

LAPORAN UTS PRAKTIKUM



Disusun Oleh:
Dzakir Tsabit Asy Syafiq (241511071)
Jurusan Teknik Komputer dan Informatika

Program Studi D-3 Teknik Informatika
Politeknik Negeri Bandung
10/10/2025

Kelas : 2C-D3

BAGIAN A

1. Pada gambar 1 diagram kelas employee, solusi keterbasan multiple inheritance di Java dapat dilihat pada penggunaan interface. Java tidak mendukung multiple inheritance secara langsung melalui kelas, tetapi memungkinkan kelas untuk mengimplementasikan beberapa interface. Dalam diagram tersebut, kelas `Employee` di bagian Fulltime nya, ada multiple inheritance yang menunjuk 2 class yaitu koperasi <<interface>> dan menunjuk employee <<class>>. sehingga dapat mewarisi metode dari kedua interface tersebut tanpa mengalami konflik yang biasanya terjadi pada multiple inheritance melalui kelas.

Contoh potongan source code nya adalah sebagai berikut:

file koperasi.java :

```
1 interface Koperasi {
2     void loanMonthly();
3 }
4
```

File Fulltime.java

```
abstract class Fulltime extends Employee implements Koperasi {
    double salary;

    public Fulltime(String name, int id, double salary) {
        super(name, id);
        this.salary = salary;
    }

    @Override
    public void loanMonthly() {
        // Implementasi metode dari interface Koperasi
        System.out.println(name + " is getting a loan from the cooperative.");
    }

    public void displayFulltimeInfo() {
        displayInfo();
        System.out.println("Salary: " + salary);
    }
}
```

File Employee.java :

```

public class Employee {

    String name;
    int id;

    public Employee(String name, int id) {
        this.name = name;
        this.id = id;
    }

    public void displayInfo() {
        System.out.println("Name: " + name + ", ID: " + id);
    }
}

```

2. Pada gambar 1 diagram kelas employee, agregasi has-a relationship dapat dilihat pada hubungan antara kelas `Employee` dan kelas `department`. Dalam hal ini, setiap objek `Employee` memiliki fungsi `departmentName()` karakteristik dari agregasi, di mana `Employee` "memiliki" atau "terdiri dari" satu atau lebih `department`, tetapi `department` dapat eksis secara independen tanpa bergantung pada `Employee`. Ini menunjukkan bahwa `Employee` dan `department` memiliki hubungan yang longgar, di mana `Employee` dapat memiliki referensi ke `department`, tetapi `department` tidak bergantung pada keberadaan `Employee`.

Contoh potongan source code nya adalah sebagai berikut:

File Department.java:

```
W8-UTS > Soal-Diagram > BAGIAN-A > Department.java > Department
1  public class Department {
2
3      String departmentName;
4
5      public Department(String departmentName) {
6          this.departmentName = departmentName;
7      }
8
9      public String getDepartmentName() {
10         return departmentName;
11     }
12 }
13
```

File employee.java (updated) :

```
W8-UTS > Soal-Diagram > BAGIAN-A > Employee.java > Employee
1
2  public class Employee {
3
4      String name;
5      int id;
6      Department department; // Agregasi has-a
7
8      public Employee(String name, int id, Department department) {
9          this.name = name;
10         this.id = id;
11         this.department = department; // Employee memiliki Department
12     }
13
14     public void displayInfo() {
15         System.out.println("Name: " + name + ", ID: " + id + ", Department: " + department.getDepartmentName());
16     }
17 }
18
```

BAGIAN B

1. lengkapi kelas kelas pada diagram kelas yang terdapat pada kelas employee
2. Terapkan enkapsulasi pada kelas-kelas yang akan anda buat!
3. buatlah base salary terkait gaji pokok yang diterapkan pada kelas employee
4. buatlah main kelas untuk menampilkan total gaji ujang dan asep di bulan maret 2025 yang dibayarkan tanggal 1 dibulan april 2025, dengan ketentuan tambahan sebagai berikut
 - Asep adalah staf programmer yang telah bekerja selama 3 tahun, telah mendapatkan tunjangan jabatan, punya 2 anak, dan memiliki cicilan pinjaman koperasi sebesar 500.000 per bulan, sedangkan ujang adalah pegawai parttime yang belum menikah
 - pada tanggal 23 maret 2025 asepe lembur dari pukul 09:00-12:00 kemudian ujang lembur dari pukul 13:00-18:00
 - hitung dan tampilkan total gaji mereka di bulan april

File employee.java :

```
W8-UTS > Soal-Diagram > BAGIAN-B > Employee.java > Employee > Employee(String name, String pos
1
2  abstract class Employee {
3
4      private String name;      Field name can be final
5      private String position;  Field position can be final
6      private Department department; // Agregasi has-a
7      protected int baseSalary;
8
9      public Employee(String name, String position, Department departemen) {
10         this.name = name;
11         this.position = position;
12         this.department = department; // Employee memiliki Department  Assign
13         setBaseSalary();
14     }
15
16     private void setBaseSalary() {
17         switch (position.toLowerCase()) {  Convert to switch expression
18             case "manager":
19                 baseSalary = 5000000;
20                 break;
21             case "programmer":
22                 baseSalary = 3000000;
23                 break;
24             case "analisis":
25                 baseSalary = 3000000;
```

```
W8-UTS > Soal-Diagram > BAGIAN-B > Employee.java > Employee
1
2  abstract class Employee {
3
4      private String name;      Field name can be final
5      private String position;  Field position can be final
6      private Department department;
7      protected int baseSalary;
8
9      public Employee(String name, String position, Department departemen) {
10         this.name = name;
11         this.position = position;
12         this.department = department;  Assignment To Itself
13         setBaseSalary();
14     }
15
16     private void setBaseSalary() {
17         switch (position.toLowerCase()) {  Convert to switch expression
18             case "manager":
19                 baseSalary = 5000000;
20                 break;
21             case "programmer":
22                 baseSalary = 3000000;
23                 break;
24             case "analisis":
25                 baseSalary = 3000000;
```

```

abstract class Employee {
    private void setBaseSalary() {
        break;
        default:
            baseSalary = 0;
    }
}

public String getName() {
    return name;
}

public String getPosition() {
    return position;
}

public Department getDepartemen() {
    return department;
}

public int getBaseSalary() {
    return baseSalary;
}

public abstract int getSalary();

```

Code :

```

abstract class Employee {

    private String name;
    private String position;
    private Department department; // Agregasi has-a
    protected int baseSalary;

    public Employee(String name, String position, Department departemen) {
        this.name = name;
    }
}

```

```

        this.position = position;
        this.department = department; // Employee memiliki Department
        setBaseSalary();
    }

    private void setBaseSalary() {
        switch (position.toLowerCase()) {
            case "manager":
                baseSalary = 5000000;
                break;
            case "programmer":
                baseSalary = 3000000;
                break;
            case "analis":
                baseSalary = 3000000;
                break;
            default:
                baseSalary = 0;
        }
    }

    public String getName() {
        return name;
    }

    public String getPosition() {
        return position;
    }

    public Department getDepartemen() {
        return department;
    }

    public int getBaseSalary() {
        return baseSalary;
    }

    public abstract int getSalary();
}

```


File Department.java

```
W8-UTS > Soal-Diagram > BAGIAN-B > Department.java > Department
1 public class Department {
2
3     private String departemenName;    Field departemenName can be final
4
5     public Department(String name) {
6         this.departemenName = name;
7     }
8
9     public String getDepartemenName() {
10        return departemenName;
11    }
12 }
13
```

File Fulltime.java :

```
W8-UTS > Soal-Diagram > BAGIAN-B > Fulltime.java > Fulltime > Fulltime(String name, String position, Department departemen, int anak, int lemburJam, Koperasi kop
1
2 public class Fulltime extends Employee {
3
4     private int tunjanganJabatan;
5     private int tunjanganKomunikasi = 500000;    Field tunjanganKomunikasi can be final
6     private int tunjanganAnak;
7     private int lemburJam;    Field lemburJam can be final
8     private Koperasi koperasi;    Field koperasi can be final
9
10    public Fulltime(String name, String position, Department departemen, int anak, int lemburJam, Koperasi koperasi) {
11        super(name, position, departemen);
12        setTunjanganJabatan();
13        setTunjanganAnak(anak);
14        this.lemburJam = lemburJam;
15        this.koperasi = koperasi;
16    }
17
18    private void setTunjanganJabatan() {
19        switch (getPosition().toLowerCase()) {    Convert to switch expression
20            case "manager":
21                tunjanganJabatan = 5000000;
22                break;
23            case "programmer":
24                tunjanganJabatan = 2000000;
25                break;
26            " " :
27                tunjanganJabatan = 0;
28                break;
29        }
30    }
31}
```

```
W8-UTS > Soal-Diagram > BAGIAN-B > Fulltime.java > Fulltime > Fulltime(String
2   public class Fulltime extends Employee {
18      private void setTunjanganJabatan() {
27          tunjanganJabatan = 3000000;
28          break;
29      default:
30          tunjanganJabatan = 0;
31      }
32  }
33
34      private void setTunjanganAnak(int anak) {
35          tunjanganAnak = Math.min(anak, 2) * 500000;
36      }
37
38      private int getTunjanganLembur() {
39          return lemburJam * 30000;
40      }
41
42      @Override
43      public int getSalary() {
44          return getBaseSalary()
45              + getTunjanganLembur()
46              + tunjanganJabatan
47              + tunjanganAnak
48              + tunjanganKomunikasi
49              - koperasi.getLoanMonthly();
50      }
51  }
```

Code :

```
public class Fulltime extends Employee {

    private int tunjanganJabatan;
    private int tunjanganKomunikasi = 500000;
    private int tunjanganAnak;
    private int lemburJam;
    private Koperasi koperasi;
```

```

    public Fulltime(String name, String position, Department departemen, int anak, int lemburJam, Koperasi
koperasi) {

        super(name, position, departemen);

        setTunjanganJabatan();

        setTunjanganAnak(anak);

        this.lemburJam = lemburJam;

        this.koperasi = koperasi;

    }

    private void setTunjanganJabatan() {

        switch (getPosition().toLowerCase()) {

            case "manager":

                tunjanganJabatan = 5000000;

                break;

            case "programmer":

                tunjanganJabatan = 2000000;

                break;

            case "analis":

                tunjanganJabatan = 3000000;

                break;

            default:

                tunjanganJabatan = 0;

        }

    }

    private void setTunjanganAnak(int anak) {

        tunjanganAnak = Math.min(anak, 2) * 500000;

    }

    private int getTunjanganLembur() {

        return lemburJam * 30000;

    }

    @Override

    public int getSalary() {

        return getBaseSalary()

            + getTunjanganLembur()

            + tunjanganJabatan

            + tunjanganAnak

            + tunjanganKomunikasi

            - koperasi.getLoanMonthly();

    }

```

```
}
```

File main.java :

```
W8-UTS > Soal-Diagram > BAGIAN-B > Main.java > Main > main(String[] args)
1  public class Main {
3  public static void main(String[] args) {
5
6      // Data Asep (Fulltime Programmer)
7      Koperasi koperasiAsep = new Koperasi(500000);
8      Fulltime asepp = new Fulltime(
9          "Asep", "Programmer", itDept,
10         2, // anak
11         3, // lembur jam (09:00-12:00)
12         koperasiAsep
13     );
14
15     // Data Ujang (Parttime Programmer)
16     Parttime ujang = new Parttime(
17         "Ujang", "Programmer", itDept,
18         5 // lembur jam (13:00-18:00)
19     );
20
21     System.out.println("Gaji bulan April 2025:");
22     System.out.println("Asep (Fulltime): Rp " + asepp.getSalary());
23     System.out.println("Ujang (Parttime): Rp " + ujang.getSalary());
24 }
25 }
26
```

```
public class Main {
```

```
    public static void main(String[] args) {
```

```
        Department itDept = new Department("IT");
```

```
        // Data Asep (Fulltime Programmer)
```

```
        Koperasi koperasiAsep = new Koperasi(500000);
```

```
        Fulltime asepp = new Fulltime(
```

```
            "Asep", "Programmer", itDept,
```

```
            2, // anak
```

```
            3, // lembur jam (09:00-12:00)
```

```
            koperasiAsep
```

```
        );
```

```
        // Data Ujang (Parttime Programmer)
```

```
        Parttime ujang = new Parttime(
```

```
            "Ujang", "Programmer", itDept,
```

```
            5 // lembur jam (13:00-18:00)
```

```
        );
```

```

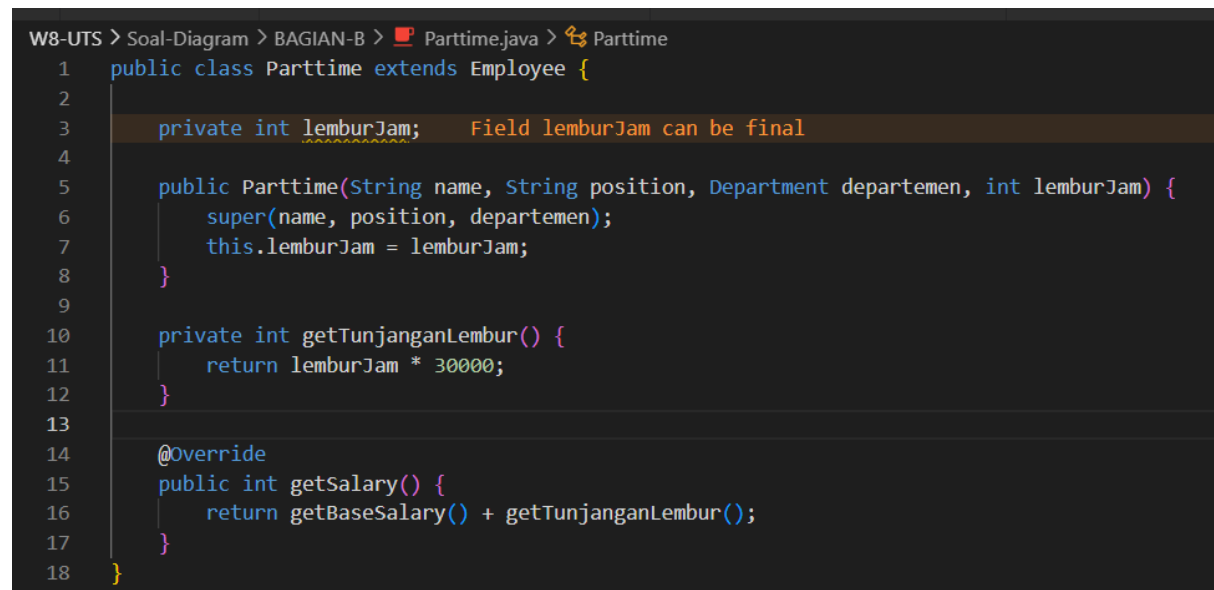
        System.out.println("Gaji bulan April 2025:");

        System.out.println("Asep (Fulltime): Rp " + asep.getSalary());

        System.out.println("Ujang (Parttime): Rp " + ujang.getSalary());
    }
}

```

File partime.java :



```

W8-UTS > Soal-Diagram > BAGIAN-B > Parttime.java > Parttime
1  public class Parttime extends Employee {
2
3      private int lemburJam;      Field lemburJam can be final
4
5      public Parttime(String name, String position, Department departemen, int lemburJam) {
6          super(name, position, departemen);
7          this.lemburJam = lemburJam;
8      }
9
10     private int getTunjanganLembur() {
11         return lemburJam * 30000;
12     }
13
14     @Override
15     public int getSalary() {
16         return getBaseSalary() + getTunjanganLembur();
17     }
18 }

```

Code :

```

public class Parttime extends Employee {

    private int lemburJam;

    public Parttime(String name, String position, Department departemen, int lemburJam) {
        super(name, position, departemen);
        this.lemburJam = lemburJam;
    }

    private int getTunjanganLembur() {
        return lemburJam * 30000;
    }

    @Override
    public int getSalary() {
        return getBaseSalary() + getTunjanganLembur();
    }
}

```

HASIL OUTPUT:

```
PS D:\SEMESTER 3 TEKNIK INFORMATIKA\BERORIENTASI-OBJECT\GIT\Object-Oriented>
Gaji bulan April 2025:
Asep (Fulltime): Rp 6090000
Ujang (Parttime): Rp 3150000
PS D:\SEMESTER 3 TEKNIK INFORMATIKA\BERORIENTASI-OBJECT\GIT\Object-Oriented>
```