REPORT MANUAL OF JOBSHEET PRACTICUM, TASKS AND QUESTIONS (PEMOGRAMAN JARINGAN)

Name	**:**	**Brian Sayudha**
Class / NIM	**:**	**3G / 1841720158**
Major	**:**	**D-IV Informatics Enginering**

Praktikum 1

Class Mahasiswa

Code :

```
public class Mahasiswa {

    private String name, nim;

    public Mahasiswa(String name, String nim) {
        this.name = name;
        this.nim = nim;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getNim() {
        return nim;
    }

    public void setNim(String nim) {
        this.nim = nim;
    }

    @Override
    public String toString() {
        return "Mahasiswa{" + "name=" + name + ", nim=" + nim + '}';
    }

}
```

Class Praktikum1

Code :

```
public class Praktikum1 {
    private static void writeObject(Object o) throws Exception {
        try {
            ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("mhs.obj"));
            oos.writeObject(o);
            oos.flush();
            oos.close();
        } catch (FileNotFoundException ex) {
            throw ex;
        } catch (IOException ex) {
            throw ex;
        }
    }

    private static Object readObject() throws Exception {
        ObjectInputStream ois = new ObjectInputStream(new FileInputStream("mhs.obj"));
        try {
            Object readObject = ois.readObject();
            ois.close();
            return readObject;
        } catch (ClassNotFoundException ex) {
            throw ex;
        }
    }

    public static void main(String[] args) {
        Mahasiswa m = new Mahasiswa("075410099", "AL Ayubbi");
        try {
            writeObject(m);
            Mahasiswa readObject = (Mahasiswa) readObject();
            System.out.println("" + readObject);
        } catch (Exception ex) {
            Logger.getLogger(Praktikum1.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

}
```

Result

(it will cause error because don’t implements the serialization)

```
java.io.NotSerializableException: Praktikum1.Mahasiswa
        at java.base/java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1185)
        at java.base/java.io.ObjectOutputStream.writeObject(ObjectOutputStream.java:349)
        at Praktikum1.Praktikum1.writeObject(Praktikum1.java:25)
        at Praktikum1.Praktikum1.main(Praktikum1.java:49)
```

Add implements java.io.Serializable at Mahasiswa Class

```
public class Mahasiswa implements java.io.Serializable {

    private String name, nim;

    public Mahasiswa(String name, String nim) {
        this.name = name;
        this.nim = nim;
    }
```

Result

```
run-single:
Mahasiswa{name=075410099, nim=AL Ayubbi}
BUILD SUCCESSFUL (total time: 1 second)
```

Question

1. Why before when your program was run error?
Answer : Because we didn't call the java.io.Serializable at our model Class, Serializable is used to read and write the object .

2. What is the function of adding serializable implements?
Answer : the serializale implements is a java interface that make our class to support object serialization.

3. Add the attributes of the department, study program, and IPK to the Student class

Mahasiswa Class Code

```
private String name, nim,jurusan,programStudi;
private double ipk;

public Mahasiswa(String name, String nim, String jurusan, String programStudi, double ipk) {
    this.name = name;
    this.nim = nim;
    this.jurusan = jurusan;
    this.programStudi = programStudi;
    this.ipk = ipk;
}
```

```
public String getJurusan() {
    return jurusan;
}

public void setJurusan(String jurusan) {
    this.jurusan = jurusan;
}

public String getProgramStudi() {
    return programStudi;
}

public void setProgramStudi(String programStudi) {
    this.programStudi = programStudi;
}

public double getIpk() {
    return ipk;
}

public void setIpk(double ipk) {
    this.ipk = ipk;
}

@Override
public String toString() {
    return "Mahasiswa{" + "name= " + name + ", nim= " + nim + ", jurusan= " + jurusan + ", programStudi= " + programStudi + ", ipk= " + ipk + '}';
}
```

Praktikum1 at main method
Code

```
public static void main(String[] args) {
    Mahasiswa m = new Mahasiswa("075410099", "AL Ayubbi", "Teknologi Informasi", "D4 Teknik Informatika", 4.00);
    try {
        writeObject(m);
        Mahasiswa readObject = (Mahasiswa) readObject();
        System.out.println("" + readObject);
    } catch (Exception ex) {
        Logger.getLogger(Praktikum1.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

Result

```
Mahasiswa{name= 075410099, nim= AL Ayubbi, jurusan= Teknologi Informasi, programStudi= D4 Teknik Informatika, ipk= 4.0}
BUILD SUCCESSFUL (total time: 1 second)
```

Praktikum 2

MataKuliah Class
Code

```java
public class MataKuliah implements java.io.Serializable {

    private static final long serialVersionUID = 5559055602375093688L;

    private String kodeMK;
    private transient String nama;
    private static byte sks;

    public MataKuliah(String kodeMK, String nama, byte sks) {
        this.kodeMK = kodeMK;
        this.nama = nama;
        this.sks = sks;
    }

    public String getKodeMK() {
        return kodeMK;
    }

    public void setKodeMK(String kodeMK) {
        this.kodeMK = kodeMK;
    }

    public String getNama() {
        return nama;
    }

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

    public static byte getSks() {
        return sks;
    }

    public static void setSks(byte sks) {
        MataKuliah.sks = sks;
    }
```

```java
@Override
public String toString() {
    return "MataKuliah{" + "kodeMK= " + kodeMK + ", nama= " + nama + ", sks= " + sks + '}';
}
```

Praktikum2 Class

```
public class Praktikum2 {
    public static void main(String[] args) {
        MataKuliah mk = new MataKuliah("001", "Pemrograman Jaringan", (byte) 3);
        try {
            System.out.println("" + mk);
            ObjectOutputStream oos = new ObjectOutputStream(new FileOutputStream("mk.obj"));
            oos.writeObject(mk);
            oos.flush();
            oos.close();
            mk.setSks((byte) 2);

            ObjectInputStream ois = new ObjectInputStream(new FileInputStream("mk.obj"));
            try {
                MataKuliah mk1 = (MataKuliah) ois.readObject();
                System.out.println("" + mk1);
            } catch (ClassNotFoundException ex) {
                Logger.getLogger(Praktikum2.class.getName()).log(Level.SEVERE, null, ex);
            }
        } catch (FileNotFoundException ex) {
            Logger.getLogger(Praktikum2.class.getName()).log(Level.SEVERE, null, ex);
        } catch (IOException ex) {
            Logger.getLogger(Praktikum2.class.getName()).log(Level.SEVERE, null, ex);
        }
    }

}
```

Result

```
MataKuliah{kodeMK= 001, nama= Pemrograman Jaringan, sks= 3}
MataKuliah{kodeMK= 001, nama= null, sks= 2}
BUILD SUCCESSFUL (total time: 1 second)
```

Question

1. After running the above code, what is the difference between transient and static modifiers in the class that implements Serialize?

Answer :
Transient = The data will not serialized, so the value become null or 0.
Static = this attribute when serialized or object being writed it can be changed back to the attribute (can be inputted new value).

2. What can you conclude with practicum 1 and practicum 2?
Answer : In practicum 1 we can do read and write the object serialization into our output and then in the practicum 2 we know that there are attribute to be used in

serialization. We can add some attributes for the data that we don't want to serialized.

Tugas

Mahasiswa Class

```
public class Mahasiswa implements java.io.Serializable{
    String nim,nama,jurusan,prodi,ipk;


    public Mahasiswa(String nim, String nama, String jurusan, String prodi, String ipk) {
        this.nim = nim;
        this.nama = nama;
        this.jurusan = jurusan;
        this.prodi = prodi;
        this.ipk = ipk;
    }

    public String getNim() {
        return nim;
    }

    public void setNim(String nim) {
        this.nim = nim;
    }

    public String getNama() {
        return nama;
    }

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

    public String getJurusan() {
        return jurusan;
    }

    public void setJurusan(String jurusan) {
        this.jurusan = jurusan;
    }
```

Frontend Class

Code

```
public void clearText(){
    NIMField.setText("");
    nameField.setText("");
    jurusanField.setText("");
    prodiField.setText("");
    ipkfield.setText("");
}

public void dataResult(){

    String [] column = {"No", "NIM", "Nama", "Jurusan", "Prodi", "IPK"};
    tblMahasiswa.setModel(new DefaultTableModel(new Object[][]{}, column));
}
```

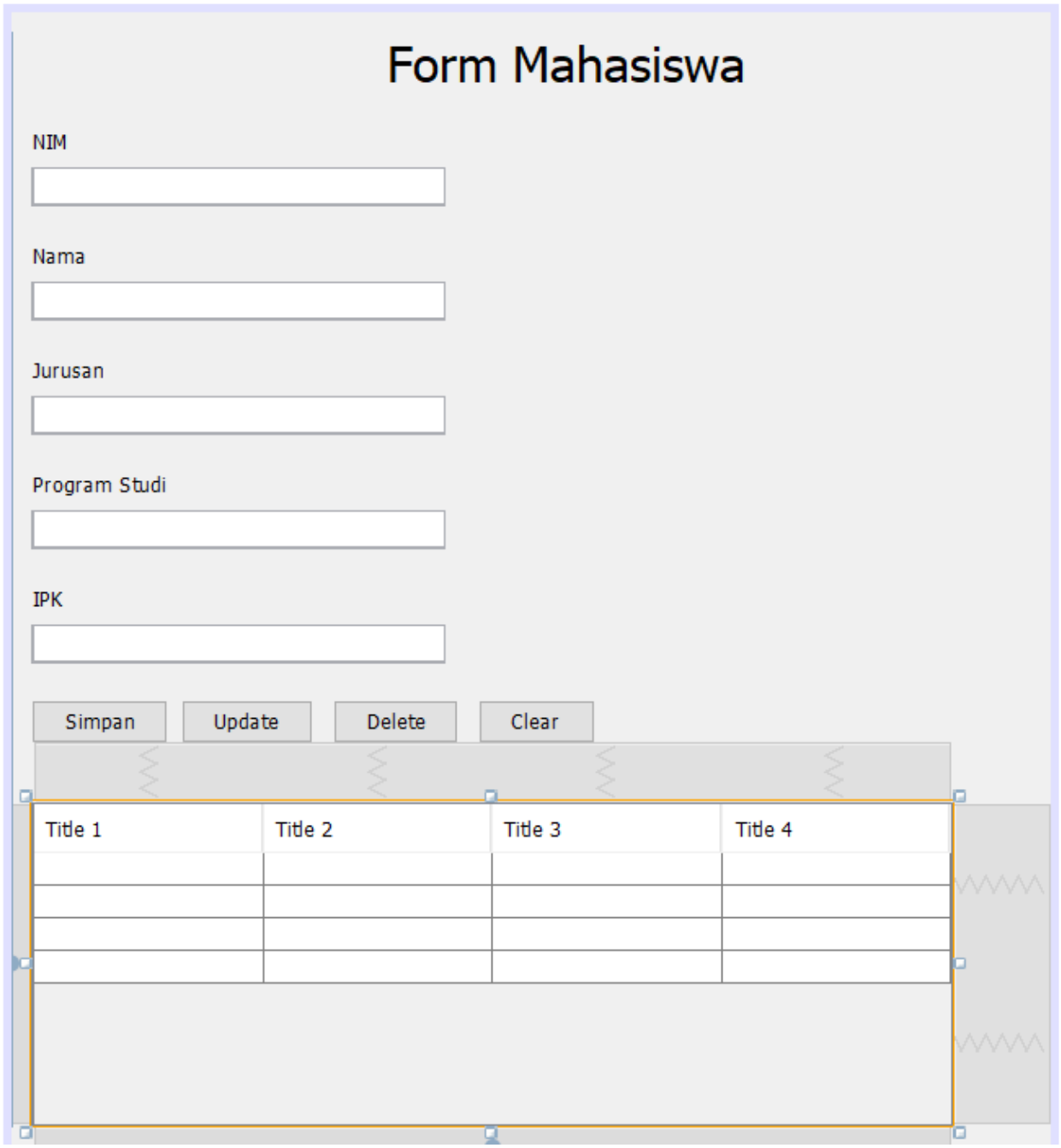
```
private void btnClearActionPerformed(java.awt.event.ActionEvent evt) {
    clearText();
}

private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
    DefaultTableModel model = (DefaultTableModel) tblMahasiswa.getModel();
    int row = tblMahasiswa.getSelectedRow();
    model.removeRow(row);
}

private void tblMahasiswaMouseClicked(java.awt.event.MouseEvent evt) {
    DefaultTableModel model = (DefaultTableModel) tblMahasiswa.getModel();
    int row = tblMahasiswa.getSelectedRow();
    int column = tblMahasiswa.getSelectedColumn();

    NIMField.setText((String) tblMahasiswa.getValueAt(row, column));
    nameField.setText((String) tblMahasiswa.getValueAt(row, column));
    jurusanField.setText((String) tblMahasiswa.getValueAt(row, column));
    prodiField.setText((String) tblMahasiswa.getValueAt(row, column));
    ipkfield.setText((String) tblMahasiswa.getValueAt(row, column));
}
```

Design


Form Mahasiswa
NIM
Nama
Jurusan
Program Studi
IPK
Simpan
Update
Delete
Clear
Title 1
Title 2
Title 3
Title 4


Form Mahasiswa
NIM
Nama
Jurusan
Program Studi
IPK
Simpan
Update
Delete
Clear
No
NIM
Nama
Jurusan
Prodi
IPK