bagian program. Cara untuk melindungi data yaitu dengan menggunakan *access modifiers* (hak akses). Ada 4 hak akses yang tersedia, yaitu default, public, protected, private.

Untuk lebih jelasnya, silahkan lihat kedua table berikut ini :

No	Modifier	Pada class dan interface	Pada method dan variabel
1	Default (tidak ada modifier)	Dapat diakses oleh yang sepaket	Diwarisi oleh subkelas dipaket yang sama, dapat diakses oleh method-method yang sepaket
2	Public	Dapat diakses dimanapun	Diwarisi oleh subkelasnya, dapat diakses dimanapun
3	Protected	Tidak bisa diterapkan	Diwarisi oleh subkelasnya, dapat diakses oleh method-method yang sepaket
4	private	Tidak bisa diterapkan	Tidak dapat dikases dimanapun kecuali oleh method-method yang ada dalam kelas itu sendiri

Aksesabilitas	public	private	protected	default
Dari kelas yang sama	Ya	Ya	Ya	Ya
Dari sembarang kelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari sembarang kelas di luar paket	Ya	Tidak	Tidak	Tidak
Dari subkelas dalam paket yang sama	Ya	Tidak	Ya	Ya
Dari subkelas di luar paket	Ya	Tidak	Ya	Tidak

Perhatikan keyword "***this***" di bawah ini (lihat pada class Enkapsulasi). Untuk membedakan variabel alas pada parameter dan variabel alas pada atribut class Enkapsulasi, digunakanlah keyword "*this*". Sehingga untuk menggunakan atribut alas pada class Enkapsulasi digunakan : *this.*alas

Contoh:

Nama file : Enkapsulasi.java

```java
public class Enkapsulasi {
    private int alas, tinggi;
    private double luasSegitiga;

    public void setAlas(int alas) {
        this.alas = alas;
    }

    public int getAlas() {
        return alas;
    }

    public void setTinggi(int tinggi) {
        this.tinggi = tinggi;
    }

    public int getTinggi() {
        return tinggi;
    }

    public void setLuasSegitiga(int alas, int tinggi) {
        luasSegitiga = 0.5 * (double)(alas * tinggi);
    }

    public double getLuasSegitiga() {
        return luasSegitiga;
    }
}
```

Nama file : MainEnkapsulasi.java

```
1  public class MainEnkapsulasi {
2      public static void main (String args[])
3      {   Enkapsulasi ob = new Enkapsulasi();
4          ob.setAlas(5);
5          ob.setTinggi(7);
6          System.out.println("Alas Segitiga : "+ob.getAlas());
7          System.out.println("Tinggi Segitiga : "+ob.getTinggi());
8          ob.setLuasSegitiga(ob.getAlas(), ob.getTinggi());
9          System.out.println("Luas Segitiga : "+ob.getLuasSegitiga());
10     }
11 }
```

Silahkan dicoba untuk melihat hasilnya !

D. Overloading

Overloading adalah diperbolehkannya dalam sebuah class memiliki lebih dari satu nama function/method yang sama tetapi memiliki parameter/argument yang berbeda.

Contoh :

```
1  public class Overloading {
2      public void Tampil(){
3          System.out.println("I love Java");
4      }
5      public void Tampil(int i){
6          System.out.println("Method dengan 1 parameter = "+i);
7      }
8      public void Tampil(int i, int j){
9          System.out.println("Method dengan 2 parameter = "+i+" & "+j);
10     }
11     public void Tampil(String str){
12         System.out.println(str);
13     }
14
15     public static void main(String a[]){
16         Overloading objek = new Overloading();
17         objek.Tampil();
18         objek.Tampil(8);
19         objek.Tampil(6,7);
20         objek.Tampil("Hello world");
21     }
22 }
```

Silahkan dicoba untuk melihat hasilnya !

MODUL 2 - LATIHAN

Latihan Soal

1. Seorang penjual alat tulis menjual 10 bolpoint, 10 pensil dan 10 penghapus. 1 biji bolpoint harganya Rp. 2000, 1 biji pensil harganya Rp. 1.000 dan 1 penghapus harganya Rp. 500. Gunakanlah objek untuk menyelesaikan soal dibawah ini!
   a. Buatlah method untuk memasukkan *(setter)* nama, stok, harga satuan, dan harga (stok x harga satuan) alat tulis tersebut! (40 point)
   b. Buatlah method untuk menampilkan *(getter)* nama, stok, harga satuan, dan harga (stok x harga satuan) alat tulis tersebut! (40 point)
   c. Buatlah method Total Harga *(setter getter)* untuk menampilkan uang yang diterima penjual jika semua alat tulis tersebut terjual semuanya! (20 point)

Catatan untuk asisten:

a. Soal dibuat berbeda antara hari senin, selasa, dan rabu. Untuk hari senin bisa memakai soal diatas, untuk hari selasa, dan rabu dapat diubah jumlah, harga dan jenisnya (misalnya mobil Toyota, Kijang, Panther, Karimun) terserah. Boleh lebih dari 3 variable.

b. *Output*

```
Nama Alat Tulis : Bolpoint
Stok : 10
Harga Satuan: 2000
Harga Bolpoint : 20000

Nama Alat Tulis : Pensil
Stok : 10
Harga Satuan: 1000
Harga Pensil : 10000

Nama Alat Tulis : Penghapus
Stok : 10
Harga Satuan: 500
Harga Penghapus : 5000

Total Harga: 35000
```

MODUL 3

ARRAY DIMENSI 1

A. Array

Array atau Larik adalah variable yang digunakan untuk menyimpan data-data yang mempunyai tipe data yang sama. Di Java, Array dimulai dari index ke-0 sampai ke ke-n. Ada dua macam cara deklarasi array dimensi 1 di Java, yaitu :

a) tipeData[] namaArray;
   Misal: String[] mhs;
b) tipeData namaArray[];
   Misal: String mhs[];

Ada dua macam cara membuat array dimensi 1 di Java, yaitu:

a) tipeData[] namaArray = new tipeData[ukuranArray];
   Misal:
   String[] mhs = new String[50];
b) tipeData[] namaArray = {isiArray, isiArray,...,isiArray};
   Misal:
   String[] mahasiswa = {"Dono","Danu","Dana","Dini","Doni"};

Contoh:
Nama file : Array.java

```java
public class Array {
    private String[] mhs;
    private int[] deret;
    private int hasilPenjumlahan;

    public void setMhs(String[] mhs) {
        this.mhs = mhs;
        mhs = null; // menghapus variable parameter dari memory
    }

    public String[] getMhs() {
        return mhs;
    }

    public void setDeret(int[] deret) {
        this.deret = deret;
        deret = null; // menghapus variable parameter dari memory
    }

    public int[] getDeret() {
        return deret;
    }

    public void setPenjumlahan(int[] deret) {
        hasilPenjumlahan = 0;
        for (int i = 0; i < deret.length; i++) {
            hasilPenjumlahan += deret[i];
        }
        deret = null; // menghapus variable parameter dari memory
    }

```

```
    public int getPenjumlahan() {
        return hasilPenjumlahan;
    }

    public void tampil(String a) {
        System.out.println(a);
        a = null;   // menghapus variable dari memory
    }

    public void tampil(String a[]) {
        String data = "";
        for (int i = 0; i < a.length; i++) {
            if (i == 0)
            {   data += a[i];
            } else
            {   data += ", "+a[i];
            }
        }
        System.out.println(data);
        a = null;   // menghapus variable dari memory
        data = null;
    }

    public void tampil(int a) {
        System.out.println(a);
    }

    public void tampil(int a[]) {
        String data = "";
        for (int i = 0; i < a.length; i++) {
            if (i == 0)
            {   data += a[i];
            } else
            {   data += ", "+a[i];
            }
        }
        System.out.println(data);
        a = null;   // menghapus variable dari memory
        data = null;
    }

    public void hapus()
    {   // menghapus variable dari memory untuk optimasi program
        mhs = null;
        deret = null;
        hasilPenjumlahan = 0;
    }

}
```

Nama file : MainArray.java

```
public class MainArray {
    public static void main (String args[])
    {   Array ob = new Array();
        String mahasiswa[]={"Dono","Danu","Dana","Dini","Doni"};
        int deret[] = {2,5,6,9,7};
        ob.tampil("Daftar Mahasiswa : ");
        ob.setMhs(mahasiswa);
        ob.tampil(ob.getMhs());
        ob.tampil("==========================");
        ob.tampil("Deret : ");
        ob.setDeret(deret);
        ob.tampil(ob.getDeret());
        ob.tampil("Hasil Penjumlahan Deret : ");
        ob.setPenjumlahan(deret);
        ob.tampil(ob.getPenjumlahan());

        // menghapus variable dari memory untuk optimasi program
        ob.hapus();
        mahasiswa = null;
        deret = null;
        ob = null;
    }
}
```

MODUL 3
ARRAY - LATIHAN

1) Terdapat 5 buah bilangan yaitu -5, -3, -6, -3, -4.

   Dengan menggunakan objek dan enkapsulasi *(setter getter)*, ...

   a) Carilah nilai rata-rata!

   b) Carilah nilai maksimum!

   c) Carilah nilai minimum!

   d) Angka -3 pada bilangan tersebut terdapat pada index keberapa?

MODUL 3
ARRAY - LATIHAN 2

1. Terdapat bilangan integer (2,5,3,5,9,5). Dengan menggunakan objek dan enkapsulasi *(setter getter)*, ubah angka 5 menjadi angka 1!
2. Hasil dari bilangan no. 1 adalah bilangan (2,1,3,1,9,1). Dengan menggunakan objek dan enkpasulasi *(setter getter)*, kalikan bilangan tersebut dengan 0.5!

MODUL 4

ARRAY DIMENSI 2

A. Array

Array atau Larik adalah variable yang digunakan untuk menyimpan data-data yang mempunyai tipe data yang sama. Di Java, Array dimulai dari index ke-0 sampai ke ke-n.
Ada dua macam cara deklarasi array dimensi 2 di Java, yaitu :

a) tipeData[][] namaArray;
   Misal: String[][] mhs;
b) tipeData namaArray[][];
   Misal: String mhs[][];

Ada dua macam cara membuat array dimensi 2 di Java, yaitu:

a) tipeData[][] namaArray = new tipeData[ukuranBaris][ukuranKolom];
   Misal:
   String[][] mhs = new String[5][2];
b) tipeData[][] namaArray = { {isiArray, isiArray,isiArray},{isiArray, isiArray,isiArray} };
   Misal:

```
String[][] mhs = {
    //  {kolom1,kolom2}
        {"NIM","NAMA"},         // baris ke-0
        {"17650123","David"},   // baris ke-1
        {"17650124","Ahmad"},   // baris ke-2
        {"17650125","Ratih"},   // baris ke-3
        {"17650126","Dina"}     // baris ke-4
    };
```

Percobaan:

Nama file : Array2.java

```
1   public class Array2
2   {   private String[][] Mahasiswa;
3       private int[][] data,hasil;
4
5       public void setMahasiswa(String[][] Mahasiswa) {
6           this.Mahasiswa = Mahasiswa;
7           Mahasiswa = null; // menghapus variable parameter dari memory
8       }
9
10      public String[][] getMahasiswa() {
11          return Mahasiswa;
12      }
13
14      public void setData(int[][] data) {
15          this.data = data;
16          data = null; // menghapus variable parameter dari memory
17      }
18
19      public int[][] getData() {
20          return data;
21      }
22
23      public void setPerkalianSkalar(int[][] data,int a) {
24          hasil = data;
25          int i, j;   // i = baris, j = kolom
26          for (i=0;i<data.length;i++)
27          {   for (j=0;j<data[i].length; j++)
28              {   hasil[i][j]=a*data[i][j];
29              }
30          }
31          data = null; // menghapus variable parameter dari memory
32      }
33
34      public int[][] getPerkalianSkalar() {
35          return hasil;
36      }
37
38      public void tampil(String a)
39      {   System.out.println(a);
40          a = null;
41      }
42
43      public void tampil(String data[][])
44      {   // mendeklarasikan baris dan kolom
45          int i, j;   // i = baris, j = kolom
46          for (i=0;i<data.length; i++)
47          {   for (j=0;j<data[i].length; j++)
48              {   System.out.print(data[i][j]+"   ");
49              }
50              System.out.println();
51          }
52          data = null; // menghapus variable parameter dari memory
53      }
54
55      public void tampil(int data[][])
56      {   int i, j;   // i = baris, j = kolom
57          for (i=0; i<data.length; i++)
58          {   for (j=0; j<data[i].length; j++)
59              {   System.out.print(data[i][j]+"   ");
60              }
61              System.out.println();
62          }
63          data = null; // menghapus variable parameter dari memory
64      }
65
66      public void hapus()
67      {   // menghapus variable private dari memory untuk optimasi program
68          Mahasiswa = null;
69          data = null;
70          hasil = null;
71      }
72  }
```

Nama file : MainArray2.java

```java
public class MainArray2 {
    public static void main(String[] args)
    {   Array2 ob = new Array2();
        String [][] Mhs = {
        //  {kolom0,kolom1}
            {"NIM","NAMA"},         // baris ke-0
            {"17650123","David"},   // baris ke-1
            {"17650124","Ahmad"},   // baris ke-2
            {"17650125","Ratih"},   // baris ke-3
            {"17650126","Dina"}     // baris ke-4
        };
        ob.tampil("Data Mahasiswa : ");
        ob.setMahasiswa(Mhs);
        ob.tampil(ob.getMahasiswa());
        ob.tampil("==================");
        int[][] data = {
        //  {kolom0,kolom1}
            {1,2},  // baris ke-0
            {3,4},  // baris ke-1
            {5,6}   // baris ke-2
        };
        int pengali = 2;
        ob.tampil("Data Matrik : ");
        ob.setData(data);
        ob.tampil(ob.getData());
        ob.tampil("Data Matrik X "+pengali+" : ");
        ob.setPerkalianSkalar(data,pengali);
        ob.tampil(ob.getPerkalianSkalar());

        // menghapus variable dari memory untuk optimasi program
        ob.hapus();
        Mhs = null;
        data = null;
        ob = null;
    }
}
```

MODUL 4

ARRAY II - TUGAS

1. Terdapat matrik

A	1	2
	3	5
	6	7

B	8	9
	10	11
	12	13

Dengan menggunakan objek dan enkapsulasi *(setter getter)*, jumlahkan matrik A dan B sehingga menghasilkan matrik

C	9	11
	13	16
	18	20

2. Dengan menggunakan objek dan enkapsulasi *(setter getter)*, kalikan matrik C pada no. 1 dengan 0.5! (Gunakan method *setter* dengan 2 parameter yaitu matrik C dan angka 0.5)

Catatan:

Soal dibuat berbeda, hari senin dan rabu penjumlahan, sedangkan hari selasa pengurangan.

MODUL 5

INHERITANCE

5.1 Inheritance

Inheritance merupakan proses pewarisan data dan method dari suatu class yang telah ada kepada suatu class baru. Class yang mewariskan disebut dengan **superclass / parent class / base class**, sedangkan class yang mewarisi (class yang baru) disebut dengan **subclass / child class / derived class**.

Subclass tidak dapat mewarisi anggota private dari superclass-nya. Dengan inheritance, class yang baru (subclass) akan mirip dengan class yang lama (superclass) namun memiliki karakteristik yang baru. Dalam Java, subclass hanya bisa memiliki satu superclass (single inheritance) sedangkan superclass bisa memiliki satu subclass atau lebih. Untuk menerapkan inheritance, gunakan statement "***extends***".

Keyword "***super***" digunakan oleh subclass untuk memanggil constructor, atribut dan method yang ada pada superclass-nya.

Contoh untuk memanggil constructor milik superclass-nya :

```
super()
super(parameter)
```

Contoh untuk memanggil atribut dan method milik superclass-nya :

```
super.namaAtribut
super.namaMethod(parameter)
```

5.2. Method Overriding

Overriding method adalah kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya. Contoh overriding method dapat dilihat pada subclass "Mobil" yang mendefinisikan kembali method keterangan() dan hapus() dari class "Kendaraan".

5.3 Input Data

Untuk menginputkan data dari keyboard ada 2 cara, yaitu :

a) Input dari mode console, yaitu dengan memanfaatkan class **BufferedReader** dan **InputStreamReader**.

Untuk bisa mengakses class BufferedReader maka perlu mengimpor dari package **java.io.*** dan menambahkan statemen **throws IOException** pada header method main.

Contoh :

```
import java.io.*;
class CobaInput1
{
 public static void main (String []args) throws IOException
 {
  BufferedReader br = new BufferedReader (new InputStreamReader(System.in));
  String nama, kota;
  System.out.print("Nama Anda : ");
  nama = br.readLine();
  System.out.print("Kota Asal : ");
  kota = br.readLine();
  System.out.println("Selamat Datang "+ nama +" dari "+ kota);
 }
}
```

b) Inputan dengan memanfaatkan class **JOptionPane**.

Untuk bisa menggunakan class JoptionPane, maka perlu mengimpor dari package **javax.swing.*** dan gunakan method **showInputDialog()** yang terdapat pada class JOptionPane.

Contoh :

```
import javax.swing.*;
class CobaInput2
{ public static void main (String [] args)
   {  String nama, kota;
      nama = JOptionPane.showInputDialog("Nama Anda :"); kota =
      JOptionPane.showInputDialog("Kota Asal :");
      System.out.println("Selamat Datang "+ nama +" dari "+ kota);
      System.exit(0);
   }
}
```

Catatan :

Semua data yang diinputkan dianggap sebagai suatu nilai String meskipun data tersebut hanya terdiri atas angka saja. Untuk menampung data yang diinputkan ke dalam variabel bertipe numerik (misal : int, long, double), maka data harus terlebih dahulu diubah ke tipe data numerik.

Contoh :

```
String sAngka;
int a = Integer.parseInt(sAngka);
long b = Long.parseLong(sAngka);
double c = Double.parseDouble(sAngka);
```

Silahkan coba source code dibawah ini:

Kendaraan.java

```
1  public class Kendaraan {
2      private String merk,warna;
3      protected String namaClass = "Kendaraan";
4
5      protected void setMerk(String merk) {
6          this.merk = merk;
7          merk = null; // menghapus variable parameter dari memory
8      }
9
10     protected String getMerk() {
11         return merk;
12     }
13
14     protected void setWarna(String warna) {
15         this.warna = warna;
16         warna = null; // menghapus variable parameter dari memory
17     }
18
19     protected String getWarna() {
20         return warna;
21     }
22
23     protected void tampil(String a)
24     {   System.out.println(a);
25         a = null;
26     }
27
28     protected String keterangan()
29     {   return ("Ini adalah class "+namaClass);
30     }
31
32     protected void hapus()
33     {   // menghapus variable private dari memory
34         warna = null;
35         merk = null;
36         namaClass = null;
37     }
38 }
```

Mobil.java

```java
public class Mobil extends Kendaraan {
    private long harga;
    protected String namaClass = "Mobil";

    protected void setHarga(String harga) {
        this.harga = Long.parseLong(harga);
        harga = null;
    }

    protected long getHarga() {
        return harga;
    }

    protected String keterangan()
    {   return (namaClass+" : Ini adalah class "+namaClass);
    }

    public String keterangan_Kendaraan()
    {   // mengakses atribut/variable & method parent (class Kendaraan)
        return super.namaClass+" : "+super.keterangan();
    }

    protected void hapus()
    {   // menghapus variable private dari memory
        harga = 0;
        // menghapus variable private parent (class Kendaraan)
        super.hapus();
    }
}
```

MainMobil.java

```java
import javax.swing.JOptionPane;

public class MainMobil extends Mobil {
    public static void main (String []args) {
        MainMobil ob = new MainMobil();
        String merk,warna,harga ;

        merk = JOptionPane.showInputDialog("Merk Mobil :");
        ob.setMerk(merk);
        ob.tampil("Merk Mobil : "+ob.getMerk());

        warna = JOptionPane.showInputDialog("Warna Mobil :");
        ob.setWarna(warna);
        ob.tampil("Warna Mobil : "+ob.getWarna());

        harga = JOptionPane.showInputDialog("Harga Mobil :");
        ob.setHarga(harga);
        ob.tampil("Harga Mobil : "+ob.getHarga());

        ob.hapus();
        merk=null;
        warna=null;
        harga=null;
        ob = null;

        System.exit(0);
    }
}
```

MODUL 5
LATIHAN SOAL

1. Buatlah aplikasi rental VCD sederhana sebagai berikut:

Masukkan data VCD Film sebanyak X kali

Judul : judul film.

Aktor : nama-nama aktor di dalam film tersebut.

Sutradara : nama sutradara film tersebut.

Publisher : yang memproduksi film tersebut.

Kategori : SU = Semua Umur, D = Dewasa, R = Remaja, A = Anak-anak

Stok : jumlah stok VCD film tersebut.

Judul, Aktor, Sutradara, Publisher, Kategori dan Stok dilooping sebanyak X kali.

Desainlah aplikasi rental VCD tersebut dengan konsep inheritance dan tentukan parent class serta child class-nya. Setelah itu, implementasikan class-class yang telah didesain dengan membuat program sederhana yang memiliki fasilitas entri data VCD film dan melihat daftar VCD film yang telah dientrikan.

MODUL 6
CONSTRUCTOR

6.1 Constructor

Constructor adalah method yang secara otomatis dipanggil/dijalankan pada saat sebuah class diinstansiasi. Jika dalam sebuah class tidak terdapat constructor maka secara otomatis Java akan membuatkan sebuah default constructor. Nama constructor harus sama dengan nama class dan tidak boleh memiliki tipe return value.

Sama halnya dengan method, constructor dapat memiliki satu atau banyak parameter maupun tanpa parameter. Constructor biasanya digunakan untuk memberi nilai awal dari atribut-atribut class tersebut.

Keyword "***super***" digunakan oleh subclass untuk memanggil constructor, atribut dan method yang ada pada superclass-nya.

Contoh untuk memanggil constructor milik superclass-nya :

```
super()
super(parameter)
```

Contoh untuk memanggil atribut dan method milik superclass-nya :

```
super.namaAtribut
super.namaMethod(parameter)
```

6.2 Static Method dan Static Property vs Variable Instant

Static property dan static method adalah property (variabel) dan method (function) yang melekat kepada class, bukan kepada objek. Konsep static property memang 'agak keluar' dari konsep objek sebagai tempat melakukan proses, karena sebenarnya class hanya merupakan '*blueprint*' saja.

Untuk membuat static property dan static method, kita menambahkan keyword ***'static'*** setelah penulisan akses level property atau method, seperti contoh berikut:

```
public static String pemilik;
public static void keterangan_pemilik(String pemilik) { ..... }
```

Dalam contoh diatas menggunakan hak akses public, tetapi kita juga bisa menggunakan hak akses lain seperti private dan protected untuk static property dan static method.

Karena static property dan static method adalah milik class, maka kita tidak perlu membuat objek untuk mengaksesnya, tapi langsung menyebutkan nama class, berikut adalah contoh pengaksesan static property dan static method dari sebuah class.

```
ConstructorMotor.pemilik = "Ahmad Afif";
ConstructorMotor.keterangan_pemilik(ConstructorMotor.pemilik);
```

Variabel-variabel yang dideklarasikan dengan tidak menggunakan keyword "*static*", maka variabel tersebut disebut dengan instant variabel (atau variabel instant).

a) Jika sebuah variable merupakan variable instant, maka masing-masing objek dari class tersebut akan memiliki variable yang sama dengan variable instant tersebut, perubahan nilai yang terjadi pada variable instant di satu objek tidak akan berpengaruh pada variable instant di objek yang berbeda.
b) Jika sebuah variable merupakan variable static (pada suatu class), maka variabel static tersebut adalah variabel yang sama di semua objek dari class tersebut. Sehingga perubahan nilai pada variabel static tersebut di suatu objek akan berpengaruh juga terhadap objek yang lainnya.
c) Nilai suatu variabel static akan selalu sama untuk semua *instant of class* (atau objek) dari sebuah class.

Sebuah variable dideklarasikan static apabila variable tersebut bersifat global bagi semua objek dari suatu class. Contohnya adalah variable yang menyimpan nilai jumlah objek yang telah dibuat.

Silahkan coba source code dibawah ini :

ConstructorKendaraan.java

```java
public class ConstructorKendaraan {
    private String merk;  // ini adalah instant variable
    private static String pemilik; // ini adalah static variable

    // ini adalah constructor tanpa parameter
    protected ConstructorKendaraan() {
        merk = null;
    }

    // overloading constructor
    // ini adalah constructor dengan parameter merk
    protected ConstructorKendaraan(String merk) {
        this.merk = merk;
        merk = null;
    }

    protected void setMerk(String merk) {
        this.merk = merk;
    }

    protected String getMerk() {
        return merk;
    }

    // ini adalah static method
    protected static void setPemilik(String pemilik) {
        ConstructorKendaraan.pemilik = pemilik;
    }

    // ini adalah static method
    protected static String getPemilik() {
        return ConstructorKendaraan.pemilik;
    }

    protected void tampil(String a)
    {   System.out.println(a);
        a = null;
    }

    protected void hapus()
    {   // menghapus variable private dari memory
        merk = null;
        pemilik = null;
    }
}
```

ConstructorMotor.java

```java
// ConstructorMotor turunan dari class ConstructorKendaraan
public class ConstructorMotor extends ConstructorKendaraan {
    private String warna; // ini adalah instant variable

    // ini adalah constructor dengan parameter merk & warna
    protected ConstructorMotor(String merk,String warna){
        // memanggil constructor parent (class ConstructorKendaraan)
        // dengan keyword super dan parameter merk
        super(merk);

        this.warna = warna;

        // menghapus variable parameter dari memory
        merk = null;
        warna = null;
    }

    protected String getWarna() {
        return warna;
    }

    protected void hapus()
    {   // menghapus variable private dari memory
        warna = null;
        // memanggil method hapus class parent (class ConstructorKendaraan)
        // dengan keyword super
        super.hapus();
    }
}
```

MainConstructorMotor.java

```java
public class MainConstructorMotor {
    public static void main(String args[])
    {   String pemilik = "Ahmad Afif";
        String merk = "Honda";
        String warna = "Merah";

        // cara akses static variable dan static method dapat dengan
        // memanggil class (ConstructorKendaraan) secara langsung
        ConstructorKendaraan.setPemilik(pemilik);
        System.out.println("Pemilik Kendaraan = "+ConstructorKendaraan.getPemilik());
        System.out.println("==================================");

        // variable merk menjadi parameter Constructor pada saat
        // instansiasi/membuat objek baru (ob)
        ConstructorKendaraan ob = new ConstructorKendaraan(merk);
        ob.tampil("Merk Kendaraan = "+ob.getMerk());
        // cara akses static variable dan static method dapat juga dengan
        // objek (ob) pada method getPemilik()
        ob.tampil("Pemilik Kendaraan = "+ob.getPemilik());
        System.out.println("==================================");

        // instansiasi/membuat objek baru (ob2) tanpa parameter
        ConstructorKendaraan ob2 = new ConstructorKendaraan();
        // bandingkan akses untuk menampilkan nilai pada instant variable (merk)
        // dan static variable (pemilik) melalui method getter setelah membuat
        // objek baru "ob2", dimana sebelumnya juga sudah membuat objek "ob".
        // instant variable (merk) nilainya akan hilang,
        // sedangkan static variable (pemilik) nilainya tidak hilang/berubah
        ob2.tampil("Merk Kendaraan (instant variable) = "+ob2.getMerk());
        ob2.tampil("Pemilik Kendaraan (static variable) = "+ob2.getPemilik());
        System.out.println("==================================");

        // variable merk dan warna menjadi parameter Constructor pada saat
        // instansiasi/membuat objek baru (ob3)
        ConstructorMotor ob3 = new ConstructorMotor(merk,warna);
        ob3.tampil("Merk Motor = "+ob3.getMerk());
        ob3.tampil("Warna Motor = "+ob3.getWarna());
        ob3.tampil("Pemilik Motor = "+ob3.getPemilik());

        pemilik = null;
        merk = null;
        warna = null;
        ob.hapus();
        ob = null;
        ob2 = null;
        ob3 = null;
    }
}
```

MODUL 6

LATIHAN SOAL

1. Buatlah aplikasi Data Mahasiswa sebagai berikut:
   a) Masukkan Nama Universitas! Buat static variable dan static method *setter getter*-nya!
   b) Masukkan data mahasiswa
      1) NIM
      2) NAMA
      3) ALAMAT
      4) JURUSAN : 61 = MATEMATIKA, 62 = BIOLOGI, 63 = KIMIA, 64 = FISIKA, 65 = TEKNIK INFORMATIKA, 66 = TEKNIK ARSITEKTUR

      Apakah Anda ingin memasukkan data lagi ? (Y) Ya ; (T) Tidak

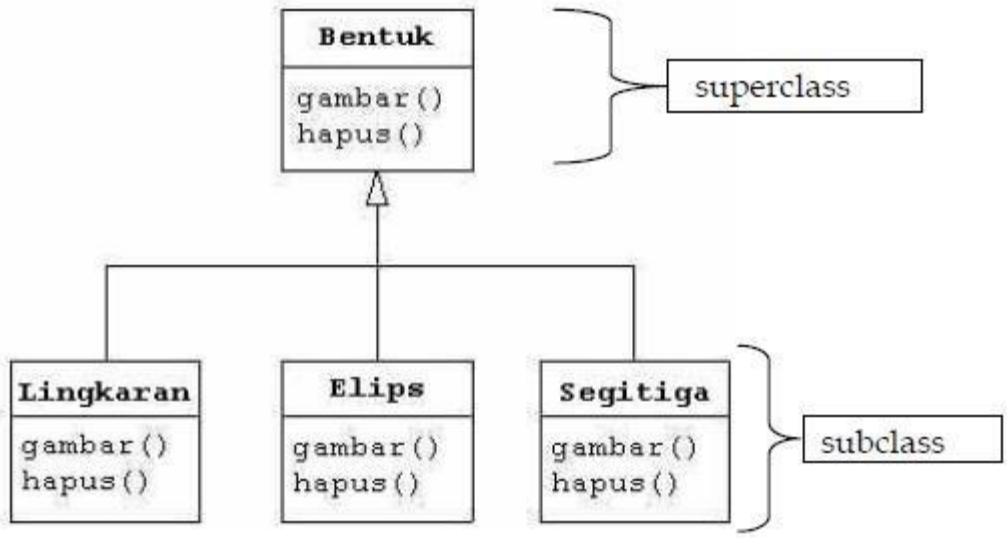
      Jika *user* memasukkan "Y", maka lanjut untuk memasukkan data mahasiswa lagi dan jika *user* memasukkan data "T", maka aplikasi berhenti.

   Desainlah aplikasi Data Mahasiswa tersebut dengan konsep enkapsulasi, constructor dan inheritance (tentukan parent class dan child class). Setelah itu, implementasikan class-class yang telah didesain dengan membuat program sederhana yang memiliki fasilitas entri data Mahasiswa dan melihat daftar Mahasiswa yang telah dimasukkan.

MODUL 7
POLIMORFISME

7.1 Polimorfisme

Polimorfisme mempunyai makna sesuatu yang memiliki banyak bentuk, yaitu memiliki nama sama, tetapi memiliki kelakuan (*behaviour*) yang berbeda.



Perhatikan gambar diagram di atas !

Class Bentuk yang merupakan class induk (*superclass*) dari class Lingkaran, Elips dan Segitiga mempunyai method **gambar()** dan **hapus()**. Class-class anak (*subclass*) juga mempunyai method **gambar()** dan **hapus()**. Meskipun keempat class tersebut mempunyai nama method yang sama, tetapi isi (source code/yang dilakukan/output) dari masing-masing method tersebut berbeda.

Jika kita menginginkan sebuah objek yang dapat memanggil setiap method (yaitu method gambar & hapus) yang ada pada setiap class (pada superclass maupun subclass), maka gunakanlah teknik **Polimorfisme**. Polimorfisme hanya berlaku pada method dan tidak berlaku untuk atribut.

Untuk mendapatkan operasi Polimorfisme dari suatu method, maka method tersebut haruslah merupakan method yang ada di class induk (lihat diagram diatas bahwa method gambar() dan hapus(), selain terdapat di class-class turunan class Bentuk, juga terdapat di class Bentuk).

Contoh source code implementasi Polimorfisme :

Bentuk.java

```
1  public class Bentuk {
2      protected void gambar() {
3        System.out.println("superclass -> Menggambar ");
4       }
5
6       protected void hapus() {
7        System.out.println("superclass -> Menghapus Gambar");
8       }
9  }
```

Lingkaran.java

```
1  public class Lingkaran extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Lingkaran");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Lingkaran");
8      }
9  }
```

Segitiga.java

```
1  public class Segitiga extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Segitiga");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Segitiga");
8      }
9  }
```

Elips.java

```
1  public class Elips extends Bentuk {
2      protected void gambar() {
3       System.out.println("subclass -> Menggambar Elips");
4      }
5
6      protected void hapus() {
7       System.out.println("subclass -> Menghapus Gambar Elips");
8      }
9
10 }
```

Cetakgambar.java

```
1  public class Cetakgambar extends Bentuk {
2
3      private void tampil(Bentuk[] obj) {
4          // Polimorfisme
5          // Memanggil method yang sama yaitu method gambar() dan hapus()
6          // pada masing-masing class
7          for (int i=0;i<obj.length;i++)
8           {  obj[i].gambar();
9              obj[i].hapus();
10             System.out.println("==================");
11          }
12     }
13
14     public static void main (String []args) {
15          Bentuk[] obj = {  new Lingkaran(),
16                            new Elips(),
17                            new Segitiga()
18                         };
19          Cetakgambar cetak = new Cetakgambar();
20
21          // Menampilkan method gambar() & hapus() pada class Bentuk (superclass)
22          cetak.gambar();
23          cetak.hapus();
24          System.out.println("==================");
25
26          // Overriding
27          // Menumpuk method gambar() & hapus() pada class Bentuk (superclass)
28          // dengan method gambar() & hapus() pada subclass-nya
29          // yaitu class Lingkaran, Elips dan Segitiga
30          cetak.tampil(obj);
31      }
32 }
```

Pada class Cetakgambar terdapat variabel/objek *obj* yang bertipe *class Bentuk*. Maka dapat dikatakan bahwa variabel *obj* dapat berperan sebagai *Lingkaran*, *Elips*, atau *Segitiga*. Hal ini didasarkan bahwa pada kenyataannya setiap objek dari class Induk (*superclass*) dapat berperan sebagai class-class turunannya sebagaimana sepeda motor adalah kendaraan, pelajar dan mahasiswa adalah orang/manusia.

Perhatikan bahwa Polimorfisme tidak sama dengan overloading !!!

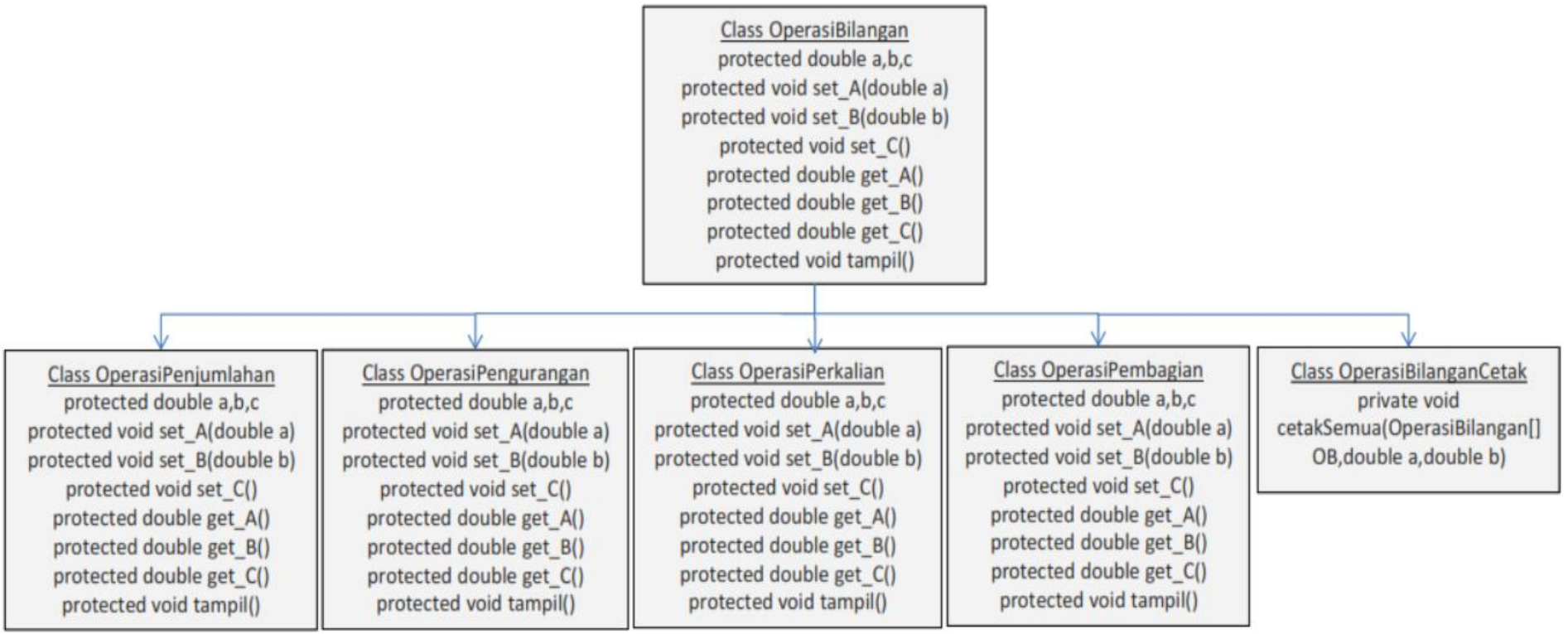
7.2. Method Overriding

Overriding method adalah kemampuan dari subclass untuk memodifikasi method dari superclass-nya, yaitu dengan cara menumpuk (mendefinisikan kembali) method superclass-nya. Contoh overriding method dapat dilihat pada class-class turunan dari class Bentuk yang mendefinisikan kembali method gambar() dan method hapus() dari class induknya.

MODUL 7

LATIHAN SOAL

1. Buatlah aplikasi penjumlahan, pengurangan, perkalian dan pembagian menggunakan konsep polimorfisme dengan memasukkan parameter bilangan A = 10.5 dan bilangan B = 0.5 !



MODUL 8
FINAL & ABSTRACT

8.1 Final Method dan Final Class

Keyword "final" digunakan untuk mencegah suatu class diturunkan atau suatu method di overridding atau suatu variable diubah.

Sintaks penggunaan keyword "*final*" pada class :

```
akses_modifier final namaClass
```

Sintaks penggunaan keyword "*final*" pada method :

```
akses_modifier final tipeMethod namaMethod()
{
      ..... // definisi method
}
```

Sintaks penggunaan keyword "*final*" pada variable (konstanta) :

```
akses_modifier final tipeData namaVariable
```

8.2. Abstract Class dan Abstract Method

Abstract Class adalah sebuah class yang tidak bisa di-*instansiasi* (tidak bisa dibuat menjadi objek) dan berperan sebagai '*kerangka dasar'* bagi class turunannya. Di dalam *abstract class* umumnya akan memiliki *abstract method*. Cara untuk membuat sebuah *abstract class* adalah :

```
akses_modifier abstract class namaClassAbstrak
{
      ......... // definisi class
}
```

Abstract Method adalah sebuah '*method dasar*' yang harus diimplementasikan ulang di dalam class anak (*child class*). *Abstract method* ditulis tanpa isi dari *method*, melainkan hanya '**signature'**-nya saja. **Signature** dari sebuah *method* adalah bagian method yang terdiri dari nama method dan parameternya (jika ada). Cara untuk membuat sebuah *abstract method* adalah :

```
abstract void move(double x, double y);
```

Abstract class digunakan di dalam *inheritance* (*pewarisan class*) untuk '*memaksakan'* implementasi method yang sama bagi seluruh class yang diturunkan dari *abstract class*. *Abstract class* digunakan untuk membuat struktur logika penurunan di dalam pemrograman objek.

Java memiliki aturan-aturan dalam penggunaan method abstrak dan class abstrak sebagai berikut:

a) Class yang di dalamnya terdapat abstract method harus dideklarasikan sebagai abstract class.
b) Abstract class tidak dapat diinstansi, tetapi harus di turunkan.
c) Abstract class tidak dapat diinstansi (menjadi objek dari class abstract), tetapi kita dapat mendeklarasikan suatu variable yang bertipe abstract class dan membuat instansi dari variabel tersebut yang bertipe class turunan dari abstract class tersebut (*teknik polymorphism*).
d) Sebuah class dapat dideklarasikan sebagai abstract class meskipun class tersebut tidak memiliki abstract method.
e) Abstract method tidak boleh mempunyai body method dan demikian juga sebaliknya bahwa method yang tidak ditulis body methodnya maka harus dideklarasikan sebagai abstract method.

Contoh source code :

Laptop.java

```
1  // abstract class
2  public abstract class Laptop {
3      // attribut class
4      protected String merk,pemilik;
5
6      // abstract method
7      protected abstract void setMerk(String merk);
8      protected abstract String getMerk();
9      protected abstract void setPemilik(String pemilik);
10     protected abstract String getPemilik();
11     protected abstract void tampil();
12     protected abstract void hapus();
13
14     // method "biasa"
15     protected void hidupkanLaptop(){
16        System.out.println("Hidupkan Laptop");
17    }
18 }
```

LaptopAsus.java

```java
public class LaptopAsus extends Laptop {
    // constructor
    LaptopAsus(String merk)
    {   setMerk(merk);
        merk = null;
    }

    protected void setMerk(String merk) {
        this.merk = merk;
        merk = null;
    }
    protected String getMerk() {
        return merk;
    }

    protected void setPemilik(String pemilik)
    {   this.pemilik = pemilik;
        pemilik = null;

    }
    protected String getPemilik()
    {   return this.pemilik;
    }

    protected void tampil()
    {   System.out.println(getPemilik()+" memiliki laptop merk "+getMerk());
    }
    protected void hapus()
    {   merk = null;
        pemilik = null;
    }
}
```

LaptopDell.java

```java
public class LaptopDell extends Laptop {
    // constructor
    LaptopDell(String merk)
    {   setMerk(merk);
        merk = null;
    }

    protected void setMerk(String merk) {
        this.merk = merk;
        merk = null;
    }
    protected String getMerk() {
        return merk;
    }

    protected void setPemilik(String pemilik)
    {   this.pemilik = pemilik;
        pemilik = null;

    }
    protected String getPemilik()
    {   return this.pemilik;
    }

    protected void tampil()
    {   System.out.println(getPemilik()+" memiliki laptop merk "+getMerk());
    }
    protected void hapus()
    {   merk = null;
        pemilik = null;
    }
}
```

LaptopToshiba.java

```java
public class LaptopToshiba extends Laptop {
    // constructor
    LaptopToshiba(String merk)
    {   setMerk(merk);
        merk = null;
    }

    protected void setMerk(String merk) {
        this.merk = merk;
        merk = null;
    }
    protected String getMerk() {
        return merk;
    }

    protected void setPemilik(String pemilik)
    {   this.pemilik = pemilik;
        pemilik = null;

    }
    protected String getPemilik()
    {   return this.pemilik;
    }

    protected void tampil()
    {   System.out.println(getPemilik()+" memiliki laptop merk "+getMerk());
    }
    protected void hapus()
    {   merk = null;
        pemilik = null;
    }
}
```

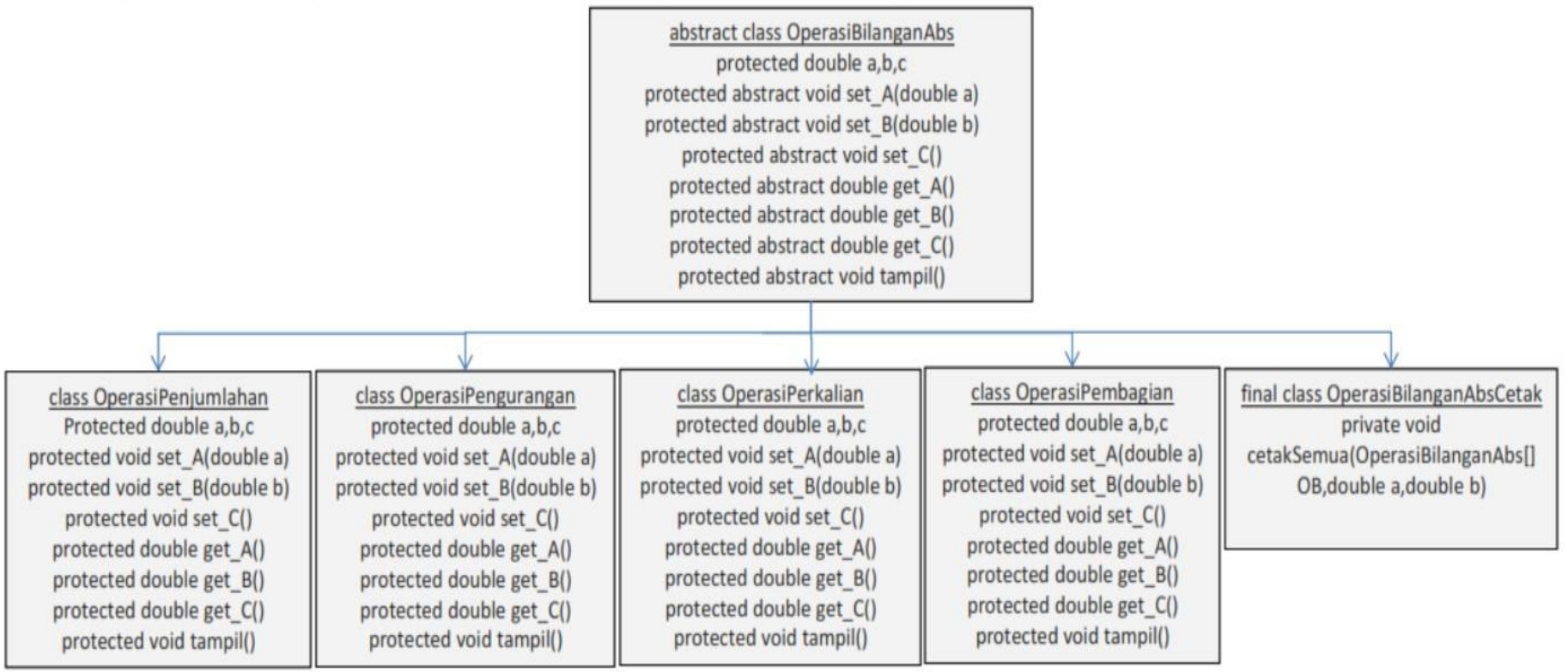
LaptopCetak.java

```java
// final class
public final class LaptopCetak {
    // final variable/konstanta
    private final String barang = "Laptop";
    // final method
    private final void cetak(Laptop[] ob, String pemilik) {
        System.out.println("Nama Barang : "+barang);
        System.out.println("");
        // polimorfisme
        for (int i=0;i<ob.length;i++)
         {  ob[i].getMerk();
            ob[i].setPemilik(pemilik);
            ob[i].getPemilik();
            ob[i].tampil();
            ob[i].hapus();
            System.out.println("########################");
         }
        ob = null;
        pemilik = null;
    }

    public static void main (String []args) {
        String pemilik ="Ahmad";
        Laptop[] ob =  {  new LaptopAsus("Asus"),
                          new LaptopDell("Dell"),
                          new LaptopToshiba("Toshiba")
                       };

        LaptopCetak abc = new LaptopCetak();
        abc.cetak(ob,pemilik);

        pemilik = null;
        ob = null;
        abc = null;
     }
}
```

MODUL 8

LATIHAN SOAL

1. Buatlah aplikasi penjumlahan, pengurangan, perkalian dan pembagian menggunakan konsep abstract, final dan polimorfisme dengan memasukkan parameter bilangan A = 6.5 dan bilangan B = 0.5 !



MODUL 9
INTERFACE

9.1 Pengertian Object Interface

Secara sederhana, Object Interface adalah sebuah ‘kontrak’ atau perjanjian implementasi method. Bagi *class* yang menggunakan *object interface, class* tersebut harus mengimplementasikan ulang seluruh *method* yang ada di dalam *interface*. Dalam pemrograman objek, penyebutan *object interface* sering disingkat dengan ‘*Interface’* saja.

Jika anda telah mempelajari *abstract class*, maka *interface* bisa dikatakan sebagai bentuk lain dari *abstract class*. Walaupun secara konsep teoritis dan tujuan penggunaannya berbeda. Sama seperti *abstract class*, *interface* juga hanya berisi *signature* dari *method*, yakni hanya nama *method* dan *parameter*nya saja (jika ada). Isi dari method akan dibuat ulang di dalam *class* yang menggunakan *interface*.

Jika kita menganggap *abstract class* sebagai ‘*kerangka’* atau ‘*blue print’* dari class-class lain, maka *interface* adalah implementasi method yang harus ‘*tersedia’* dalam sebuah objek. *Interface* tidak bisa disebut sebagai ‘*kerangka’ class*.

Jika terdapat *class* komputer, *interface* bisa dicontohkan dengan ‘*mouse’*, atau ‘*keyboard’*. Di dalam *interface mouse*, kita bisa membuat method seperti *klik_kiri(), klik_kanan(),* dan *double_klik()*. Jika class laptop ‘*menggunakan’* interface *mouse*, maka class tersebut harus membuat ulang method *klik_kiri(), klik_kanan(),* dan *double_klik()*.

9.2. Deklarasi interface

Untuk mendeklarasikan sebuah interface gunakan sintaks :

```
interface namaInterface
{     ........ //isi dari interface namaInterface
}
```

Berikut ini adalah contoh membuat interface mouse:

```
public interface mouse
{  public void klik_kanan();
   public void klik_kiri();
   public void double_klik();
}
```

9.3. Implementasi interface

Cara menggunakan suatu interface adalah dengan mengimplementasikan interface tersebut pada class yang menggunakannya. Selain itu, anda juga harus mendefinisikan secara detail method-method yang ada pada interface tersebut.

```
public class NamaClass implements namaInterface
{     ........ //method dan isi method pada namaInterface
}
```

9.4. Perbedaan Interface dengan Abstract Class

Salah satu yang membedakan interface dengan abstract class adalah kita tidak bisa membuat method normal / biasa di dalam Interface.

Perbedaan lain antara Interface dengan Abstract Class adalah: Sebuah class bisa menggunakan lebih dari 1 interface, sedangkan untuk abstract class, kita hanya bisa menggunakan
1 abstract class dalam sekali penurunan class.

9.5. Fungsi Interface

Interface lebih berperan untuk *menyeragamkan method*. Ia tidak masuk kedalam struktur class seperti *abstract class*. Jika kita menggunakan *abstract class komputer* sebagai '*konsep class'* untuk kemudian diturunkan kepada class lain seperti *class laptop, class pc*, dan *class netbook*, maka interface hanya '*penyedia method*'. *Interface* tidak termasuk kedalam pewarisan class.

Contoh source code:

Mouse.java

```
1 public interface Mouse {
2    String jenis = "Laser Mouse";
3    // Method Interface harus public
4    public void klik_kanan();
5    public void klik_kiri();
6    // Tidak boleh ada method normal / biasa
7    // yang terdapat isi method-nya
8 }
```

MouseBaru.java

```
1 // Interface dapat diturunkan (Inherit)
2 public interface MouseBaru extends Mouse {
3     public void double_klik();
4 }
```

Keyboard.java

```
1 public interface Keyboard {
2     public void tekan_enter();
3 }
```

Komputer.java

```
1  public class Komputer implements MouseBaru {
2
3     public void klik_kanan(){
4       System.out.println("Mouse Komputer : Klik Kanan...");
5     }
6
7     public void klik_kiri(){
8       System.out.println("Mouse Komputer : Klik Kiri...");
9     }
10
11    public void double_klik(){
12      System.out.println("Mouse Komputer : Double Klik...");
13    }
14
15 }
```

PC.java

```
1  public class PC extends Komputer implements Keyboard {
2
3     public void tekan_enter(){
4       System.out.println("Keyboard PC : Tekan Enter...");
5     }
6
7  }
```

Laptop.java

```
// sebuah class dapat menggunakan banyak interface
public class Laptop extends Komputer implements MouseBaru, Keyboard {

    public void jenis(){
      System.out.println("Jenis Mouse Laptop : "+jenis);
    }

    public void klik_kanan(){
      System.out.println("Mouse Laptop : Klik Kanan...");
    }

    public void klik_kiri(){
      System.out.println("Mouse Laptop : Klik Kiri...");
    }

    public void double_klik(){
      System.out.println("Mouse Laptop : Double Klik...");
    }

    public void tekan_enter(){
      System.out.println("Keyboard Laptop : Tekan Enter...");
    }
}
```

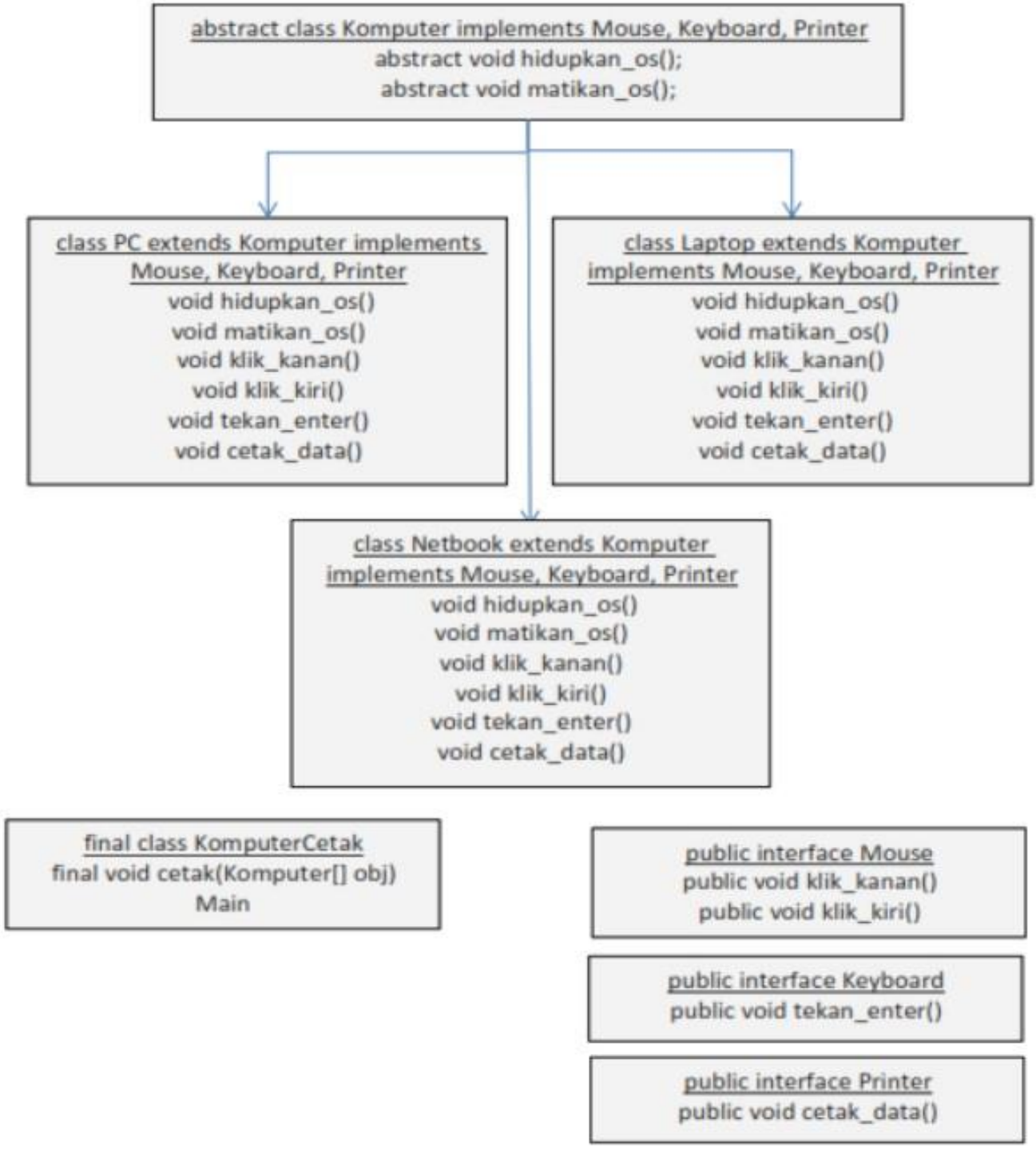
Cetak.java

```
public class Cetak {
    public static void main (String []args) {
        System.out.println("Komputer : ");
        Komputer komputer_baru = new Komputer();
        komputer_baru.klik_kanan();
        komputer_baru.klik_kiri();
        komputer_baru.double_klik();
        System.out.println("==================");

        System.out.println("PC : ");
        PC pc_baru = new PC();
        pc_baru.tekan_enter();
        System.out.println("==================");

        System.out.println("Laptop : ");
        Laptop laptop_baru = new Laptop();
        laptop_baru.jenis();
        laptop_baru.klik_kanan();
        laptop_baru.klik_kiri();
        laptop_baru.double_klik();
        laptop_baru.tekan_enter();
    }
}
```

MODUL 9

LATIHAN SOAL

a. Masukkanlah source code pada modul 9 ke dalam konsep interface, abstract, final dan polimorfisme !



MODUL 10
Graphical User Interface (GUI) Java NetBeans
& Aplikasi CRUD (Create, Read, Update, Delete)

10.1 NetBeans

NetBeans IDE adalah GUI Editor yang memudahkan programmer untuk melakukan pembelajaran dan pengembangan software atau dalam tahapan membuat software atau aplikasi, yang sebelumnya banyak GUI Editor dalam bahasa pemrograman seperti notepad atau text editor, yang sebenarnya jika kita memakai notepad atau text editor maka pengetahuan kita akan lebih berkembang dibanding memakai GUI Editor.

NetBeans mengacu pada dua hal, yakni platform untuk pengembangan aplikasi desktop java dan sebuah Integrated Development Environment (IDE) yang dibangun menggunakan platform NetBeans.

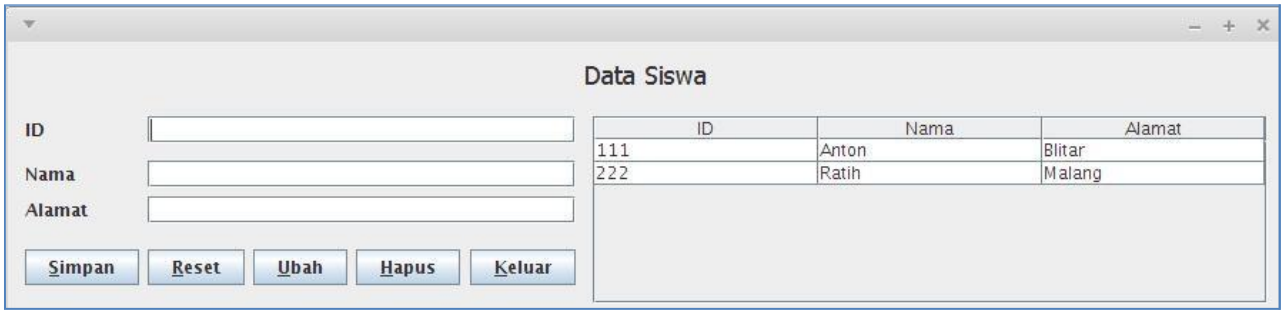
NetBeans IDE adalah sebuah lingkungan pengembangan untuk pemrogram, menulis, mengompilasi, mencari kesalahan dan menjalankan program. NetBeans IDE ditulis dalam bahasa Java, namun dapat mendukung bahasa pemrograman lain. Terdapat banyak modul untuk memperluas NetBeans IDE. NetBeans IDE adalah sebuah produk bebas dengan tanpa batasan.

NetBeans adalah Integrated Development Environment (IDE) berbasiskan Java yang berawal pada tahun 1997 yang diawali dari Xelfi, sebuah proyek mahasiswa di bawah bimbingan Fakultas Matematika dan Fisika Universitas Charles, Praha dan akhirnya Sun Microsystems membelinya. Pada tahun 1999, dikembangkan NetBeans Integrated Development Environment (IDE) berbasiskan Java dari Sun Microsystems yang berjalan di atas Swing. Swing sebuah teknologi Java untuk pengembangan aplikasi Desktop yang dapat bejalan di berbagai macam platforms seperti Windows, Linux, Mac OS X and Solar dengan kata lain, java dengan NetBeans dapat dikatakan sebagai pemrograman multi platforms karena fleksibel dengan segala sistem operasi.

Pada tahun 2000, sun menjadikan NetBeans sebagai software development yang Open Source, dengan kata lain software ini gratis tanpa biaya karena software ini di bawah pengembangan bersama. Software NetBeans dapat di download di http://www.netbeans.com atau juga bisa di download di situs yang lain.

10.2 Aplikasi CRUD

Aplikasi CRUD sederhana ini memiliki logika yang sangat sederhana, yaitu pertama saat aplikasi dijalankan, maka aplikasi CRUD akan meload semua data dan menampilkannya di tabel, serta mengisi text field dengan data – data yang sama pada tabel yang sudah siap untuk diubah. Selanjutnya jika melakukan perubahan berarti memerintahkan aplikasi untuk mengupdate data ke database, sedangkan jika melakukan penambahan data maka aplikasi akan menambah data tersebut dengan perintah insert ke database. Dan terakhir melakukan perintah delete, baris yang dipilih pada tabel akan dihapus secara permanent. Aplikasi CRUD pada data siswa seperti gambar di bawah ini.

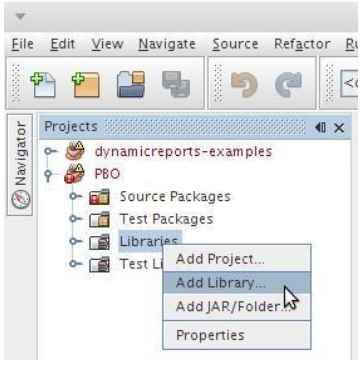


10.3 Latihan

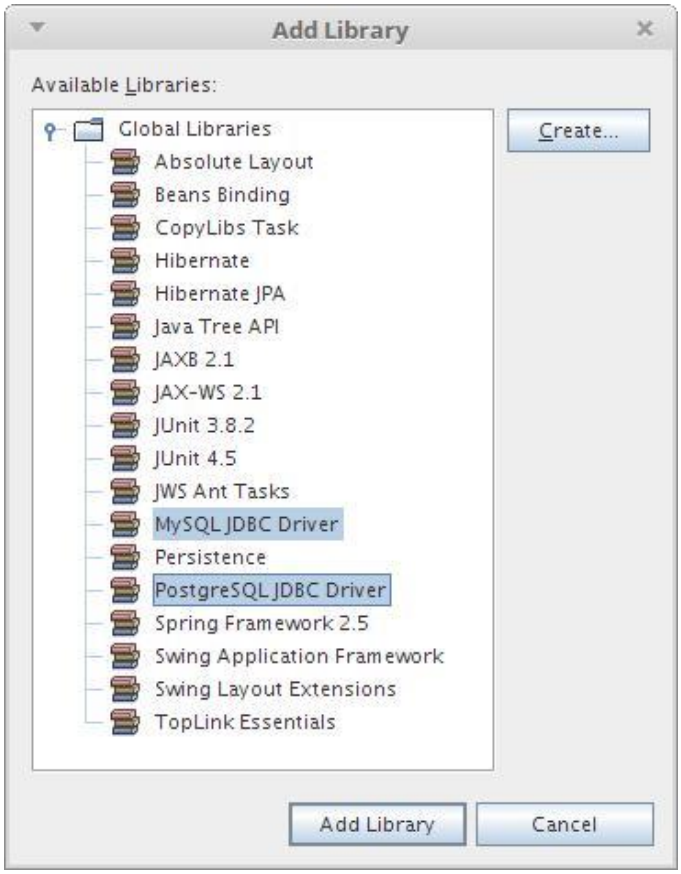
A. Buat database mysql dengan nama crud+nim misalnya ‘crud15650001’ dan buat tabel ‘siswa’!

```
CREATE TABLE siswa (
  id varchar(20) NOT NULL,
  nama varchar(100) DEFAULT NULL,
  alamat text,
  PRIMARY KEY (id)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

insert  into siswa(id,nama,alamat) values
('111','Anton','Blitar'),('222','Ratih','Malang');
```

B. Buka NetBeans dan klik menu “**Window > Projects”.** Pilih salah satu project lalu klik kanan “**Libraries > Add Library”**.



Pilih “**MySQL JDBC Driver”** untuk driver MySQL dan pilih “**PostgreSQL JDBC Driver**’ untuk driver PostgreSQL. Klik tombol “**Add Library**”.



C. Source code koneksi MySQL di NetBeans :

Nama file : KoneksiMysql.java

```
1  package crud;
2  import java.sql.Connection;
3  import java.sql.DriverManager;
4  import java.sql.SQLException;
5  public class KoneksiMysql {
6     private Connection connect;
7     private String driverName = "com.mysql.jdbc.Driver"; // Driver Untuk Koneksi Ke MySQL
8     private String jdbc = "jdbc:mysql://";
9     private String host = "localhost:"; // Bisa Menggunakan IP Anda, Cnth : 192.168.100.100
10    private String port = "3306/"; // Port ini port MySQL
11    private String database = "crud15650001"; // Ini Database yang akan digunakan
12    private String url = jdbc + host + port + database;
13    private String username = "root"; // username default mysql
14    private String password = "";
15    public Connection getKoneksi() throws SQLException {
16      if (connect == null) {
17        try {
18          Class.forName(driverName);
19          System.out.println("Class Driver Ditemukan");
20          try {
21            connect = DriverManager.getConnection(url, username, password);
22            System.out.println("Koneksi Database Sukses");
23          } catch (SQLException se) {
24            System.out.println("Koneksi Database Gagal : " + se);
25            System.exit(0);
26          }
27        } catch (ClassNotFoundException cnfe) {
28          System.out.println("Class Driver Tidak Ditemukan, Terjadi Kesalahan Pada : " + cnfe);
29          System.exit(0);
30        }
31      }
32      return connect;
33    }
34 }
```

Untuk database Postgresql :

Nama file : KoneksiPostgresql.java

```java
package crud;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
public class KoneksiPostgresql {
   private Connection connect;
   private String driverName = "org.postgresql.Driver"; // Driver Untuk Koneksi Ke PostgreSQL
   private String jdbc = "jdbc:postgresql://";
   private String host = "localhost:"; // Host ini Bisa Menggunakan IP Anda, Contoh : 192.168.100.100
   private String port = "5432/"; // Port Default PostgreSQL
   private String database = "crud15650001"; // Ini Database yang akan digunakan
   private String url = jdbc + host + port + database;
   private String username = "postgres"; //
   private String password = "";
   public Connection getKoneksi() throws SQLException {
     if (connect == null) {
       try {
         Class.forName(driverName);
         System.out.println("Class Driver Ditemukan");
         try {
           connect = DriverManager.getConnection(url, username, password);
           System.out.println("Koneksi Database Sukses");
         } catch (SQLException se) {
           System.out.println("Koneksi Database Gagal : " + se);
           System.exit(0);
         }
       } catch (ClassNotFoundException cnfe) {
         System.out.println("Class Driver Tidak Ditemukan, Terjadi Kesalahan Pada : " + cnfe);
         System.exit(0);
       }
     }
     return connect;
   }
}
```

D. Source code class CRUD

Nama file : CRUD.java

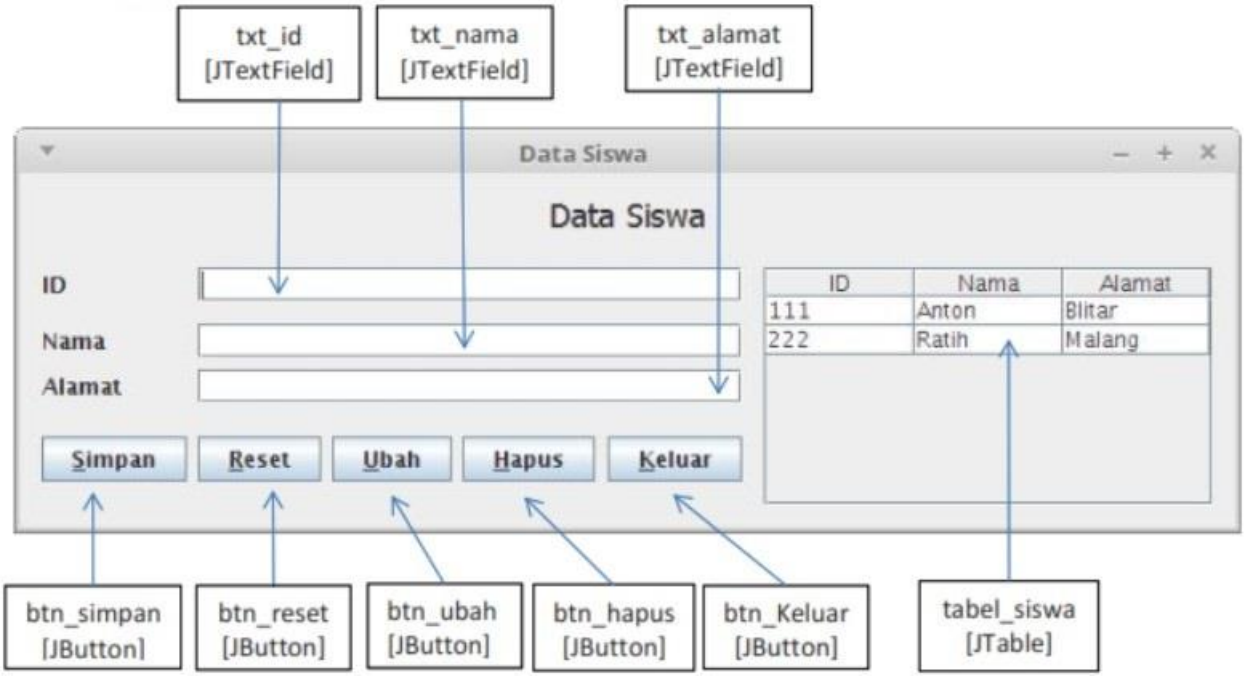
```java
package crud;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.Statement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class CRUD {
    private String id, nama, alamat;
    private Connection CRUDkoneksi;
    private PreparedStatement CRUDpsmt;
    private Statement CRUDstat;
    private ResultSet CRUDhasil;
    private String CRUDquery;

    public CRUD(){
        try {   KoneksiMysql connection = new KoneksiMysql();
                CRUDkoneksi = connection.getKoneksi();
        } catch(SQLException ex){
            System.out.println(ex);
        }
    }

    public void setID(String id)
    {   this.id = id;
    }

    public String getID()
    {   return id;
    }

    public void setNama(String nama)
    {   this.nama = nama;
    }

```

```
37      public String getNama()
38      {   return nama;
39      }
40
41      public void setAlamat(String alamat)
42      {   this.alamat = alamat;
43      }
44
45      public String getAlamat()
46      {   return alamat;
47      }
48
49      public ResultSet tampilData()
50      {   CRUDquery = "select * from siswa";
51          try {   CRUDstat = CRUDkoneksi.createStatement();
52                  CRUDhasil = CRUDstat.executeQuery(CRUDquery);
53          } catch (Exception e) {
54          }
55          return CRUDhasil;
56      }
57
58      public void simpanData(String id,String nama,String alamat)
59      {   CRUDquery = "insert into siswa values(?,?,?)";
60          try {
61              CRUDpsmt = CRUDkoneksi.prepareStatement(CRUDquery);
62              CRUDpsmt.setString(1, id);
63              CRUDpsmt.setString(2, nama);
64              CRUDpsmt.setString(3, alamat);
65              CRUDpsmt.executeUpdate();
66              CRUDpsmt.close();
67          } catch (Exception e) {
68              System.out.println(e);
69          }
70      }
71
72      public void ubahData(String id,String nama,String alamat)
73      {   CRUDquery = "update siswa set nama=?, alamat=? where id=?";
74          try {
75              CRUDpsmt = CRUDkoneksi.prepareStatement(CRUDquery);
76              CRUDpsmt.setString(1, nama);
77              CRUDpsmt.setString(2, alamat);
78              CRUDpsmt.setString(3, id);
79              CRUDpsmt.executeUpdate();
80              CRUDpsmt.close();
81          } catch (Exception e) {
82              System.out.println(e);
83          }
84      }
85
86      public void hapusData(String id)
87      {   CRUDquery = "delete from siswa where id=?";
88          try {
89              CRUDpsmt = CRUDkoneksi.prepareStatement(CRUDquery);
90              CRUDpsmt.setString(1, id);
91              CRUDpsmt.executeUpdate();
92              CRUDpsmt.close();
93          } catch (Exception e) {
94              System.out.println(e);
95          }
96      }
97
98
99  }
100
```

E. Buat Form Siswa dengan JFrame, klik kanan package "**crud > New > JFrame Form**" dan beri nama dengan Form_Siswa. Lihat catatan JFrame pada halaman 10.

Nama file : Form_Siswa.java

TabDesign



TabSource

```
package crud;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

public class Form_Siswa extends javax.swing.JFrame {
    private DefaultTableModel tabmode;
    private ResultSet hasil;

    // membuat objek baru dari class CRUD yang sudah terkoneksi dengan database
    CRUD aa = new CRUD();

    public Form_Siswa() throws SQLException {
        initComponents();
        tampil_database();
    }

    // method untuk menampilkan data ke tabel_siswa [JTable]
    public void tampil_database(){
        Object [] baris = {"ID","Nama","Alamat"};
        tabmode = new DefaultTableModel(null, baris);
```

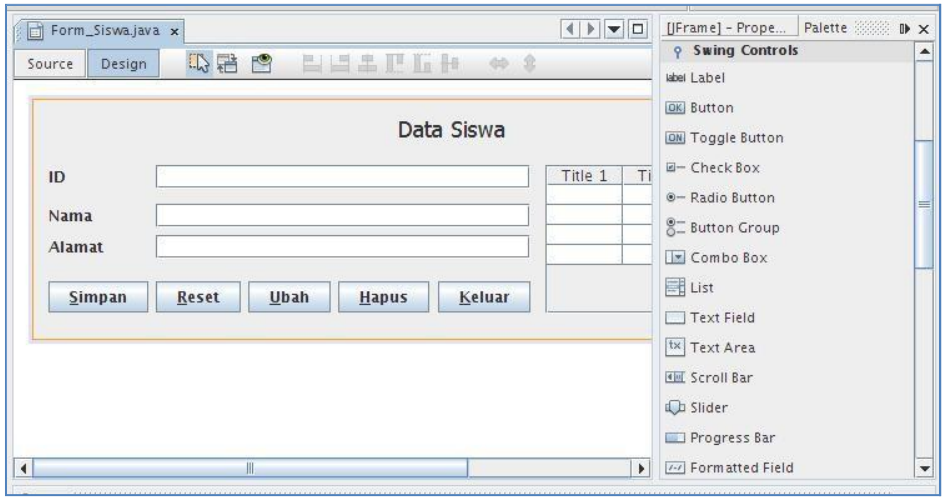
```
        tabel_siswa.setModel(tabmode);
        try {
            hasil = aa.tampilData();
            while (hasil.next()){
                aa.setID(hasil.getString("id"));
                aa.setNama(hasil.getString("nama"));
                aa.setAlamat(hasil.getString("alamat"));
                String[] data = {aa.getID(), aa.getNama(), aa.getAlamat()};
                tabmode.addRow(data);
            }
        } catch (Exception e) {
        }
    }

 // method untuk reset text
    public void reset_text() {
        txt_id.setText("");
        txt_nama.setText("");
        txt_alamat.setText("");
    }

// klik double tombol btn_simpan untuk menyimpan data
private void btn_simpanActionPerformed(java.awt.event.ActionEvent evt) {
        if (txt_id.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, ID belum diisi !");
            txt_id.requestFocus();
        } else if (txt_nama.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, Nama belum diisi !");
            txt_nama.requestFocus();
        } else if (txt_alamat.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, Alamat belum diisi !");
            txt_alamat.requestFocus();
        } else {
            try {   aa.setID(txt_id.getText());
                    aa.setNama(txt_nama.getText());
                    aa.setAlamat(txt_alamat.getText());
                    aa.simpanData(aa.getID(), aa.getNama(), aa.getAlamat());

                    JOptionPane.showMessageDialog(null, "Data berhasil tersimpan" , "Informasi",
JOptionPane.INFORMATION_MESSAGE);
                    tampil_database();
                    reset_text();
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, "Data gagal tersimpan" , "Informasi",
JOptionPane.INFORMATION_MESSAGE);
            }
        }
}
```

```
// klik double tombol btn_reset untuk reset text
private void btn_resetActionPerformed(java.awt.event.ActionEvent evt) {
        reset_text();
}

// klik double tombol btn_ubah untuk mengubah data
private void btn_ubahActionPerformed(java.awt.event.ActionEvent evt) {
        if (txt_id.getText().trim().equals("")){
           JOptionPane.showMessageDialog(null,"Maaf, ID belum diisi !");
           txt_id.requestFocus();
        } else if (txt_nama.getText().trim().equals("")){
           JOptionPane.showMessageDialog(null,"Maaf, Nama belum diisi !");
           txt_nama.requestFocus();
        } else if (txt_alamat.getText().trim().equals("")){
           JOptionPane.showMessageDialog(null,"Maaf, Alamat belum diisi !");
           txt_alamat.requestFocus();
        } else {
           try { aa.setID(txt_id.getText());
                 aa.setNama(txt_nama.getText());
                 aa.setAlamat(txt_alamat.getText());
                 aa.ubahData(aa.getID(), aa.getNama(), aa.getAlamat());

                 JOptionPane.showMessageDialog(null, "Data berhasil diubah" , "Informasi",
JOptionPane.INFORMATION_MESSAGE);
                 tampil_database();
           } catch (Exception e) {
              JOptionPane.showMessageDialog(null, "Data gagal diubah" , "Informasi",
JOptionPane.INFORMATION_MESSAGE);
           }
        }
}

// klik double tombol btn_hapus untuk menghapus data
private void btn_hapusActionPerformed(java.awt.event.ActionEvent evt) {
        if (txt_id.getText().trim().equals("")){
           JOptionPane.showMessageDialog(null,"Maaf, ID belum diisi !");
           txt_id.requestFocus();
        } else {
           if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan menghapus data
ini ?","Warning",2) == JOptionPane.YES_OPTION){
              String id = "";
              try {   aa.setID(txt_id.getText());
                    aa.hapusData(aa.getID());

                    JOptionPane.showMessageDialog(null, "Data berhasil dihapus" , "Informasi",
JOptionPane.INFORMATION_MESSAGE);
                    tampil_database();
                    reset_text();
```

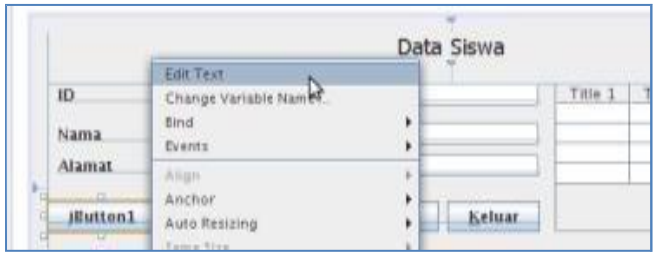
```
            } catch (Exception e) {
                JOptionPane.showMessageDialog(null, "Data gagal dihapus" , "Informasi", JOptionPane.INFORMATION_MESSAGE);
            }
        }
    }
}

// klik kanan tabel_siswa [JTable] > Events > Mouse > MouseClicked
// fungsinya ketika di klik salah satu data maka akan muncul di [JTextField] id, nama, alamat
private void tabel_siswaMouseClicked(java.awt.event.MouseEvent evt) {
    try {   int row = tabel_siswa.rowAtPoint(evt.getPoint());

            String id = tabel_siswa.getValueAt(row, 0).toString();
            String nama = tabel_siswa.getValueAt(row, 1).toString();
            String alamat = tabel_siswa.getValueAt(row, 2).toString();

            txt_id.setText(String.valueOf(id));
            txt_nama.setText(String.valueOf(nama));
            txt_alamat.setText(String.valueOf(alamat));

            } catch (Exception e) {
        }
   }

// klik double tombol btn_keluar untuk keluar program
private void btn_keluarActionPerformed(java.awt.event.ActionEvent evt) {
    if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan keluar ?","Warning",2) == JOptionPane.YES_OPTION){
            System.exit(0);
    }
}

} // End of Form_Siswa class
```

Catatan JFrame:

1. Desainlah Form Data Siswa pada tab “Design” menggunakan Palette, klik menu **Window > Palette**. Drag JLabel, JButton, JTextField dan JTable ke area JFrame.

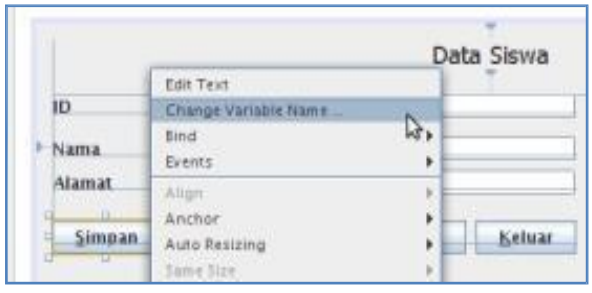


2. Untuk edit text, misalnya tombol “jButton1” ingin diubah menjadi tombol “Simpan”, klik kanan tombol “jButton1” dan pilih “Edit Text”.

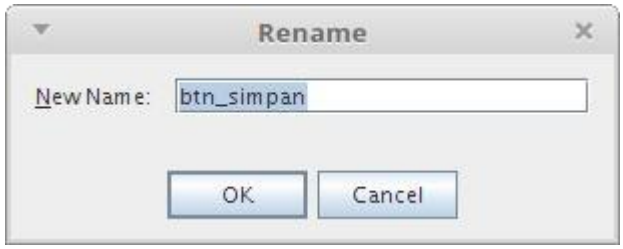


Ubah “jButton1” menjadi “Simpan”. Terapkan pula pada semua JLabel, JButton, JTextField dan JTable !

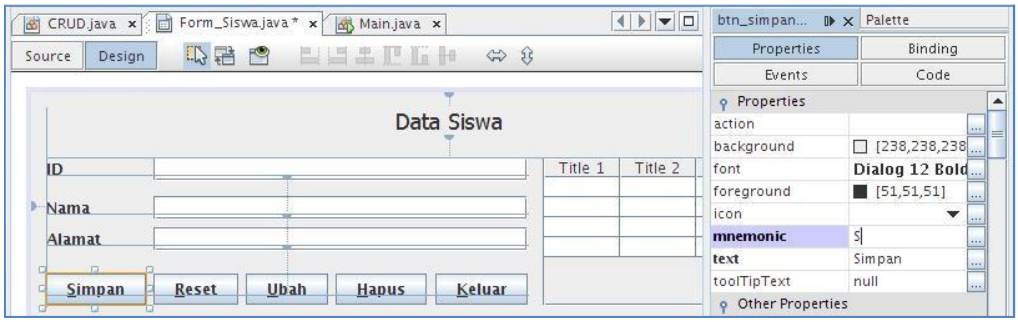
3. Untuk memberi nama variable, misalnya variable tombol “Simpan”, klik kanan tombol “Simpan” dan pilih “Change Variable Name”.



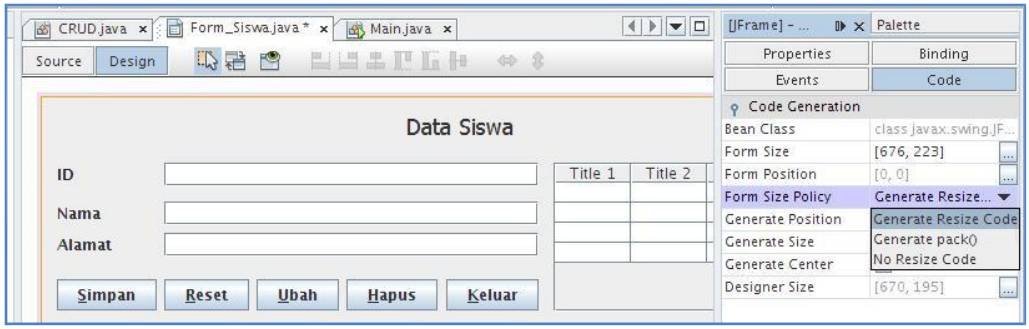
Ganti “New Name” yang awalnya “jButton1” menjadi “btn_simpan”. Terapkan pula pada semua JLabel, JButton, JTextField dan JTable !



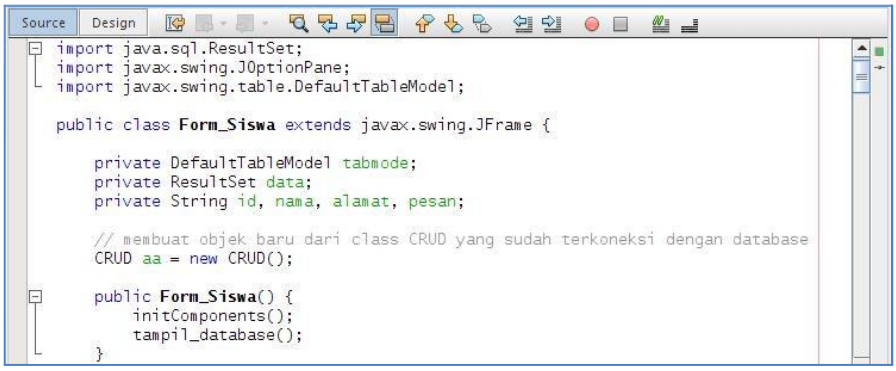
4. Agar tombol memiliki shortcut keyboard, misalnya tombol “Simpan” akan dijalankan ketika mengetikkan Alt+S. Caranya, klik tombol “Simpan”. Klik Menu **Window > Properties.** Klik Tab “Properties” dan pada “mnemonic” tambahkan huruf “S”. Tambahkan shortcut keyboard pula pada tombol yang lain.



5. Agar JFrame berada di posisi tengah, klik area JFrame. Klik Menu **Window > Properties.** Klik Tab “Code” dan pada “Form Size Policy” pilih “Generate Resize Code”.



6. Tulis source code yang berwarna biru pada tab “Source”.



F. Class Main untuk memanggil Form Siswa

Nama file : Main.java

```java
package crud;

public class Main {
    public static void main(String[] args) {
        try{
            Form_Siswa form = new Form_Siswa();
            form.setVisible(true);
        }catch(Exception ex){
            System.out.println(ex.toString());
        }
    }
}
```

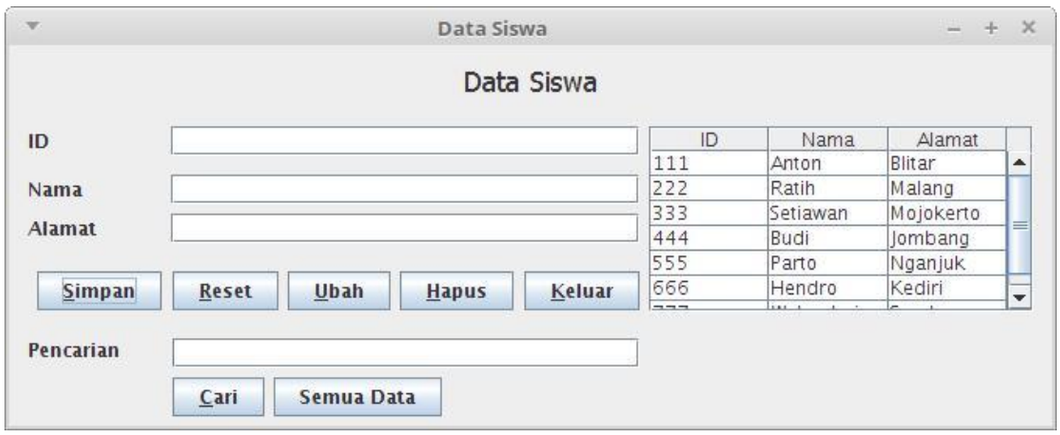
MODUL 10

Graphical User Interface (GUI) Java NetBeans

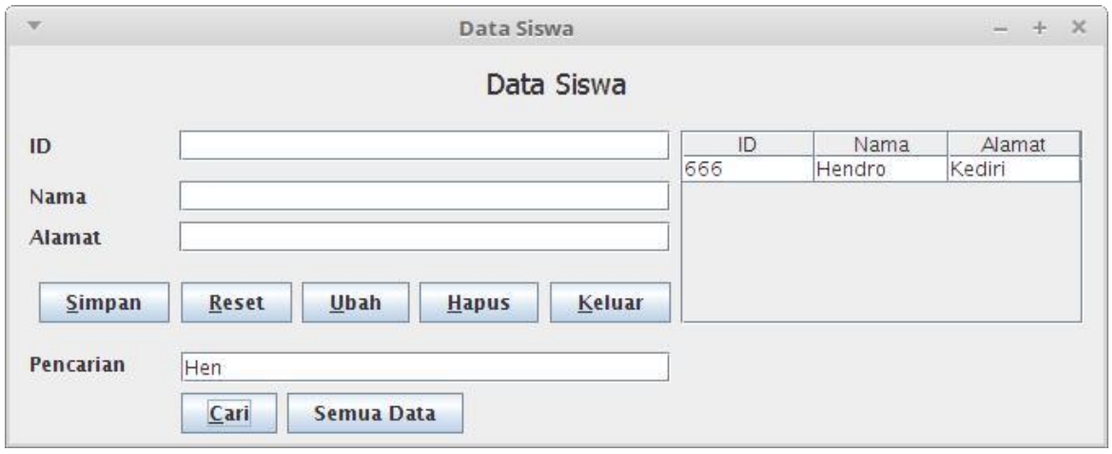
& Aplikasi CRUD (Create, Read, Update, Delete)

LATIHAN 1

1. Ubahlah method simpanData, ubahData dan hapusData pada class CRUD dari void menjadi String. Nilai String yang dikembalikan adalah pesan sukses jika data berhasil disimpan/diubah/dihapus atau pesan gagal jika data gagal disimpan/diubah/dihapus. String pesan sukses/gagal tadi akan ditangkap oleh JOptionPane.showMessageDialog pada class Form_Siswa.
2. Tambahkan Pencarian ID atau Nama pada Form Siswa dengan GUI sebagai berikut:



Masukkan ID atau Nama Siswa, contoh “Hen”, pada pencarian lalu klik tombol “Cari” sehingga muncul di tabel siswa seperti berikut:



Query yang digunakan untuk pencarian “Hen” adalah sebagai berikut:

SELECT * FROM siswa WHERE id LIKE ‘%Hen%’ OR nama LIKE ‘%Hen%’

Klik tombol “Semua Data” untuk menampilkan semua data seperti diawal ketika program dijalankan.

3. Buatlah class Abstract yang diwarisi oleh class CRUD, dimana class Abstract tersebut memuat semua method setter dan getter pada class CRUD !
4. Buatlah Interface yang diimplementasikan pada class CRUD, dimana Inteface tersebut memuat semua method yang berhubungan dengan database seperti method simpan/tampil/ubah/hapus/cari data pada class CRUD !

MODUL 11
LOGIN DAN MENU GUI JAVA NETBEANS

11.1 Login

A. Masih menggunakan package dan koneksi database mysql yang sama dengan CRUD pada modul sebelumnya, buat tabel ‘log_login’ dan ‘user’!

```sql
CREATE TABLE log_login (
  id bigint(18) NOT NULL AUTO_INCREMENT,
  id_user varchar(50),
  waktu_login timestamp DEFAULT CURRENT_TIMESTAMP,
  waktu_logout timestamp,
  PRIMARY KEY (id)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

CREATE TABLE user (
  id_user varchar(50),
  password char(32),
  nama varchar(100),
  PRIMARY KEY (id_user)
) ENGINE=MyISAM DEFAULT CHARSET=utf8;

insert  into user(id_user,password,nama) values
    ('admin',md5('pass'),'Administrator');
```

B. Source code class Session
Nama file : Session.java

```java
package crud;

public class Session {
    private static String UserID,Nama,StatusLogin;

    public static void setUserID(String UserID)
    {   Session.UserID = UserID;
    }

    public static String getUserID()
    {   return UserID;
    }

    public static void setNama(String Nama)
    {   Session.Nama = Nama;
    }

    public static String getNama()
    {   return Nama;
    }

    public static void setStatusLogin(String StatusLogin)
    {   Session.StatusLogin = StatusLogin;
    }

    public static String getStatusLogin()
    {   return StatusLogin;
    }
}
```

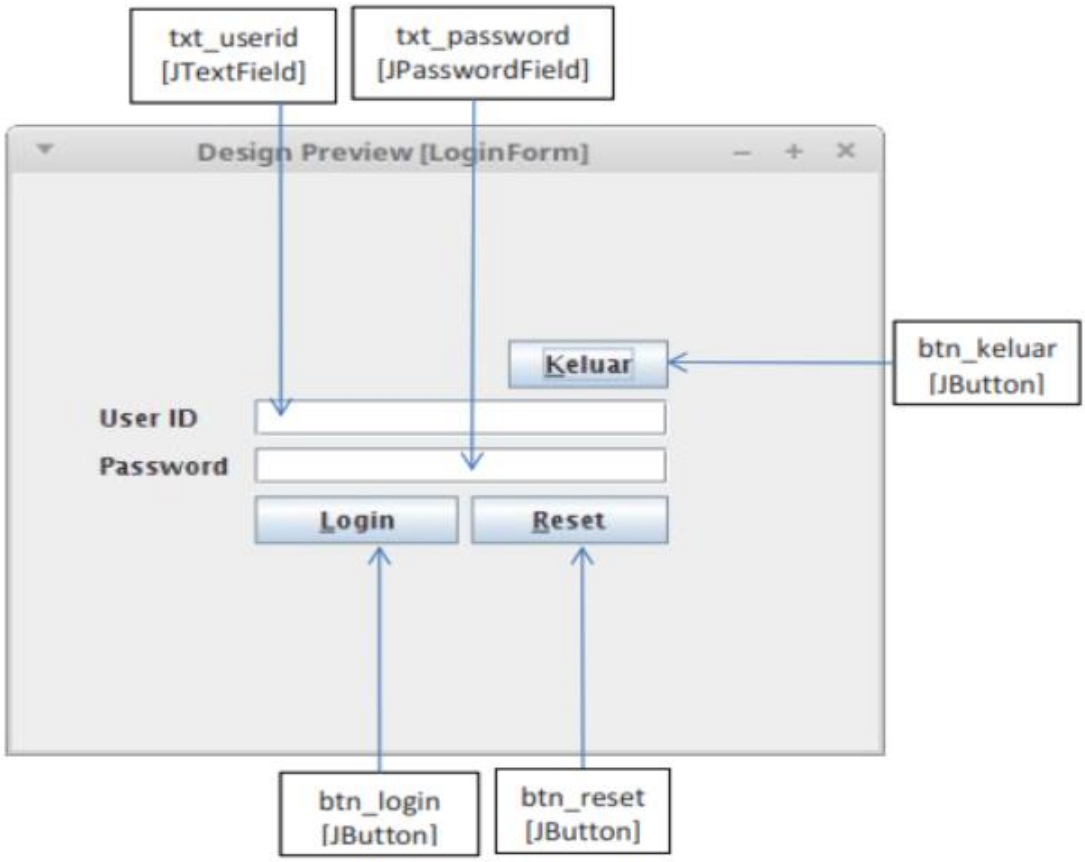
C. Source code class Login
Nama file : Login.java

```
package crud;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class Login {
   private Connection koneksi;
   private PreparedStatement psmt;
   private ResultSet dataUser;
   private String query,userID,password,pesan;

   public Login(){
      KoneksiMysql connection = new KoneksiMysql();
      try {   koneksi = connection.getKoneksi();
      } catch (SQLException ex) {
      }
   }

   public void setUserID(String userID)
   {   this.userID = userID;
   }

   public String getUserID()
   {   return userID;
   }

   public void setPassword(String password)
   {   this.password = password;
   }

   public String getPassword()
   {   return password;
   }

   public String cekLogin(String userID, String password)
   {   query = "SELECT nama FROM user WHERE id_user=? AND password=md5(?)";
      try {   psmt = koneksi.prepareStatement(query);
            psmt.setString(1, userID);
            psmt.setString(2, password);
            dataUser = psmt.executeQuery();

            if (dataUser.next()){
               Session.setUserID(userID) ;
               Session.setNama (dataUser.getString("nama"));
               Session.setStatusLogin("AKTIF");
```

```
                query = "INSERT INTO log_login (id_user) VALUES (?)";
                try {       psmt = koneksi.prepareStatement(query);
                            psmt.setString(1, userID);
                            psmt.executeUpdate();
                            psmt.close();
                } catch (Exception e) {
                            pesan = "Gagal Simpan Log Login";
                }
            } else {
                pesan = "Gagal Login";
            }
        } catch (Exception e) {
            pesan = "Gagal Login, Query Error";
        }
        return pesan;
    }

    public void Logout(String userID)
    {     query = "UPDATE log_login SET waktu_logout=CURRENT_TIMESTAMP() WHERE
id_user=? ORDER BY id DESC LIMIT 1";
        try {   psmt = koneksi.prepareStatement(query);
            psmt.setString(1, userID);
            psmt.executeUpdate();
            psmt.close();

            // memutus koneksi database
            koneksi.close();

            // hapus Session
            Session.setUserID(null);
            Session.setNama(null);
            Session.setStatusLogin(null);
        } catch (Exception e) { }
    }

}
```

D. Buat Form Login dengan JFrame dan beri nama dengan LoginForm.
Nama file : LoginForm.java

TabDesign



TabSource

```
package crud;
import javax.swing.JOptionPane;

public class LoginForm extends javax.swing.JFrame {
    private String pesan;
    // membuat objek baru dari class Login yang sudah terkoneksi dengan database
    Login bb = new Login();

    public LoginForm() {
        initComponents();
    }

     // method untuk mereset text
    public void reset_text() {
        txt_userid.setText("");
        txt_password.setText("");
    }
```

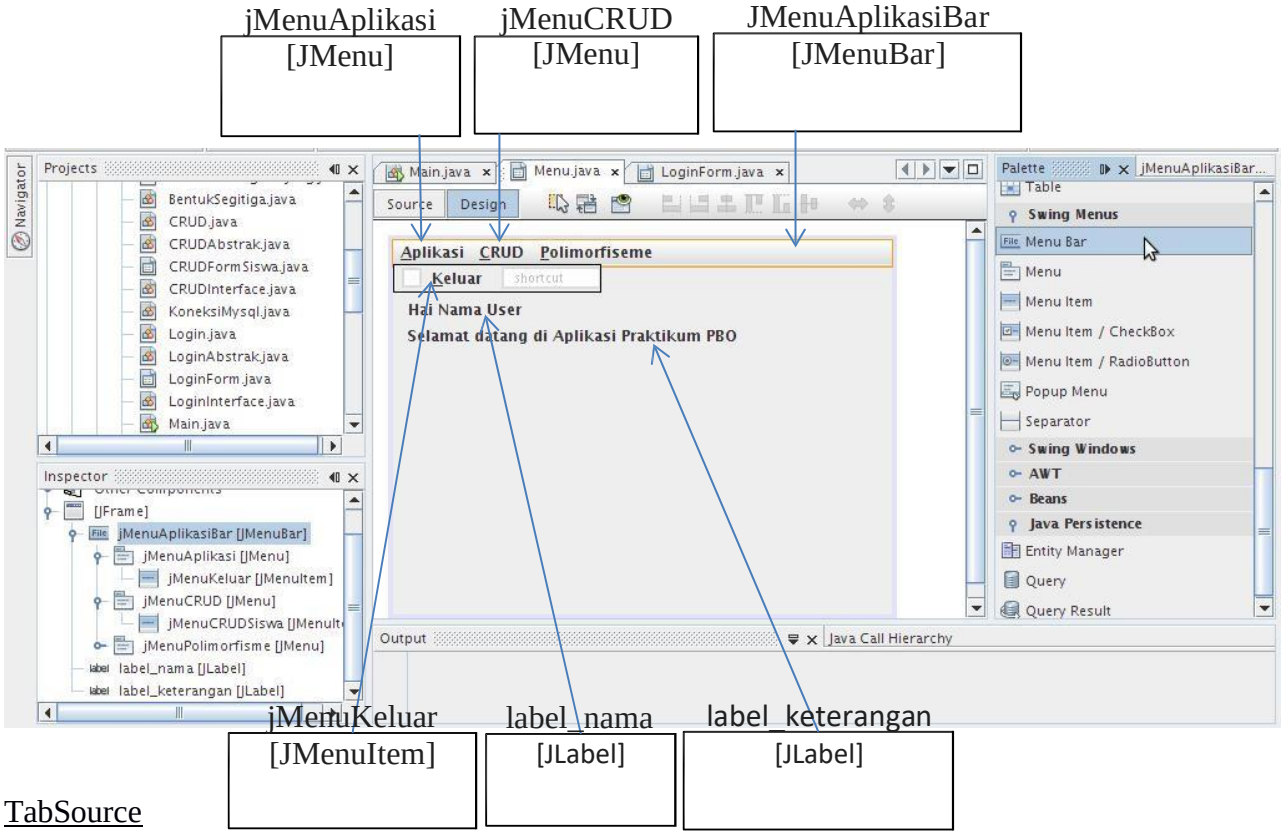
```
    private void btn_loginActionPerformed(java.awt.event.ActionEvent evt) {
        if (txt_userid.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, User ID belum diisi !");
            txt_userid.requestFocus();
        } else if (txt_password.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, Password belum diisi !");
            txt_password.requestFocus();
        } else { bb.setUserID(txt_userid.getText());
            bb.setPassword(txt_password.getText());
            pesan = bb.cekLogin(bb.getUserID(), bb.getPassword());

            if (Session.getStatusLogin() == "AKTIF")
            {  // menutup form login tanpa keluar dari aplikasi
               dispose();

               // memanggil form menu
               Menu form = new Menu();
               form.setVisible(true);
            } else {
                    JOptionPane.showMessageDialog(null,pesan,"Informasi",JOptionPane.INFORMAT
            ION_MESSAGE);
            }
        }
    }

    private void btn_resetActionPerformed(java.awt.event.ActionEvent evt) {
        reset_text();
    }

    private void btn_keluarActionPerformed(java.awt.event.ActionEvent evt) {
        if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan keluar ?","Warning",2)
== JOptionPane.YES_OPTION){
                System.exit(0);
        }
    }

} // End of LoginForm class
```

11.2. Menu

A. Buatlah Menu dengan menggunakan Menu Bar, Menu dan Menu Item.

Nama file : Menu.java

TabDesign



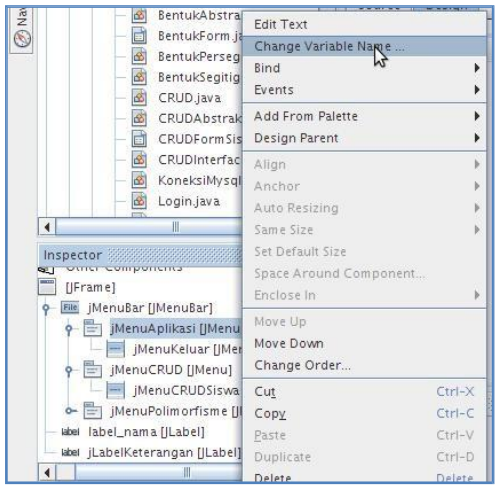
TabSource

```
package crud;
import javax.swing.JOptionPane;

public class Menu extends javax.swing.JFrame {
   // membuat objek baru dari class Login yang sudah terkoneksi dengan database
   Login bb = new Login();

   public Menu() {
      // mengecek sudah login atau belum
      // jika belum login akan di redirect ke form login
      if (Session.getStatusLogin() == "AKTIF")
      {   initComponents();
         setSessionNama();
      } else {
         // menutup menu tanpa keluar dari aplikasi
         dispose();
         // memanggil form login
         LoginForm form = new LoginForm();
         form.setVisible(true);
      }
```

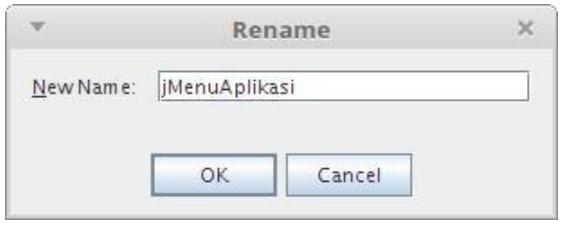
```
    }
    private void setSessionNama()
    {   label_nama.setText("Hai "+Session.getNama());
    }

    private void jMenuCRUDSiswaActionPerformed(java.awt.event.ActionEvent evt) {
        try{
            Form_Siswa form = new Form_Siswa();
            form.setVisible(true);
        }catch(Exception ex){
        }
    }

    private void jMenuKeluarActionPerformed(java.awt.event.ActionEvent evt) {
        if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan keluar ?","Warning",2)
== JOptionPane.YES_OPTION){
            bb.Logout(Session.getUserID());
            System.exit(0);
        }
    }

}  // End of file Menu class
```

Catatan :

1. Untuk mengubah nama variable menu, klik kanan JMenu, pilih **Change Variable Name**.

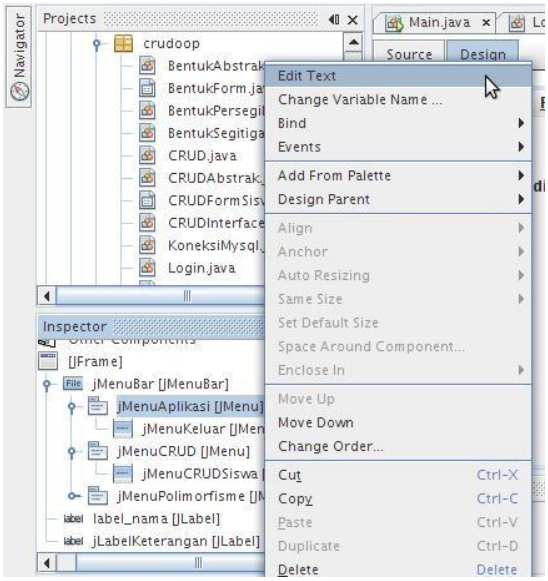


Ubah nama variable menu menjadi "**jMenuAplikasi**".

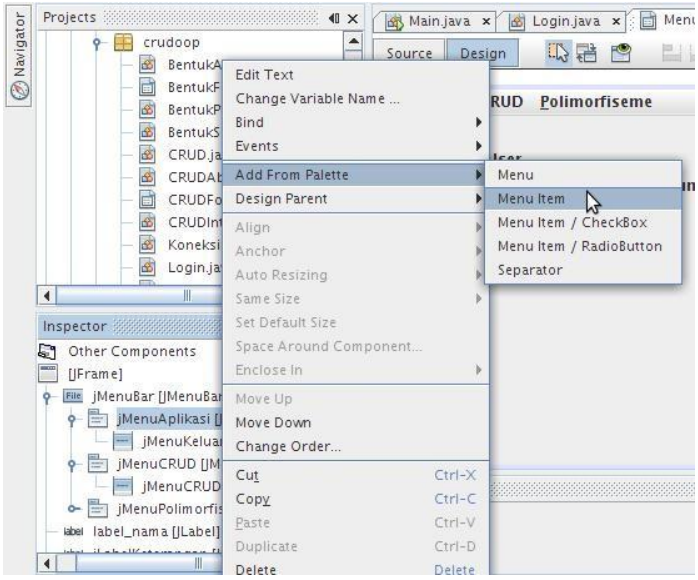


Selain jMenuAplikasi**,** tambahkan pula jMenuCRUD.

2. Untuk mengubah teks menu, klik kanan JMenu, pilih **Edit Text.** Ubah menjadi "**Aplikasi**".

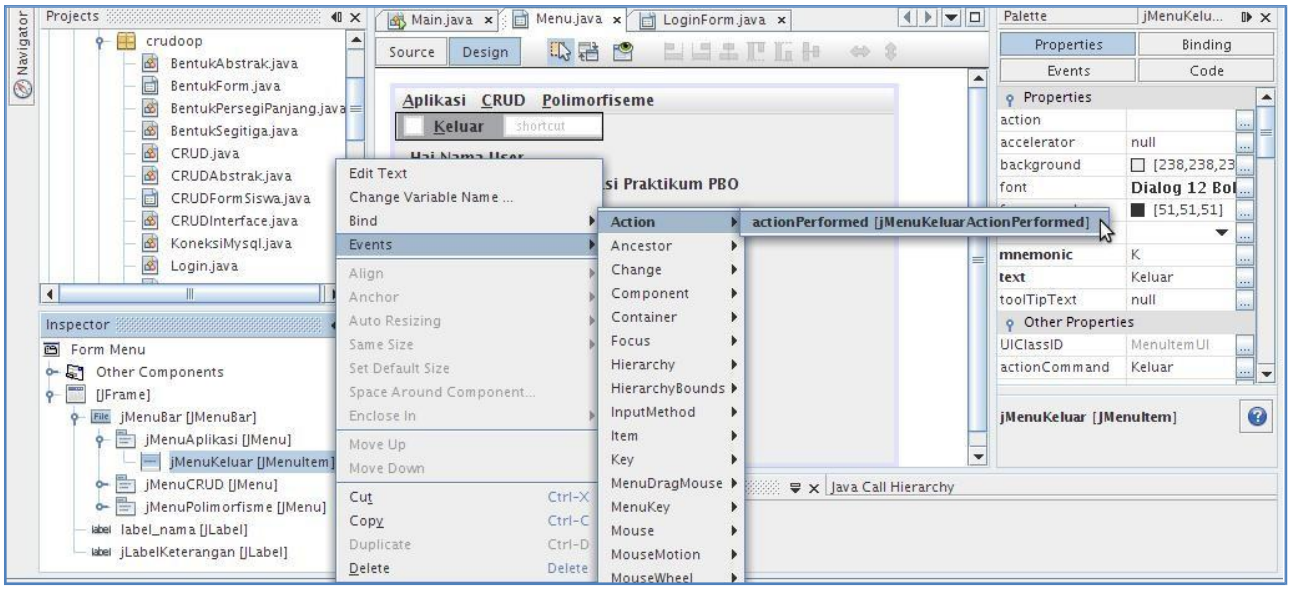


3. Untuk menambah menu item, klik kanan JMenu, pilih **Add From Palette > Menu Item**.



Buatlah Menu Item "Keluar" pada menu Aplikasi dan Menu Item "CRUD Siswa" pada menu CRUD! Untuk mengganti nama variable dan mengubah teks pada Menu Item, caranya sama dengan langkah-langkah pada nomor 1 dan 2.

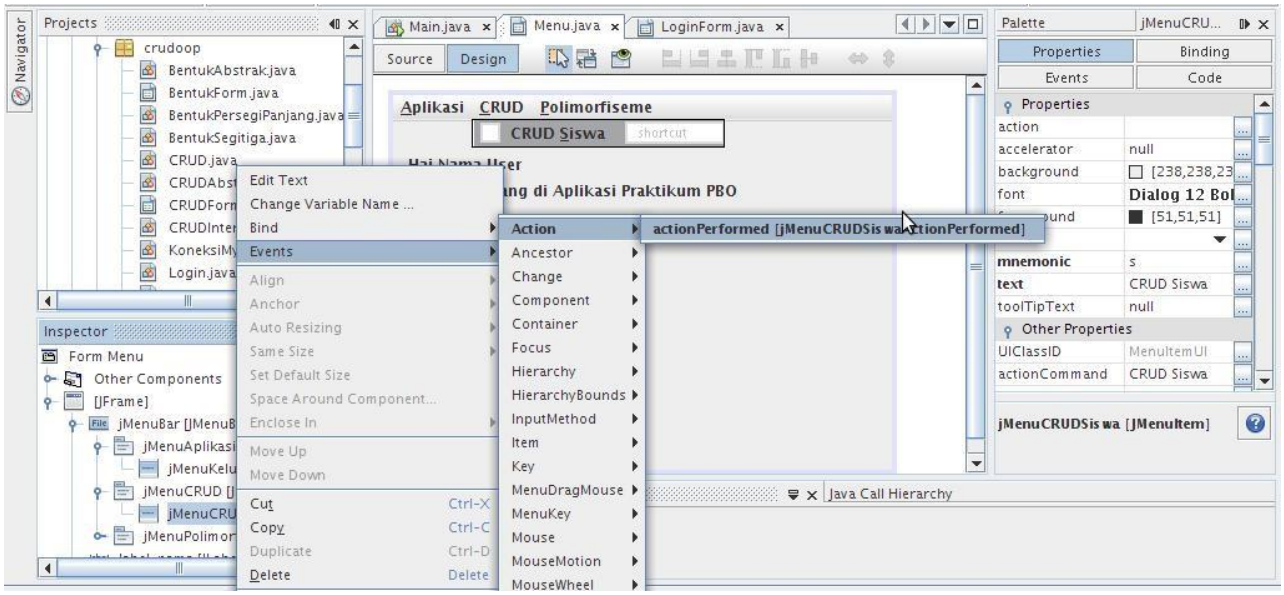
4. Untuk memberikan aksi pada Menu Item **"Keluar"** ketika di klik oleh pengguna, caranya adalah klik kanan Menu Item **"Keluar"**, pilih **Events > Action > actionPerformed.**



Tambahkan source code:

```
private void jMenuKeluarActionPerformed(java.awt.event.ActionEvent evt) {
      if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan keluar ?","Warning",2)
== JOptionPane.YES_OPTION){
            bb.Logout(Session.getUserID());
            System.exit(0);
      }
 }
```

5. Untuk memberikan aksi pada Menu Item **"CRUD Siswa"** ketika di klik oleh pengguna, caranya adalah klik kanan Menu Item **"CRUD Siswa"**, pilih **Events > Action > actionPerformed.**



Tambahkan source code:

```
private void jMenuCRUDSiswaActionPerformed(java.awt.event.ActionEvent evt) {
      try{
         // memanggil form siswa
         Form_Siswa form = new Form_Siswa();
         form.setVisible(true);
      }catch(Exception ex){
      }
}
```

11.3. Perubahan pada Form_Siswa.java

A. Pada Form_Siswa, ubah source code Constructor-nya:

```
   public Form_Siswa() throws SQLException {
      initComponents();
      tampil_database();
   }
```

Menjadi:

```
   public Form_Siswa() throws SQLException {
      // mengecek sudah login atau belum
      // jika belum login akan di redirect ke form login
      if (Session.getStatusLogin() == "AKTIF")
      {   initComponents();
         tampil_database();
      } else {
          // menutup form tanpa keluar dari aplikasi
         dispose();
         // memanggil form login
         LoginForm form = new LoginForm();
         form.setVisible(true);
      }
   }
```

B. Aksi pada tombol “Keluar” juga diubah:

```
private void btn_keluarActionPerformed(java.awt.event.ActionEvent evt) {
      if (JOptionPane.showConfirmDialog(null, "Apakah Anda yakin akan keluar ?","Warning",2)
== JOptionPane.YES_OPTION){
            System.exit(0);
      }
}
```

Menjadi :

```
private void btn_keluarActionPerformed(java.awt.event.ActionEvent evt) {
         dispose();
}
```

11.4. Class Main untuk memanggil Form Login
Nama file : Main.java

```
package crud;
public class Main {
   public static void main(String[] args) {
      try{  // Form_Siswa form = new Form_Siswa ();
            LoginForm form = new LoginForm();
            form.setVisible(true);
      }catch(Exception ex){
         System.out.println(ex.toString());
      }
   }
}
```

Tugas:

1. Buatlah class Abstract yang diwarisi oleh class Login, dimana class Abstract tersebut memuat method-method Setter dan Getter pada class Login.
2. Buatlah Interface yang diimplementasikan pada class Login, dimana Inteface tersebut memuat method-method yang berhubungan dengan database, yaitu method cek login dan method logout pada class Login.
3. Silahkan lihat tabel “log_login” di dalam database setelah melakukan proses login dan logout! Pastikan userID, waktu login dan waktu logout tersimpan di dalam tabel “log_login” tersebut!

MODUL 12
POLIMORFISME GUI JAVA NETBEANS

12.1 Bentuk

Dalam bentuk ini akan dibahas tentang perhitungan luas berbagai macam bentuk seperti persegi panjang, segitiga, belah ketupat, jajar genjang dan layang-layang. Class bentuk akan menjadi class parent bagi class persegi panjang, class segitiga, class belah ketupat, class jajar genjang dan class layang-layang.

A. Source code class Bentuk
Nama file : Bentuk.java

```
1  package crud;
2
3  public class Bentuk {
4      protected double a,b,c;
5      protected String luas;
6
7      protected void set_A(double a) {
8          this.a = a;
9      }
10
11     protected double get_A() {
12         return a;
13     }
14
15     protected void set_B(double b) {
16         this.b = b;
17     }
18
19     protected double get_B() {
20         return b;
21     }
22
23     protected void set_C() {
24     }
25
26     protected double get_C() {
27         return c;
28     }
29
30     protected String cetak() {
31         return luas;
32     }
33 }
```

B. Source code class BentukPersegiPanjang
Nama file : BentukPersegiPanjang.java

```
1  package crud;
2
3  public class BentukPersegiPanjang extends Bentuk {
4
5      protected void set_C() {
6          c = a * b;
7      }
8
9      protected String cetak() {
10         luas = "Luas Persegi Panjang : "+get_A()+" * "+get_B()+" = "+get_C();
11         return luas;
12     }
13
14 }
```

C. Source code class BentukSegitiga
Nama file : BentukSegitiga.java

```java
package crud;

public class BentukSegitiga extends Bentuk {

    protected void set_C() {
        c = a * b * 0.5;
    }

    protected String cetak() {
        luas = "Luas Segitiga : "+get_A()+" * "+get_B()+" * 0.5  = "+get_C();
        return luas;
    }

}
```

D. Source code class BentukBelahKetupat
Nama file : BentukBelahKetupat.java

```java
package crud;

public class BentukBelahKetupat extends Bentuk {

    protected void set_C() {
        c = a * b * 0.5;
    }

    protected String cetak() {
        luas = "Luas Belah Ketupat : "+get_A()+" * "+get_B()+" * 0.5  = "+get_C();
        return luas;
    }

}
```

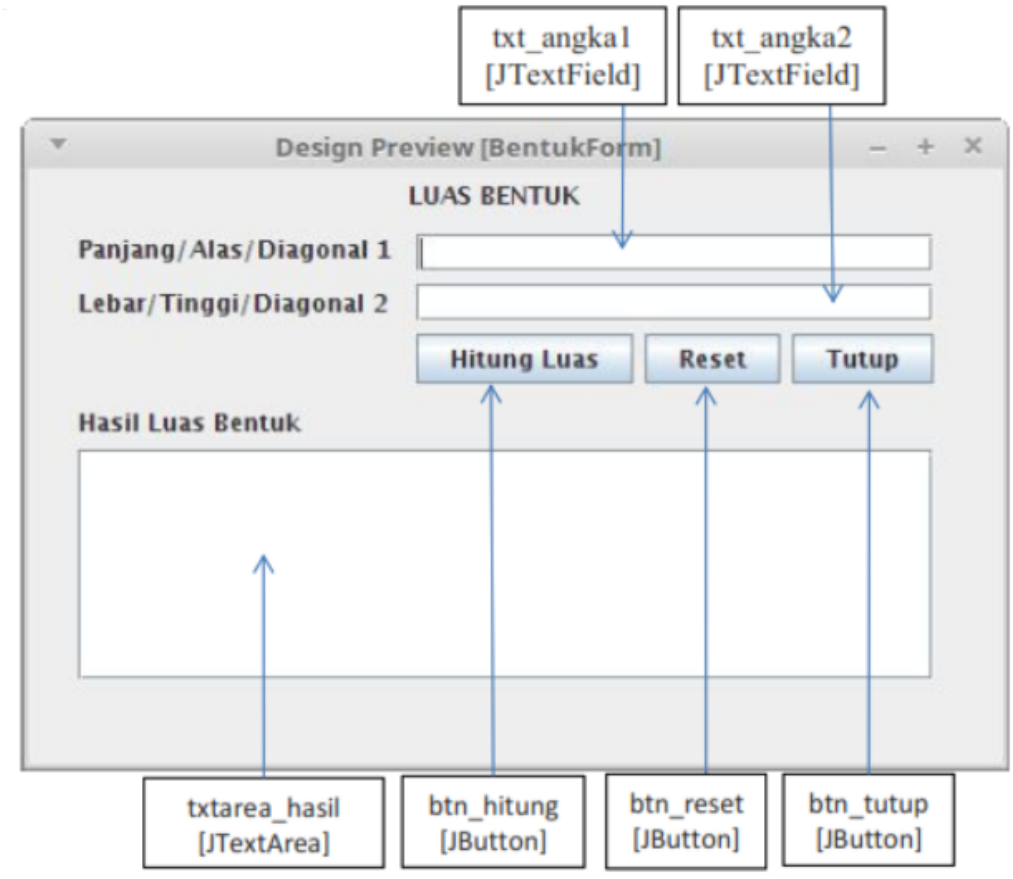
E. Source code class BentukJajarGenjang
Nama file : BentukJajarGenjang.java

```java
package crud;

public class BentukJajarGenjang extends Bentuk {

    protected void set_C() {
        c = a * b;
    }

    protected String cetak() {
        luas = "Luas Jajar Genjang : "+get_A()+" * "+get_B()+" = "+get_C();
        return luas;
    }

}
```

F. Source code class BentukLayangLayang
Nama file : BentukLayangLayang.java

```
1  package crud;
2
3  public class BentukLayangLayang extends Bentuk {
4
5      protected void set_C() {
6          c = a * b * 0.5;
7      }
8
9      protected String cetak() {
10         luas = "Luas Layang - Layang : "+get_A()+" * "+get_B()+" * 0.5  = "+get_C();
11         return luas;
12     }
13
14 }
```

G. Buat Form Bentuk dengan JFrame dan beri nama dengan BentukForm.
Nama file : BentukForm.java
TabDesign



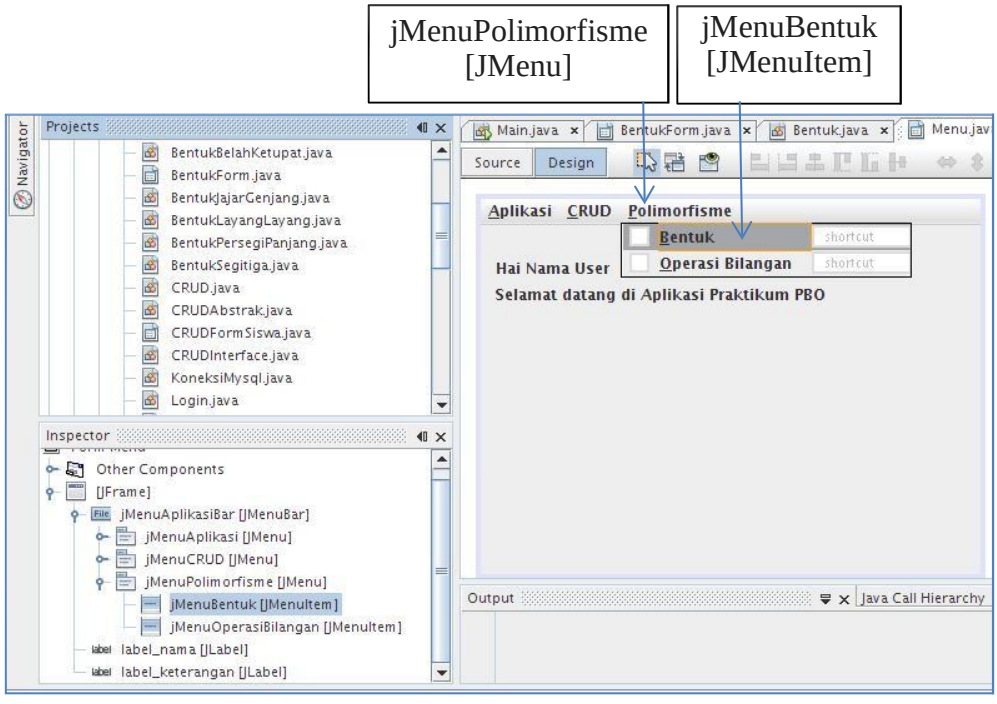
TabSource

```
package crud;
import javax.swing.JOptionPane;

public class BentukForm extends javax.swing.JFrame {
   private String Hasil;

   public BentukForm() {
      // mengecek sudah login atau belum
      // jika belum login akan di redirect ke form login
      if (Session.getStatusLogin() == "AKTIF")
      {   initComponents();
          txtarea_hasil.disable();
      } else {
         // menutup form menu tanpa menutup aplikasi
         dispose();
         // memanggil form login
         LoginForm form = new LoginForm();
         form.setVisible(true);
      }
   }

   public void reset_text() {
      txt_angka1.setText("");
      txt_angka2.setText("");
   }

   private void polimorfisme(Bentuk[] B,double a,double b) {
      Hasil="";
      for (int i=0;i<B.length;i++)
       {   B[i].set_A(a);
          B[i].set_B(b);
          B[i].set_C();
          Hasil += B[i].cetak()+"\n";
       }
   }

   public void cetakHasil(String Hasil) {
      txtarea_hasil.setText(Hasil);
   }

  private void btn_tutupActionPerformed(java.awt.event.ActionEvent evt) {
      dispose();
   }

   private void btn_resetActionPerformed(java.awt.event.ActionEvent evt) {
      reset_text();
   }
```

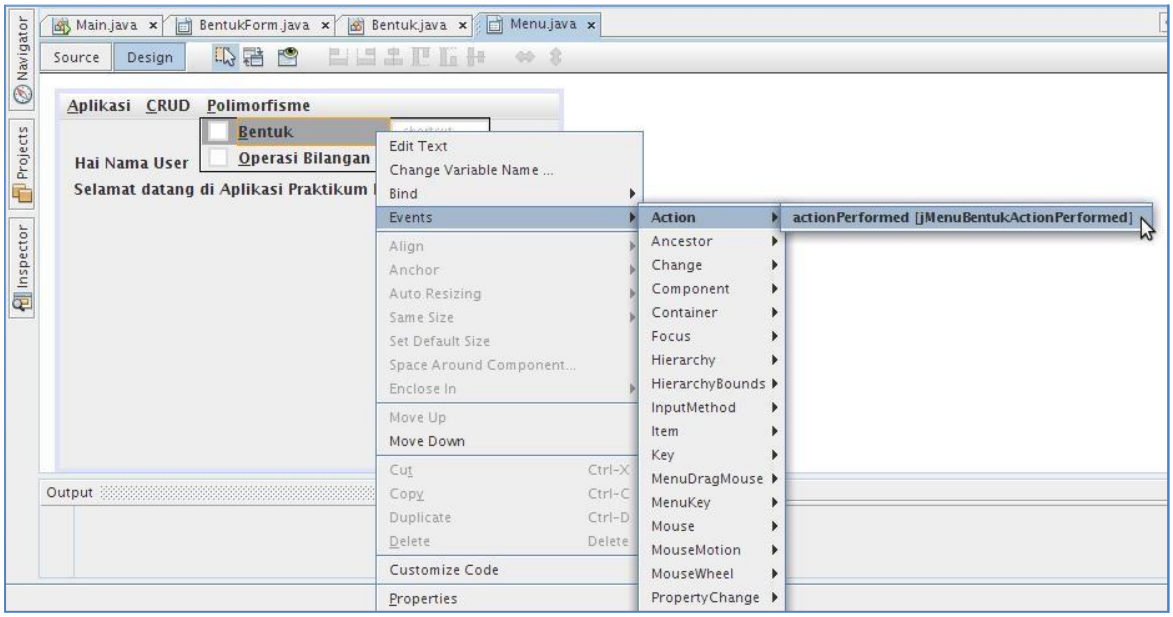
```
    private void btn_hitungActionPerformed(java.awt.event.ActionEvent evt) {
        if (txt_angka1.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, Panjang/Tinggi belum diisi !");
            txt_angka1.requestFocus();
        } else if (txt_angka2.getText().trim().equals("")){
            JOptionPane.showMessageDialog(null,"Maaf, Lebar belum diisi !");
            txt_angka2.requestFocus();
        } else {
            String angka1 = txt_angka1.getText();
            Double angka_1 = Double.parseDouble(angka1);

            String angka2 = txt_angka2.getText();
            Double angka_2 = Double.parseDouble(angka2);

            Bentuk[] B = {  new BentukPersegiPanjang(),
                            new BentukJajarGenjang(),
                            new BentukSegitiga(),
                            new BentukLayangLayang(),
                            new BentukBelahKetupat()
                        };

            polimorfisme(B,angka_1,angka_2);
            cetakHasil(Hasil);
        }
    }
```

} // End of BentukForm class

H. Pada menu.java, buatlah menu [JMenu] Polimorfisme dan menu item [JMenuItem] Bentuk.



Klik kanan [JMenuItem] Bentuk, pilih **Events > Action > actionPerformed.**



Tambahkan source code:

```
private void jMenuBentukActionPerformed(java.awt.event.ActionEvent evt) {
        try{        // memanggil form bentuk
                    BentukForm form = new BentukForm();
                    form.setVisible(true);
        }catch(Exception ex){
        }
}
```

I. Jalankan Main.java untuk mencoba aplikasinya.

Tugas:

1. Buatlah class Abstract yang diwarisi oleh class Bentuk, dimana class Abstract tersebut memuat seluruh method setter, getter dan cetak pada class Bentuk!

MODUL 12
POLIMORFISME GUI JAVA NETBEANS
LATIHAN 1

1. Buatlah tabel "log_aktifitas" di dalam database!

```
CREATE TABLE log_aktifitas (
  id BIGINT(18) NOT NULL AUTO_INCREMENT,
  waktu TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  id_user VARCHAR(50),
  aktifitas VARCHAR(50),
  keterangan TEXT,
  PRIMARY KEY (id)
) ENGINE=MYISAM DEFAULT CHARSET=utf8;
```

2. Buat void method "LogAktifitas" di dalam class "Bentuk" yang mencatat (insert) semua perhitungan luas bentuk yang dilakukan oleh pengguna ke dalam tabel "log_aktifitas" di database! **Masukkan method "LogAktifitas" tersebut ke dalam polimorfisme!** Method tersebut adalah:

   public void LogAktifitas(String userID,String aktifitas,String keterangan)

   Keterangan:
   a) **userID** = Session.getUserID()
   b) **aktifitas** = "LUAS BENTUK"
   c) **Keterangan** = Hasil perhitungan luas bentuk yang ditampilkan di JTextArea, misalnya:
   Luas Persegi Panjang : 6.0 * 4.0 = 24.0
   Luas Jajar Genjang : 6.0 * 4.0 = 24.0
   Luas Segitiga : 6.0 * 4.0 * 0.5 = 12.0
   Luas Layang - Layang: 6.0 * 4.0 * 0.5 = 12.0
   Luas Belah Ketupat : 6.0 * 4.0 * 0.5 = 12.0

3. Buatlah Interface yang diimplementasikan pada class Bentuk, dimana Inteface tersebut memuat method yang berhubungan dengan database, yaitu method "LogAktifitas" pada class "Bentuk"!
4. Silahkan lihat tabel "log_aktifitas" di dalam database setelah melakukan proses perhitungan luas bentuk! Pastikan userID dan semua hasil perhitungan luas bentuk tersimpan di dalam tabel "log_aktifitas" tersebut!