**Tugas 8 OOP**

Nama : Muhammad Ahlizh Ulinnuha

NRP : 2110147051

Mata Kuliah : Praktikum PBO

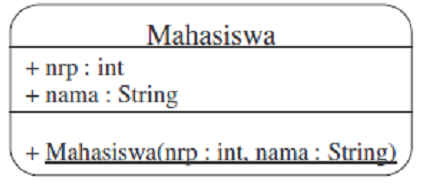
Materi : Enkapsulasi

**Tugas : “Kerjakan percobaan 1 pada modul praktikum 8 Enkapsulasi”**

**PERCOBAAN**

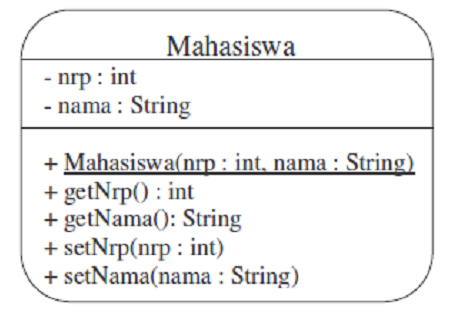
***Percobaan 1 : Melakukan enkapsulasi pada suatu class***

Implementasikan UML class diagram Mahasiswa sebelum dan setelah dilakukan enkapsulasi!

****

*Gambar 1 UML class diagram Mahasiswa sebelum enkapsulasi*

Jika enkapsulasi dilakukan pada class diagram diatas, maka akan berubah menjadi:



*Gambar 2 UML class diagram Mahasiswa setlah enkapsulasi*

Analisa :

Pada UML Class Diagram sebelum dilakukan enkapsulasi, variable-variable yang berada pada Class Mahasiswa bersifat public. Ini artinya bahwa variable tersebut dapat diakses oleh semua class. Kemudian konstruktor mahasiswa memiliki dua parameter yaitu parameter nrp yang bertipe integer dan parameter nama yang bertipe data String. Pada konstruktor Mahasiswa dilakukan set value pada variable nrp dan nama di mana valuenya mengambil dari parameter pada konstruktor Mahasiswa.

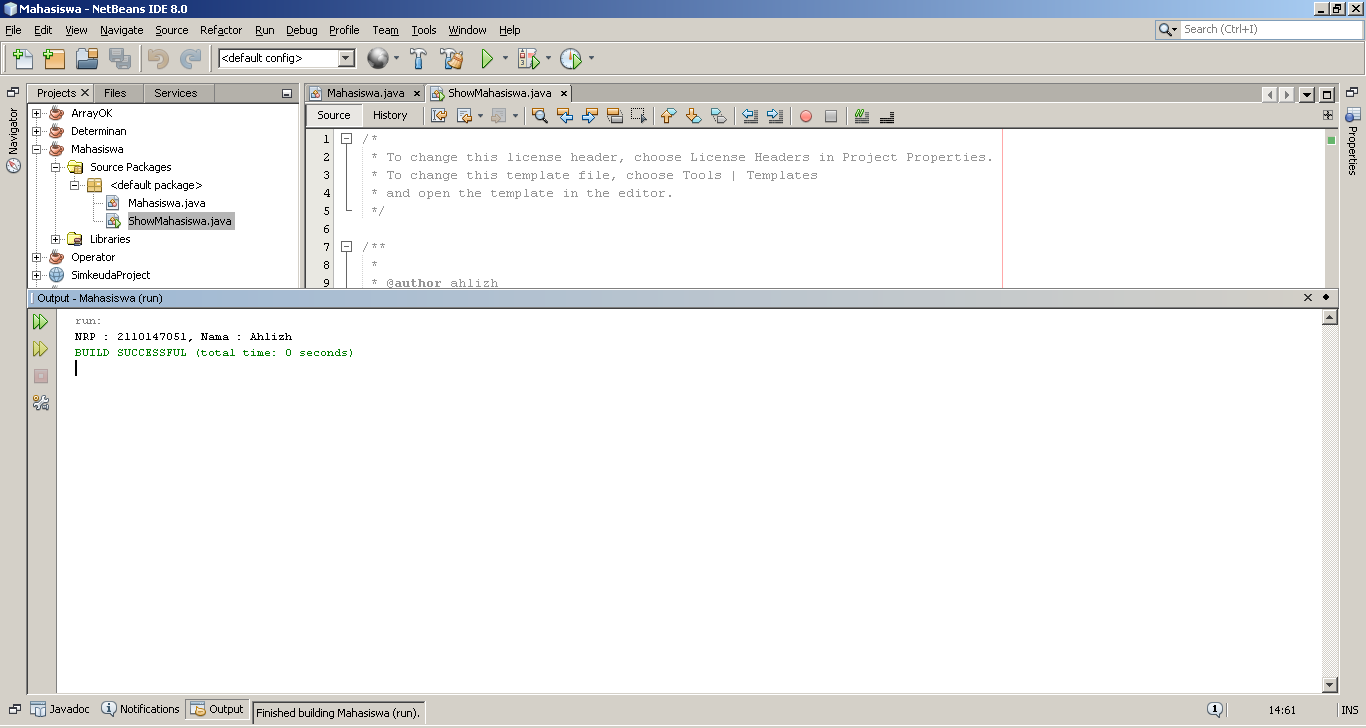
Jika UML Class Diagram sebelum dienkapsulasi di atas, seperti pada Gambar 1 diimplementasikan adalah sebagai berikut :

|  |
| --- |
| Source Class Mahasiswa sebelum enkapsulasi |
| public class Mahasiswa {  public int nrp;  public String nama;    public Mahasiswa(int nrp, String nama) {  this.nrp = nrp;  this.nama = nama;  }  } |

Kemudian dilakukan pengakses kedua variable tersebut pada class lain. Seperti pada source code di bawah

|  |
| --- |
| Source Class Show Mahasiswa |
| public class ShowMahasiswa {    public static void main(String[] args) {  Mahasiswa mhs = new Mahasiswa(2110147051, "Ahlizh");    System.out.println("NRP : " + mhs.nrp + ", Nama : " + mhs.nama);  }    } |

Variable nrp dan nama tetap akan dapat diakses dari kelas manapun karena bersifat public. Hasil outputnya adalah seperti pada gambar di bawah ini.



Pada UML Class Diagram yang kedua, dilakukan metode enkapsulasi yaitu penyembunyian detail dari suatu class. Di mana yang disembunyikan adalah variable nrp dan nama. Kemudian untuk mengakses variable tersebut dari class lain dibuatlah suatu interface yaitu method getNrp dan setNrp serta getNama dan setNama. Jadi class lain tidak dapat mengakses secara langsung variable tersebut. Untuk konstruktor mahasiswa tetap terdapat dua parameter yaitu nrp dan nama. Kemudian pada konstruktor mahasiswa tersebut dilakukan set value pada nrp dan nama.

Jika UML Class Diagram pada Gambar 2 diimplementasikan ke dalam bahasa pemrograman java adalah seperti pada gambar di bawah ini.

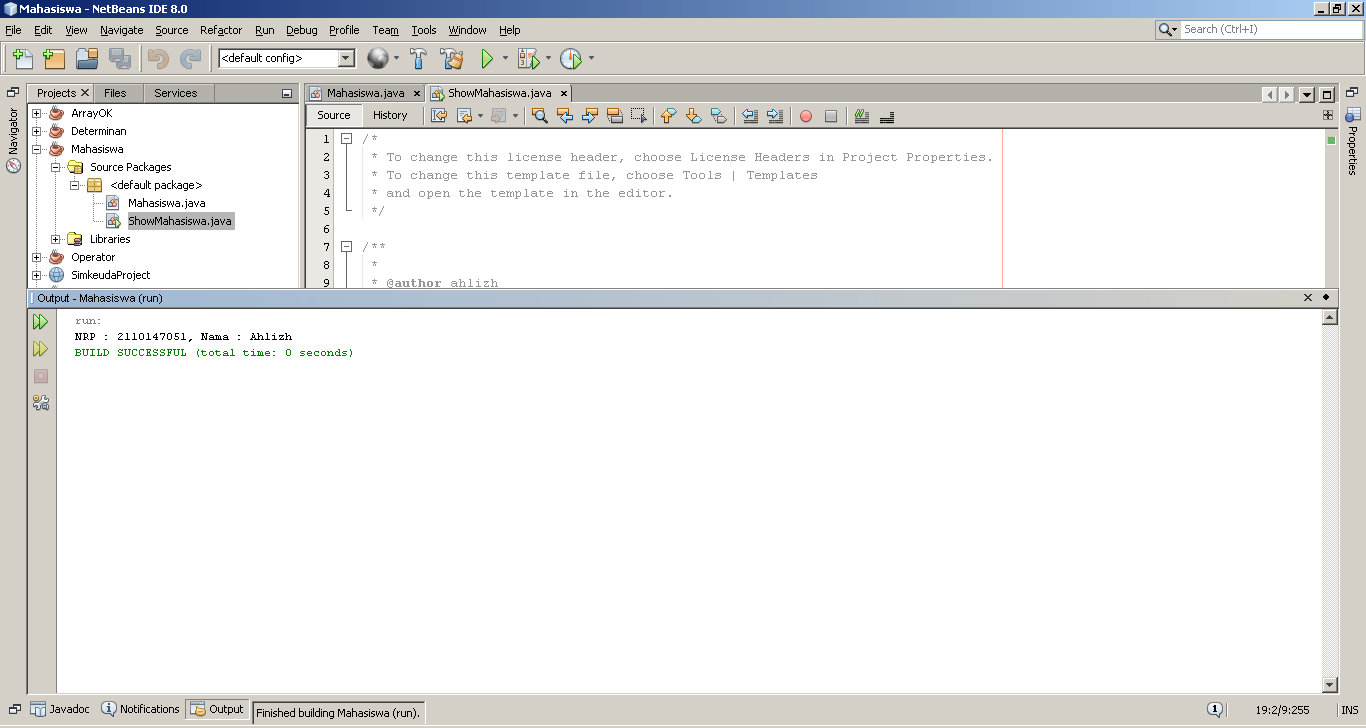
|  |
| --- |
| Source Code |
| public class Mahasiswa {  private int nrp;  private String nama;    public Mahasiswa(int nrp, String nama) {  this.nrp = nrp;  this.nama = nama;  }    public int getNrp() {  return nrp;  }  public void setNrp(int nrp) {  this.nrp = nrp;  }  public String getNama() {  return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

Terlihat pada source di atas telah diterapkan methode getter dan setter untuk masing-masing variable. Method ini nantinya yang digunakan untuk mengakses variable tersebut dari class lain walaupun tipe sifatnya adalah private bukan public.

Source code di bawah ini adalah contoh pengaksesannya dari class lain.

|  |
| --- |
| Source Code untuk mengakses enkapsulasi |
| public class ShowMahasiswa {    public static void main(String[] args) {  Mahasiswa mhs = new Mahasiswa(2110147051, "Ahlizh");    System.out.println("NRP : " + mhs.getNrp() + ", Nama : " + mhs.getNama());  }    } |

Hasil program di atas adalah seperti pada gambar di bawah ini.



Pada contoh di atas digunakan method getNrp dan getNama untuk mengambil nilainya. Tidak lagi seperti pada class yang belum mengimplementasikan enkapsulasi yang untuk mengaksesnya langsung berhubungan dengan variable yang dimaksud.