Name: Deras Pangestu Gusti (08)
Class: SIB-1G
NIM: 254107060114

JOBSHEET II
OBJECT

1. Purpose

After doing this practicum, students will be able to:

1. Getting to know objects and classes as fundamental concepts in object-oriented programming

2. Declare classes, attributes, and methods

3. Create an object (instantiate)

4. Access attributes and methods of an object

5. Implement constructors

1. Practicum

2.1 Practicum 1: Class Declarations, Attributes and Methods

Time : 50 Minutes

In practicum 1, a class was created along with its attributes and methods. Consider the following Class Diagram:

Mahasiswa
nim: String
nama: String
kelas: String
ipk: double
tampilkanInformasi(): void
ubahKelas(kelasBaru: String): void
updateIpk(ipkBaru: double): void
nilaiKinerja(ipk: double): String

Based on the diagram class, a program will be created using the Java language.

2.1.1 Practicum Steps

1. Open the text editor. Create a new file, name it **Mahasiswa< attendance number >.java**

2. Complete the **Mahasiswa** class with the attributes that have been described in the class diagram

```
String nama;
String nim;
String kelas;
double ipk;
```

3. Complete the **Mahasiswa** class with the method that has been described in the class diagram.

```java
void tampilkanInformasi() {
    System.out.println("Nama: " + nama);
    System.out.println("NIM: " + nim);
    System.out.println("IPK: " + ipk);
    System.out.println("Kelas: " + kelas);
}

void ubahKelas(String kelasBaru) {
    kelas = kelasBaru;
}

void updateIpk(double ipkBaru) {
    ipk = ipkBaru;
}

String nilaiKinerja() {
    if (ipk >= 3.5) {
        return "Kinerja sangat baik";
    } else if (ipk >= 3.0) {
        return "Kinerja baik";
    } else if (ipk >= 2.0) {
        return "Kinerja cukup";
    } else {
        return "Kinerja kurang";
    }
}
```
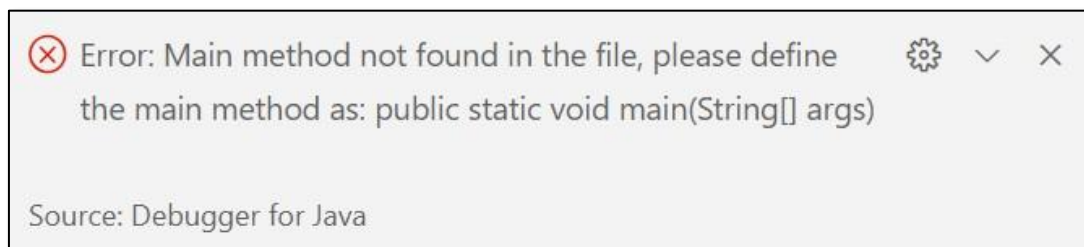
4. Compile dan run program.

```
1   package praktikum.jobsheet2.vscode;
2
3   public class Student8
4   {
        Run | Debug | Run main | Debug main
5       public static void main(String[] args)
6       {
7
8       }
9   }
10
11  class Student
12  {
13      String name;
14      String nim;
15      String class_student;
16      float ipk;
17
18      void displayInformation()
19      {
20          System.out.println("Nama: " + name);
21          System.out.println("NIM: " + nim);
22          System.out.println("IPK: " + ipk);
23          System.out.println("Kelas: " + class_student);
24      }
25
26      void changeClass(String newClass)
27      {
28          class_student = newClass;
29      }
30
31      void updateIPK(float newIpk)
32      {
33          ipk = newIpk;
34      }
35
36      String performaceScore()
37      {
38          if (ipk >= 3.5)
39          {
40              return "Excellent performance";
41          }
42          else if (ipk >= 3.0)
43          {
44              return "Good performance";
45          }
46          else if (ipk >= 2.5)
47          {
48              return "Satisfactory performance";
49          }
50          else
51          {
52              return "Poor performance";
53          }
54      }
55
56  }
```

2.1.2 Verification of Practicum Results

Match the results of your program code compile with the following image.

Error: Main method not found in the file, please define the main method as: public static void main(String[] args)

Source: Debugger for Java

2.1.3 Questions

1. Name two characteristics of a class or object!

Class: General, Object: Specific.

2. Pay attention to the **Mahasiswa** class in Practicum 1, how many attributes do the Student class have? Mention what the attributes are!

   4 attributes: name, nim, class_student, ipk.

3.   How many methods does the class have? Mention what the methods are!

4 methods: displayInformation, changeClass, updateIPK, performanceScore.

4.   Pay attention to the method **updateIpk** contained in the **Mahasiswa** class. Modify the content of the method so that the GPA entered is valid, namely first checking whether the GPA entered is in the range of 0.0 to 4.0 (0.0 <= GPA <= 4.0). If the GPA is not in that range, the message is issued: "GPA is invalid. Must be between 0.0 and 4.0".

```java
36      String performaceScore()
37      {
38          if (ipk < 0.0 || ipk > 4.0)
39          {
40              return "GPA is invalid. Must be between 0.0 and 4.0";
41          }
42          else if (ipk >= 3.5)
43          {
44              return "Excellent performance";
45          }
46          else if (ipk >= 3.0)
47          {
48              return "Good performance";
49          }
50          else if (ipk >= 2.5)
51          {
52              return "Satisfactory performance";
53          }
54          else
55          {
56              return "Poor performance";
57          }
58      }
```

5.   Explain how the **nilaiKinerja()** method works in evaluating student performance, what criteria are used to determine the performance value, and what is returned by the **nilaiKinerja()** method?

6.   **Commit and push program code to Github**

https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/blob/main/src/com/jobsheet2/Student8.java

2.2 Practicum 2: Object Institutionalization, and Accessing Attributes and Methods

Time: 50 Minutes

Up to this stage, **the Student class** has been successfully created in Experiment 1. Furthermore, if the Student class wants to be used and accessed by its attributes and methods, it is necessary to

create an object/instance of the **Student class** first through an institutionalization process.

2.2.1 Practicum Steps

1. Create a new file, name it **StudentMain< attendance number >.java**

2. Write down the basic structure of the Java programming language consisting of **the main() function**

3. In **the main() function,** instantize, then proceed to access the attributes and methods of the object that has been formed.

```
Mahasiswa mhs1 = new Mahasiswa();
mhs1.nama = "Muhammad Ali Farhan";
mhs1.nim = "2241720171";
mhs1.kelas = "SI 2J";
mhs1.ipk = 3.55;

mhs1.tampilkanInformasi();
mhs1.ubahKelas("SI 2K");
mhs1.updateIpk(3.60);
mhs1.tampilkanInformasi();
```

4. Compile dan run program.

```
1    package src.jobsheet2;
2
3    public class StudentMain8
4    {
     Run | Debug | Run main | Debug main
5        public static void main(String[] args)
6        {
7            Student std1 = new Student();
8            std1.name = "Muhammad Ali Farhan";
9            std1.nim = "2241720171";
10           std1.class_student = "SI 2J";
11           std1.ipk = 3.55f;
12
13           std1.displayInformation();
14           std1.changeClass(newClass: "SI 2K");
15           std1.updateIPK(newIpk: 3.60f);
16           std1.displayInformation();
17
18       }
19   }
```

```
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.55
Kelas: SI 2J
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.6
Kelas: SI 2K
```

5.   **Commit and push program code to Github**

https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/blob/main/src/com/jobsheet2/StudentMain8.java

2.2.2   Verification of Practicum Results

Match the results of your program code compile with the following image.

```
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.55
Kelas: SI 2J
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.6
Kelas: SI 2K
```

2.2.3   Questions

1.   In the **StudentMain class**, indicate the line of program code used for the instance process!

What is the name of the resulting object?

```
5         public static void main(String[] args)
6         {
7             Student std1 = new Student();
8             std1.name = "Muhammad Ali Farhan";
9             std1.nim = "2241720171";
10            std1.class_student = "SI 2J";
11            std1.ipk = 3.55f;
```

Name variable object: std1

2.   How do I access the attributes and methods of an object?
By using a dot after the object variable, example:
Attribute: objectName.AttributeName = value;
Method: objectName.MethodName ();

3.   Why are the output output results of the **method tampilkanInformasi()** first and second different?

In the first method displayInformation(), the program will generate a value attribute in the main method before it is changed. The second method displayInformation() is the result of the value that has been changed with the object method that was previously created.

2.3 Practicum 3: Constructor

Time: 60 Minutes

In this experiment, program code was created to implement various constructors based on their parameters.

2.3.1   Practicum Steps

1.   Re-open **the Mahasiswa** Class. Add two constructors in the **Mahasiswa** class , consisting of

one default constructor and one parameter constructor. Constructors are privileged methods, the placement of program code for constructors can be treated the same as any other method (after attributes).

```java
public Mahasiswa() {
}

public Mahasiswa(String nm, String nim, double ipk, String kls) {
    nama = nm;
    this.nim = nim;
    this.ipk = ipk;
    kelas = kls;
}
```

*Note: If the parameter name is the same as the attribute name, then to refer to the attribute variable the syntax **of this** is added in front of the attribute name*

2.   Re-open **the Mahasiswa Class.** Create another object named **mhs2** using a parameterized constructor.

```java
Mahasiswa mhs1 = new Mahasiswa();
mhs1.nama = "Muhammad Ali Farhan";
mhs1.nim = "2241720171";
mhs1.kelas = "SI 2J";
mhs1.ipk = 3.55;

mhs1.tampilkanInformasi();
mhs1.ubahKelas("SI 2K");
mhs1.updateIpk(3.60);
mhs1.tampilkanInformasi();

Mahasiswa mhs2 = new Mahasiswa("Annisa Nabila","2141720160", 3.25,"TI 2L");
mhs2.updateIpk(3.30);
mhs2.tampilkanInformasi();
```

3.   Compile dan run program.

```java
package src.jobsheet2;

public class Student8
{
    Run | Debug | Run main | Debug main
    public static void main(String[] args)
    {

    }
}

class Student
{
    String name;
    String nim;
    String class_student;
    float ipk;

    void displayInformation()
    {
        System.out.println("Nama: " + name);
        System.out.println("NIM: " + nim);
        System.out.println("IPK: " + ipk);
        System.out.println("Kelas: " + class_student);
    }

    void changeClass(String newClass)
    {
        class_student = newClass;
    }

    void updateIPK(float newIpk)
    {
        ipk = newIpk;
    }
```

```java
36          String performaceScore()
37          {
38              if (ipk < 0.0 || ipk > 4.0)
39              {
40                  return "GPA is invalid. Must be between 0.0 and 4.0";
41              }
42              else
43              {
44                  if (ipk >= 3.5)
45                  {
46                      return "Excellent performance";
47                  }
48                  else if (ipk >= 3.0)
49                  {
50                      return "Good performance";
51                  }
52                  else if (ipk >= 2.5)
53                  {
54                      return "Satisfactory performance";
55                  }
56                  else
57                  {
58                      return "Poor performance";
59                  }
60              }
61          }
62
63          public Student()
64          {
65
66          }
67
68          public Student(String name, String nim, float ipk, String class_student)
69          {
70              this.name = name;
71              this.nim = nim;
72              this.ipk = ipk;
73              this.class_student = class_student;
74          }
75
76  }
```

```java
1   package src.jobsheet2;
2
3   public class StudentMain8
4   {
        Run | Debug | Run main | Debug main
5       public static void main(String[] args)
6       {
7           Student std1 = new Student();
8           std1.name = "Muhammad Ali Farhan";
9           std1.nim = "2241720171";
10          std1.class_student = "SI 2J";
11          std1.ipk = 3.55f;
12
13          std1.displayInformation();
14          std1.changeClass(newClass: "SI 2K");
15          std1.updateIPK(newIpk: 3.60f);
16          std1.displayInformation();
17
18          Student std2 = new Student(name: "Annisa Nabila", nim: "2141720160", ipk: 3.25f, class_student: "TI 2L");
19          std2.updateIPK(newIpk: 3.30f);
20          std2.displayInformation();
21      }
22  }
```

```
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.55
Kelas: SI 2J
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.6
Kelas: SI 2K
Nama: Annisa Nabila
NIM: 2141720160
IPK: 3.3
Kelas: TI 2L
```

4. **Commit and push program code to Github**

   https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/tree/main/src/com/jobsheet2

2.3.2   Verification of Practicum Results

Match the results of your program code compile with the following image.

```
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.55
Kelas: SI 2J
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.6
Kelas: SI 2K
Nama: Annisa Nabila
NIM: 2141720160
IPK: 3.3
Kelas: TI 2L
```

2.3.3   Questions

1. In the **Mahasiswa** class in practicum 3, indicate the program code line used to declare a parameterized constructor!

```java
public Student(String name, String nim, float ipk, String class_student)
{
    this.name = name;
    this.nim = nim;
    this.ipk = ipk;
    this.class_student = class_student;
}
```

2. Pay attention to **the StudentMain** class. What exactly does the following line of program do?

```java
Mahasiswa mhs2 = new Mahasiswa("Annisa Nabila","2141720160", 3.25,"TI 2L");
```

Create a new object from the Student class with specific data, then save it to the std2 variable

so it can access the attributes and methods of an object.

3. Remove the default constructor on the **Mahasiswa class**, and then compile and run the program. How did it turn out? Explain why this is the case!

    The code will generate a compilation error because the constructor without parameters no longer exists.

4.   After instantiating the object, do the methods in the **Mahasiswa** class have to be accessed sequentially? Explain why!

No, because every function can be called at any time as long as there is an object.

5.   Create a new object with the name **mhs<StudentName>** using the parameterized constructor from the **Mahasiswa class**!

```
Student stdDeras = new Student(name: "Deras Pangestu Gusti", nim: "254107060114", ipk: 3.8f, class_student: "LPR2_
stdDeras.displayInformation();
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.55
Kelas: SI 2J
Nama: Muhammad Ali Farhan
NIM: 2241720171
IPK: 3.6
Kelas: SI 2K
Nama: Annisa Nabila
NIM: 2141720160
IPK: 3.3
Kelas: TI 2L
Nama: Deras Pangestu Gusti
NIM: 254107060114
IPK: 3.8
Kelas: LPR2_7T
```

6.   **Commit and push program code to Github**

https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/tree/main/src/com/jobsheet2
=

2.4 Practicum Exercises

Time : 150 Minutes

1.   A class diagram from the Course class is given as follows:

MataKuliah
kodeCourse: String name: String credits: Int numberOfHours: int
showInformation(): void updateSKS(sksNew: int): void addHour(hours: int): void reduceHours (hours: int): void

Create a program to implement a Course class based on the class diagram above, which consists of:

2.   Course Class (Course< attendance number >.java)

3.   Class ClassMain (CourseMain< attendance number >.java)

In the Main Course, create at least 2 objects. Use the default constructor and the parameterized constructor when instilling objects. Then call all the methods that have been made in the Course class.

The explanation of the attributes and methods in the Course class is as follows:

1.   Attributes

•   kodeCourse(String): a unique code for the course.

•   name (String): the full name of the course

•   SKS (int): credit Semester

•   numberOfHours (int): total number of meeting hours per week for courses

2.   Method

•   DisplayInformation(): This method is used to display all the information related to the course.

•   changeSKS(int sksNew): This method allows you to change the credit score for a course. After changing the value, it notifies the user that the credits have been changed.

- addHours(int hours): This method adds an additional number of hours to the existing number of hours for the course.

- reduceHours(int hours): This method serves to reduce the number of hours of the course. Before reducing, this method checks to make sure that the number of hours left is sufficient to be reduced. If the number of hours is insufficient (the number of initial hours is less than the reduction hours), this method will notify the user that the reduction cannot be made. If the subtraction is successful, it updates the number of hours and prints the value of the new number of hours.

```java
1    package com.jobsheet2;
2
3    public class Course8
4    {
        Run | Debug | Run main | Debug main
5        public static void main(String[] args)
6        {
7
8        }
9    }
10
11   class Course
12   {
13       String codeCourse;
14       String name;
15       int credits;
16       int numberOfHours;
17
18       void showInformation()
19       {
20           System.out.println("Code Course: " + codeCourse);
21           System.out.println("Name: " + name);
22           System.out.println("Credits: " + credits);
23           System.out.println("Number of Hours: " + numberOfHours + " hours\n");
24       }
25
26       void updateCredits(int newCredits)
27       {
28           credits = newCredits;
29       }
30
31       void addHour(int hours)
32       {
33           numberOfHours += hours;
34       }
35
```

```java
36        void reduceHours(int hours)
37        {
38            if (hours > numberOfHours)
39            {
40                System.out.println(x: "Cannot reduce hours. Not enough hours available.\n");
41            }
42            else
43            {
44                numberOfHours -= hours;
45            }
46        }
47
48        Course()
49        {
50
51        }
52
53        Course(String codeCourse, String name, int credits, int numberOfHours) {
54            this.codeCourse = codeCourse;
55            this.name = name;
56            this.credits = credits;
57            this.numberOfHours = numberOfHours;
58        }
59  }
```

```java
1   package com.jobsheet2;
2
3   public class CourseMain8
4   {
        Run | Debug | Run main | Debug main
5       public static void main(String[] args)
6       {
7           Course course1 = new Course();
8
9           course1.codeCourse = "PASD01";
10          course1.name = "Algorithms & Data Structures Practicum";
11          course1.credits = 2;
12          course1.numberOfHours = 6;
13
14          course1.showInformation();
15
16          course1.updateCredits(newCredits: 3);
17          course1.addHour(hours: 2);
18          course1.showInformation();
19
20          course1.reduceHours(hours: 4);
21          course1.showInformation();
22
23          course1.reduceHours(hours: 10);
24
25          Course course2 = new Course(codeCourse: "PBO02", name: "Object Oriented Programming", credits: 3, numberOfHour
26          course2.showInformation();
27
28          course2.updateCredits(newCredits: 1);
29          course2.addHour(hours: 3);
30          course2.showInformation();
31
32          course2.reduceHours(hours: 4);
33          course2.showInformation();
34
35          course2.reduceHours(hours: 5);
36      }
37
38  }
```

```
Code Course: PASD01
Name: Algorithms & Data Structures Practicum
Credits: 2
Number of Hours: 6 hours

Code Course: PASD01
Name: Algorithms & Data Structures Practicum
Credits: 3
Number of Hours: 8 hours

Code Course: PASD01
Name: Algorithms & Data Structures Practicum
Credits: 3
Number of Hours: 4 hours

Cannot reduce hours. Not enough hours available.

Code Course: PBO02
Name: Object Oriented Programming
Credits: 3
Number of Hours: 4 hours

Code Course: PBO02
Name: Object Oriented Programming
Credits: 1
Number of Hours: 7 hours

Code Course: PBO02
Name: Object Oriented Programming
Credits: 1
Number of Hours: 3 hours

Cannot reduce hours. Not enough hours available.
```

https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/tree/main/src/com/jobsheet2

1.   A class diagram from the Lecturer class is given as follows:

Lecturer
idLecturer: String Name: String ActiveStatus: boolean yearOfEntry: int ExpertiseCompetency: String
showInformation(): void setStatusActive(status: boolean): void calculateTimeWork(YearNow: int): int changeSkill(Skill: String): void

Create a program to implement the Lecturer class based on the class diagram above, which consists of:

1.   Class Lecturer (Lecturers<NoAbsen>.java)

2.   TeacherMain Class (DosenMain<NoAbsen>.java)

In the LecturerMain class, make at least 2 objects. Use the default constructor and the parameterized constructor when instilling objects. Then call all the methods that have been made in the Lecturer's class

The explanation of the attributes and methods in the lecturer class is as follows:

1. Attributes

- idLecturer (String): a unique id for each lecturer.

- name (String): the full name of the lecturer.

- ActiveStatus (boolean): indicates whether the lecturer is active (true) or inactive (false) in carrying out his duties.

- yearOfEntry (int): the year when the lecturer starts joining the college

- ExpertiseCompetency (String): the field of expertise of the lecturer, which describes the specialization or academic focus of the lecturer

2. Method

- displayInformation(): this method is used to display complete information about the lecturer

- setActiveStatus(status: boolean): this method is used to set the active status of the lecturer. If the status parameter is set to true, it means that the lecturer is active. On the other hand, if it is false, the lecturer is declared inactive.

- calculateTimeWork (YearNow: int): this method calculates and returns the lecturer's working period in the year, based on the year of joining and the current year (yrSkrg) which is the input parameter of this method. The results of the calculation provide information about the length of time lecturers work in universities.

- changeSkill (Skill: String): This method is used to change the lecturer's field of expertise.

```java
1    package com.jobsheet2;
2
3    public class Lecturers8
4    {
         Run | Debug | Run main | Debug main
5        public static void main(String[] args)
6        {
7
8        }
9    }
10
11   class Lecturers
12   {
13       String idLecturer;
14       String name;
15       boolean activeStatus;
16       int yearOfEntry;
17       String expertiseCompetency;
18       int timeWork;
19
20       void displayInformation()
21       {
22           System.out.println("ID Lecturer : " + idLecturer);
23           System.out.println("Name : " + name);
24           System.out.println("Active Status : " + setActiveStatus(activeStatus));
25           System.out.println("Year of Entry : " + yearOfEntry);
26           System.out.println("Time Work : " + timeWork + " years");
27           System.out.println("Expertise Competency : " + expertiseCompetency + "\n");
28       }
29
30       String setActiveStatus(boolean status)
31       {
32           activeStatus = status;
33
34           if(status == true)
35           {
36               return "Active";
37           }
38           else
39           {
40               return "Inactive";
41           }
42       }
43
```

```
44        int calculateTimeWork(int yearNow)
45        {
46            timeWork = yearNow - yearOfEntry;
47            return timeWork;
48        }
49
50        void changeSkill(String skill)
51        {
52            expertiseCompetency = skill;
53        }
54
55        public Lecturers()
56        {
57
58        }
59
60        public Lecturers(String idLecturer, String name, boolean activeStatus, int yearOfEntry, String expertiseCompet
61        {
62            this.idLecturer = idLecturer;
63            this.name = name;
64            setActiveStatus(activeStatus);
65            this.yearOfEntry = yearOfEntry;
66            this.expertiseCompetency = expertiseCompetency;
67        }
68    }
```

```
1   package com.jobsheet2;
2
3   public class LecturersMain8
4   {
        Run | Debug | Run main | Debug main
5       public static void main(String[] args)
6       {
7           Lecturers lecturer1 = new Lecturers();
8
9           lecturer1.idLecturer = "VAL001";
10          lecturer1.name = "Vivin Ayu Lestari, B.Ed., M.Comp.Sc";
11          lecturer1.activeStatus = true;
12          lecturer1.yearOfEntry = 2015;
13          lecturer1.expertiseCompetency = "Software Engineer";
14
15          lecturer1.calculateTimeWork(yearNow: 2026);
16          lecturer1.displayInformation();
17
18          lecturer1.setActiveStatus(status: false);
19          lecturer1.changeSkill(skill: "Data Scientist");
20          lecturer1.displayInformation();
21
22          Lecturers lecturer2 = new Lecturers(idLecturer: "WIS001", name: "Wilda Imama Sabilla, B.Comp.Sc., M.Comp.Sc.", activeStatus: true, yearOfEntry: 2018, ex… "Data Scie
23
24          lecturer2.calculateTimeWork(yearNow: 2026);
25          lecturer2.displayInformation();
26
27          lecturer2.setActiveStatus(status: false);
28          lecturer2.changeSkill(skill: "Network Engineer");
29          lecturer2.displayInformation();
30      }
31  }
```

```
ID Lecturer : VAL001
Name : Vivin Ayu Lestari, B.Ed., M.Comp.Sc
Active Status : Active
Year of Entry : 2015
Time Work : 11 years
Expertise Competency : Software Engineer

ID Lecturer : VAL001
Name : Vivin Ayu Lestari, B.Ed., M.Comp.Sc
Active Status : Inactive
Year of Entry : 2015
Time Work : 11 years
Expertise Competency : Data Scientist

ID Lecturer : WIS001
Name : Wilda Imama Sabilla, B.Comp.Sc., M.Comp.Sc.
Active Status : Active
Year of Entry : 2018
Time Work : 8 years
Expertise Competency : Data Scientist

ID Lecturer : WIS001
Name : Wilda Imama Sabilla, B.Comp.Sc., M.Comp.Sc.
Active Status : Inactive
Year of Entry : 2018
Time Work : 8 years
Expertise Competency : Network Engineer
```

https://github.com/guramedadar/Praktikum-Algoritma-Struktur-Data/tree/main/src/com/jobsheet2