MODUL PRAKTIKUM

OBJECT ORIENTED PROGRAMMING

BAHASA PEMROGRAMAN JAVA

DOSEN PENGAMPU

NURUL KHAIRINA

UNIVERSITAS MEDAN AREA

MEDAN

2020

KATA PENGANTAR

Alhamdulillah, segala puji bagi Allah SWT. Atas rahmat dan karunianya, Modul Praktikum *Object Oriented Programming* Bahasa Pemrograman Java ini dapat diselesaikan dengan baik. Modul ini merupakan pengembangan dari modul yang telah dibuat sebelumnya pada modul praktikum Bahasa pemrograman Java di salah satu PTS di Kota Medan. Modul ini terdiri dari 10 pertemuan yaitu : Memahami Dasar-Dasar Pemrograman Java, Memberikan Input dari Keyboard, Struktur Pengendali Proses, Java Array, Membuat Class Sendiri, Abstraksi *(Abstraction)* & Pembungkusan *(Encapsulation),* Pewarisan *(Inheritance)* & Polimorfisme *(Polymorphism), Exception Handling, Graphical User Interface* (GUI), dan Aplikasi Java CRUD dengan Database MySQL. Modul ini diharapkan dapat membantu mahasiswa dalam memahami praktikum *object oriented programming* (OOP) yang akan dibangun dengan bahasa pemrograman java dengan *software* Netbeans 8.2. Kritik dan saran yang membangun akan sangat diharapkan untuk perkembangan modul praktikum ini.

Medan, 6 April 2020

Nurul Khairina

MODUL PRAKTIKUM

Mata Kuliah Praktikum	**: Praktikum Pemrograman Berorientasi Objek**
Kode Mata Kuliah Praktikum	**: TIF16025**
SKS	**: 1 SKS**
Program Studi	**: Teknik Informatika**
Semester	**: IV**

DISETUJUI OLEH	DIPERIKSA OLEH	DIBUAT OLEH
Ka. Prodi	Ka. Lab	Dosen Pengampu
Rizki Muliono, S.Kom, M.Kom	Muhathir, S.Kom.,M.Kom	Nurul Khairina, S.Kom, M.Kom

DAFTAR ISI

MODUL 1

MEMAHAMI DASAR - DASAR PEMROGRAMAN JAVA

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat mengenal Java melalui *Class, Object, Method, Constructor*.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. Perangkat Lunak Netbeans

C. TEORI

Pemrograman Berorientasi Objek (PBO) atau *Object Oriented Programming* (OOP) menggunakan Bahasa Pemrograman Java. Bahasa Pemrograman Java memiliki beberapa komponen, antara lain :

- *Class* : tempat untuk mendeklarasikan tipe data
- *Object* : entiti yang memiliki keadaan/ tingkah laku
- *Attribute* : elemen dari sebuah objek yang berisi informasi tentang objek
- *Method* : tingkah laku dari objek/ sub program
- *Construction* : method yang digunakan untuk membuat objek baru

Bahasa pemrograman java dikenal dengan *case sensitive*, artinya ada banyak aturan dalam Pemrograman Java yang harus benar-benar diperhatikan. Berikut ini beberapa aturan dalam pemberian nama *class* :

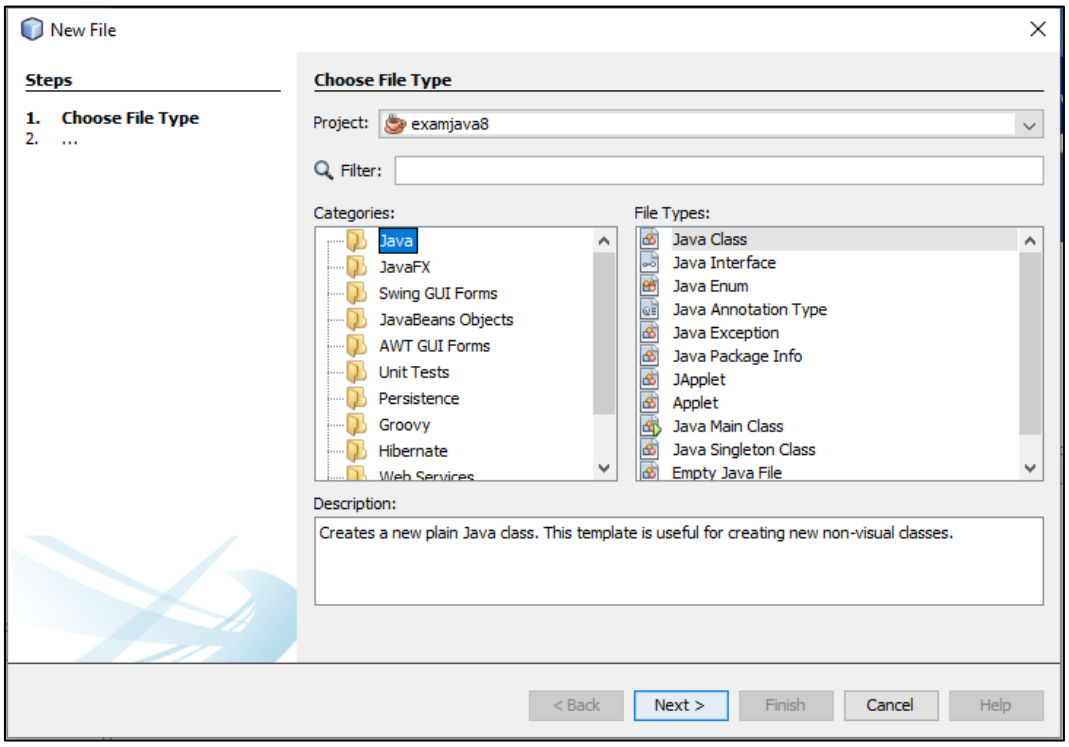
a. Nama file **halaman kerja java** harus sama dengan **nama class**
b. Nama class boleh terdiri dari huruf kecil dan huruf besar
c. Nama class yang terdiri dari dua suku kata, dapat dituliskan dengan tanda underscore (_) sebagai pemisah. Contoh : Belajar_Java
d. Nama class tidak boleh terdiri dari angka
e. Nama class boleh ditulis dengan huruf abjad dan diikuti dengan angka.
   Contoh : Belajar_Java1

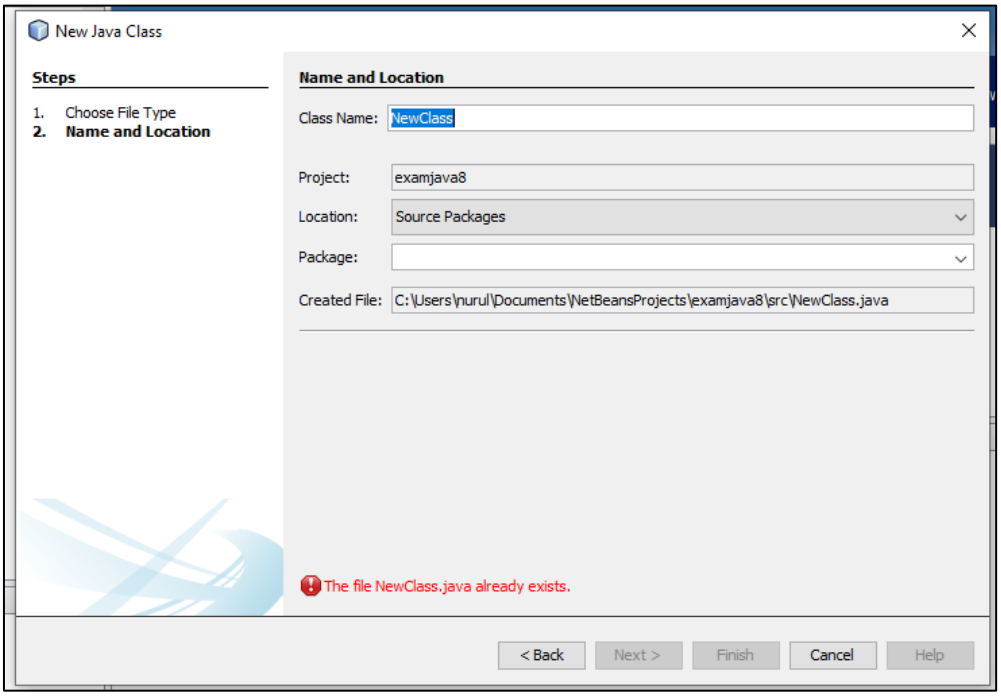
f. Nama class tidak boleh menggunakan keyword (*public*, *class*, *private*, dan sebagainya)

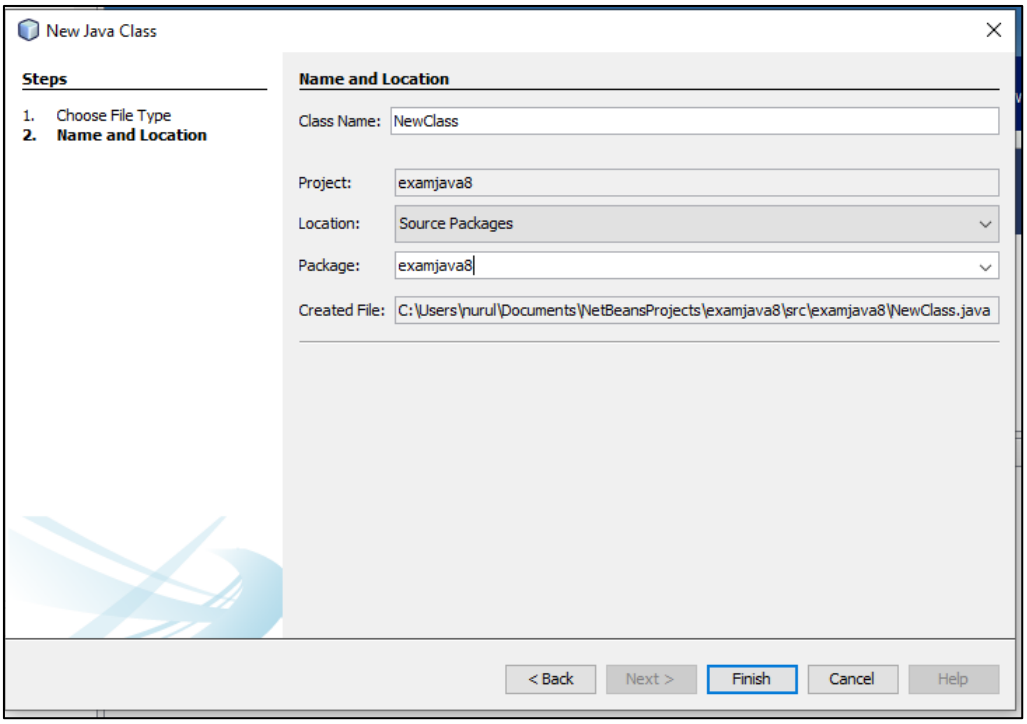
D. PRAKTIKUM

1. Memulai Aplikasi Netbeans

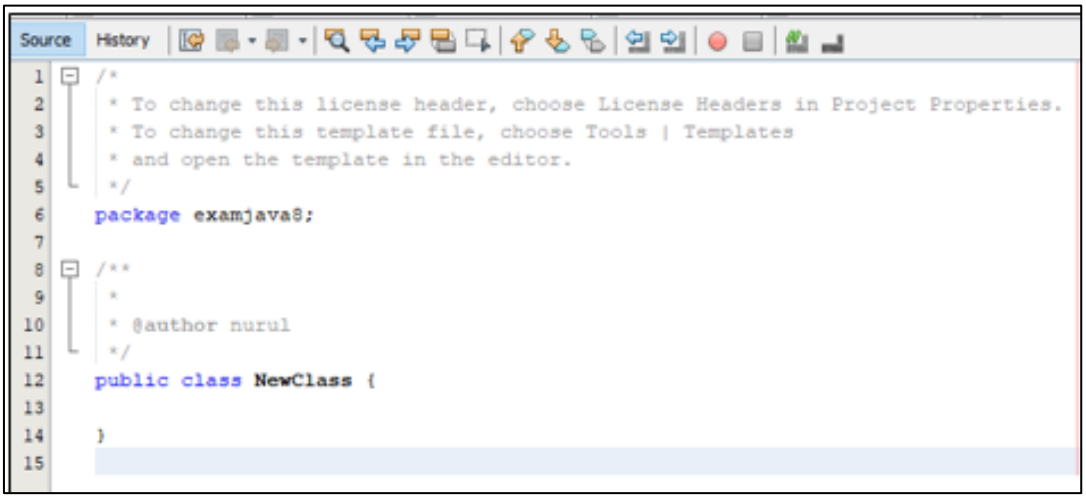
Praktikum Pemrograman Berorientasi Objek (PBO) ini akan menggunakan software Java NetBeans. Halaman baru pada Java NetBeans dapat dimulai dari menu **New – New File – Java Class**. Kemudian berilah nama project, nama class, dan nama package. Nama package dapat diisi sesuai dengan nama project.




New Java Class
Steps
1. Choose File Type
2. Name and Location
Name and Location
Class Name: NewClass
Project: examjava8
Location: Source Packages
Package:
Created File: C:\Users\nurul\Documents\NetBeansProjects\examjava8\src\NewClass.java
The file NewClass.java already exists.
< Back
Next >
Finish
Cancel
Help

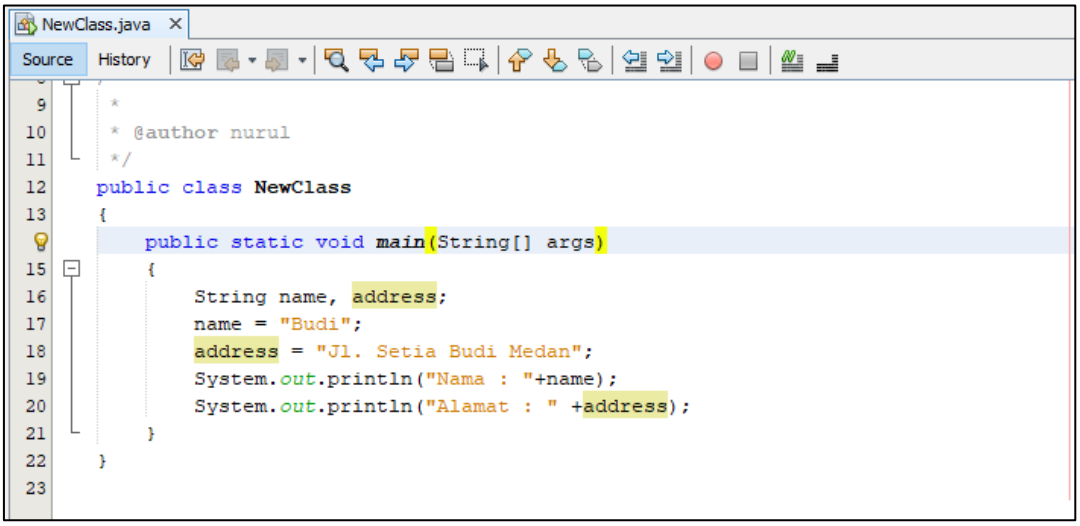

New Java Class
Steps
1. Choose File Type
2. Name and Location
Name and Location
Class Name: NewClass
Project: examjava8
Location: Source Packages
Package: examjava8
Created File: C:\Users\nurul\Documents\NetBeansProjects\examjava8\src\examjava8\NewClass.java
< Back
Next >
Finish
Cancel
Help

Tampilan awal halaman kerja java dengan NewClass :

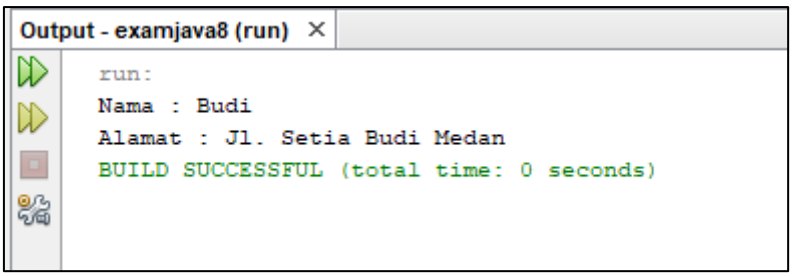


2. Membuat Class

❖ Code Pemrograman:



❖ Hasil Pemrograman :



3. Membuat *Object*

❖ Code Pemrograman :

```
12
13  class variabel
14  {
15      int lebar;
16      int panjang;
17  }
18
19  class NewClass2
20  {
21      public static void main(String[] args)
22      {
23          double luas;
24          variabel p = new variabel();
25          p.panjang = 20;
26          p.lebar = 10;
27          luas = p.panjang * p.lebar;
28          System.out.println("Luas Persegi : " +luas);
29      }
30  }
```

❖ Hasil Pemrograman :

```
Output - examjava8 (run)
run:
Luas Persegi : 200.0
BUILD SUCCESSFUL (total time: 0 seconds)
```

4. Membuat *Method*

❖ Code :

```
10      * @author nurul
11      */
12
13  class kali
14  {
15      double a;
16      double b;
17
18      void hasil()
19      {
20          System.out.println("Hasil perkalian = " + (a*b));
21      }
22  }
```

```
Source  History

23
24      public class HasilPerkalian
25      {
26          public static void main(String[] args)
27          {
28              kali variabel1, variabel2;
29
30              variabel1 = new kali();
31              variabel2 = new kali();
32
33              variabel1.a = 3;
34              variabel1.b = 6;
35
36              variabel2.a = 5;
37              variabel2.b = 10;
38
39              variabel1.hasil();
40              variabel2.hasil();
41          }
42      }
```

❖ Output :

```
Output - examjava8 (run)

run:
Hasil perkalian = 18.0
Hasil perkalian = 50.0
BUILD SUCCESSFUL (total time: 1 second)
```

5. Membuat *Constructor*

❖ Code :

```
Source  History

12
13      class konstruktor
14      {
15          int panjang;
16          int lebar;
17
18          konstruktor()
19          {
20              panjang = 15;
21              lebar = 4;
22          }
23
24          int luas()
25          {
26              return(panjang*lebar);
27          }
28      }
```

```
30  public class hitung
31  {
32      public static void main(String[] args)
33      {
34          konstruktor k1, k2;
35          k1 = new konstruktor();
36          k2 = new konstruktor();
37
38          System.out.println("Luas Persegi Panjang k1 : " +k1.luas());
39          System.out.println("Luas Persegi panjang k2 : " +k2.luas());
40      }
41  }
```

❖ Output :

```
Output - examjava8 (run)
run:
Luas Persegi Panjang k1 : 60
Luas Persegi panjang k2 : 60
BUILD SUCCESSFUL (total time: 1 second)
```

E. TUGAS

1. Dengan menerapkan object dan method, buatlah program java menghitung :
   a. Luas isi tabung
   b. Luas persegi panjang
   c. Luas trapesium
2. Buatlah program prosedural dan Program Berorientasi Objek (OOP) yang dapat menghitung volume Tabung. Tunjukkan dimana letak perbedaan pemrograman prosedural dan OOP.

MODUL 2

MEMBERIKAN INPUT DARI KEYBOARD

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat membuat program yang mampu memperoleh masukan dari keyboard.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software* Netbeans

C. TEORI

Dalam pemrograman Java, mendapatkan inputan dari java dapat menggunakan bantuan library tertentu, antara lain seperti : Scanner, BufferedReader, JoptionPane.

- Scanner menggunakan import java.util.Scanner, dan membuat kelas Scanner tersendiri, yaitu : Scanner sc = new Scanner(System.in)
- BufferedReader menggunakan import java.io.BufferedReader dan import java.io.InputStreamReader, dan membuat kelas BufferedReader tersendiri, yaitu : InputStreamReader isr = new InputStreamReader(System.in) dan BufferedReader br = new BufferedReader(isr)
- JOptionPane menggunakan import javax.swing.JoptionPane, dan membuat kelas JoptionPane tersendiri, yaitu : JOptionPane.showInputDialog

D. PRAKTIKUM

1. Input dengan Scanner

❖ Code :

```java
import java.util.Scanner;

public class belajar_input
{
    public static void main(String[] args)
    {
        String nama, jurusan, fakultas, universitas;

        Scanner input = new Scanner(System.in);
        System.out.print("Nama : ");
        nama = input.nextLine();

        System.out.print("Jurusan : ");
        jurusan = input.nextLine();

        System.out.print("Fakultas : ");
        fakultas = input.nextLine();

        System.out.print("Universitas : ");
        universitas = input.nextLine();
    }
}
```

❖ Output :

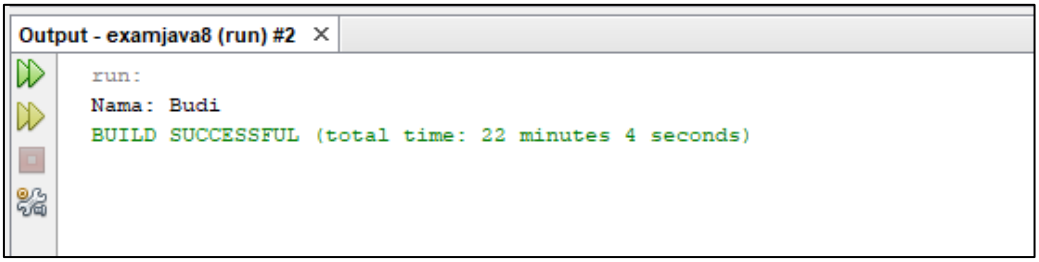
```
Output - examjava8 (run) #2
run:
Nama : Budi
Jurusan : Teknik Informatika
Fakultas : Teknik
Universitas : Universitas Medan Area
BUILD SUCCESSFUL (total time: 1 minute 36 seconds)
```

2. Input dengan BufferedReader

❖ Code :

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class belajar_input2
{
    public static void main(String[] args) throws IOException
    {
        String nama;

        InputStreamReader isr = new InputStreamReader(System.in);

        BufferedReader br = new BufferedReader(isr);

        System.out.print("Nama: ");
        nama = br.readLine();
    }
}
```

❖ Output :

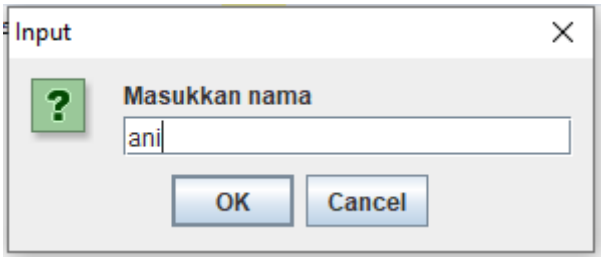


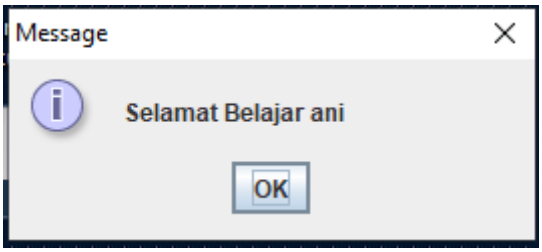
3. Input dengan JoptionPane

❖ Code :

```
8  import javax.swing.JOptionPane;
9
10 public class belajar_input3
11 {
12     public static void main(String[] args)
13     {
14         String nama = " ";
15         nama = JOptionPane.showInputDialog("Masukkan nama");
16         String msg = "Selamat Belajar " +nama;
17         JOptionPane.showMessageDialog(null,msg);
18     }
19 }
20
```

❖ Output :





E. TUGAS

Dengan menggunakan 3 jenis cara membuat inputan (Scanner, BufferedReader, JoptionPane). Buatlah program sederhana yang memiliki inputan untuk menghitung :

a. Keliling lingkaran
b. Luas trapesium
c. Luas prisma segitiga

MODUL 3
STRUKTUR PENGENDALI PROSES

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat membuat program pengendali proses seperti if - else, switch – case, for, while, do-while, break, dan continue.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software* Netbeans

C. TEORI

Pengendalian proses memiliki beberapa jenis, yaitu :

- If – Else :

  Apabila kondisi if benar, maka perintah akan tetap berjalan. Apabila kondisi if tidak benar, maka perintah yang berjalan adalah yang berada di dalam else.
- Switch – Case :

  Kondisi kontrol terdapat pada switch, case menyediakan beberapa pilihan yang dapat dipilih sesuai kondisi
- For :

  Perulangan tetap terjadi apabila syarat perulangan tetap terpenuhi.
- While :

  Perulangan dilakukan dengan terlebih dahulu menguji sebuah pernyataan
- Do - While :

  Perulangan dilakukan selama pernyataan masih bernilai benar
- Break :

  Sebuah statement yang dapat menghentikan kondisi perulangan
- Continue :

  Statement yang digunakan untuk melewati proses perulangan

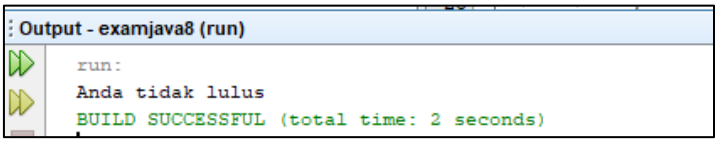
D. PRAKTIKUM

1. IF - Else

❖ **Code :**

```
8   public class Belajar_if_else
9   {
10      public static void main(String [] args)
11      {
12          String kategori = " ";
13          int nilai_UAS_PBO;
14
15          nilai_UAS_PBO = 79;
16
17          if(nilai_UAS_PBO > 85)
18          {
19              kategori = "Anda Lulus";
20          }
21
22          else
23          {
24              kategori = "Anda tidak lulus";
25          }
26
27          System.out.println(kategori);
28      }
29  }
```

❖ **Output :**



2. Switch - Case

❖ **Code :**

```
8   public class Belajar_Switch
9   {
10      public static void main(String[] args)
11      {
12          int pilihan;
13          pilihan = 3;
14
15          switch(pilihan)
16          {
17              case 1 :
18                  System.out.println("Universitas Sumatera Utara");
19                  break;
20
21              case 2:
22                  System.out.println("Universitas Negeri Medan");
23                  break;
24
25              case 3 :
26                  System.out.println("Universitas Islam Negeri Sumatera Utara");
27                  break;
28
29              default:
30              System.out.println("Universitas Medan Area");
31          }
32      }
33  }
```

❖ **Output :**

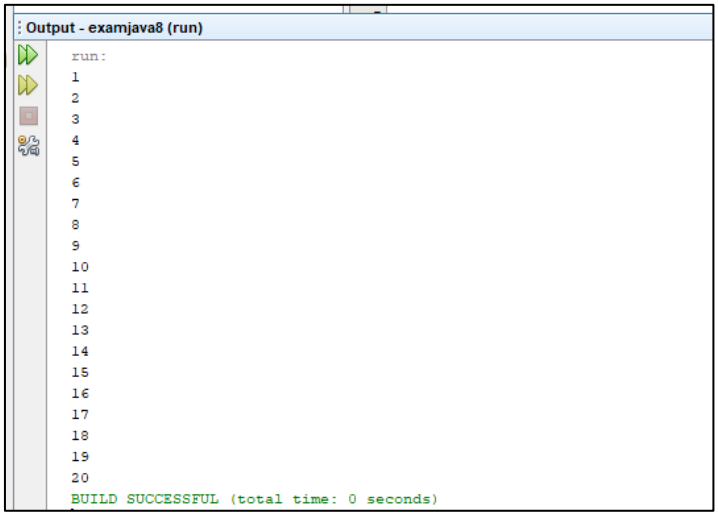
```
Output - examjava8 (run)
run:
Universitas Negeri Medan
BUILD SUCCESSFUL (total time: 0 seconds)
```

3. For – Loop

❖ **Code :**

```
8   public class Belajar_for_loop
9   {
10      public static void main(String[] args)
11      {
12          for(int a=1; a<=20; a++)
13          {
14              System.out.println(a);
15          }
16
17      }
18  }
```

❖ **Output :**

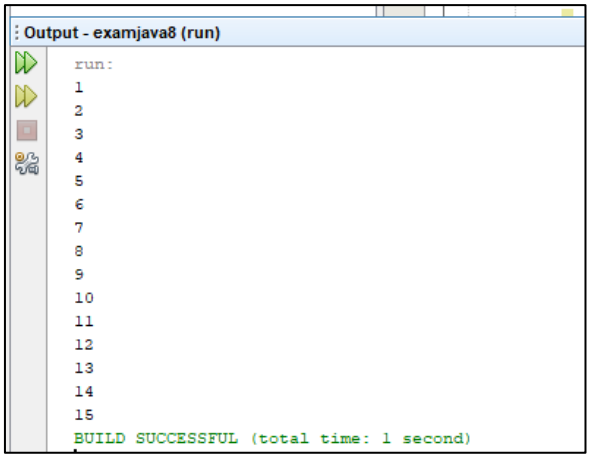


4. While – Loop

❖ **Code :**

```
8   public class Belajar_While
9   {
10      public static void main(String[] args)
11      {
12          int a;
13          a = 1;
14          while(a <= 15)
15          {
16              System.out.println(a);
17              a++;
18          }
19      }
20  }
```

❖ **Output :**



5. Do – While Loop

❖ **Code :**

```
8   public class Belajar_Do_while
9   {
10      public static void main (String[] args)
11      {
12          int a;
13          a=1;
14
15          do
16          {
17              System.out.println(a);
18              a++;
19          }
20
21          while(a<=10);
22      }
23  }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
1
2
3
4
5
6
7
8
9
10
BUILD SUCCESSFUL (total time: 0 seconds)
```

6. Break

❖ **Code :**

```
9   public class Belajar_Break
10  {
11      public static void main(String[] args)
12      {
13          for(int a=1; a<=10; a++)
14          {
15              System.out.println("Selamat Belajar Bahasa Pemrograman Java OOP");
16
17              a = a+1;
18              if(a == 5)
19                  break;
20          }
21
22          System.out.println("Proses berhenti saat perlangan ke-5");
23      }
24  }
```

❖ **Ouput :**

```
Output - examjava8 (run)
run:
Selamat Belajar Bahasa Pemrograman Java OOP
Selamat Belajar Bahasa Pemrograman Java OOP
Selamat Belajar Bahasa Pemrograman Java OOP
Selamat Belajar Bahasa Pemrograman Java OOP
Selamat Belajar Bahasa Pemrograman Java OOP
Proses berhenti saat perlangan ke-5
BUILD SUCCESSFUL (total time: 0 seconds)
```

7. Continue

❖ **Code :**

```
8   public class Belajar_Break
9   {
10      public static void main(String[] args)
11      {
12          int a;
13          int b;
14
15          medan:
16
17          for(a=1; a<=10; a++)
18          {
19              System.out.println("Perulangan pertama");
20
21              a = a+1;
22              if(a == 5)
24              continue medan;
25          }
26
27          System.out.println("Terjadi penghentian dan dilanjutkan ke");
28
29          for(b=1; b<3; b++)
30          {
31              System.out.println("Perulangan kedua");
32          }
33      }
    }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
Perulangan pertama
Perulangan pertama
Perulangan pertama
Perulangan pertama
Perulangan pertama
Terjadi penghentian dan dilanjutkan ke
Perulangan kedua
Perulangan kedua
BUILD SUCCESSFUL (total time: 0 seconds)
```

E. TUGAS

Buatlah program dengan menggunakan for-loop, while-loop, do-while loop untuk mencari :

a. Bilangan prima antara 1-1000
b. Bilangan ganjil dan genap antara 1-1000

MODUL 4
JAVA ARRAY

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan membuat program array dan menerapkannya dalam matriks dengan bahasa pemrograman java.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software Netbeans*

C. TEORI

Array merupakan sekumpulan variabel dengan tipe yang sama. Variabel array adalah lokasi memori tertentu yang memiliki satu nama sebagai identifier, akan tetapi ia dapat menyimpan lebih dari sebuah nilai *(value).* Array dapat menyimpan beberapa item data yang memiliki tipe data yang sama di dalam memori yang berdekatan dan kemudian dibagi menjadi beberapa slot. Array dalam Java terbagi menjadi array satu dimensi dan array dua dimensi.

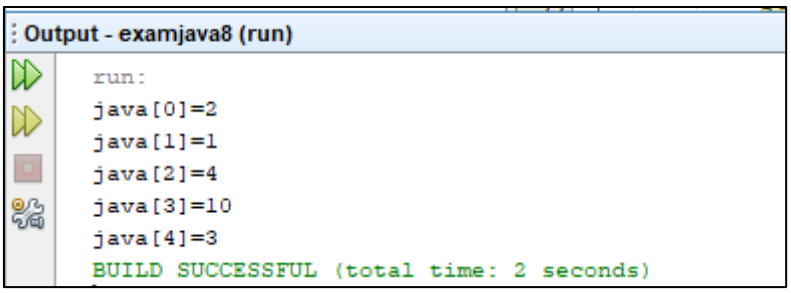
D. PRAKTIKUM

1. Array Satu Dimensi

- **Code 1 :**

```java
public class Array
{
    public static void main(String[] args)
    {
        int java[];
        java = new int[5];

        int i;

        java[0] = 2;
        java[1] = 1;
        java[2] = 4;
        java[3] = 10;
        java[4] = 3;

        for(i=0; i<java.length;i++)
        {
            System.out.println("java["+i+"]="+java[i]);
        }
    }
}
```

❖ **Output Code 1:**



❖ **Code 2 :**

```java
public class rata_rata_array
{
    public static void main(String[] args)
    {
        int javaku[];
        javaku = new int[8];
        int i;
        int bd;
        float total;
        float average;
        bd = 8;
        total = 0;

        javaku[0] = 3;
        javaku[1] = 2;
        javaku[2] = 8;
        javaku[3] = 10;
        javaku[4] = 14;
        javaku[5] = 22;
```

```
32              for(i = 0; i<bd; i++)
33              {
34                  System.out.println("javaku["+i+"] = "+javaku[i]);
35                  total = total + javaku[i];
36              }
37                  average = total/bd;
38                  System.out.println("Total = "+total);
39                  System.out.println("Rata-rata = "+average);
40      }
41  }
42
```

❖ **Ouput Code 2 :**

```
Output - examjava8 (run)
run:
javaku[0] = 3
javaku[1] = 2
javaku[2] = 8
javaku[3] = 10
javaku[4] = 14
javaku[5] = 22
javaku[6] = 0
javaku[7] = 0
Total = 59.0
Rata-rata = 7.375
BUILD SUCCESSFUL (total time: 1 second)
```

2. Array Dua Dimensi

❖ **Code :**

```
13  import java.util.Scanner;
14
15  public class Penjumlahan_Matriks
16  {
17      public static void main(String[] args)
18      {
19       Scanner inputan= new Scanner (System.in);
20          int A[][]=new int[2][2];
21          int B[][]=new int[2][2];
22          int C[][]=new int[2][2];
23
24          System.out.println("Masukkan Nilai Matriks X");
25          System.out.println("========================");
26          for(int i=0;i<2;i++)
27          {
28              for(int j=0;j<2;j++)
29              {
30                  System.out.print("[" +(i+1)+ "][" +(j+1)+ "]:");
31                  A[i][j]=inputan.nextInt();
32              }
33          }
```

```
35              System.out.println("\nMasukkan Nilai Matriks Y");
36              System.out.println("==========================");
37              for(int i=0;i<2;i++)
38              {
39                  for(int j=0;j<2;j++)
40                  {
41                      System.out.print("[" +(i+1)+ "][" +(j+1)+ "]:");
42                      B[i][j]=inputan.nextInt();
43                  }
44              }
45
46              /* Melakukan penjumlahan matriks*/
47              for(int i=0;i<2;i++)
48              {
49                  for(int j=0;j<2;j++)
50                  {
51                      C[i][j]=A[i][j]+B[i][j];
52                  }
53              }
```

```
55              System.out.println("\nHasil penjumlahan Matriks");
56              System.out.println("==========================");
57              for(int i=0;i<2;i++)
58              {
59                  for(int j=0;j<2;j++)
60                  {
61                      System.out.print(+(C[i][j])+" ");
62                  }
63                  System.out.println(" ");
64              }
65          }
66      }
```

Output :

```
Output - examjava8 (run)
run:
Masukkan Nilai Matriks X
========================
[1][1]:2
[1][2]:4
[2][1]:5
[2][2]:12

Masukkan Nilai Matriks Y
==========================
[1][1]:4
[1][2]:6
[2][1]:7
[2][2]:3

Hasil penjumlahan Matriks
==========================
6 10
12 15
BUILD SUCCESSFUL (total time: 27 seconds)
```

E. TUGAS

Buatlah program Java sederhana yang dapat menghitung perkalian dan pengurangan antara matriks X dan Y yang memiliki ordo 2x2. Gunakan switch-case dalam pemilihan perkalian dan pengurangan matriks.

MODUL 5
MEMBUAT CLASS SENDIRI

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan membuat class sendiri, *overloading method*, dan *overloading constructor* dengan bahasa pemrograman java.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. Software *Netbeans*

C. TEORI

- ***Overloading Method***

  Overloading method merupakan sebuah kondisi dimana sebuah class memiliki 2 atau lebih method dengan nama yang sama, namun memiliki jumlah parameter yang berbeda, tipe paramater dan urutan dari tipe data paramater yang berbeda.

- ***Overloading Constructor***

  Overloading constructor merupakan sebuah kondisi dimana terdapat konstruktor yang lebih dari satu dalam sebuah Class, namun memiliki parameter yang berbeda, jumlah parameternya yang berbeda, dan type data parameter yang berbeda.

- **Keyword this**

  Keyword this merupakan keyword yang berfungsi untuk mereferensikan atau mengacu ke objek yang sedang aktif. Keyword this sering digunakan pada overload method dan overload constructor.

D. PRAKTIKUM

1. Membuat Class sendiri :

❖ **Code Class StudentRecord :**

```
12  public class StudentRecord
13  {
14      private String name;
15      private String address;
16      private int age;
17      private double mathGrade;
18      private double englishGrade;
19      private double scienceGrade;
20      private double average;
21
22      private static int studentCount;
23
24      /**
25      * Menghasilkan nama dari Siswa
26      */
27
29      public String getName()
30      {
31          return name;
32      }
33
34      /**
35      * Mengubah nama siswa
36      */
37      public void setName( String temp )
38      {
39          name = temp;
40      }
```

```
41      // area penulisan kode lain
42      /**
43      * Menghitung rata - rata nilai Matematik, Bahasa Inggris, * * Ilmu
44      Pasti
45      */
46      public double getAverage(){
47      double result = 0;
48      result = ( mathGrade+englishGrade+scienceGrade )/3;
49
50      return result;
51      }
52
53      /**
54      * Menghasilkan jumlah instance StudentRecord
55      */
56
57      public static int getStudentCount()
58      {
59          return studentCount;
60      }
61  }
```

❖ **Code StudentRecordExample :**

```
13  public class StudentRecordExample
14  {
15      public static void main(String[] args)
16      {
17          //membuat 3 object StudentRecord
18          StudentRecord annaRecord = new StudentRecord();
19          StudentRecord beahRecord = new StudentRecord();
20          StudentRecord crisRecord = new StudentRecord();
21
22          //Memberi nama siswa
23          annaRecord.setName("Anna");
24          beahRecord.setName("Beah");
25          crisRecord.setName("Cris");
26
27          //Menampilkan nama siswa "Anna"
28          System.out.println(annaRecord.getName());
29
30          //Menampilkan jumlah siswa
31          System.out.println("Count="+StudentRecord.getStudentCount());
32      }
33  }
```

❖ ***Output :***

```
Output - examjava8 (run)
run:
Anna
Count=0
BUILD SUCCESSFUL (total time: 2 seconds)
```

2. Overloading Method

❖ **Code :**

Source History

```
12  public class StudentRecord2
13  {
14      private String name;
15      private String address;
16      private int age;
17      private double mathGrade;
18      private double englishGrade;
19      private double scienceGrade;
20      private double average;
21
22      public void setName(String temp)
23      {
24          name = temp;
25      }
26
27      public String getName()
28      {
29          return name;
30      }
31
32      public void setAddress(String temp)
33      {
34          address = temp;
35      }
```

```
37      public void setAge(int temp)
38      {
39          age = temp;
40      }
41
42      public void setMathGrade(double temp)
43      {
44          mathGrade = temp;
45      }
46
47      public double getMathGrade()
48      {
49          return mathGrade;
50      }
51
52      public void setEnglishGrade(double temp)
53      {
54          englishGrade = temp;
55      }
56
57      public double getEnglishGrade()
58      {
59          return englishGrade;
60      }
```

```
62      public void setScienceGrade(double temp)
63      {
64          scienceGrade = temp;
65      }
66
67      public double getScienceGrade()
68      {
69          return scienceGrade;
70      }
71
72      public double getAverage()
73      {
75          double result = 0;
76          result = ( mathGrade+englishGrade+scienceGrade )/3;
77
78          return result;
79      }
80
81      public void print(String temp)
82      {
83       System.out.println("Name:" + name);
84       System.out.println("Address:" + address);
85       System.out.println("Age:" + age);
87      }
88      public void print(double eGrade, double mGrade, double sGrade)
89      {
90       System.out.println("Name:" + name);
91       System.out.println("Math Grade:" + mGrade);
92       System.out.println("English Grade:" + eGrade);
93       System.out.println("Science Grade:" + sGrade);
94       }
    }
```

❖ ***Output :***

```
Output - examjava8 (run)
run:
Name:Anna
Address:Philippines
Age:15
Name:Anna
Math Grade:80.0
English Grade:95.5
Science Grade:100.0
BUILD SUCCESSFUL (total time: 5 seconds)
```

3. ***Overloading Constructor :***

❖ **Code :**

```
12    class StudentData
13    {
14        private int stuID;
15        private String stuName;
16        private int stuAge;
17
18        StudentData()
19        {
20            //Default constructor
21            stuID = 100;
22            stuName = "New Student";
23            stuAge = 18;
24        }
25        StudentData(int num1, String str, int num2)
26        {
27            //Parameterized constructor
28            stuID = num1;
29            stuName = str;
30            stuAge = num2;
31        }
```

```
32        //Getter and setter methods
33        public int getStuID()
34        {
35            return stuID;
36        }
37
38        public void setStuID(int stuID)
39        {
40            this.stuID = stuID;
41        }
42
43        public String getStuName()
44        {
45            return stuName;
46        }
47
48        public void setStuName(String stuName)
49        {
50            this.stuName = stuName;
51        }
52
53        public int getStuAge()
54        {
55            return stuAge;
56        }
57
```

```
58      public void setStuAge(int stuAge)
59      {
60          this.stuAge = stuAge;
61      }
62
63      public static void main(String args[])
64      {
65          //This object creation would call the default constructor
66          StudentData data1 = new StudentData();
67          System.out.println("Student Name is: "+data1.getStuName());
68          System.out.println("Student Age is: "+data1.getStuAge());
69          System.out.println("Student ID is: "+data1.getStuID());
70
71          /*This object creation would call the parameterized
72           * constructor StudentData(int, String, int)*/
73          StudentData data2 = new StudentData(555, "Chaitanya", 25);
74          System.out.println("Student Name is: "+data2.getStuName());
75          System.out.println("Student Age is: "+data2.getStuAge());
76          System.out.println("Student ID is: "+data2.getStuID());
77      }
78  }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
Student Name is: New Student
Student Age is: 18
Student ID is: 100
Student Name is: Chaitanya
Student Age is: 25
Student ID is: 555
BUILD SUCCESSFUL (total time: 1 second)
```

E. TUGAS

1. Buatlah program *overloading method* yang dapat menampilkan identitas mahasiswa dan prestasi akademiknya, seperti gambaran berikut :

Identitas Mahasiswa	Prestasi Akademik
Nama	Nama
Nim	IP Semester
Jurusan	IP Kumulatif
Fakultas	Riwayat Beasiswa

2. Buatlah program *overloading constructor* yang dapat menampilkan identitas mahasiswa seperti gambaran berikut :

Identitas Mahasiswa
Nama
Nim
Jurusan
Fakultas
IP Semester
IP Kumulatif

MODUL 6

ABSTRAKSI *(ABSTRACTION)* &

PEMBUNGKUSAN *(ENCAPSULATION)*

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat membuat program sederhana yang menerapkan abstraksi *(abstraction)* dan pembungkusan *(encapsulation).*

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. Software *Netbeans*

C. TEORI

- Abstraksi *(Abstraction)* merupakan proses seleksi terhadap aspek-aspek tertentu pada sebuah masalah. Abstraksi sering digunakan untuk menyembunyikan kerumitan dari sebuah proses. Pemahaman tentang pewarisan *(inheritance)* sangat dibutuhkan untuk dapat memahami cara kerja abstraksi.
- Pembungkusan *(Encapsulation)* merupakan proses membuat data objek bersama dengan metodenya menjadi dalam satu paket yang sama.

D. PRAKTIKUM

1. Abstraksi *(Abstraction)*

❖ **Code 1 :**

```
import java.*;
import java.io.*;

abstract class P
{
    abstract public void method2();
    int x,y,z;

    public void method1()
    {
        System.out.println("Method nyata dari class P");
        System.out.println("Nilai x,y ditentukan dalam class P");
        x = 12;
        y = 10;
    }
}
```

```
32    class Q extends P
33    {
34        public void method2()
35        {
36            System.out.println("Method abstract yang sudah menjadi nyata dalam class P");
37            z = x + y;
38
39            System.out.println("Hasil perhitungan = "+z);
40            System.out.println(" ");
41        }
42    }
43
44    class Abstraksi_sederhana
45    {
46        public static void main(String [] args)
47        {
48            Q object = new Q();
49            object.method1();
50            object.method2();
51        }
52    }
```

❖ **Output 1 :**

```
Output - examjava8 (run)
run:
Method nyata dari class P
Nilai x,y ditentukan dalam class P
Method abstract yang sudah menjadi nyata dalam class P
Hasil perhitungan = 22

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

❖ **Code 2 :**

➢ **Class MainMakhlukHidup :**

```
12    public class MainMakhlukHidup
13    {
14        public void cekMakhlukHidup(MakhlukHidup mHidup)
15        {
16            mHidup.berdiri();
17        }
18
19        public static void main(String[] args)
20        {
21            MainMakhlukHidup mh = new MainMakhlukHidup();
22
23            mh.cekMakhlukHidup(new Manusia("Dua Kaki"));
24        }
25    }
```

➢ **Class MakhlukHidup :**

```
public abstract class MakhlukHidup
{
    public abstract void berdiri();
}
```

➢ **Class Manusia :**

```
public class Manusia extends MakhlukHidup
{
        private String duaKaki;
        private String bernafas;

        public Manusia(String duaKaki)
        {
            this.duaKaki = duaKaki;
        }

        public void berdiri()
        {
            System.out.println("Manusia berdiri menggunakan : "+duaKaki);
        }
}
```

❖ **Output Code 2 :**

```
Output - examjava8 (run)
run:
Manusia berdiri menggunakan : Dua Kaki
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Pembungkusan *(Enccapsulation)*

❖ **Code :**

➢ **Class Belajar_Enkapsulasi**

```
public class Belajar_Enkapsulasi
{
    public static void main(String[] args)
    {
        Encapsulation objek = new Encapsulation();
        objek.ModifNama("Aulia Fitri");
        objek.ModifAddress("Jalan Gatot Subroto No.15 Medan");
        objek.ModifNim(191232009);
        System.out.println("Nama : "+objek.getNama());
        System.out.println("Address : "+objek.getAddress());
        System.out.println("Nama : "+objek.getNim());
    }
}
```

➢ **Class Encapsulation**

```
12  public class Encapsulation
13  {
14      private String nama;
15      private String address;
16      private int nim;
17
18      public String getNama()
19      {
20          return this.nama;
21      }
22
23      public String getAddress()
24      {
25          return this.address;
26      }
27
28      public int getNim()
29      {
30          return this.nim;
31      }
32
33      public void ModifNama(String nama)
34      {
35          this.nama = nama;
36      }
```

```
38      public void ModifAddress(String address)
39      {
40          this.address = address;
41      }
42
43      public void ModifNim(int nim)
44      {
45          this.nim = nim;
46      }
47  }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
Nama : Aulia Fitri
Address : Jalan Gatot Subroto No.15 Medan
Nama : 191232009
BUILD SUCCESSFUL (total time: 1 second)
```

E. TUGAS

1. Buatlah program abstraksi sederhana yang dapat menghitung :
   a. Volume balok
   b. Volume bola
2. Buatlah program enkapsulasi sederhana yang dapat menampilkan biodata anda, seperti : nama, nim, jurusan, fakultas, universitas, alamat, email, pekerjaan, hobi, keahlian, dan karya.

MODUL 7

PEWARISAN *(INHERITANCE)* & POLIMORFISME *(POLYMORPHISM)*

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan membuat program pewarisan *(inheritance)* dan polimorfisme *(polymorphism)* sederhana.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software Netbeans*

C. TEORI

- Pewarisan *(Inheritance)* merupakan sebuah kondisi dimana terdapat kelas induk *(super class)* dan kelas anak/ kelas turunan *(sub class)*. Melalui pewarisan, kelas induk akan menurunkan sifat-sifatnya kepada beberapa kelas anak. Sifat-sifat yang dimaksud, dapat berupa variabel, type data, dan method.
- Polimorfisme *(Polymorphism)* merupakan sebuah kondisi dimana sebuah objek dapat mendefenisikan beberapa hal yang berbeda dengan cara yang sama. Dengan adanya polimorfisme, kita dapat melihat beberapa kesamaan antara sebuah kelas dengan kelas yang lain.

D. PRAKTIKUM

1. Pewarisan *(Inheritance)*

❖ **Code :**

➢ **Class Program_Inheritance :**

```
7  import java.util.Scanner;
8
9  public class Program_Inheritance
10 {
11     public static void main(String [] args)
12     {
13         kelasA superClass = new kelasA();
14         kelasB subClass = new kelasB();
15         Scanner input = new Scanner(System.in);
16
17         System.out.println(" SuperClass adalah kelas A");
18         superClass.x = 100;
19         superClass.y = 125;
20         superClass.tampilannilaixy();
21
22         System.out.println("\n SubClass adalah kelas B");
23         subClass.x = 13;
24         subClass.y = 20;
25         subClass.tampilannilaixy();
```

```
27         System.out.print("\n Masukkan nilai z : ");
28         subClass.z = input.nextInt();
29         subClass.tampilkanjumlah();
30     }
31 }
```

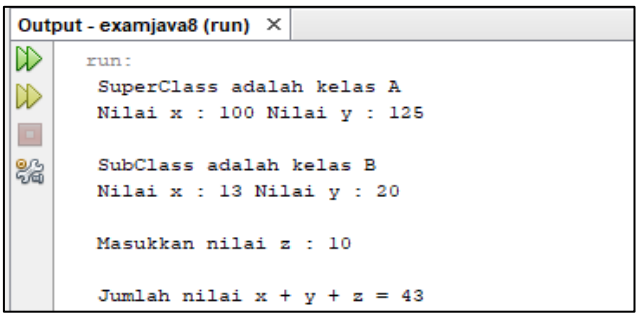
- **Class kelasA :**

```
   class kelasA
13 {
14     int x;
15     int y;
16
17     void tampilannilaixy()
18     {
19         System.out.println(" Nilai x : "+x+" Nilai y : "+y);
20     }
21 }
```

- **Class kelasB :**

```
12 class kelasB extends kelasA
13 {
14     int z;
15
16     void tampilkanjumlah()
17     {
18         System.out.println("\n Jumlah nilai x + y + z = " +(x+y+z));
19     }
20 }
```

❖ **Output :**



2. Polimorfisme *(Polymorphism)*

❖ **Code :**

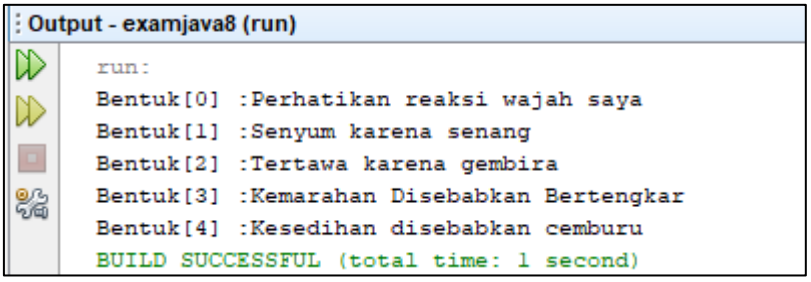
➢ **Class Program Polimorfis :**

```
import java.*;
import java.io.*;

class BentukWajah
{
    public String respons()
    {
        return("Perhatikan reaksi wajah saya");
    }
}

class Senyum extends BentukWajah
{
    public String respons()
    {
        return("Senyum karena senang");
    }
}
```

```
32      class Tertawa extends BentukWajah
33      {
34          public String respons()
35          {
36              return("Tertawa karena gembira");
37          }
38      }
39
40      class Marah extends BentukWajah
41      {
42          public String respons()
43          {
44              return("Kemarahan Disebabkan Bertengkar");
45          }
46      }
47
48      class Sedih extends BentukWajah
49      {
50          public String respons()
51          {
52              return("Kesedihan disebabkan cemburu");
53          }
54      }
```

```
56      public class ProgramPolimorpis
57      {
58          public static void main(String[] args)
59          {
60              BentukWajah objBentuk = new BentukWajah();
61              Senyum objSenyum = new Senyum();
62              Tertawa objTertawa = new Tertawa();
63              Marah objMarah = new Marah();
64              Sedih objSedih = new Sedih();
65
66              BentukWajah[] Bentuk = new BentukWajah[5];
67              Bentuk[0] = objBentuk;
68              Bentuk[1] = objSenyum;
69              Bentuk[2] = objTertawa;
70              Bentuk[3] = objMarah;
71              Bentuk[4] = objSedih;
72
73              System.out.println("Bentuk[0] :"+ Bentuk[0].respons());
74              System.out.println("Bentuk[1] :"+ Bentuk[1].respons());
75              System.out.println("Bentuk[2] :"+ Bentuk[2].respons());
76              System.out.println("Bentuk[3] :"+ Bentuk[3].respons());
77              System.out.println("Bentuk[4] :"+ Bentuk[4].respons());
78          }
79      }
```

❖ **Output :**



E. TUGAS

1. Buatlah program pewarisan *(inheritance)* sederhana. Buatlah sebuah kelas induk yang bernama bangun ruang, dimana kelas induk ini memiliki turunan beberapa kelas sederhana seperti : kubus, balok, tabung, kerucut, limas, prisma, dan bola. Gunakanlah variabel-variabel yang tepat, serta cari dan tampilkanlah nilai volume tujuh bangun ruang tersebut.
2. Buatlah program polimorfisme *(polymorphism)* sederhana yang dapat menampilkan jenis-jenis alat musik seperti : piano, biola, gitar, drum, saxophone, dan trumpet. Gunakanlah variabel dan method yang tepat, agar konsep polimorfisme dapat diterapkan dengan baik.

MODUL 8

EXCEPTION HANDLING

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan mengatasi error dengan *exception handling.*

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software Netbeans*

C. TEORI

Exception handling merupakan teknik yang diperlukan dalam mengatasi error saat program berjalan. Pada umumnya sebuah program akan berhenti secara tiba-tiba/hang apabila terdapat kesalahan/error, untuk menghindari hal ini maka diperlukanlah *exception handling*. *Exception handling* akan dapat membantu program tetap berjalan dan menampilkan hasil, walaupun ada kesalahan yang terjadi secara tiba-tiba. *Exception handling* memiliki beberapa jenis i, yaitu : *try-catch, try-catch-finally, throw, dan throws*. Secara rinci dapat dilihat pada penjelasan berikut ini :

- *try* :
  menentukan bagian program yang akan terjadi pengecualian, *try* harus diikuti dengan *catch* atau *finally*
- *catch* :
  menangani kesalahan, *catch* harus disertai dengan *try* dan *finally*
- *finally* :
  mengeksekusi code yang dianggap penting, *finally* dapat berjalan dengan baik walaupun tidak ada *exception* yang terjadi
- *throw* :
  melempar pengecualian yang terjadi, *throw* digunakan di dalam body code
- *throws* :
  mendeklarasikan pengecualian yang terjadi pada fungsi tertentu

D. PRAKTIKUM

1. Exception Handling dengan Try - Catch

❖ **Code :**

```
12    class NestedTryDemo2
13    {
14        static void nestedTry(String args[])
15        {
16            try
17            {
18                int a = Integer.parseInt(args[0]);
19                int b = Integer.parseInt(args[1]);
20                System.out.println(a/b);
21            }
22
23            catch (ArithmeticException e)
24            {
25                System.out.println("Divide by zero error!");
26            }
27        }
```

```
29        public static void main(String args[])
30        {
31            try
32            {
33                nestedTry(args);
34            }
35
36            catch (ArrayIndexOutOfBoundsException e)
37            {
38                System.out.println("2 parameters are required!");
39            }
40        }
41    }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
2 parameters are required!
BUILD SUCCESSFUL (total time: 1 second)
```

2. **Exeption Handling dengan Try-Catch-Finally**

❖ **Code :**

```
12      class FinallyDemo
13      {
14          static void myMethod(int n) throws Exception
15          {
16              try
17              {
18                  switch(n)
19                  {
20                      case 1: System.out.println("case pertama");
21                      return;
22
23                      case 3: System.out.println("case ketiga");
24
25                      throw new RuntimeException("demo case ketiga");
26
27                      case 4: System.out.println("case keempat");
28
29                      throw new Exception("demo case keempat");
30
31                      case 2: System.out.println("case Kedua");
32                  }
33              }
```

```
35              catch (RuntimeException e)
36              {
37                  System.out.print("RuntimeException terjadi: ");
38                  System.out.println(e.getMessage());
39              }
40
41              finally
42              {
43                  System.out.println("try-block entered.");
44              }
45          }
```

```
47          public static void main(String args[])
48          {
49              for (int i=1; i<=4; i++)
50              {
51                  try
52                  {
53                      FinallyDemo.myMethod(i);
54                  }
55
56                  catch (Exception e)
57                  {
58                      System.out.print("Exception terjadi: ");
59                      System.out.println(e.getMessage());
60                  }
61              System.out.println();
62              }
63          }
64      }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
case pertama
try-block entered.

case Kedua
try-block entered.

case ketiga
RuntimeException terjadi: demo case ketiga
try-block entered.

case keempat
try-block entered.
Exception terjadi: demo case keempat

BUILD SUCCESSFUL (total time: 1 second)
```

3. **Exception Handling dengan Throw**

❖ **Code :**

```
12    class ThrowDemo
13    {
14        public static void main(String args[])
15        {
16            String input = "invalid input";
17            try
18            {
19                if (input.equals("invalid input"))
20                {
21                    throw new RuntimeException("throw demo");
22                }
23                else
24                {
25                    System.out.println(input);
26                }
27
28                System.out.println("After throwing");
29            }
30            catch (RuntimeException e)
31            {
32                System.out.println("Exception caught here.");
33                System.out.println(e);
34            }
35        }
36    }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
Exception caught here.
java.lang.RuntimeException: throw demo
BUILD SUCCESSFUL (total time: 2 seconds)
```

4. **Exception Handling dengan Throws**

❖ **Code :**

➢ **Class BalokDenganException :**

```
12      public class BalokDenganException
13      {
14          private double panjang, lebar, tinggi;
15          private static int jumlahObjek = 0;
16
17          public BalokDenganException(double panjang, double lebar, double tinggi)
18          {
                setPanjang(panjang);
                setLebar(lebar);
                setTinggi(tinggi);
22              jumlahObjek++;
23          }
24
25          public static int getJumlahObjek()
26          {
27              return jumlahObjek;
28          }
29
30          //set dan get method untuk panjang balok
31          public double getPanjang()
32          {
33              return panjang;
34          }
```

```
36          public void setPanjang(double panjang) throws IllegalArgumentException
37          {
38              if (panjang >= 0)
39              this.panjang = panjang;
40
41              else
42              throw new IllegalArgumentException("Nilai panjang dari persegi panjang tidak boleh negatif");
43          }
44
45          //set dan get method untuk lebar balok
46          public double getLebar()
47          {
48              return lebar;
49          }
50
51          public void setLebar(double lebar)throws IllegalArgumentException
52          {
53              if(lebar >= 0)
54              this.lebar = lebar;
55
56              else
57              throw new IllegalArgumentException("Nilai lebar dari persegi panjang tidak boleh negatif");
58      }
```

```
60      //set dan get method untuk lebar balok
61      public double getTinggi()
62      {
63         return tinggi;
64      }
65
66      public void setTinggi(double tinggi)
67      {
68       if(tinggi >= 0)
69         this.tinggi = tinggi;
70
71       else
72         throw new IllegalArgumentException("Nilai tinggi dari persegi panjang tidak boleh negatif");
73      }
74    }
```

➢ **Class TestBalokDenganException :**

```
13    public class TestBalokDenganException
14    {
15     public static void main(String[] args)
16     {
17      try
18      {
19       BalokDenganException balok1 = new BalokDenganException(8.7, 2.8, 1.7);
20       BalokDenganException balok2 = new BalokDenganException(5.2, 3.6, 1.5);
21       BalokDenganException balok3 = new BalokDenganException(3.9, 2.7, 1.7);
22       BalokDenganException balok4 = new BalokDenganException(9.8, 6.7, 3.4);
23       BalokDenganException balok5 = new BalokDenganException(-5.6, 3.9, 1.6);
24      }
25
26      catch(IllegalArgumentException ex)
27      {
28       System.out.println(ex);
29      }
30
31      System.out.println("Jumlah objek yang dibuat: " + BalokDenganException.getJumlahObjek());
32     }
33    }
```

❖ **Output :**

```
Output - examjava8 (run)
run:
java.lang.IllegalArgumentException: Nilai panjang dari persegi panjang tidak boleh negatif
Jumlah objek yang dibuat: 4
BUILD SUCCESSFUL (total time: 1 second)
```

E. TUGAS

Buatlah program sederhana yang menerapkan *exception handling* :

a. ArithmeticException

b. IndexOutOfBoundsException

MODUL 9

GRAPHICAL USER INTERFACE (GUI)

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan membuat program java sederhana dengan GUI.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software Netbeans*

C. TEORI

Graphical User Interface (GUI) merupakan aplikasi dalam java yang berbasis grafik. Bahasa pemrograman java yang dibagun dengan NetBeans memiliki dua kelas GUI, yaitu :

a. AWT *(Abstract Windowing Toolkit)* terdapat dalam package java.awt

b. Swing, terdapat dalam package javax.swing

Dua package berikut ini merupakan package yang unik dan sudah memiliki *event handling* (mekanisme penanganan).

D. PRAKTIKUM

Berikut ini langkah-langkah membuat Kalkulator sederhana dengan java Swing :

1. File - New Project - JFrame Form
2. Buatlah layout seperti berikut ini :

3. Buatlah properties seperti berikut ini :

Komponen	Properties	Text	Edittable	Horizontal Aligmnment
jTextField1	values	0	Hilangkan centang	RIGHT
jButton1	values	<		
jButton2	values	7		
jButton3	values	8		
jButton4	values	9		
jButton5	values	+		
jButton6	values	4		
jButton7	values	5		
jButton8	values	6		
jButton9	values	-		
jButton10	values	1		
jButton11	values	2		
jButton12	values	3		
jButton13	values	x		
jButton14	values	0		
jButton15	values	.		
jButton16	values	=		
jButton17	values	/		
jButton18	values	C		

4. Untuk menambahkan coding pada setiap button dan textfield, dapat dilakukan dengan “klik kanan” pada setiap button/ textfield - Events - Mouse - MouseClicked
5. Pada halaman source, tambahkan setiap coding berikut untuk setiap button :

- Class Kalkulator :

```
12    public class Kalkulator extends javax.swing.JFrame {
13
14        String bil = "";
15        double jumlah, bil1, bil2;
16        int pilih;
```

- jButton1 :

```
293    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
294        // TODO add your handling code here:
295        String nilaiBaru;
296        nilaiBaru = jTextField1.getText().substring(0, jTextField1.getText().length() - 1);
297        bil = nilaiBaru;
298        jTextField1.setText(nilaiBaru);
299        if (jTextField1.getText().length() == 0)
300        {
301            jTextField1.setText("0");
302            bil = "";
303        }
304    }
```

- jButton2 :

```
306    private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
307        // TODO add your handling code here:
308        bil += "7";
309        jTextField1.setText(bil);
310    }
```

- jButton3 :

```
313    private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
314        // TODO add your handling code here:
315        bil += "8";
316        jTextField1.setText(bil);
317    }
```

- jButton4 :

```
319    private void jButton4MouseClicked(java.awt.event.MouseEvent evt) {
320        // TODO add your handling code here:
321        bil += "9";
322        jTextField1.setText(bil);
323    }
```

- jButton5 :

```
325     private void jButton5MouseClicked(java.awt.event.MouseEvent evt) {
326         // TODO add your handling code here:
327         bill = Double.parseDouble(jTextField1.getText());
328         //jTextField1.setText("0");
329         bil = "";
330         pilih = 1;
331     }
```

- jButton6 :

```
333     private void jButton6MouseClicked(java.awt.event.MouseEvent evt) {
334         // TODO add your handling code here:
335         bil += "4";
336         jTextField1.setText(bil);
337     }
```

- jButton7 :

```
339     private void jButton7MouseClicked(java.awt.event.MouseEvent evt) {
340         // TODO add your handling code here:
341         bil += "5";
342         jTextField1.setText(bil);
343     }
```

- jButton8 :

```
345     private void jButton8MouseClicked(java.awt.event.MouseEvent evt) {
346         // TODO add your handling code here:
347         bil += "6";
348         jTextField1.setText(bil);
349     }
```

- jButton9 :

```
351     private void jButton9MouseClicked(java.awt.event.MouseEvent evt) {
352         // TODO add your handling code here:
353         bill = Double.parseDouble(jTextField1.getText());
354         //jTextField1.setText("0");
355         bil = "";
356         pilih = 2;
357     }
```

- jButton10 :

```
359     private void jButton10MouseClicked(java.awt.event.MouseEvent evt) {
360         // TODO add your handling code here:
361         bil += "1";
362         jTextField1.setText(bil);
363     }
```

- jButton11 :

```
365     private void jButton11MouseClicked(java.awt.event.MouseEvent evt) {
366         // TODO add your handling code here:
367         bil += "2";
368         jTextField1.setText(bil);
369     }
```

- jButton12 :

```
371     private void jButton12MouseClicked(java.awt.event.MouseEvent evt) {
372         // TODO add your handling code here:
373         bil += "3";
374         jTextField1.setText(bil);
375     }
```

- jButton13 :

```
377     private void jButton13MouseClicked(java.awt.event.MouseEvent evt) {
378         // TODO add your handling code here:
379         bill = Double.parseDouble(jTextField1.getText());
380         //jTextField1.setText("0");
381         bil = "";
382         pilih = 3;
383     }
```

- jButton14 :

```
385     private void jButton14MouseClicked(java.awt.event.MouseEvent evt) {
386         // TODO add your handling code here:
387         if (!jTextField1.getText().equals("0"))
388         {
389             bil += "0";
390             jTextField1.setText(bil);
391         }
392     }
```

- jButton15 :

```
394     private void jButton15MouseClicked(java.awt.event.MouseEvent evt) {
395         // TODO add your handling code here:
396         if (!jTextField1.getText().contains("."))
397         {
398             if (jTextField1.getText().equals("0"))
399             {
400                 bil += "0.";
401                 jTextField1.setText(bil);
402             }
403             else
404             {
405                 bil += ".";
406                 jTextField1.setText(bil);
407             }
408         }
409     }
```

- jButton16 :

```
411     private void jButton16MouseClicked(java.awt.event.MouseEvent evt) {
412         // TODO add your handling code here:
413         bil2 = Double.parseDouble(jTextField1.getText());
414         double hasil = 0;
415 
416         switch (pilih)
417         {
418             case 1:
419                 hasil = bil1 + bil2;
420                 break;
421             case 2:
422                 hasil = bil1 - bil2;
423                 break;
424             case 3:
425                 hasil = bil1 * bil2;
426                 break;
427             case 4:
428                 hasil = bil1 / bil2;
429                 break;
430         }
431 
432         bil = "";
433         jTextField1.setText(String.valueOf(hasil));
434     }
```

- jButton17 :

```
436     private void jButton17MouseClicked(java.awt.event.MouseEvent evt) {
437         // TODO add your handling code here:
438         bil1 = Double.parseDouble(jTextField1.getText());
439         //jTextField1.setText("0");
440         bil = "";
441         pilih = 4;
442     }
```

- jButton18 :

```
444     private void jButton18MouseClicked(java.awt.event.MouseEvent evt) {
445         // TODO add your handling code here:
446         double hasil;
447         bil = "";
448         bil1 = 0.0;
449         bil2 = 0.0;
450         hasil = 0.0;
451         jTextField1.setText("0");
452     }
```

- Main Program :

```
461     public static void main(String args[])
462     {
463         /* Set the Nimbus look and feel */
464         Look and feel setting code (optional)
485
486         /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable()
488     {
            public void run() {
490         new Kalkulator().setVisible(true);
491         }
492     });
493 }
```

6. Output :

   Pada bagian design kalkulator, klik Run - Run File :

E. TUGAS

Buatlah kalkulator sederhana dengan model anda sendiri, serta tambahkanlah beberapa button berikut :

a. Persen (%)

b. Modulo (mod)

c. Pangkat kuadrat (x^2)

d. Pangkat kubik (x^3)

e. Akar kuadrat ($\sqrt{}$)

MODUL 10

APLIKASI JAVA CRUD DENGAN DATABASE MYSQL

A. TUJUAN

Setelah praktikum ini, praktikan diharapkan dapat memahami dan membuat program aplikasi java CRUD sederhana dengan database MySQL.

B. PERALATAN DAN BAHAN

1. Personal Komputer
2. *Software* Netbeans

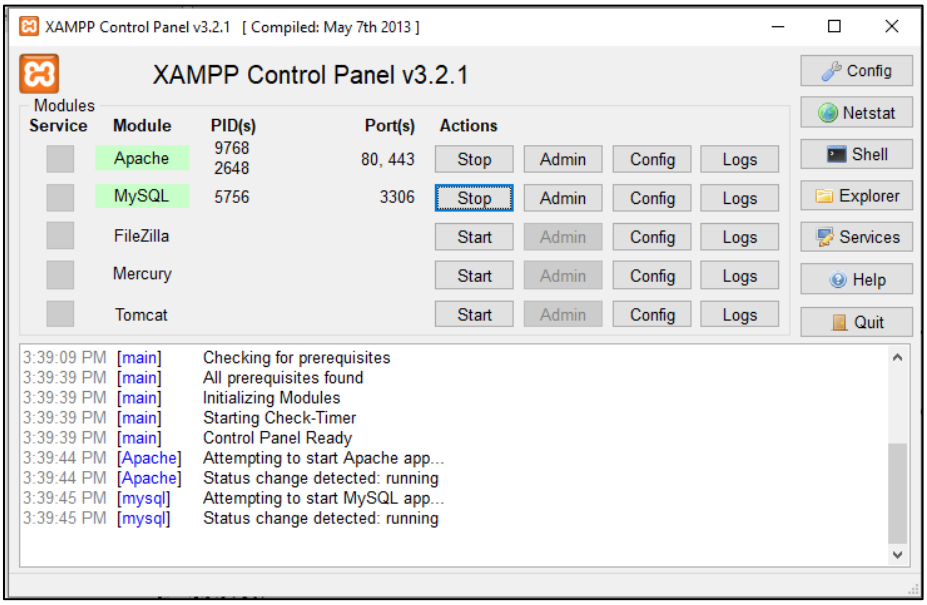
C. TEORI

Java GUI berbasis Swing mampu membuat aplikasi java yang terkoneksi dengan database. Pada bab ini kita akan menggunakan database MySQL dan software XAMPP. Adanya koneksi antara java dengan database memungkinkan kita dapat menambah data, mengupdate data/ edit data, serta menghapus data yang ada pada database. Dalam menghubungkan coding pada java dan database MySQL kita membutuhkan library tambahan yaitu MySQL JDBC *(Java Data Base Connectivity).*

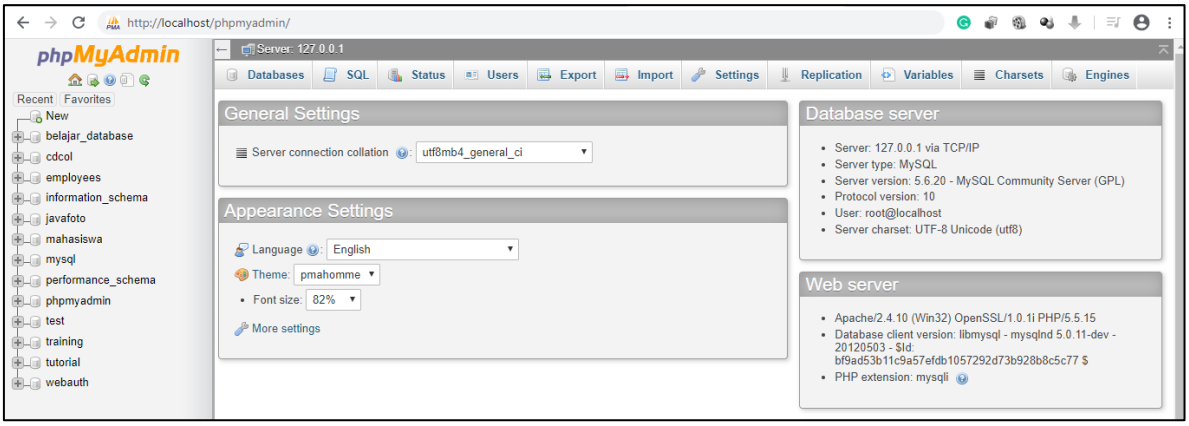
D. PRAKTIKUM

Berikut ini langkah-langkah membuat Aplikasi CRUD (Create, Update, Delete) pada Java dengan koneksi database MySQL.

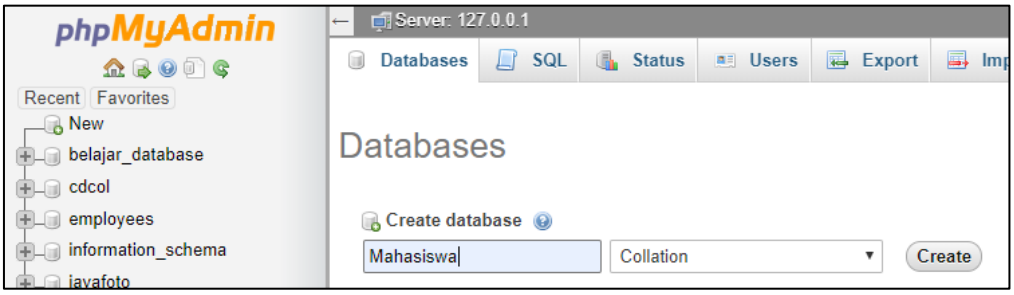
1. Aktifkan XAMPP, kemudian aktifkan Apache dan MySQL dengan menekan tombol Start :



2. Aktifkan Browser, dan ketikkan : http://localhost/phpmyadmin/



3. Buatlah database dengan nama Mahasiswa

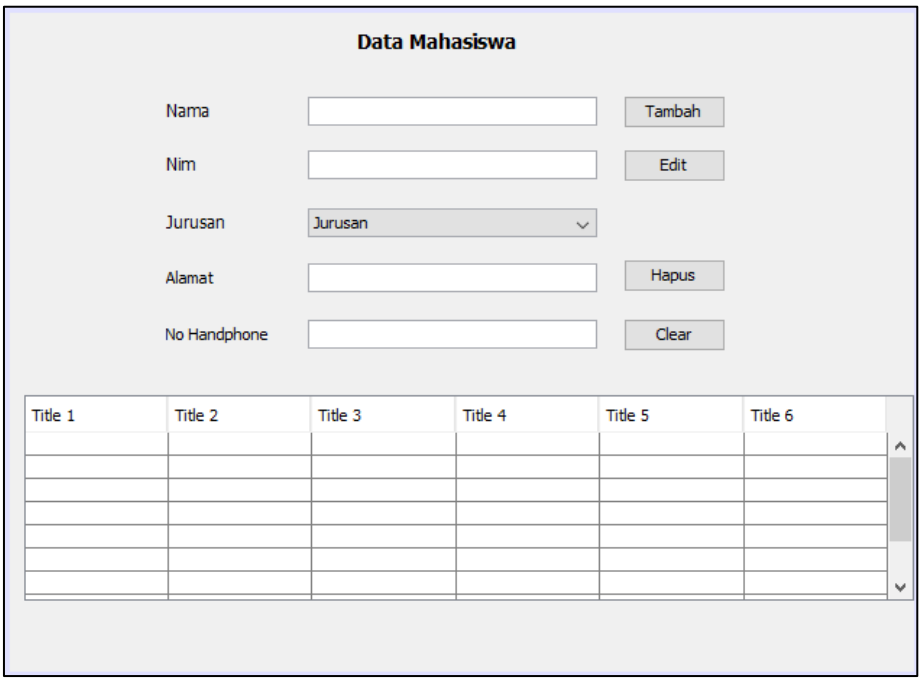


4. Buat tabel dengan nama table_mhs yang memiliki 5 kolom dengan struktur sebagai berikut :

Browse | Structure | SQL | Search | Insert | Export | Import | Privilege

#	Name	Type	Collation	Attributes	Null	Default	Extra	Action
1	nama	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary
2	nim	int(20)			No	None		Change Drop Primary
3	jurusan	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary
4	alamat	varchar(225)	latin1_swedish_ci		No	None		Change Drop Primary
5	no_telpon	varchar(20)	latin1_swedish_ci		No	None		Change Drop Primary

5. Buatlah project baru dengan cara klik File - New Project - Java - Java Application. Berilah nama project dengan CRUD_Java
6. Buatlah Jframe baru pada project CRUD_Java dengan cara klik kanan pada project CRUD_Java - New - Jframe Form. Berilah nama file dengan Data_Mahasiswa
7. Buatlah tampilan GUI seperti berikut ini :



8. Tambahkan library MySQL JDBC pada project CRUD_Java dengan cara klik kanan pada library - add library - MySQL JDBC Driver
9. Pada bagian source umum pada class Data_Mahasiswa, tambahkan coding berikut :

```
13  import java.sql.Connection;
14  import java.sql.DriverManager;
15  import java.sql.SQLException;
16  import javax.swing.JOptionPane;
17  import javax.swing.table.DefaultTableModel;
```

```
26  public class Data_Mahasiswa extends javax.swing.JFrame {
27
28      private static Connection mysqlconfig;
29
30      public static Connection configDB()throws SQLException
31      {
32          try {
33              String url="jdbc:mysql://localhost:3306/Mahasiswa";
34              String user="root"; //user database
35              String pass=""; //password database
36              DriverManager.registerDriver(new com.mysql.jdbc.Driver());
37              mysqlconfig=DriverManager.getConnection(url, user, pass);
38          }
            catch (Exception e)
40          {
41              System.err.println("koneksi gagal "+e.getMessage());
42          }
43          return mysqlconfig;
44      }
```

10. Mengaktifkan button Tambah dapat dilakukan dengan cara klik kanan button tambah - Event - Action - ActionPerformed, lalu tambahkan coding berikut :

```
262             try
263             {
264                 String sql = "INSERT INTO table_mhs VALUES ('"+jTextField1.getText()+"','"+
265                                                             jTextField2.getText()+"','"+
266                                                             jComboBox1.getSelectedItem()+"','"+
267                                                             jTextField3.getText()+"','"+
268                                                             jTextField4.getText()+"')";
269                 java.sql.Connection conn=(Connection)Data_Mahasiswa.configDB();
270                 java.sql.PreparedStatement pst=conn.prepareStatement(sql);
271                 pst.execute();
272                 JOptionPane.showMessageDialog(null, "Penyimpanan Data Berhasil");
273             }
                catch (Exception e)
275             {
276                 JOptionPane.showMessageDialog(this, e.getMessage());
277             }
```

11. Menampilkan hasil penambahan data ke dalam tabel dapat dilakukan dengan menambahkan coding berikut pada source umum di kelas Data_Mahasiswa :

```
341        private void load_table()
342        {
343            // membuat tampilan model tabel
344            DefaultTableModel model = new DefaultTableModel();
345            model.addColumn("No");
346            model.addColumn("Nama");
347            model.addColumn("NIM");
348            model.addColumn("Jurusan");
349            model.addColumn("Alamat");
350            model.addColumn("Phone");
351
352            try //menampilkan data database kedalam tabel
353            {
354                int no=1;
355                String sql = "select * from table_mhs";
356                java.sql.Connection conn=(Connection)Data_Mahasiswa.configDB();
357                java.sql.Statement stm=conn.createStatement();
358                java.sql.ResultSet res=stm.executeQuery(sql);
359                while(res.next())
360                {
361                    model.addRow(new Object[]{no++,res.getString(1),res.getString(2),
362                                              res.getString(3),res.getString(4),
363                                              res.getString(5)});
364                }
```

```
365                jTable2.setModel(model);
366            }
               catch (Exception e)
368            {
369
370            }
```

12. Tambahkan class load_table ke dalam class Mahasiswa, seperti pada coding berikut ini :

```
23         public Data_Mahasiswa()
24         {
25             initComponents();
26             load_table();
```

13. Untuk menghapus Text Field setelah menambahkan data, dapat dilakukan dengan menambahkan coding berikut ini :

```
373        private void kosong()
374        {
375            jTextField1.setText(null);
376            jTextField2.setText(null);
377            jTextField3.setText(null);
378            jTextField4.setText(null);
379            jComboBox1.setSelectedItem(this);
380        }
```

14. Tambahkan class kosong ke dalam class Mahasiswa, seperti coding berikut ini :

```
23      public Data_Mahasiswa()
24      {
25          initComponents();
26          load_table();
27          kosong();
28      }
```

15. Untuk menambahkan data otomatis dan menghapus isi Text Field secara otomatis, dapat dilakukan dengan menambahkan class load_table dan class kosong pada coding button Tambah seperti dibawah ini :

```
262             try
263             {
264                 String sql = "INSERT INTO table_mhs VALUES ('"+jTextField1.getText()+"','"+
265                                                              jTextField2.getText()+"','"+
266                                                              jComboBox1.getSelectedItem()+"','"+
267                                                              jTextField3.getText()+"','"+
268                                                              jTextField4.getText()+"')";
269                 java.sql.Connection conn=(Connection)Data_Mahasiswa.configDB();
270                 java.sql.PreparedStatement pst=conn.prepareStatement(sql);
271                 pst.execute();
272                 JOptionPane.showMessageDialog(null, "Penyimpanan Data Berhasil");
273             }
                catch (Exception e)
275             {
276                 JOptionPane.showMessageDialog(this, e.getMessage());
277             }
278
279             load_table();
280             kosong();
```

16. Mengedit data pada tabel dapat dilakukan dengan klik kanan pada tabel - Event - Mouse - MouseClicked, kemudian tambahkan coding berikut :

```
328             int baris = jTable2.rowAtPoint(evt.getPoint());
329             String nama = jTable2.getValueAt(baris, 1).toString();
330             jTextField1.setText(nama);
331             String nim = jTable2.getValueAt(baris,2).toString();
332             jTextField2.setText(nim);
333             String jr = jTable2.getValueAt(baris, 3).toString();
334             jComboBox1.setSelectedItem(jr);
335             String alamat=jTable2.getValueAt(baris, 4).toString();
336             jTextField3.setText(alamat);
337             String no_telpon = jTable2.getValueAt(baris, 5).toString();
338             jTextField4.setText(no_telpon);
```

17. Mengaktifkan tombol Edit dapat dilakukan dengan klik kanan pada tombol Edit - Event - Action - actionPerformed, kemudian tambahkan coding berikut :

```
290             try
291             {
292                 String sql = "UPDATE table_mhs SET nama = '"+jTextField1.getText()+"', nim = '"+
293                                                     jTextField2.getText()+"', jurusan = '"+
294                                                     jComboBox1.getSelectedItem()+"',alamat= '"+
295                                                     jTextField3.getText()+"',no_telpon = '"+
296                                                     jTextField4.getText()+"' WHERE nim = '"+
297                                                     jTextField2.getText()+"'";
298
299                 java.sql.Connection conn=(Connection)Data_Mahasiswa.configDB();
300                 java.sql.PreparedStatement pst=conn.prepareStatement(sql);
301                 pst.execute();
302                 JOptionPane.showMessageDialog(null, "data berhasil di edit");
303             }
304             catch (Exception e)
305             {
306                 JOptionPane.showMessageDialog(null, "Perubahan Data Gagal"+e.getMessage());
307             }
308             load_table();
309             kosong();
```

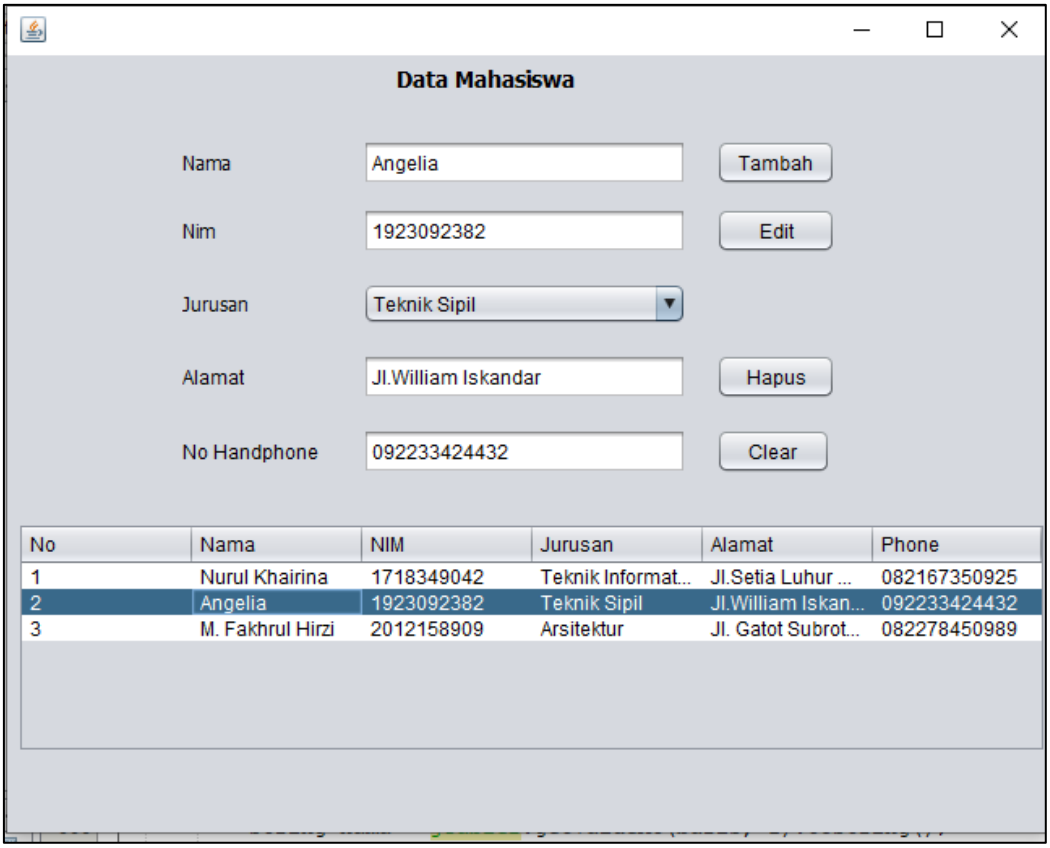
18. Mengaktifkan tombol Hapus dapat dilakukan dengan klik kanan pada tombol Hapus - Event - actionPerformed, kemudian tambahkan coding berikut :

```
314             try {
315                 String sql ="delete from table_mhs where nim='"+jTextField2.getText()+"'";
316                 java.sql.Connection conn=(Connection)Data_Mahasiswa.configDB();
317                 java.sql.PreparedStatement pst=conn.prepareStatement(sql);
318                 pst.execute();
319                 JOptionPane.showMessageDialog(this, "berhasil di hapus");
320             } catch (Exception e) {
321                 JOptionPane.showMessageDialog(this, e.getMessage());
322             }
323             load_table();
324             kosong();
```

19. Mengaktifkan tombol Clear dapat dilakukan dengan klik kanan pada tombol Clear - Event - actionPerformed, kemudian tambahkan coding berikut :

```
327     private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
328         // TODO add your handling code here:
329         kosong();
330     }
```

20. Berikut ini hasil tampilan aplikasi CRUD :



E. TUGAS

Dari program CRUD Data Mahasiswa yang telah anda buat, tambahkanlah data :

1. Fakultas
2. Universitas
3. Bidang Peminatan

Penambahan data pada aplikasi CRUD Data_Mahasiswa meliputi :

a. Penambahan kolom fakultas, universitas, bidang_peminatan pada database melalui http://localhost/phpmyadmin/
b. Penambahan label Fakultas, Universitas, Bidang Peminatan pada GUI Java Jframe
c. Penambahan jTextField5, jTextField6, jTextField7 untuk masing-masing data Fakultas, Universitas, dan Bidang Peminatan
d. Penambahan 3 kolom tabel untuk Fakultas, Universitas, dan Bidang Peminatan
e. serta penyesuaian coding dan penambahan coding-coding tertentu yang diperlukan.

DAFTAR PUSTAKA

Avestro, Joyce. 2007. JENI : Pengenalan Pemrograman I. Jardiknas.

Brett Spell. 2000. Proffesional Java Programming. United States : Wrox Press Ltd.

Siallagan, Sariadin. 2009. Pemrograman Java : Dasar-dasar pengenalan dan Pemahaman. Yogyakarta : Penerbit Andi.

Xiaoping Jia. 2000. Object Oriented Software Development Using Java. United States of America : Addion Wesley Logman, Inc.

https://www.guru99.com/java-exception-handling.html

https://socs.binus.ac.id/2018/12/05/exception-handling/

http://ilmukita.org/membuat-kalkulator-sederhana-dengan-java-netbeans/

https://www.malasngoding.com/cara-membuat-crud-dengan-java-mysql-part-1/

https://www.malasngoding.com/cara-membuat-crud-dengan-java-mysql-part-2/

LAMPIRAN

Format Pengumpulan Tugas :

Nama :

NPM :

Coding :

Screenshot Running Program :

Tugas setiap bab pertemuan diupload dalam versi PDF (Max 2 MB) melalui Elearning UMA dengan format nama file :

Nama_NPM_Tugas[Bab xx]

Contoh nama file Tugas Bab 1:

Budi_1812158909_Tugas[1]