Laporan Praktikum Pemrograman Berbasis Objek

Ke – 2

Modul 4 : Pengenalan Pemrograman Berorientasi Objek



Oleh :

Risyad Pangestu

140810170003

TEKNIK INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS PADJADJARAN

2018

**A. Tujuan Pembelajaran**

1. Mendeklarasikan suatu class
2. Mendeklarasikan suatu atribut
3. Mendeklarasikan suatu method
4. Mengakses anggota suatu obyek
5. Mendeklarasikan static
6. Mendeklarasikan package dan import

**B. Dasar Teori**

Deklarasi class dapat dilakukan dengan sintaks sebagai berikut:

<modifier> class <nama\_class> {

[deklarasi\_atribut]

[deklarasi\_konstruktor]

[deklarasi\_metode]

}

Contoh:

public class Mahasiswa {

...

}

Deklarasi atribut dapat dilakukan dengan sintaks sebagai berikut:

<modifier> <tipe> <nama\_atribut> ;

Contoh:

public class Mahasiswa {

public String npm;

public String nama;

}

Deklarasi metode dapat dilakukan dengan sintaks sebagai berikut:

<modifier> <return\_type> <nama\_metode> ([daftar\_argumen]) {

[<statement>]

}

public class Mahasiswa {

private String npm;

private String nama;

public static void cetak(){

System.out.println("Mahasiswa Teknik Informatika") ;

}

public static void main(String args[]){

cetak();

}

}

Untuk dapat mengakses anggota-anggota dari suatu obyek, maka harus dibuat instance dari class tersebut terlebih dahulu. Berikut ini adalah contoh pengaksesan anggota-anggota dari class Mahasiswa:

public class Mahasiswa {

private String npm;

private String nama;

public void cetak(){

System.out.println("Mahasiswa Teknik Informatika") ;

}

public static void main(String args[]){

Mahasiswa mhs =new Mahasiswa();

mhs.cetak();

}

}

Kita dapat menyembunyikan information dari suatu class sehingga anggota- anggota class tersebut tidak dapat diakses dari luar. Adapun caranya adalah cukup dengan memberikan akses kontrol private ketika mendeklarasikan suatu atribut atau method.

Contoh:

private String NPM;

**Modifier**

Modifier digunakan untuk mengatur hak akses antar/inter class dan member class. Modifier terdiri dari private, protected, public, dan internal. Ada istilah non- modifier, kondisi ini memungkinkan class dapat mengakses class lainnya yang berada dalam satu package tetapi tidak berlaku pada sub class.

**Enkapsulasi**

Encapsulation (Enkapsulasi) adalah suatu cara untuk menyembunyikan implementasi detail dari suatu class. Enkapsulasi mempunyai dua hal mendasar, yaitu :

1. information hiding

2. menyediakan suatu perantara (method) untuk pengaksesan data

**Static**

Static (atau disebut “class variable”) adalah sebuah keyword dalam java yang memungkinkan member class dapat diakses oleh kelas yang berbeda tanpa melalui instansiasi.

**Package**

Package adalah suatu cara untuk mengatur class-class yang kita buat. Package akan sangat bermanfaat jika class-class yang kita buat sangat banyak sehingga perlu dikelompokkan berdasarkan kategori tertentu.

Yang perlu kita perhatikan pada saat deklarasikan package, bahwa class tersebut harus disimpan pada suatu direktori yang sama dengan nama package-nya. Suatu class dapat

meng-import class lainnya sesuai dengan nama package yang dipunyainya. Contoh:

import tes.Mahasiswa;

public class TesMahasiswa {

...

}

Satu hal yang perlu kita ketahui, pada saat kita ingin meng-import suatu class dalam suatu package, pastikan letak package tersebut satu direktori dengan class yang ingin meng-import.

Kata kunci this sangat berguna untuk menunjukkan suatu member dalam class- nya sendiri. This dapat digunakan baik untuk data member maupun untuk function member, serta dapat juga digunakan untuk konstruktor. Adapun format penulisannya adalah:

this.data\_member ->merujuk pada data member

this.function\_member() ->merujuk pada function member

this() ->merujuk pada konstruktor

Contoh:

class Parent {

public int x = 5;

}

class Child extends Parent {

public int x = 10;

public void Info() {

System.out.println(super.x);

}

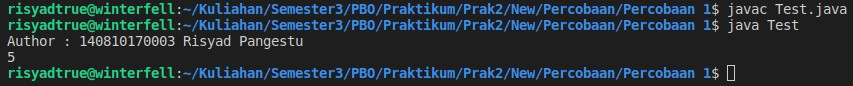
}

Ketika kita memakai konsep inheritance, maka yang harus kita ketahui adalah konstruktor dari parent class tidak dapat diwariskan ke subclass-nya. Sebagai konsekuensinya adalah setiap kali kita membuat suatu subclass, maka kita harus memanggil konstruktor parent class di konstruktor subclass. Jika kita tidak mendeklarasikannya secara eksplisit, maka kompiler Java akan menambahkan deklarasi pemanggilan kontruktor parent class di konstruktor subclass.

Percobaan :

Percobaan 1 :

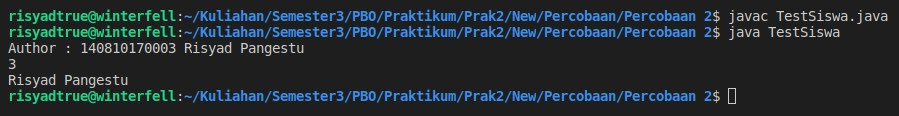
Output :



Analisa : Menjalankan program dengan class main yang ditempatkan berbeda lembar dengan mebua objek dari class tersebut.

Percobaan 2 :

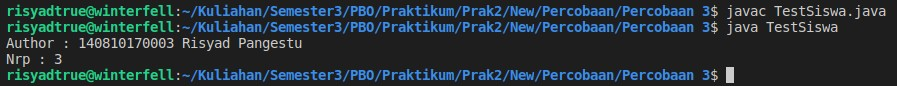
Output :



Analisa : Hampir sama dengan lat 1 namun proses pemanggilan menggunakan method void.

Percobaan 3 :

Output :



Analisa :

Sama seperti tujuan latihan sebelumnya, nanum dalam program in terdapat konstruktor dan method yang berbeda.

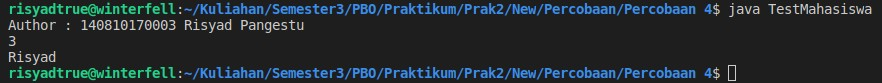
Percobaan 4 :

Code :

|  |
| --- |
| public class Mahasiswa      {          private int nrp;          private String nama;          public Mahasiswa(int nrp, String nama)              {                  this.nrp = nrp;                  this.nama = nama;              }          public int getNrp(){              return nrp;          }            public String getNama(){              return nama;          }          public void setNrp(int i){              nrp = i;          }          public void setNama(String j){              nama = j;          }        } |

|  |
| --- |
| public class TestMahasiswa      {          public static void main(String[] args)              {                  System.out.println("Author : 140810170003 Risyad Pangestu");                  Mahasiswa mhs = new Mahasiswa(03, "Risyad Pangestu");                  mhs.setNrp(03);                  mhs.setNama("Risyad");                  mhs.getNrp();                  mhs.getNama();                  System.out.println(mhs.getNrp());                  System.out.println(mhs.getNama());              }      } |

Output :



Analisa : Sama seperti sebelumnya, yaitu mengakses suatu class namun terlebih

dahulu dilakukan enkapsulasi menggunakan konstruktor yang berfungsi untuk

menyembunyikan implementasi detail dari suatu class

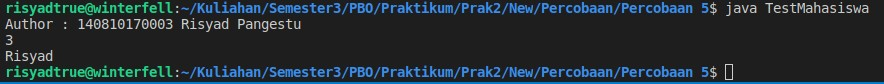
Percobaan 5 :

Code :

|  |
| --- |
| public class Mahasiswa{          public int npm;          public String nama;            public Mahasiswa(int npm, String nama){              this.npm = npm;              this.nama = nama;          }          public void cetak(){              System.out.println(this.npm);              System.out.println(this.nama);          }  } |

|  |
| --- |
| public class TestMahasiswa      {          public static void main(String[] args)              {                  System.out.println("Author : 140810170003 Risyad Pangestu");                  Mahasiswa mhs = new Mahasiswa(03,"Risyad");                  mhs.cetak();              }      } |

Output :



Analisa : Penggunaan “this” pada konstruktor untuk merujuk pada variable global

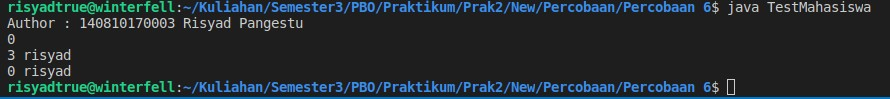
Percobaan 6 :

Code :

|  |
| --- |
| public class Mahasiswa      {          private int npm;          private String nama;          public Mahasiswa(){                  this(0,"");              }          public Mahasiswa(String nama){                  this(0,nama);              }          public Mahasiswa(int npm, String nama){                  this.npm = npm;                  this.nama = nama;              }          // Print Constructor          public void cetak(){              System.out.print(this.npm + " ");              System.out.print(this.nama);              System.out.print("\n");          }      } |

|  |
| --- |
| public class TestMahasiswa{          public static void main(String[] args)              {                  System.out.println("Author : 140810170003 Risyad Pangestu");                  Mahasiswa mhs2 = new Mahasiswa();                  mhs2.cetak();                  Mahasiswa mhs1 = new Mahasiswa(03,"risyad");                  mhs1.cetak();                  Mahasiswa mhs3 = new Mahasiswa("risyad");                  mhs3.cetak();              }      } |

Output :



Analisa : Penggunaan konstruktor overloading (banyak parameter)

Percobaan 7 :

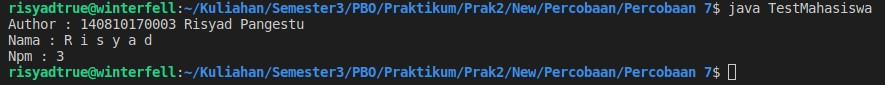
Code :

|  |
| --- |
| package sekolah;  public class Kelas      {          private int kodekelas;          private String namakelas;          private Mahasiswa mahasiswa;          public Kelas(int kode, String nama)              {                  this.kodekelas = kode;                  this.namakelas = nama;              }          public void setMhs(Mahasiswa mhs)              {                  this.mahasiswa = mhs;              }          public void cetakMhs()              {                  System.out.println("Nama : " + this.mahasiswa.getNama());                  System.out.println("Npm : " + this.mahasiswa.getNpm());              }      } |

|  |
| --- |
| package sekolah;  public class Mahasiswa      {          private int npm;          private String nama;          public Mahasiswa(int npm, String nama)              {                  this.npm = npm;                  this.nama = nama;              }          public int getNpm()              {                  return npm;              }          public String getNama()              {                  return nama;              }      } |

|  |
| --- |
| import sekolah.\*;  public class TestMahasiswa{          public static void main(String[] args)              {                  System.out.println("Author : 140810170003 Risyad Pangestu");                  Kelas kls = new Kelas(14, "Informatika!");                  kls.setMhs(new Mahasiswa(03, "R i s y a d"));                  kls.cetakMhs();              }  } |

Output :



Analisa : Penggunaan package untuk memanggil class class yang nantinya bersatu dalam satu program.

Latihan

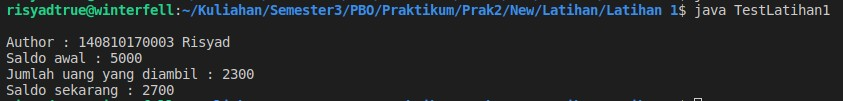
Latihan 1 :

Code :

|  |
| --- |
| public class Tabungan{          public int saldo;          public Tabungan(int initsaldo){                  saldo = initsaldo;              }          public int getSaldo(){                  return saldo;              }          public void simpanUang(int jumlah){                  saldo = saldo+jumlah;              }          public boolean ambilUang(int jumlah){                  int temp = saldo - jumlah;                  if (temp < 0)                      return false;                  else                      {                          saldo = saldo-jumlah;                              return true;                      }              }  } |

|  |
| --- |
| public class TestLatihan1      {          public static void main(String[] args)              {                  System.out.println("\nAuthor : 140810170003 Risyad");                  Tabungan tabungan = new Tabungan(5000);                  System.out.println("Saldo awal : " + tabungan.saldo);                  tabungan.ambilUang(2300);                  System.out.println("Jumlah uang yang diambil : 2300");                  System.out.println("Saldo sekarang : " + tabungan.saldo);              }      } |

Output :



Analisa : Mengakses dari class yang berbeda dengan menggunakan operator kondisi

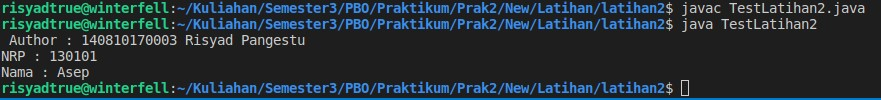
Latihan 2 :

Code :

|  |
| --- |
| public class Mahasiswa{  private int nrp;  private String nama;  public Mahasiswa(int i, String n){  nrp = i;  nama = n;  }  public int getNrp(){  return nrp;  }  public String getNama(){  return nama;  }  } |

|  |
| --- |
| public class TestLatihan2{  public static void main(String args[]){  System.out.println(" Author : 140810170003 Risyad Pangestu ");  Mahasiswa mhs = new Mahasiswa(130101, "Asep");  System.out.println("NRP : " + mhs.getNrp());  System.out.println("Nama : " + mhs.getNama());  }  } |

Output :



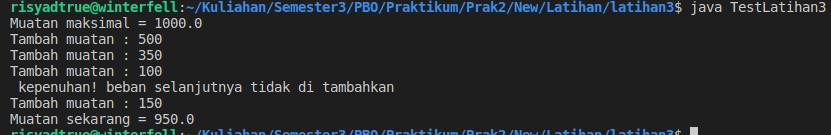
Analisa : Mengakses dari class yang berbeda menggunakan konstruktor

Latihan 3 :

Code :

|  |
| --- |
| public class Truk{  private double muatan;  private double muatanmaks;  public Truk(double beratmaks){  muatanmaks = beratmaks;  muatan = 0;  }  public double getMuatan(){  return muatan;  }  public double getMuatanMaks(){  return muatanmaks;  }  public void tambahMuatan(double berat){  double temp;  temp = muatan + berat;  if (temp > muatanmaks){  System.out.println(" kepenuhan! beban selanjutnya tidak di tambahkan");  }  else  muatan = muatan + berat;  }  }  // lupa Author |

Output :



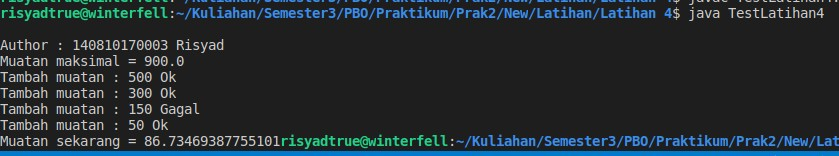
Analisa : Mengakses dari class lain.

Latihan 4 :

Code :

|  |
| --- |
| public class Truck      {          private double muatan;          private double muatanmaks;          public Truck(double beratmaks)              {                  muatanmaks = beratmaks;                  muatan=0;              }          public double getMuatan()              {                  return this.newtsToKilo(this.muatan);              }          public double getMuatanMaks()              {                  return muatanmaks;              }          public boolean tambahMuatan(double berat)              {                  double temp = muatan + berat;                  if (temp > muatanmaks)                      {                          return false;                      }                  else                      {                          muatan = muatan + berat;                          return true;                      }              }          public double newtsToKilo(double berat)              {                  return 1/9.8\*berat;              }          public double kiloToNewts(double berat)              {                  return 9.8\*berat;              }      } |

Output :



Analisa : Mengakses dari class lain dan juga menggunakan operator kondisi

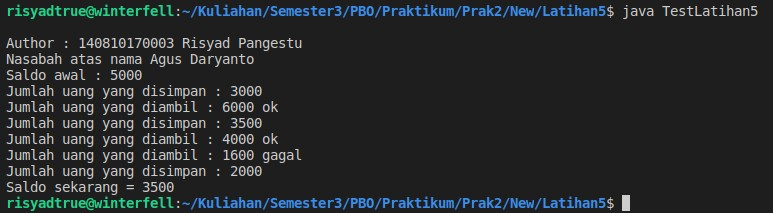
Latihan 5 :

Code :

|  |
| --- |
| package perbankan;  public class Nasabah      {          private String namaAwal;          private String namaAkhir;          private Tabungan tabungan;          public Nasabah(String namaAwal, String namaAkhir)              {                  this.namaAwal = namaAwal;                  this.namaAkhir = namaAkhir;              }          public String getNamaAwal()              {                  return namaAwal;              }          public String getNamaAkhir()              {                  return namaAkhir;              }          public Tabungan getTabungan()              {                  return tabungan;              }          public void setTabungan(Tabungan tabungan)              {                  this.tabungan = tabungan;              }      } |

|  |
| --- |
| package perbankan;  public class Tabungan      {          public int saldo;          public Tabungan(int saldo)              {                  this.saldo = saldo;              }          public int getSaldo()              {                  return saldo;              }          public void simpanUang(int jumlah)              {                  saldo = saldo + jumlah;              }          public boolean ambilUang(int jumlah)              {                  int temp = saldo - jumlah;                  if (temp < 0)                      return false;                  else                      {                          saldo = saldo-jumlah;                          return true;                      }              }      } |

Output :



Analisa : Menggunakan package untuk mengakses dari foder yang berbeda

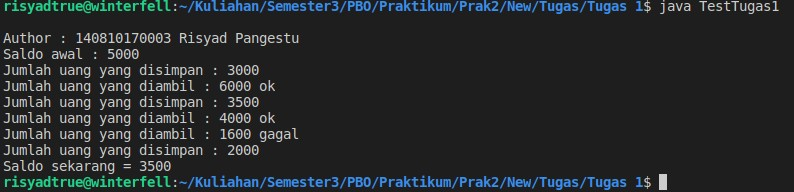
Tugas

Tugas 1 :

Code :

|  |
| --- |
| public class Tabungan      {          public int saldo;          public Tabungan(int initsaldo)              {                  saldo = initsaldo;              }          public int getSaldo()              {                  return saldo;              }          public void simpanUang(int jumlah)              {                  saldo = saldo + jumlah;              }          public boolean ambilUang(int jumlah)              {                  int temp = saldo - jumlah;                  if (temp < 0)                      return false;                  else                      {                          saldo = saldo-jumlah;                          return true;                      }              }      } |

Output :

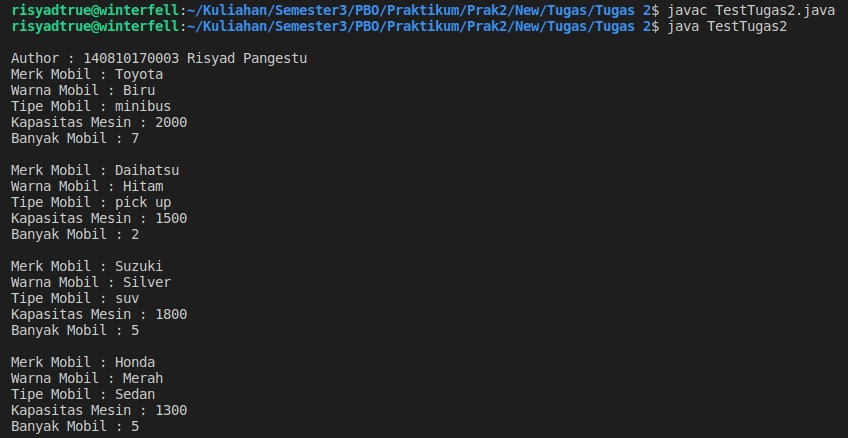


Analisa : Mengakses dair class lain dan menggunakan operator kondisi

Tugas 2 :

Code :

|  |
| --- |
| public class TestTugas2  {  public static void main(String[] args)  {  System.out.println("\nAuthor : 140810170003 Risyad Pangestu");  Mobil mobil1 = new Mobil();  mobil1.setMerkMobil("Toyota");  mobil1.setWarnaMobil("Biru");  mobil1.setTipeMobil("minibus");  mobil1.setKapasitasMesin(2000);  mobil1.setBanyakMobil(7);  System.out.println(mobil1.infoMobil());  Mobil mobil2 = new Mobil("Daihatsu");  mobil2.setWarnaMobil("Hitam");  mobil2.setTipeMobil("pick up");  mobil2.setKapasitasMesin(1500);  mobil2.setBanyakMobil(2);  System.out.println(mobil2.infoMobil());  Mobil mobil3 = new Mobil("Suzuki", 5);  mobil3.setWarnaMobil("Silver");  mobil3.setTipeMobil("suv");  mobil3.setKapasitasMesin(1800);  System.out.println(mobil3.infoMobil());  Mobil mobil4 = new Mobil("Honda", "Sedan", 5);  mobil4.setWarnaMobil("Merah");  mobil4.setKapasitasMesin(1300);  System.out.println(mobil4.infoMobil());  }  } |

Output :

Analisa : Mengakses dari class lain dengan overload konstruktor

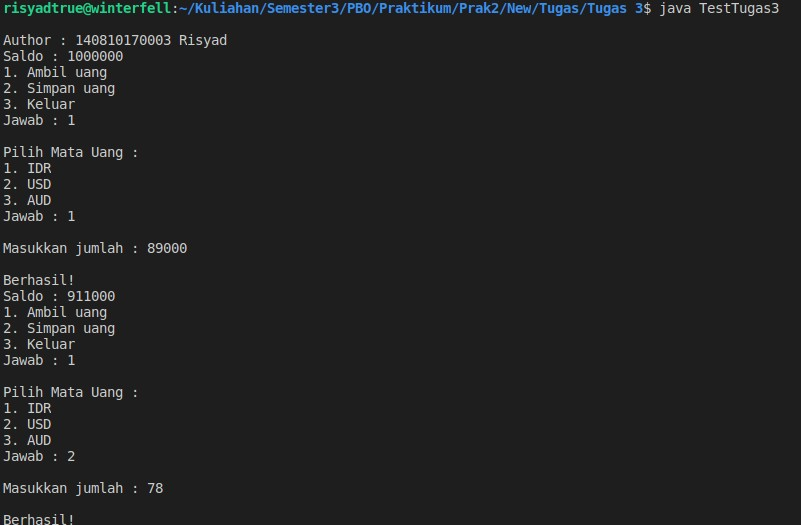
Tugas 3 :

Code :

|  |
| --- |
| import java.util.Scanner;  public class TestTugas3{          public static void main(String[] args){                  System.out.println("\nAuthor : 140810170003 Risyad");                  int pilih, id, jumlah;                  boolean status;                    Scanner in = new Scanner(System.in);                  Tabungan Tabungan1 = new Tabungan(1000000);                    while (true)                      {                          System.out.println("Saldo : " + Tabungan1.getSaldo());                          System.out.println("1. Ambil uang");                          System.out.println("2. Simpan uang");                          System.out.println("3. Keluar");                          System.out.print("Jawab : ");                          pilih = in.nextInt();                          if(pilih == 3) break;                              System.out.println("\nPilih Mata Uang :");                              System.out.println("1. IDR");                              System.out.println("2. USD");                              System.out.println("3. AUD");                              System.out.print("Jawab : ");                              id = in.nextInt();                              System.out.print("\nMasukkan jumlah : ");                              jumlah = in.nextInt();                              switch(pilih)                                  {                                      case 1:                                          status = Tabungan1.ambilUang(jumlah, id);                                            if (status)                                              {                                                  System.out.println("\nBerhasil!");                                              }                                          else                                              {                                                  System.out.println("\nGagal!");                                              }                                          break;                                      case 2:                                          Tabungan1.simpanUang(jumlah, id);                                          System.out.println("\nBerhasil menyimpan uang!");                                          break;                                  }                      }              }  } |

|  |
| --- |
| public class Tabungan      {          private int saldo;          public Tabungan(int initsaldo)              {                  saldo = initsaldo;              }          public int getSaldo()              {                  return saldo;              }          public void simpanUang(int jumlah, int id)              {                  if(id == 2)                      jumlah = this.usdToIdr(jumlah);                  else if (id == 3)                      jumlah = this.audToIdr(jumlah);                      saldo = saldo+jumlah;              }          public boolean ambilUang(int jumlah, int id)              {                  if(id == 2)                      jumlah = this.usdToIdr(jumlah);                  else if (id == 3)                      jumlah = this.audToIdr(jumlah);                  int temp = saldo - jumlah;                    if (temp < 0)                      return false;                  else                      {                          saldo = saldo-jumlah;                          return true;                      }              }          public int audToIdr(int jumlah)              {                  return jumlah\*10000;              }            public int usdToIdr(int jumlah)              {                  return jumlah\*9000;              }      } |

Output :



Analisa : Sama seperti tugas 1, namun ditambah menggunakan konsep enkapsulasi terlebih

dahulu

Tugas 4 :

Code :

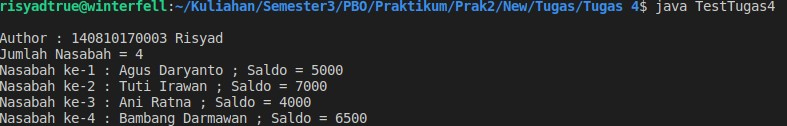
|  |
| --- |
| package perbankan;  public class Tabungan      {          private int saldo;          public Tabungan(int saldo)              {                  this.saldo = saldo;              }          public int getSaldo()              {                  return saldo;              }          public void simpanUang(int jumlah)              {                  saldo = saldo+jumlah;              }          public boolean ambilUang(int jumlah)              {                  int temp = saldo - jumlah;                  if (temp < 0)                      return false;                  else                      {                          saldo = saldo-jumlah;                          return true;                      }              }      } |

|  |
| --- |
| package perbankan;  public class Nasabah      {          private String namaAwal;          private String namaAkhir;          private Tabungan tabungan;          public Nasabah(String namaAwal, String namaAkhir)              {                  this.namaAwal = namaAwal;                  this.namaAkhir = namaAkhir;              }          public String getNamaAwal()              {                  return namaAwal;              }          public String getNamaAkhir()              {                  return namaAkhir;              }          public Tabungan getTabungan()              {                  return tabungan;              }          public void setTabungan(Tabungan tabungan)              {                  this.tabungan = tabungan;              }      } |

|  |
| --- |
| package perbankan;  public class Bank      {          private Nasabah[] nasabah;          private int jumlahNasabah;          public Bank()              {                  jumlahNasabah=0;                  nasabah = new Nasabah[4];              }          public void tambahNasabah(String namaAwal, String namaAkhir)              {                  Nasabah nasabahBaru = new Nasabah(namaAwal, namaAkhir);                  nasabah[jumlahNasabah] = nasabahBaru;                  jumlahNasabah++;              }          public int getJumlahNasabah()              {                  return jumlahNasabah;              }          public Nasabah getNasabah(int indeks)              {                  return this.nasabah[indeks];              }      } |

|  |
| --- |
| import perbankan.Bank;  import perbankan.Tabungan;  public class TestTugas4      {          public static void main(String[] args)              {                  System.out.println("\nAuthor : 140810170003 Risyad");                  Bank bank = new Bank();                    bank.tambahNasabah("Agus", "Daryanto");                  bank.getNasabah(0).setTabungan(new Tabungan(5000));                    bank.tambahNasabah("Tuti", "Irawan");                  bank.getNasabah(1).setTabungan(new Tabungan(7000));                    bank.tambahNasabah("Ani", "Ratna");                  bank.getNasabah(2).setTabungan(new Tabungan(4000));                    bank.tambahNasabah("Bambang", "Darmawan");                  bank.getNasabah(3).setTabungan(new Tabungan(6500));                    System.out.println("Jumlah Nasabah = " + bank.getJumlahNasabah());                    for (int i=0; i<bank.getJumlahNasabah(); i++)                      {                          System.out.println("Nasabah ke-" + (i+1) + " : " + bank.getNasabah(i).getNamaAwal() + " " + bank.getNasabah(i).getNamaAkhir() + " ; Saldo = " + bank.getNasabah(i).getTabungan().getSaldo());                      }              }      } |

Output :



Analisa : Menggunakan package untuk mengakses dari folder lain