

# **LAPORAN 8**

## **Pemrograman Berorientasi Objek**

**“Pewarisan & Koleksi  
Heterogen”**

**Dosen Pengampu : Paulina Heruningsih Prima Rosa.**



**DIBUAT OLEH :**

**Nama : Yohanis Calvin D.P.U.Pati**

**NIM : 245314033**

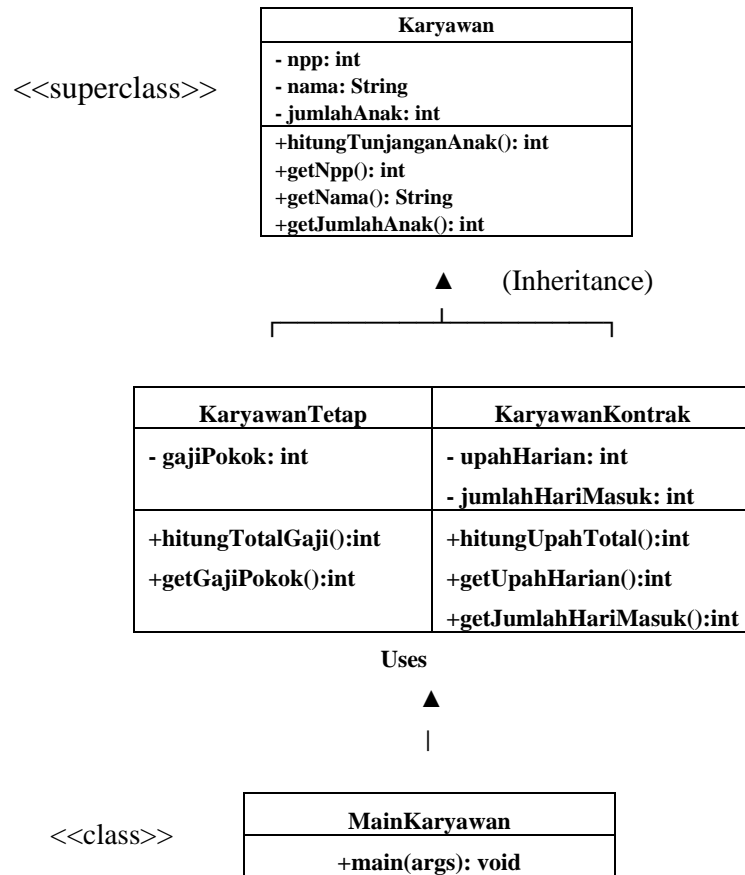
**KELAS : BP**

**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS SANATA DHARMA  
YOGYAKARTA  
2024**

# Kasus 1

## A. DIAGRAM CLASS

Penggambaran diagram kelas



## B. LISTING PROGRAM

```

8 class Karyawan {
9     // deklarasi atribut private
10    private int npp; // Nomor Pegawai
11    private String nama; // Nama Karyawan
12    private int jumlahAnak; // Jumlah anak karyawan
13
14    // Variabel static untuk subsidi per anak (nilai sama untuk semua karyawan)
15    public static int subsidiPerAnak = 100000;
16
17    // Constructor untuk inisialisasi objek Karyawan
18    public Karyawan(int npp, String nama, int jumlahAnak) {
19        this.npp = npp; // Set nilai npp
20        this.nama = nama; // Set nilai nama
21        this.jumlahAnak = jumlahAnak; // Set nilai jumlahAnak
22    }
23
24    // Method menghitung tunjangan anak
25    public int hitungTunjanganAnak() {
26        return jumlahAnak * subsidiPerAnak; // Hitung total subsidi anak
27    }
28
29    // Getter methods untuk mengakses atribut private
30    public int getNpp() {
31        return npp;
32    }
33
34    public String getNama() {
35        return nama;
36    }
37
38    public int getJumlahAnak() {
39        return jumlahAnak;
40    }
41
42
43
44
45

```

```

8 /**
9  * Author Calvin Pati
10  */
11 class KaryawanKontrak extends Karyawan {
12     // Upah per hari
13     private int upahHari;
14     private int jumlahHariMasuk; // Jumlah hari kerja
15
16     // Constructor Karyawan Kontrak
17     public KaryawanKontrak(int npp, String nama, int jumlahAnak, int upahHari, int jumlahHariMasuk) {
18         super(npp, nama, jumlahAnak); // Panggil constructor parent class
19         this.upahHari = upahHari; // Set nilai upahHari
20         this.jumlahHariMasuk = jumlahHariMasuk; // Set nilai jumlahHariMasuk
21     }
22
23     // Method hitung total upah (upah harian * hari kerja + tunjangan anak)
24     public int hitungUpahTotal() {
25         return upahHari * jumlahHariMasuk + super.hitungTunjanganAnak();
26     }
27
28     // Getter methods
29     public int getUpahHari() {
30         return upahHari;
31     }
32
33     public int getJumlahHariMasuk() {
34         return jumlahHariMasuk;
35     }
36
37
38
39
40

```

```

8 /**
9  * Author Calvin Pati
10  */
11 class KaryawanTetap extends Karyawan {
12     // Gaji pokok karyawan tetap
13     private int gajiPokok;
14
15     // Constructor karyawan tetap
16     public KaryawanTetap(int npp, String nama, int jumlahAnak, int gajiPokok) {
17         super(npp, nama, jumlahAnak); // Panggil constructor parent class
18         this.gajiPokok = gajiPokok; // Set nilai gajiPokok
19     }
20
21     // Method hitung total gaji (gaji pokok + tunjangan anak)
22     public int hitungTotalGaji() {
23         return gajiPokok + super.hitungTunjanganAnak();
24     }
25
26     public int getGajiPokok() {
27         return gajiPokok;
28     }
29
30
31
32
33

```

```

11 /**
12  * Author Calvin Pati
13  */
14 public class MainKaryawan {
15     // Static void main(String[] args) {
16     Scanner scanner = new Scanner(System.in); // Buat objek Scanner
17     ArrayList<Karyawan> daftarKaryawan = new ArrayList<>(); // Buat koleksi heterogen
18
19     // Input jumlah karyawan
20     System.out.println("Masukkan jumlah karyawan: ");
21     int jumlah = scanner.nextInt();
22     scanner.nextLine(); // Bersihkan newline di buffer
23
24     // Loop untuk input data setiap karyawan
25     for(int i=0; i<jumlah; i++) {
26         System.out.println("Mute Karyawan ke-" + (i+1));
27
28         // Input jenis karyawan
29         System.out.println("Jenis Karyawan (1=Tetap, 2=Kontrak): ");
30         int jenis = scanner.nextInt();
31         scanner.nextLine(); // Bersihkan newline di buffer
32
33         // Input data umum semua karyawan
34         System.out.println("NPP: ");
35         int npp = scanner.nextInt();
36         scanner.nextLine();
37         System.out.println("Nama: ");
38         String nama = scanner.nextLine();
39         System.out.println("Jumlah anak: ");
40         int anak = scanner.nextInt();
41
42         // Input data spesifik berdasarkan jenis karyawan
43         if(jenis == 1) {
44             // Input data karyawan tetap
45             System.out.println("Gaji pokok: ");
46             int gaji = scanner.nextInt();
47             daftarKaryawan.add(new KaryawanTetap(npp, nama, anak, gaji));
48         } else {
49             // Input data karyawan kontrak
50             System.out.println("Upah harian: ");
51             int upah = scanner.nextInt();
52             System.out.println("Jumlah hari masuk: ");
53             int hari = scanner.nextInt();
54             daftarKaryawan.add(new KaryawanKontrak(npp, nama, anak, upah, hari));
55         }
56     }
57
58     // Tampilkan semua karyawan
59     System.out.println("Daftar Seluruh Karyawan:");
60     System.out.println("-----");
61     for(Karyawan k : daftarKaryawan) {
62         System.out.println("Nama: " + k.getNama());
63
64         // Cek jenis Karyawan menggunakan instanceof
65         if(k instanceof KaryawanTetap) {
66             KaryawanTetap tetap = (KaryawanTetap)k; // Downcasting
67             System.out.println("Jenis: Tetap");
68             System.out.println("Total Gaji: Rp" + tetap.hitungTotalGaji());
69         } else {
70             KaryawanKontrak kontrak = (KaryawanKontrak)k; // Downcasting
71             System.out.println("Jenis: Kontrak");
72             System.out.println("Total Upah: Rp" + kontrak.hitungUpahTotal());
73         }
74         System.out.println("-----");
75     }
76
77     // Tampilkan karyawan kontrak dengan upah < 2 juta
78     System.out.println("Karyawan Kontrak dengan Upah < Rp1.000.000:");
79     System.out.println("-----");
80     for(Karyawan k : daftarKaryawan) {
81         if(k instanceof KaryawanKontrak) {
82             KaryawanKontrak kontrak = (KaryawanKontrak)k;
83             int total = kontrak.hitungUpahTotal();
84             if(total < 1000000) {
85                 System.out.println("Nama: " + k.getNama());
86                 System.out.println("Total Upah: Rp" + total);
87                 System.out.println("-----");
88             }
89         }
90     }
91
92     scanner.close(); // Tutup Scanner
93
94
95

```

### C. OUTPUT Unit Test nya

- Setelah program nya saya perbaiki dan saya rapikan strukturnya maka program dapat tampil lebih bagus berikut utputnya:

```

Output - Run (MainKaryawan) x Atas.java x Bawah.java x Atas_priv.java x Bawah
Scanning for projects...
Building File 1.0-SNAPSHOT
from pom.xml
----- [ JAR ] -----
--- resources2.3.1:resources (default-resources) 0 Fx ---
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\Flx\src\main\resources
--- compile2.1.3:compile (default-compile) 0 Fx ---
Recompiling the module because of changed source code.
Compiling 0 source files with javac [debug release 23] to target\classes
--- exec2.1.0:exec (default-cli) 0 Fx ---
Masukkan jumlah karyawan: 2
Data Karyawan ke-1
Jenis karyawan (1=Tetap, 2=Kontrak): 1
NPP: 200
Nama: Cia
Jumlah anak: 2
Gaji pokok: 1000000
Data Karyawan ke-2
Jenis karyawan (1=Tetap, 2=Kontrak): 2
NPP: 200
Nama: delta
Jumlah anak: 2
Upah harian: 100
Jumlah hari kerja: 6
Definisi Seluruh Karyawan:
=====
Nama: Cia
Jenis: Tetap
Total Gaji: Rp1200000
-----
Nama: delta
Jenis: Kontrak
Total Upah: Rp200000
-----
Karyawan Kontrak dengan Upah < Rp1.000.000:
=====
Nama: delta
Total Upah: Rp200000
=====
BUILD SUCCESS
Total time: 01:04 min
Finished at: 2025-04-29T20:44:53+07:00
  
```

### D. ANALISA

- Class Karyawan

No	Syntax	penjelasan
1	<pre>public class Karyawan{     private int npp;     private String nama;     private int jumlahAnak;</pre>	// Atribut/variabel instan dari class Karyawan yang diset private agar hanya bisa diakses melalui method (encapsulation).
2	<pre>public static int subsidiPernk =     100;</pre>	// Atribut static berarti dimiliki bersama oleh semua objek dari class Karyawan.
3	<pre>public Karyawan(int npp, String nama, int jumlahAnak) {     this.npp = npp;     this.nama = nama;     this.jumlahAnak = jumlahAnak; }</pre>	// Constructor dengan parameter yang digunakan untuk menginisialisasi data saat objek dibuat.
4	<pre>public int hitungTunjanganAnak() { return jumlahAnak * subsidiPerAnak; }</pre>	//Method yang menghitung tunjangan anak berdasarkan jumlah anak dan subsidi per anak.
5	<pre>public int getNpp() { return npp; } public String getNama() { return nama; } public int getJumlahAnak() { return jumlahAnak; }</pre>	//Method getter untuk mengambil nilai atribut npp, nama, dan jumlahAnak.

- **Class KaryawanTetap**

No	Syntax	penjelasan
1	<code>class KaryawanTetap extends Karyawan{</code>	// KaryawanTetap adalah subclass (anak) dari Karyawan, artinya mewarisi atribut dan method dari superclass Karyawan (hubungan pewarisan/inheritance).
2	<code>Private int gajiPokok;</code>	// Atribut khusus untuk karyawan tetap, bersifat private, menyimpan nilai gaji pokok.
3	<code>public KaryawanTetap(int npp, String nama, int jumlahAnak, int gajiPokok) { super(npp, nama, jumlahAnak); this.gajiPokok = gajiPokok; }</code>	//Constructor dengan parameter. Menggunakan super(...) untuk memanggil constructor superclass, lalu menyetel nilai gajiPokok.
4	<code>public int hitungTotalGaji() { return gajiPokok + super.hitungTunjanganAnak(); }</code>	//Method untuk menghitung total gaji, yaitu gaji pokok ditambah tunjangan anak yang dihitung dari superclass.
5	<code>public int getGajiPokok() { return gajiPokok; }</code>	//Getter untuk mengakses nilai atribut gajiPokok. Mengikuti prinsip encapsulation.

- **Class KaryawanKontrak**

No	Syntax	penjelasan
1	<code>class KaryawanKontrak extends Karyawan {</code>	// Mendeklarasikan kelas KaryawanKontrak yang merupakan subclass dari kelas Karyawan, sehingga mewarisi atribut dan metode dari kelas Karyawan.
2	<code>private int upahHarian;</code>	// Mendeklarasikan atribut upahHarian, yang menyimpan informasi tentang upah per hari untuk karyawan kontrak, bersifat private.
3	<code>private int jumlahHariMasuk;</code>	//Mendeklarasikan atribut jumlahHariMasuk, yang menyimpan jumlah hari kerja bagi karyawan kontrak, bersifat private.

4	public KaryawanKontrak(int npp, String nama, int jumlahAnak, int upahHarian, int jumlahHariMasuk) {	//Constructor untuk kelas KaryawanKontrak dengan parameter yang menerima data seperti nomor pokok pegawai (NPP), nama, jumlah anak, upah harian, dan jumlah hari masuk. Menggunakan super(...) untuk memanggil konstruktor kelas induk Karyawan dan menginisialisasi atribut kelas.
5	super(npp, nama, jumlahAnak);	//Panggilan konstruktor superclass Karyawan untuk menginisialisasi atribut npp, nama, dan jumlahAnak pada kelas induk.
6	this.upahHarian = upahHarian;	//Menginisialisasi atribut upahHarian dengan nilai yang diterima sebagai parameter pada konstruktor.
7	this.jumlahHariMasuk = jumlahHariMasuk;	//Menginisialisasi atribut jumlahHariMasuk dengan nilai yang diterima sebagai parameter pada konstruktor.
8	public int hitungUpahTotal() {	//Mendeklarasikan metode hitungUpahTotal() yang digunakan untuk menghitung total upah karyawan kontrak.
9	return (upahHarian * jumlahHariMasuk) + super.hitungTunjanganAnak();	//Menghitung total upah dengan mengalikan upah harian dengan jumlah hari masuk, lalu menambahkan tunjangan anak yang dihitung dari superclass Karyawan.
10	public int getUpahHarian() {	//Mendeklarasikan metode getter untuk mengakses nilai upahHarian.
11	return upahHarian;	//Mengembalikan nilai dari atribut upahHarian.
12	public int getJumlahHariMasuk() {	//Mendeklarasikan metode getter untuk mengakses nilai jumlahHariMasuk.
13	return jumlahHariMasuk;	//Mengembalikan nilai dari atribut jumlahHariMasuk.

- **Class MainKaryawan**

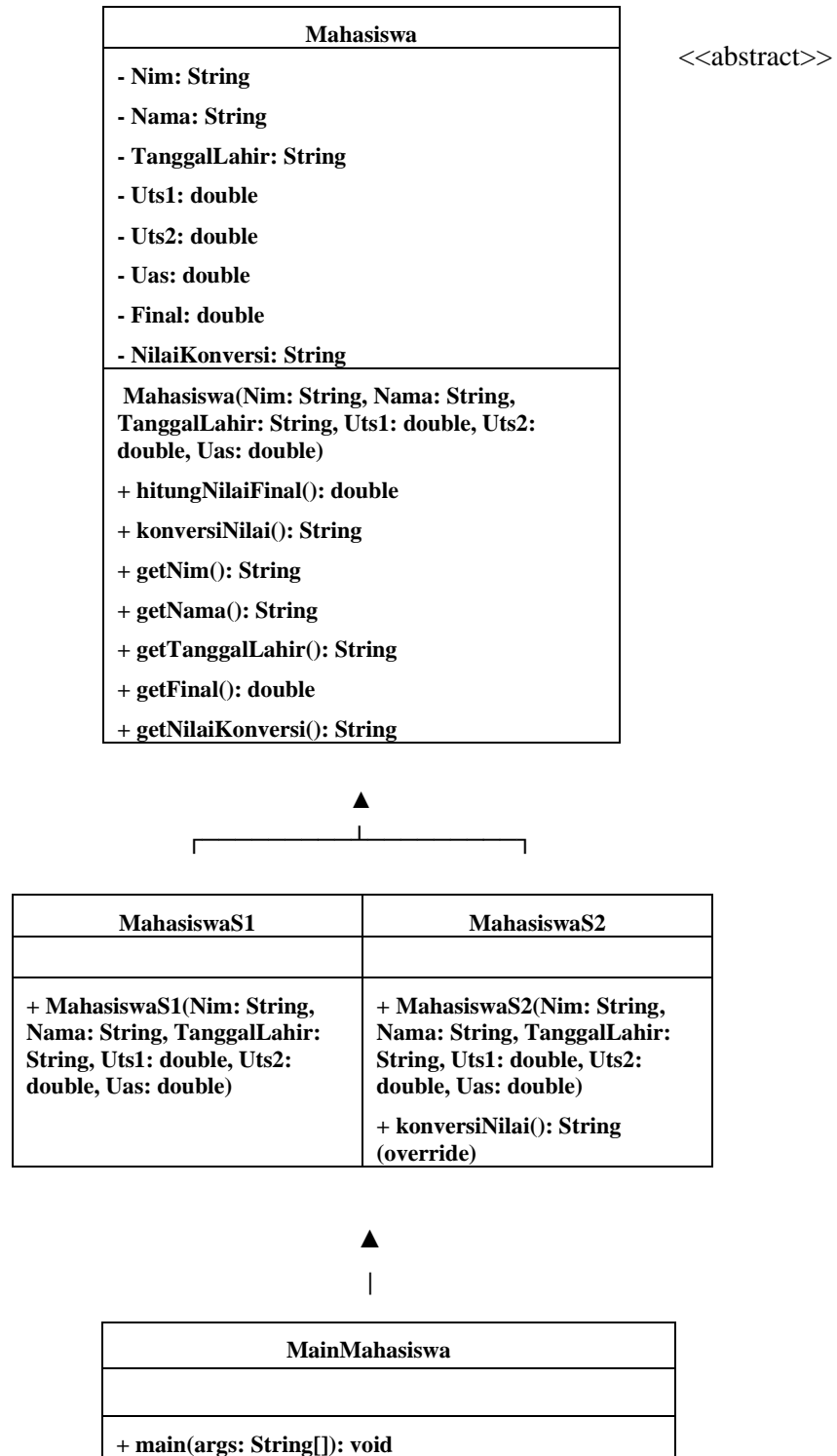
No	Syntax	penjelasan
1	public class MainKaryawan {	// Mendeklarasikan kelas utama MainKaryawan tempat program dieksekusi.
2	Scanner scanner = new Scanner(System.in);	// Membuat objek Scanner untuk membaca input dari pengguna.
3	ArrayList<Karyawan> daftarKaryawan = new ArrayList<>();	//Membuat list daftarKaryawan yang dapat menyimpan objek dari subclass Karyawan, seperti KaryawanTetap dan KaryawanKontrak.
4	System.out.print("Masukkan jumlah karyawan: ");	//Minta input jumlah karyawan.
5	int jumlah = scanner.nextInt();	//Membaca jumlah karyawan dari input.
6	scanner.nextLine();	//Membersihkan karakter newline di buffer agar tidak mengganggu input berikutnya.
7	for(int i=0; i<jumlah; i++) { ... }	//Loop untuk menginput data setiap karyawan.
8	System.out.print("Jenis karyawan (1=Tetap, 2=Kontrak): ");	//Meminta input jenis karyawan.
9	int jenis = scanner.nextInt();	//Membaca input jenis karyawan.
10	System.out.print("NPP: "); ... System.out.print("Jumlah anak: ");	//Input data umum karyawan (NPP, Nama, dan jumlah anak).
11	if(jenis == 1) { ... } else { ... }	//Percabangan untuk membedakan input karyawan tetap dan kontrak.

12	daftarKaryawan.add(new KaryawanTetap(...));	//Menambahkan objek KaryawanTetap ke list setelah input data lengkap.
13	daftarKaryawan.add(new KaryawanKontrak(...));	//Menambah objek KaryawanKontrak ke list.
14	for(Karyawan k : daftarKaryawan) { ... }	//Loop untuk menampilkan seluruh data karyawan.
15	if(k instanceof KaryawanTetap) { ... } else { ... }	//Memakai instanceof untuk memeriksa jenis objek karyawan, lalu melakukan downcasting ke jenis aslinya.
16	System.out.println("Total Gaji: Rp" + tetap.hitungTotalGaji());	//Menampilkan total gaji untuk karyawan tetap.
17	System.out.println("Total Upah: Rp" + kontrak.hitungUpahTotal());	//Menampilkan total upah untuk karyawan kontrak.
18	for(Karyawan k : daftarKaryawan) { ... if(total < 1000000) { ... } }	//Loop untuk menampilkan hanya karyawan kontrak dengan total upah di bawah Rp1.000.000.
19	scanner.close();	//Menutup objek Scanner setelah selesai digunakan.



## Kasus 2

### A. DIAGRAM CLASS



## B. LISTING PROGRAM

```

Source History
9  * @author Calvin Pati
10  */
11  class Mahasiswa {
12      protected String Nim; // Menyimpan NIM mahasiswa
13      protected String Nama; // Menyimpan nama mahasiswa
14      protected String TanggalLahir; // Menyimpan tanggal lahir
15      protected Double Uts1; // Nilai UTS pertama
16      protected Double Uts2; // Nilai UTS kedua
17      protected Double Uas; // Nilai UAS
18      protected Double Final; // Nilai akhir (final)
19      protected String NilaiKonversi; // Nilai akhir dalam huruf (A, B, C)
20
21      // Konstruktor untuk inisialisasi objek Mahasiswa
22      public Mahasiswa(String Nim, String Nama, String TanggalLahir,
23                          Double Uts1, Double Uts2, Double Uas) {
24          this.Nim = Nim;
25          this.Nama = Nama;
26          this.TanggalLahir = TanggalLahir;
27          this.Uts1 = Uts1;
28          this.Uts2 = Uts2;
29          this.Uas = Uas;
30          this.Final = hitungNilaiFinal(); // Hitung nilai akhir saat objek dibuat
31          this.NilaiKonversi = konversiNilai(); // Konversi nilai akhir menjadi huruf
32
33          // Menghitung nilai akhir dari UTS1, UTS2, dan UAS
34          public double hitungNilaiFinal() {
35              return (0.3 * Uts1 + 0.3 * Uts2 + 0.4 * Uas);
36          }
37
38          // Mengubah nilai akhir menjadi nilai huruf (konversi untuk S1)
39          public String konversiNilai() {
40              if (Final >= 85) return "A";
41              else if (Final >= 70) return "B";
42              else if (Final >= 56) return "C";
43              else if (Final >= 45) return "D";
44              else return "E";
45          }
46
47          // Getter untuk tanggal lahir
48          public String getTanggalLahir() {
49              return TanggalLahir;
50          }
51
52          // Getter lainnya
53          public String getNim() { return Nim; }
54          public String getNama() { return Nama; }
55          public double getFinal() { return Final; }
56          public String getNilaiKonversi() { return NilaiKonversi; }
57      }
58  }

```

```

8  *
9  * @author Calvin Pati
10  */
11  class MahasiswaS1 extends Mahasiswa {
12      // Konstruktor Mahasiswa S1 memanggil konstruktor induk
13      public MahasiswaS1(String Nim, String Nama, String TanggalLahir,
14                          double Uts1, double Uts2, double Uas) {
15          super(Nim, Nama, TanggalLahir, Uts1, Uts2, Uas);
16      }
17  }
18
19

```

```

8  *
9  * @author Calvin Pati
10  */
11  class MahasiswaS2 extends Mahasiswa {
12      // Konstruktor Mahasiswa S2 memanggil konstruktor induk
13      public MahasiswaS2(String Nim, String Nama, String TanggalLahir,
14                          double Uts1, double Uts2, double Uas) {
15          super(Nim, Nama, TanggalLahir, Uts1, Uts2, Uas);
16      }
17
18      // Override konversi nilai dengan ketentuan yang berbeda dari S1
19      @Override
20      public String konversiNilai() {
21          if (Final >= 85) return "A";
22          else if (Final >= 70) return "B";
23          else if (Final >= 56) return "C";
24          else if (Final >= 45) return "D";
25          else return "E";
26      }
27  }
28

```

```

Source History
12  * @author Calvin Pati
13  */
14  public class MainMahasiswa {
15
16      public static void main(String[] args) {
17          Scanner input = new Scanner(System.in); // Membuat objek Scanner untuk input
18          ArrayList<Mahasiswa> daftarMahasiswa = new ArrayList<>(); // Menyimpan semua data mahasiswa
19
20          // Meminta jumlah mahasiswa yang akan diinput
21          System.out.print("Masukkan jumlah mahasiswa: ");
22          int jumlah = input.nextInt();
23          input.nextLine(); // Menghapus newline sisa dari nextInt
24
25          // Perulangan untuk input data tiap mahasiswa
26          for (int i = 0; i < jumlah; i++) {
27              System.out.println("Data Mahasiswa ke-" + (i + 1));
28              System.out.print("Jenjang (S1, S2): ");
29              int jenjang = input.nextInt();
30              input.nextLine(); // Membersihkan newline
31              // Input data umum
32              System.out.print("NIM: ");
33              String nim = input.nextLine();
34              System.out.print("Nama: ");
35              String nama = input.nextLine();
36              System.out.print("Tanggal Lahir (DD-MM-YYYY): ");
37              String tglLahir = input.nextLine();
38              // Input nilai
39              System.out.print("Nilai UTS 1: ");
40              double uts1 = input.nextDouble();
41              System.out.print("Nilai UTS 2: ");
42              double uts2 = input.nextDouble();
43              System.out.print("Nilai UAS: ");
44              double uas = input.nextDouble();
45
46              input.nextLine(); // Membersihkan newline
47
48              // Menambahkan objek Mahasiswa sesuai jenjang ke ArrayList
49              if (jenjang == 1) {
50                  daftarMahasiswa.add(new MahasiswaS1(nim, nama, tglLahir, uts1, uts2, uas));
51              } else {
52                  daftarMahasiswa.add(new MahasiswaS2(nim, nama, tglLahir, uts1, uts2, uas));
53              }
54
55              // Menampilkan semua data mahasiswa
56              System.out.println("\nDaftar Seluruh Mahasiswa:");
57              System.out.println("=====");
58              for (Mahasiswa mhs : daftarMahasiswa) {
59                  String jenjang = (mhs instanceof MahasiswaS1) ? "S1" : "S2";
60                  System.out.println("Nama\t\t: " + mhs.getNama());
61                  System.out.println("Jenjang\t\t: " + jenjang);
62                  System.out.println("Nilai Huruf\t: " + mhs.getNilaiKonversi());
63                  System.out.println("-----");
64              }
65
66              // Menampilkan hanya mahasiswa S1 yang lulus (nilai > 70)
67              System.out.println("\nDaftar Mahasiswa S1 yang lulus:");
68              System.out.println("-----");
69              for (Mahasiswa mhs : daftarMahasiswa) {
70                  if (mhs instanceof MahasiswaS1 && mhs.getNilaiKonversi().equals("E")) {
71                      System.out.println("Nama\t\t: " + mhs.getNama());
72                      System.out.println("Nilai Huruf\t: " + mhs.getNilaiKonversi());
73                      System.out.println("-----");
74                  }
75              }
76
77              input.close(); // Menutup scanner agar tidak terjadi kebocoran resource
78          }
79      }
80  }

```

## C. OUTPUT

- Setelah program nya saya perbaiki dan saya rapikan strukturnya maka program dapat menampilkan hasil yang lebih baik dari pada hasil kerja saa di lab:

```

Output - Run (MainMahasiswa) x Atas.java x Bawah.java x Atas_priv.java
-----< com.mycompany.Fix >-----
Building Fix 1.0-SNAPSHOT
from pom.xml

----- { jax } -----

--- resources:3.3.1:resources (default-resources) @ Fix ---
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\Fix\src\main\
--- compiler:3.3.0:compile (default-compile) @ Fix ---
Nothing to compile - all classes are up to date.
--- exec:3.1.0:exec (Default-CLI) @ Fix ---
Masukkan jumlah mahasiswa: 2

Data Mahasiswa No-1
Jenjang (1=SI, 2=SI2): 1
NIM: 245314033
Nama: oelvin
Tanggal Lahir (DD-MM-YYYY): 14-05-2006
Nilai UTS 1: 90
Nilai UTS 2: 90
Nilai UAS: 70

Data Mahasiswa No-2
Jenjang (1=SI, 2=SI2): 2
NIM: 215314022
Nama: oelvin
Tanggal Lahir (DD-MM-YYYY): 29-05-2006
Nilai UTS 1: 40
Nilai UTS 2: 90
Nilai UAS: 77

Daftar Seluruh Mahasiswa:
=====
Nama      : oelvin
Jenjang   : SI
Nilai Huruf : B
=====
Nama      : 215314022
Jenjang   : SI2
Nilai Huruf : C
=====

Daftar Mahasiswa SI yang lulus:
=====
Nama      : oelvin
Nilai Huruf : B
=====

BUILD SUCCESS
-----
Total time: 01:53 min
Finished at: 2025-04-29T21:40:41+07:00

```

## D. ANALISA

- Class Mahasiswa**

No	Syntax	penjelasan
1	<pre> java public Mahasiswa(String Nim, String Nama, String TanggalLahir, double Uts1, double Uts2, double Uas) { this.Nim = Nim; this&gt;Nama = Nama; this.TanggalLahir = TanggalLahir; this.Uts1 = Uts1; this.Uts2 = Uts2; this.Uas = Uas; this.Final = hitungNilaiFinal(); this.NilaiKonversi = konversiNilai(); } </pre>	//Konstruktor untuk menginisialisasi atribut dan langsung menghitung nilai akhir (Final) serta nilai huruf (NilaiKonversi).
2	<pre> java public double hitungNilaiFinal() </pre>	// Method untuk menghitung nilai akhir berdasarkan bobot: UTS1 30%, UTS2 30%, dan UAS 40%.

	<pre> { return (0.3 * Uts1) + (0.3 * Uts2) + (0.4 * Uas); } </pre>	
3	<pre> java public String konversiNilai() { if (Final &gt;= 80) return "A"; else if (Final &gt;= 70) return "B"; else if (Final &gt;= 56) return "C"; else if (Final &gt;= 45) return "D"; else return "E"; } </pre>	// Method untuk mengubah nilai Final menjadi huruf berdasarkan skala penilaian S1.
4	<pre> java public String getTanggalLahir() { return TanggalLahir; } </pre>	// Getter untuk mengambil nilai dari atribut TanggalLahir.
5	<pre> java public String getNim() { return Nim; } </pre>	// Getter untuk mengambil nilai dari atribut Nim.
6	<pre> java public String getNama() { return Nama; } </pre>	// Getter untuk mengambil nilai dari atribut Nama.
7	<pre> java public double getFinal() { return Final; } </pre>	// Getter untuk mengambil nilai akhir (Final).
8	<pre> java public String getNilaiKonversi() { return NilaiKonversi; } </pre>	// Getter untuk mengambil nilai huruf (NilaiKonversi).

- **Class MahasiswaS1**

No	Syntax	penjelasan
1	<pre>java public     MahasiswaS1(String         Nim, String Nama, String         TanggalLahir, double         Uts1, double Uts2, double         Uas) { super(Nim, Nama,         TanggalLahir, Uts1, Uts2,         Uas); }</pre>	// Konstruktor kelas MahasiswaS1 yang memanggil konstruktor kelas induk (Mahasiswa) menggunakan super untuk mengisi data mahasiswa.

- **Class MahasiswaS2**

No	Syntax	penjelasan
1	<pre>java public     MahasiswaS2(String         Nim, String Nama, String         TanggalLahir, double         Uts1, double Uts2, double         Uas) { super(Nim, Nama,         TanggalLahir, Uts1, Uts2,         Uas); }</pre>	// Konstruktor MahasiswaS2 yang memanggil konstruktor induk (Mahasiswa) menggunakan super untuk menginisialisasi data.
2	<pre>java @Override public     String konversiNilai() { if     (Final &gt;= 85) return "A";     else if (Final &gt;= 70)     return "B"; else if (Final     &gt;= 56) return "C"; else if     (Final &gt;= 45) return "D";     else return "E"; }</pre>	//Method override dari konversiNilai() yang mengganti aturan konversi nilai khusus untuk jenjang S2.

- **MainMahasiswa**

No	Syntax	penjelasan
1	<pre>java Scanner input = new     Scanner(System.in);</pre>	// Membuat objek Scanner untuk membaca input dari pengguna.
2	<pre>java     ArrayList&lt;Mahasiswa&gt;     daftarMahasiswa = new     ArrayList&lt;&gt;();</pre>	//Membuat ArrayList untuk menyimpan objek dari class Mahasiswa (baik S1 maupun S2).

3	<pre>javaSystem.out.print("Masukkan jumlah mahasiswa:"); int jumlah = input.nextInt(); input.nextLine();</pre>	//Menerima jumlah mahasiswa yang akan diinput, lalu membersihkan newline.
4	<pre>java for (int i = 0; i &lt; jumlah; i++) { ... }</pre>	//Melakukan pengulangan untuk input data tiap mahasiswa sebanyak jumlah yang dimasukkan.
5	<pre>java System.out.print("Jenjang pendidikan (1=S1, 2=S2): "); int jenjang = input.nextInt(); input.nextLine();</pre>	//Menerima input jenjang pendidikan (S1 atau S2), lalu membersihkan newline.
6	<pre>java System.out.print("NIM:"); String nim = input.nextLine(); ...</pre>	//Mengambil input identitas mahasiswa: NIM, Nama, Tanggal Lahir, dan nilai ujian.
7	<pre>java if (jenjang == 1) { daftarMahasiswa.add(new MahasiswaS1(...)); } else { daftarMahasiswa.add(new MahasiswaS2(...)); }</pre>	//Menambahkan objek ke ArrayList sesuai jenjang: MahasiswaS1 atau MahasiswaS2.
8	<pre>java for (Mahasiswa mhs : daftarMahasiswa) { ... }</pre>	//Menampilkan semua data mahasiswa, termasuk jenjang dan nilai huruf.
9	<pre>java if (mhs instanceof MahasiswaS1 &amp;&amp; !mhs.getNilaiKonversi().equals("E")) { ... }</pre>	//Menampilkan hanya mahasiswa S1 yang nilai hurufnya bukan "E" (berarti lulus).
10	<pre>java input.close();</pre>	Menutup objek Scanner untuk menghindari kebocoran resource.