

# **LAPORAN PEMROGRAMAN BERORIENTASI OBJEK**

## **MODUL KE – V**

<<ARRAY OF OBJECTS SEBAGAI PARAMETER DAN ARGUMEN DALANMETODE>>



Oleh :

Nama : Yohanis Calvin D. P. U. Pati

NIM : 245314033

Kelas : BP

**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS SANATA DHARMA  
SEMESTER GENAP 2025/2026**

## A. Tujuan

1. Mahasiswa mampu mempergunakan array of objects sebagai parameter dan argumen dalam metode.

## B. Pelaksanaan Praktikum

### 1. Soal nomor 1 ( soal utama praktikum)

#### a. Soal

#### **Tugas 1 : Pegawai Perusahaan**

Buatlah kelas Perusahaan dengan atribut namaPerusahaan, alamat, pemilik, NPWP, daftar pegawai dari perusahaan tersebut. Pegawai bertipe class Pegawai yang telah dibuat dalam modul pertemuan sebelumnya. Kemudian buatlah sebuah kelas bernama KelolaPerusahaan yang berisi main class untuk mengisi daftar pegawai dari suatu perusahaan. Class tersebut harus mampu:

- a. Meminta masukan data jumlah pegawai
- b. Memasukkan data pegawai baru.
- c. Mengeset daftar pegawai pada class Perusahaan.
- d. Menampilkan seluruh pegawai yang ada di perusahaan.
- e. Menampilkan pegawai yang bergaji pokok terbesar dan terkecil serta rata-rata gaji pokok pegawai yang ada.

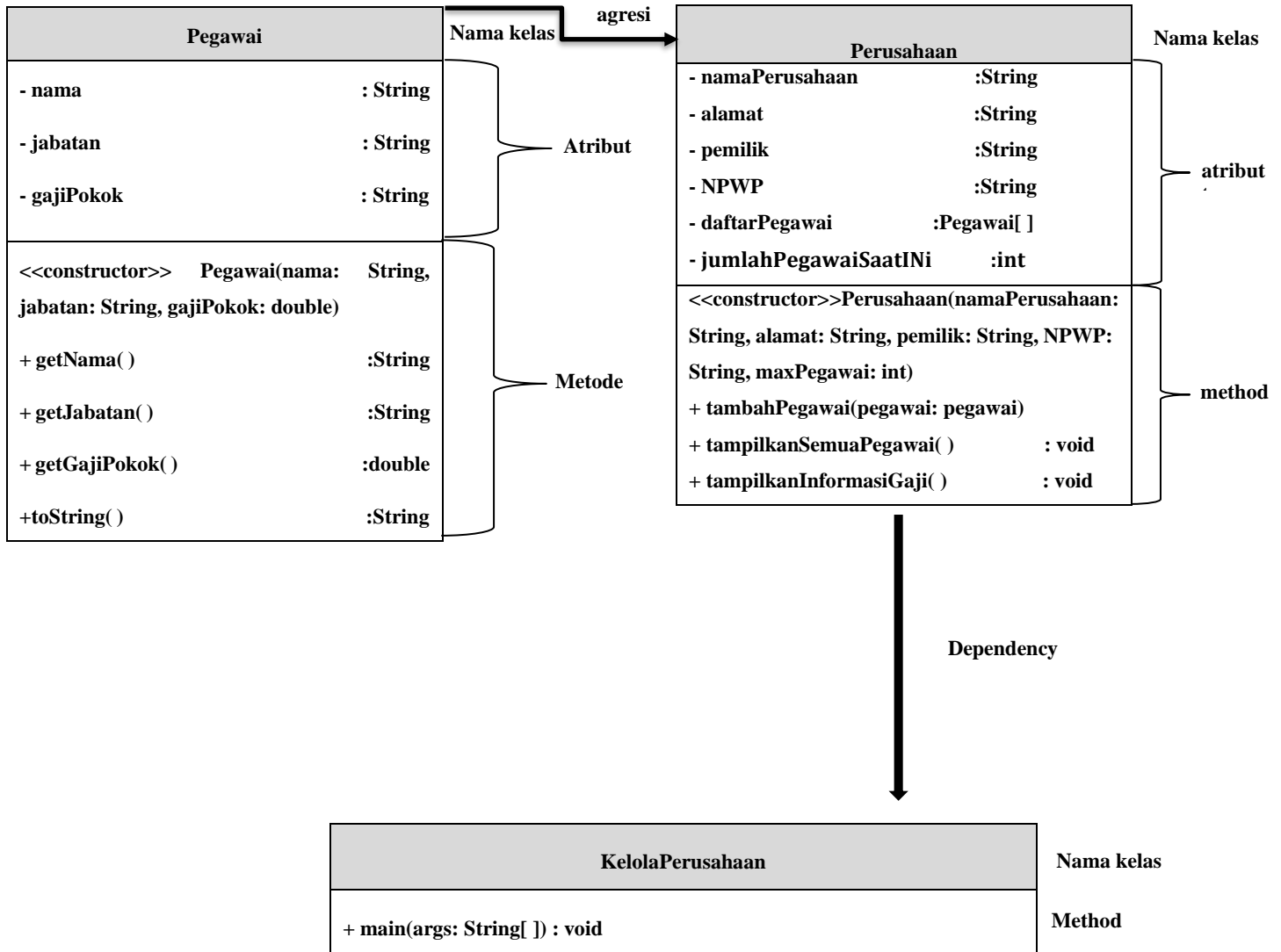
#### b. Gambaran umum:

- Penjelasan singkat kasus yang dipecahkan (1 – 2 kalimat)

Program ini berfungsi untuk mengelola data perusahaan dan pegawai, yang bisa dipakai untuk memasukkan informasi perusahaan yaitu, nama, alamat, pemilik, dan NPWP dan juga data pegawai yaitu, nama, jabatan, gaji pokok. Dan program ini juga bisa menampilkan daftar pegawai, menampilkan pegawai dengan gaji terbesar dan terkecil serta rata-rata gaji pokok pegawai dan menambah pegawai.

- Diagram kelas

Kelas atribut Mahasiswa dan class Main:



- **Penjelasan diagram kelas yang dirancang (kelas-kelas yg ada dan artinya, relasi antar kelas).**

Program ini terdiri dari tiga kelas yaitu kelas Pegawai, kelas Perusahaan, dan kelas KelolaPerusahaan.

### 1. Kelas pegawai:

#### Fungsi:

Kelas ini berfungsi untuk menyimpan dan mengelola data setiap pegawai dan informasi tentang pegawai.

#### Atribut:

- **nama:** Berfungsi untuk menyimpan nama pegawai.
- **jabatan:** ini untuk menyimpan jabatan pegawai.
- **gajiPokok:** ini berfungsi untuk menyimpan gaji pokok pegawai.

#### Constructor & Metode:

- **<<constructor>> Pegawai(nama, jabatan, gajiPokok)** konstruktor ini berfungsi untuk menginisialisasi objek Pegawai.
- **getNama( ):** metode ini berfungsi untuk membaca nama pegawai.
- **getJabatan( ):** metode ini berfungsi untuk membaca jabatan pegawai.
- **getGajiPokok( ):** metode ini berfungsi untuk membaca gaji pokok pegawai.
- **toString( ):** metode ini berfungsi untuk menampilkan informasi tentang pegawai dalam bentuk String.

### 2. Kelas Perusahaan:

#### Fungsi:

kelas ini berfungsi untuk mengelola data perusahaan dan menambah data daftar pegawai baru.

#### Atribut:

- **namaPerusahaan :** untuk simpan nama perusahaan.
- **alamat :** untuk simpan alamat perusahaan.
- **pemilik :** untuk simpan nama pemilik perusahaan.
- **NPWP :** untuk simpan NPWP perusahaan.
- **daftarPegawai :** ini adalah Array untuk simpan daftar pegawai.
- **jmlahPegawaiSaatIni :** ini untuk mengimpon jumlah pegawai yang sudah ditambahkan.

#### Konstruktor & Metode:

- **<<constructor>>Perusahaan(namaPerusahaan, alamat, pemilik, NPWP, maxPegawai)** : konstruktor ini berfungsi untuk inisialisasi objek Perusahaan.
- **tambahPegawai(pegawai)** : ini berfungsi untuk menambah pegawai ke daftar pegawai.
- **tampilkanSemuaPegawai( ):** ini berfungsi untuk menampilkan semua pegawai yang ada dalam daftar pegawai.

- **tampilkanInformasiGaji( )**: method ini berfungsi untuk menampilkan un formasi gaji pegawai (terbesar, terkecil, dan rata-rata).

### 3. **Kelas KelolaPerusahaan**

#### **Fungsi:**

Kelas ini berfungsi untuk menjalankan progeam dan menghubungkan pengguna dengan kelas Perusahaan dan Pegawai.

#### **Method:**

- **Main(args)** : ini adalah method utama untuk menjalankan program, yang berfungsi untuk meminta input dari pengguna dan mengelola Perusahaan dan Pegawai.

### 4. **Relasi antara kelas:**

Relasi antara kelas Pegawai dan kelas Perusahaan adalah Agresi, dan relasi antara kelas Pegwai,Perusahhaan, dan KelolaPerusahaan adalah Dependency.

- Relasi antara kelas Perusahaan dan Pegawai:
  - Jenis relasi: Agresi.
  - Perusahaan dan Pegawai kelas ini berhubungan secara agresi yaitu hubungan mereka bersifat longgal, yaitu objek yang dimiliki kelas Pegawai dapat berdiri sendiri tanpa objek dari Perusahaan.
- Relasi antara kelas KelolaPerusahaan dan Perusahaan:
  - Jenis relasi: Dependency.
  - KelolaPerusahaan dan Perusahaan kela ini berhubungan secara dependency yaitu besifat sementara, yaitu salah satu kelas dari kedua kelas ini dapat memakai kelas lain tanpa memiliki hubungan kepemilikan.

c. Program yang dibuat saat praktikum:

■ GajiPegawai

```
8  *
9  * @author Calvin Pati
10 */
11 public class GajiPegawai { //awal kelas pegawai
12     private String NPP, nama; //atribut NPP, nama
13     private int golongan; //atribut untuk golongan
14
15     public GajiPegawai(String NPP, String nama, int golongan) { //konstruktor GajiPegawai dengan parameter NPP, nama, golongan
16         this.NPP = NPP; //mengatur NPP
17         this.nama = nama; //mengatur nama
18         this.golongan = golongan; //mengatur golongan
19     }
20
21     public int hitungGaji() { //ini adalah metode untuk hitung gaji pegawai
22         if (golongan == 1) {
23             return 1300000; //gaji pegawai golongan 1
24         } else if (golongan == 2) {
25             return 1750000; //gaji pegawai golongan 2
26         } else if (golongan == 3) {
27             return 2000000; //gaji pegawai golongan 3
28         } else {
29             return 0; //jadi nilai akan kembali 0 jika input salah
30         }
31     }
32
33     public void tampilkanInfo() { //metode untuk menampilkan info pegawai
34         System.out.println("NPP: " + NPP + ", Nama: " + nama + ", Gaji: Rp" + hitungGaji());
35     }
36 }
```

■ Perusahaan

```
9  /**
10  *
11  * @author Calvin Pati
12  */
13 public class Perusahaan { //awal kelas perusahaan
14     private String nama; //deklarasi nama yang bersifat private
15     private GajiPegawai[] pegawaiList; //deklarasi gaji pegawai dengan array of objek
16     private int jumlahPegawai; //deklarasi jumlah pegawai bersifat private
17
18     public Perusahaan(String nama, int kapasitas) { //inikonstruktor Perusahaan dengan atribut String nama, int kapasitas
19         this.nama = nama; //untuk atur nama
20         this.pegawaiList = new GajiPegawai[kapasitas]; // untuk atur pegawai dan membuat objek baru
21         this.jumlahPegawai = 0; //untuk atur jumlah pegawai
22     }
23
24     public void tambahPegawai(GajiPegawai pegawai) { //ini metode untuk tambah pegawai
25         if (jumlahPegawai < pegawaiList.length) { //loop untuk menambah pegawai
26             pegawaiList[jumlahPegawai] = pegawai;
27             jumlahPegawai++;
28         } else { //jika melewati kapasitas maka false
29             System.out.println("Kapasitas pegawai sudah penuh!");
30         }
31     }
32
33     public void tampilkanPegawai() { //metode untuk menampilkan pegawai
34         System.out.println("\nDaftar Pegawai di " + nama + ":"); //untuk daftar pegawai
35         for (int i = 0; i < jumlahPegawai; i++) { //ini loop untuk menampilkan pegawai
36             pegawaiList[i].tampilkanInfo();
37         }
38     }
39 }
```

## ▪ KelolaPerusahaan

```

13 public class KelolaPerusahaan { //ini awal kelas KelolaPerusahaan
14     public static void main(String[] args) { //awal metode utama
15         Scanner keyboard = new Scanner(System.in); //ini untuk buat objek scanner yang di input dari keyboard
16
17         Scanner input = new Scanner(System.in); //ini buat objek scanner lagi untuk input
18
19         System.out.print("Masukkan Nama Perusahaan: "); //ini untuk input nama perusahaan
20         String namaPerusahaan = input.nextLine(); //untuk simpan inputan ke namaPerusahaan
21
22         System.out.print("Masukkan jumlah pegawai: "); //ini untuk input jumlah pegawai
23         int jumlahPegawai = input.nextInt(); //ini untuk menyimpan inputan ke jumlahPegawai
24         input.nextLine();
25
26         Perusahaan perusahaan = new Perusahaan(namaPerusahaan, jumlahPegawai); //ini buat objek perusahaan
27
28         for (int i = 0; i < jumlahPegawai; i++) { //ini adalah loop untuk input data pegawai
29             System.out.print("\nNPP: "); //untuk input Npp
30             String NPP = input.nextLine();
31             System.out.print("Nama: "); //untuk nama
32             String nama = input.nextLine();
33             System.out.print("Golongan (1-3): "); //untuk golongan
34             int golongan = input.nextInt();
35             input.nextLine();
36
37             perusahaan.tambahPegawai(new GajiPegawai(NPP, nama, golongan)); //ini untuk memanggil fungsi tambahPegawai
38         }
39
40         // Tampilkan daftar pegawai
41         perusahaan.tampilkanPegawai();
42
43         input.close();
44     }
45 }

```

Activate Windows  
Go to Settings to activate Windows.

## ▪ Output:

```

Output x Pegawai.java x Perusahaan.java x KelolaPerusahaan.java x
Run (KelolaPerusahaan) x PBOModul5 (run) x

run:
Masukkan Nama Perusahaan: pt celvin
Masukkan jumlah pegawai: 5

NPP: 200
Nama: delta
Golongan (1-3): 3

NPP: 300
Nama: cia
Golongan (1-3): 2

NPP: elvian
Nama: elvian
Golongan (1-3): 1

NPP: 201
Nama: dimas
Golongan (1-3): 3

NPP: 400
Nama: eve
Golongan (1-3): 1

Daftar Pegawai di pt celvin:
NPP: 200, Nama: delta, Gaji: Rp2000000
NPP: 300, Nama: cia, Gaji: Rp1750000
NPP: elvian, Nama: elvian, Gaji: Rp1300000
NPP: 201, Nama: dimas, Gaji: Rp2000000
NPP: 400, Nama: eve, Gaji: Rp1300000
BUILD SUCCESSFUL (total time: 1 minute 16 seconds)

```

d. Perbaiki program yang dibuat di rumah disertai penjelasan dalam bentuk komentar pada program:

- kelas Pegawai

```
9      * @author Calvin Pati
10     */
11     public class Pegawai { //awal kelas pegawai
12         // Atribut ini untuk simpan nama pegawai
13         private String nama;
14         // Atribut ini untuk simpan jabatan pegawai
15         private String jabatan;
16         // Atribut ini untuk simpan gaji pokok pegawai
17         private double gajiPokok;
18
19         // ini adalah constructor untuk inisialisasi objek Pegawai
20         public Pegawai(String nama, String jabatan, double gajiPokok) {
21             this.nama = nama; // untuk isi atribut nama
22             this.jabatan = jabatan; // untuk isi atribut jabatan
23             this.gajiPokok = gajiPokok; // untuk isi atribut gajiPokok
24         }
25
26         // Getter untuk untuk baca nama pegawai
27         public String getNama() {
28             return nama;
29         }
30
31         // Getter untuk baca jabatan pegawai
32         public String getJabatan() {
33             return jabatan;
34         }
35
36         // Getter untuk baca gaji pokok pegawai
37         public double getGajiPokok() {
38             return gajiPokok;
39         }
40         //method ini untuk menampilkan informasi pegawai
41         public String toString() {
42             // Menggunakan String.format untuk memformat gaji pokok
43             return String.format(
44                 "Nama: %s, Jabatan: %s, Gaji Pokok: %, .0f",
45                 nama, jabatan, gajiPokok
46             );
47         } //akhir metode utama
48     } //akhir kelas Pegawai
49
50
51
```



## ■ Perusahaan

```

Source History
9  * @author Calvin Pati
10 */
11 public class Perusahaan { //awal kelas Perusahaan
12     // Atribut untuk simpan nama perusahaan
13     private String namaPerusahaan;
14     // Atribut untuk simpan alamat perusahaan
15     private String alamat;
16     // Atribut untuk simpan nama pemilik perusahaan
17     private String pemilik;
18     // Atribut untuk simpan NPWP perusahaan
19     private String NPWP;
20     // Array untuk simpan daftar pegawai
21     private Pegawai[] daftarPegawai;
22     // Variabel untuk lacak jumlah pegawai yang ditambah
23     private int jumlahPegawaiSaatIni;
24
25     // Constructor ini untuk inialisasi objek Perusahaan
26     public Perusahaan(String namaPerusahaan, String alamat, String pemilik, String NPWP, int maxPegawai) {
27         this.namaPerusahaan = namaPerusahaan; // ini untuk isi atribut namaPerusahaan
28         this.alamat = alamat; // untuk isi atribut alamat
29         this.pemilik = pemilik; // untuk isi atribut pemilik
30         this.NPWP = NPWP; // untuk isi atribut NPWP
31         this.daftarPegawai = new Pegawai[maxPegawai]; // ini adalah inialisasi array dgnan ukuran terbesar
32         this.jumlahPegawaiSaatIni = 0; // nilai awal belum ada pegawai jika defuld
33     }
34
35     // Method ini untuk tambahkan pegawai ke dalam array
36     public void tambahPegawai(Pegawai pegawai) {
37         if (jumlahPegawaiSaatIni < daftarPegawai.length) { // untuk cek jika masih ada ruang di array
38             daftarPegawai[jumlahPegawaiSaatIni] = pegawai; // untuk tambah pegawai ke array
39             jumlahPegawaiSaatIni++; // untuk tmbah jumlah pegawai yang sudah ditambahkan
40         } else {
41             System.out.println("Kapasitas pegawai penuh, tidak bisa menambah pegawai lagi.");
42         }
43     }
44
45     // Method ini untuk menampilkan semua pegawai
46     public void tampilkanSemuaPegawai() {
47         if (jumlahPegawaiSaatIni == 0) { // untuk cek jika tidak ada pegawai
48             System.out.println("Tidak ada pegawai.");
49             return;
50         }
51
52         // Loop ini untuk menampilkan semua pegawai
53         for (int i = 0; i < jumlahPegawaiSaatIni; i++) {
54             System.out.println(daftarPegawai[i]); // untuk menampilkan informasi pegawai
55         }
56     }
57
58     // Method ini untuk menampilkan informasi gaji pegawai
59     public void tampilkanInformasiGaji() {
60         if (jumlahPegawaiSaatIni == 0) { // untuk cek jika tidak ada pegawai
61             System.out.println("Tidak ada pegawai.");
62             return;
63         }
64
65         // Inisialisasi pegawai dengan gaji terbesar dan terkecil
66         Pegawai pegawaiTerbesar = daftarPegawai[0];
67         Pegawai pegawaiTerkecil = daftarPegawai[0];
68         double totalGaji = 0; // Variabel untuk simpan total gaji
69
70         // Loop ini untuk cari pegawai dengan gaji terbesar, terkecil, dan hitung total gaji
71         for (int i = 0; i < jumlahPegawaiSaatIni; i++) {
72             Pegawai pegawai = daftarPegawai[i];
73             if (pegawai.getGajiPokok() > pegawaiTerbesar.getGajiPokok()) {
74                 pegawaiTerbesar = pegawai; // untuk perbaharui pegawai dengan gaji terbesar
75             }
76             if (pegawai.getGajiPokok() < pegawaiTerkecil.getGajiPokok()) {
77                 pegawaiTerkecil = pegawai; // untuk perbaharui pegawai dengan gaji terkecil
78             }
79             totalGaji += pegawai.getGajiPokok(); // untuk tambah gaji ke totalGaji
80         }
81
82         // untuk hitung rata-rata gaji
83         double rataRataGaji = totalGaji / jumlahPegawaiSaatIni;
84
85         // untuk menampilkan informasi gaji
86         System.out.println("Pegawai dengan gaji terbesar: " + pegawaiTerbesar);
87         System.out.println("Pegawai dengan gaji terkecil: " + pegawaiTerkecil);
88         System.out.println("Rata-rata gaji pokok: " + rataRataGaji);
89     }
90 } //akhir kelas Perusahaan
91
92

```

## ■ KelolaPerusahaan

```
11  * @author Calvin Pati
12  */
13  public class KelolaPerusahaan { //awal kelas Perusahaan
14      public static void main(String[] args) { //awal metode utama
15          Scanner scanner = new Scanner(System.in); // untuk buat objek Scanner untuk input
16
17          // Input data perusahaan
18          System.out.print("Masukkan nama perusahaan: ");
19          String namaPerusahaan = scanner.nextLine(); // baca nama perusahaan
20          System.out.print("Masukkan alamat perusahaan: ");
21          String alamat = scanner.nextLine(); // baca alamat perusahaan
22          System.out.print("Masukkan nama pemilik: ");
23          String pemilik = scanner.nextLine(); // baca nama pemilik
24          System.out.print("Masukkan NPWP: ");
25          String NPWP = scanner.nextLine(); // baca NPWP
26
27          // untuk input jumlah maksimum pegawai
28          System.out.print("Masukkan jumlah maksimum pegawai: ");
29          int maxPegawai = scanner.nextInt(); // untuk baca jumlah maksimum pegawai
30          scanner.nextLine(); // untuk newline
31
32          // untuk buat objek Perusahaan
33          Perusahaan perusahaan = new Perusahaan(namaPerusahaan, alamat, pemilik, NPWP, maxPegawai);
34
35          // untuk input jumlah pegawai yang mau ditambah
36          System.out.print("Masukkan jumlah pegawai yang akan ditambahkan: ");
37          int jumlahPegawai = scanner.nextInt(); // untuk baca jumlah pegawai
38          scanner.nextLine(); // untuk newline
39
40          // untuk input data pegawai
41          for (int i = 0; i < jumlahPegawai; i++) {
42              System.out.println("Masukkan data pegawai ke-" + (i + 1));
43              System.out.print("Nama: ");
44
45              String nama = scanner.nextLine(); // untuk baca nama pegawai
46              System.out.print("Jabatan: ");
47              String jabatan = scanner.nextLine(); // baca jabatan pegawai
48              System.out.print("Gaji Pokok: ");
49              double gajiPokok = scanner.nextDouble(); // baca gaji pokok pegawai
50              scanner.nextLine(); // untuk newline
51
52              // untuk buat objek Pegawai dan ditambah ke perusahaan
53              Pegawai pegawai = new Pegawai(nama, jabatan, gajiPokok);
54              perusahaan.tambahPegawai(pegawai);
55
56              // untuk menampilkan semua pegawai
57              System.out.println("\nDaftar Pegawai:");
58              perusahaan.tampilkanSemuaPegawai();
59
60              // untuk menampilkan informasi gaji pegawai
61              System.out.println("\nInformasi Gaji Pegawai:");
62              perusahaan.tampilkanInformasiGaji();
63
64              scanner.close(); // Menutup Scanner
65          } //akhir metode utama
66      } //akhir kelas KelolaPerusahaan
67
68  }
```

Activate  
Go to Setti

▪ Output:

```
Output - Run (KelolaPerusahaan) X Pegawai.java X Perusahaan.java X KelolaPerusahaan.java
--- exec:3.1.0:exec (default-cli) @ perbaikanPBOModul5 ---
Masukkan nama perusahaan: PT. Trade Ptatama
Masukkan alamat perusahaan: Maguwoharjo, Sleman, Yogyakarta, JL.Paingan II, No.43A.
Masukkan nama pemilik: Calvin Umbu Pati
Masukkan NPWP: 202709
Masukkan jumlah maksimum pegawai: 10.000
Masukkan jumlah pegawai yang akan ditambahkan: 5
Masukkan data pegawai ke-1
Nama: Max
Jabatan: Manager
Gaji Pokok: 24.000.000
Masukkan data pegawai ke-2
Nama: Bastian
Jabatan: Assistant
Gaji Pokok: 25.000.000
Masukkan data pegawai ke-3
Nama: Jonathan
Jabatan: Web Developer
Gaji Pokok: 24.000.000
Masukkan data pegawai ke-4
Nama: Cindy
Jabatan: Sekretaris
Gaji Pokok: 21.000.000
Masukkan data pegawai ke-5
Nama: Cantika
Jabatan: 22.000.000
Gaji Pokok: 22.000.000

Daftar Pegawai:
Nama: Max, Jabatan: Manager, Gaji Pokok: 24.000.000
Nama: Bastian, Jabatan: Assistant, Gaji Pokok: 25.000.000
Nama: Jonathan, Jabatan: Web Developer, Gaji Pokok: 24.000.000
Nama: Cindy, Jabatan: Sekretaris, Gaji Pokok: 21.000.000
Nama: Cantika, Jabatan: 22.000.000, Gaji Pokok: 22.000.000

Informasi Gaji Pegawai:
Pegawai dengan gaji terbesar: Nama: Bastian, Jabatan: Assistant, Gaji Pokok: 25.000.000
Pegawai dengan gaji terkecil: Nama: Cindy, Jabatan: Sekretaris, Gaji Pokok: 21.000.000
Rata-rata gaji pokok: 2.32E7

-----
BUILD SUCCESS
-----

Total time: 04:28 min
Finished at: 2025-02-22T14:18:40:07:00
```

e. Analisis perbandingan program saat praktikum dan perbaikan program:

- **perbedaan kedua program:**

Aspek Perbedaan	Program sebelum perbaikan	Program Sesudah perbaikan
Kelas	Di program pertama terdiri dari 3 kelas; gajiPegawai, Perusahaan, dan KelolaPerusahaan.	Di program ke dua terdiri dari 3 kelas: Pegawai, Perusahaan, dan KelolaPerusahaan.
Atribut Pegawai	Hanya mempunyai 3 atribut: NPP, nama, dan golongan.	Memiliki 3 atribut: nama, jabatan, dan gajiPokok.
Perhitungan gaji	Gaji dihitung sesuai dengan golongan (1, 2, atau 3).	Gaji langsung diinput sebagai gajiPokok, dan tidak ada perhitungan sesuai dengan golongan.
Fleksibilitas	Kurang fleksibel karena gaji ditentukan dengan golongan yang terbatas.	Lebih fleksibel karena gaji bisa langsung diinput oleh pengguna tanpa Batasan golongan.
Informasi yang ditampilkan	Hanya menampilkan NPP, nama, dan gaji.	Bisa menampilkan nama, jabatan, gaji pokok, dan informasi gaji terbesar, terkecil, dan rata-rata.
Kapasitas pegawai	Kapasitas pegawai ditentukan sesuai dengan input pengguna saat membuat objek Perusahaan.	Kapasitas pegawai juga ditentukan sesuai dengan input pengguna, tapi lebih terstruktur.
Error handling	Tidak ada seperti penanganan error jika input golongan tidak valid (hanya mengembalikan nilai default 0).	Tidak ada perubahan seperti itu dalam penanganan error.

- **Kekurangan program pertama (sebelum perbaikan):**

1. Keterbatasan dengan golongan:

- Yaitu gaji pegawai ditentukan sesuai dengan golongan (1,2,3), jadi kurang fleksibel karena kalau perusahaan mau memberikan gaji diluar ketentuan program bisa error.

2. Informasi yang terbatas:

- Program pertama hanya menampilkan NPP, nama, dan gaji. Dan tidak ada informasi tambahan seperti jabatan analisis gaji (terbesar, terkecil, dan rata-rata).

3. Tidak ada penanganan error:
    - Di program pertama kalau input golongan tidak valid contoh golongan = 4, program hanya mengembalikan gaji 0 tanpa pesan error yang jelas.
  4. Struktur kurang terorganisir:
    - Program pertama tidak punya method untuk menampilkan informasi gaji secara detail seperti, gaji terbesar, terkecil dan rata-rata.
- **Kelebihan program kedua (setelah perbaikan)**
    1. Fleksibilitas gaji:
      - Di program kedua ini gaji bisah langsung diinput sebagai gajiPokok, jadi tidak terbatas dengan golongan tertentu seperti program pertama.
    2. Informasi lebih lengkap:
      - Di program kedua bisa menampilkan informasi lengkap tentang pegawai, yaitu nama, jabatan, dan gaji pokok.
    3. Struktur lebih terorganisir:
      - Program kedua mempunyai method khusus untuk menampilkan informasi gaji (tampilkanInformasiGaji), jadi lebih fleksibel.
    4. Penggunaan Formatting:
      - Di program kedua ini ada memakai String.format yaitu untuk menampilkan gaji dengan pemisah ribuan, jadi lebih mudah dibaca pengguna.
    5. Kapasitas pegawai:
      - Kapasitas pegawai di program kedua bisah ditentukan dengan input pengguna, dan juga program bisah memberikan pesan jika kapasitas penuh.

- **kesimpulan yang bisa dipetik dari perbedaan itu**

jadi dapat saya lihat dan pahami bahwa dalam program pertama, sebelum perbaikan, mempunyai kekurangan yaitu seperti ketergantungan dengan system golongan untuk penentuan gaji, dan informasi yang terbatas hanya mencakup NPP, nama, dan gaji, dan juga ada kekurangan dalam penanganan error untuk input yang tidak valid. Program ini juga tidak menyediakan analisis gaji yaitu seperti gaji terbesar, terkecil, dan rata-rata, jadi program kurang informatif dan fleksibel.

Sedangkan di program kedua, versi perbaikan, program lebih fleksibel karena pengguna bisah input langsung gaji yang diinginkan tanpa Batasan golongan, dan juga menyediakan informasi lengkap seperti jabatan dan analisis gaji. Dan struktur program lebih terorganisir dan juga memakai method khusus untuk menampilkan informasi gaji, dan program kedua juga saya memberikan penggunaan formatting jadi bisa membuat output lebih mudah dipaka.

## 2. soal nomer 2 (Soal utama praktikum)

### a. soal

Buatlah program untuk mengelola data buku dan pengarangnya. Setiap buku memiliki ciri kode buku, judul, tahun terbit, dan daftar pengarang. Pengarang adalah seseorang yang memiliki NIK, nama, alamat. Setiap buku bisa memiliki lebih dari satu pengarang. Buatlah kelas main bernama KelolaBuku yang melakukan beberapa hal sebagai berikut:

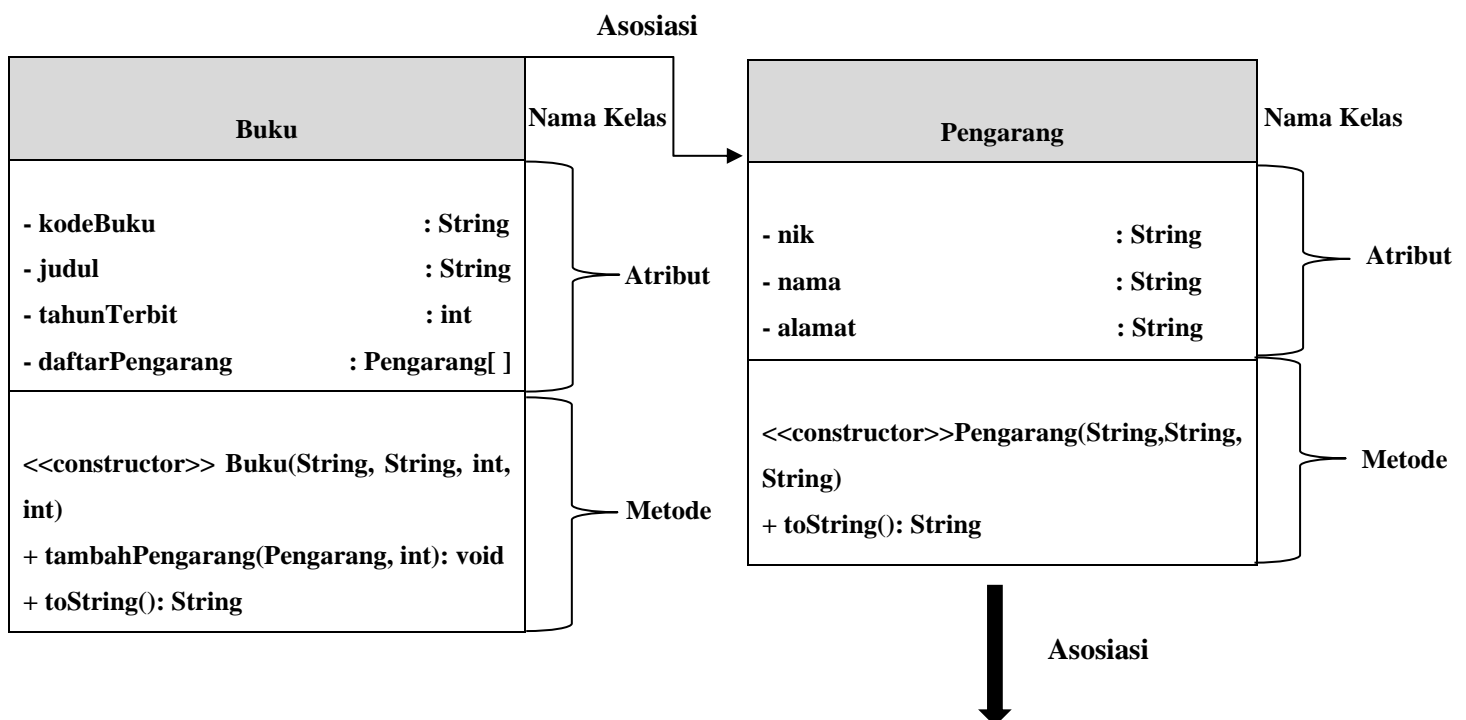
- Meminta masukan berapa jumlah buku baru yang akan disimpan.
- Untuk setiap obyek buku:
  - Membuat obyek buku dan memasukkan kode buku, judul, pengarang, tahun terbit, .
  - Meminta masukan berupa jumlah pengarang.
  - Memasukkan data-data pengarang sesuai jumlah pengarangnya.
  - Mengeset daftar pengarang pada class Buku.
- Menampilkan data buku beserta pengarangnya masing-masing
- Menampilkan jumlah buku yang diterbitkan pada tahun tertentu. Tahun yang dikehendaki dimasukkan lewat keyboard.
- Menampilkan buku terbitan paling lama.
- Menampilkan buku terbitan paling akhri.

### b. gambaran umum:

- Penjelasan singkat kasus yang dipecahkan (1 – 2 kalimat)**

Jadi program ini berfungsi untuk mengelola data buku dan pengarangnya dengan menyimpan informasi seperti kode buku, judul, tahun terbit, dan daftar pengarang. Program ini juga bisa membantu pengguna untuk menambahkan buku, mencari jumlah buku yang terbit pada tahun tertentu, dan menampilkan buku dengan tahun terbit tertua dan terbaru.

- Diagram kelas**



KelolaBuku
+ main(String[]): void

- **Penjelasan diagram kelas yang dirancang (kelas-kelas yg ada dan artinya, relasi antar kelas)**

## 1. Kelas Buku

### Fungsi:

Kelas ini berfungsi untuk menyimpan informasi buku, seperti kode buku, judul, tahun terbit, serta daftar pengarang yang menulisnya.

### Atribut:

- kodeBuku Bersifat private, bertipe String, dan berfungsi untuk menyimpan kode buku.
- judul bersifat private, bertipe String, dan berfungsi untuk menyimpan judul buku.
- tahunTerbit bersifat private, bertipe int, dan berfungsi untuk menyimpan tahun terbit.
- daftarPengarang : Pengarang[] ini berfungsi untuk menyimpan daftar pengarang dari setiap buku dalam sebuah buku.

### Metode:

- <<constructor>> Buku(String, String, int, int) ini adalah onstruktur untuk menginisialisasi objek buku dengan jumlah pengarang tertentu.
- tambahPengarang() ini berfungsi untuk menambahkan pengarang ke dalam buku.
- toString() ini untuk menampilkan informasi buku dan pengarangnya.

## 2. Kelas Pengarang

### Fungsi:

Kelas ini dipakai untuk menampilkan data pengarang, termasuk NIK, nama, dan alamatnya.

### Atribut:

- nik berfungsi untuk menyimpan nik pengarang.
- nama berfungsi untuk menyimpan nama pengarang.
- alamat berfungsi untuk menyimpan alamat pengarang.

### Metode:

- <<constructor>> Pengarang(String, String, String) Konstruktor ini untuk inisialisasi objek pengarang.
- toString() ini berfungsi untuk menampilkan informasi pengarang dalam format string.

### 3. Kelas **KelolaBuku**

#### **Fungsi:**

Kelas ini berfungsi untuk menjalankan program utama yang bisa membantu pengguna untuk menambahkan, menampilkan, dan mencari data buku serta pengarangnya.

#### **Metode:**

- + main(String[] args) bersifat public, bertipe void, dan berfungsi untuk menjalankan program, menerima input dari pengguna dan mencetak hasil proses program.

### 4. Relasi antara kelas:

- Relasi antara **Buku** dan **Pengarang**
  - Nama relasi : Asosiasi.
  - Penjelasan: Hubungan Asosiasi yang berarti setiap objek Buku bisa memiliki satu atau lebih Pengarang, yaitu Buku bisa menyimpan daftar Pengarang dalam bentuk Array ( Pengarang[ ] daftarPengarang ).
- Relasi antara **KelolaBuku** dan **Buku**
  - Nama relasi: Asosiasi.
  - Penjelasan: Relasi Asosiasi yang berarti kelas **KelolaBuku** bisa mengontrol program mengelola objek Buku dan Pengarang.



c. Kode program saat praktikum dan versi perbaikan

- **Sebelum perbaikan**
- **Kelas Buku:**

```
Source History
9      * @author Calvin Pati
10     */
11     public class Buku {
12         String kodeBuku, judul, namaPengarang;
13         int tahunTerbit;
14
15         // Konstruktor
16         public Buku(String kodeBuku, String judul, String namaPengarang, int tahunTerbit) {
17             this.kodeBuku = kodeBuku;
18             this.judul = judul;
19             this.namaPengarang = namaPengarang;
20             this.tahunTerbit = tahunTerbit;
21         }
22
23         // Getter untuk atribut buku
24         public String getKodeBuku() {
25             return kodeBuku;
26         }
27
28         public String getJudul() {
29             return judul;
30         }
31
32         public String getNamaPengarang() {
33             return namaPengarang;
34         }
35
36         public int getTahunTerbit() {
37             return tahunTerbit;
38         }
39
40         // Metode untuk menampilkan informasi buku dalam format tabel
41         public void tampilkanBuku() {
42             System.out.printf("%-10s %-20s %-15s %-10d\n", kodeBuku, judul, namaPengarang, tahunTerbit);
43         }
44     }
45
46     // Kelas Penerbit untuk menyimpan informasi tentang penerbit buku
47     class Penerbit {
48         String namaPenerbit, alamat;
49         int tahunBerdiri;
50
51         // Konstruktor
52         public Penerbit(String namaPenerbit, String alamat, int tahunBerdiri) {
53             this.namaPenerbit = namaPenerbit;
54             this.alamat = alamat;
55             this.tahunBerdiri = tahunBerdiri;
56         }
57
58         // Getter untuk atribut penerbit
59         public String getNamaPenerbit() {
60             return namaPenerbit;
61         }
62
63         public String getAlamat() {
64             return alamat;
65         }
66
67         public int getTahunBerdiri() {
68             return tahunBerdiri;
69         }
70     }
```

## ■ Kelas KelolaBuku

```
11  * @author Calvin Pati
12  */
13  public class KelolaBuku {
14  public static void main(String[] args) {
15      Scanner input = new Scanner(System.in);
16
17      // Meminta jumlah buku yang akan dimasukkan
18      System.out.print("Masukkan jumlah buku: ");
19      int jumlahBuku = input.nextInt();
20      input.nextLine(); // Membersihkan newline buffer
21
22      Buku[] daftarBuku = new Buku[jumlahBuku]; // Array untuk menyimpan daftar
23
24      // Input data buku
25      for (int i = 0; i < jumlahBuku; i++) {
26          System.out.println("\nMasukkan data buku ke-" + (i + 1));
27
28          System.out.print("Kode Buku: ");
29          String kode = input.nextLine();
30
31          System.out.print("Judul: ");
32          String judul = input.nextLine();
33
34          System.out.print("Nama Pengarang: ");
35          String pengarang = input.nextLine();
36
37          System.out.print("Tahun Terbit: ");
38          int tahun = input.nextInt();
39          input.nextLine(); // Membersihkan newline buffer
40
41          // Menyimpan objek buku ke dalam array daftarBuku
42          daftarBuku[i] = new Buku(kode, judul, pengarang, tahun);
43      }
44
45      // Menampilkan daftar buku dalam bentuk tabel
46      System.out.println("\nDaftar Buku:");
47      System.out.println("-----");
48      System.out.printf("%-10s %-20s %-15s %-10s\n", "Kode", "Judul", "Pengarang", "Tahun Terbit");
49      System.out.println("-----");
50      for (Buku buku : daftarBuku) {
51          buku.tampilkanBuku();
52      }
53      System.out.println("-----");
54
55      // Menampilkan jumlah buku yang terbit pada tahun tertentu
56      System.out.print("\nMasukkan tahun terbit yang ingin dicari: ");
57      int tahunCari = input.nextInt();
58      int jumlahTahun = 0;
59
60      System.out.println("\nBuku yang terbit pada tahun " + tahunCari + ":");
61      for (Buku buku : daftarBuku) {
62          if (buku.getTahunTerbit() == tahunCari) {
63              buku.tampilkanBuku();
64              jumlahTahun++;
65          }
66      }
67
68      if (jumlahTahun == 0) {
69          System.out.println("Tidak ada buku yang terbit pada tahun tersebut.");
70      }
71
72      // Menampilkan data buku berdasarkan pengarang tertentu
73      input.nextLine(); // Membersihkan newline buffer
74      System.out.print("\nMasukkan nama pengarang yang ingin dicari: ");
75      String pengarangCari = input.nextLine();
76      int jumlahPengarang = 0;
77
78      System.out.println("\nBuku yang ditulis oleh " + pengarangCari + ":");
79      for (Buku buku : daftarBuku) {
80          if (buku.getNamaPengarang().equalsIgnoreCase(pengarangCari)) {
81              buku.tampilkanBuku();
82              jumlahPengarang++;
83          }
84      }
85
86      if (jumlahPengarang == 0) {
87          System.out.println("Tidak ada buku yang ditulis oleh pengarang tersebut.");
88      }
89
90      input.close(); // Menutup scanner untuk menghindari kebocoran resource
91  }
92  }
```

## ■ Output

```
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\PBOSemester2
--- compiler:3.13.0:compile (default-compile) @ PBOSemester2 ---
Recompiling the module because of changed source code.
Compiling 22 source files with javac [debug release 23] to target\classes

--- exec:3.1.0:exec (default-cli) @ PBOSemester2 ---
Masukkan jumlah buku: 3

Masukkan data buku ke-1
Kode Buku: 200
Judul: belajar java
Nama Pengarang: Cia
Tahun Terbit: 2015

Masukkan data buku ke-2
Kode Buku: 203
Judul: belajar python
Nama Pengarang: celvin
Tahun Terbit: 2023

Masukkan data buku ke-3
Kode Buku: 201
Judul: belajar C++
Nama Pengarang: Delvitri
Tahun Terbit: 2020

Daftar Buku:
-----
Kode      Judul          Pengarang      Tahun Terbit
-----
200      belajar java    Cia            2015
203      belajar python  celvin         2023
201      belajar C++     Delvitri       2020
-----

Masukkan tahun terbit yang ingin dicari: 2015

Buku yang terbit pada tahun 2015:
200      belajar java    Cia            2015

Masukkan nama pengarang yang ingin dicari: celvin

Buku yang ditulis oleh celvin:
203      belajar python  celvin         2023
-----
BUILD SUCCESS
-----

Total time: 02:50 min
Finished at: 2025-03-17T13:33:32+07:00
-----
```

- Versi perbaikan
- Kelas Buku

```

8      *
9      * @author Calvin Pati
10     */
11     public class Buku { //awal kelas Buku
12         String kodeBuku, judul; // Variabel untuk simpan kode buku dan judul
13         int tahunTerbit; // Variabel untuk simpan tahun terbit buku
14         Pengarang[] daftarPengarang; // Array untuk simpan daftar pengarang buku
15
16         public Buku(String kodeBuku, String judul, int tahunTerbit, int jumlahPengarang) { // Konstruktor untuk inisialisasi bu
17             this.kodeBuku = kodeBuku; // untuk set kode buku
18             this.judul = judul; // untuk set judul
19             this.tahunTerbit = tahunTerbit; // untuk set tahunTerbit
20             this.daftarPengarang = new Pengarang[jumlahPengarang]; // untuk buat array pengarang sesuai dengan jumlah yang dibe
21         }
22
23         public void tambahPengarang(Pengarang pengarang, int index) { // Method untuk menambah pengarang ke dalam buku
24             daftarPengarang[index] = pengarang;
25         }
26
27         public String toString() { // Method untuk menampilkan data buku ke bentuk string
28             StringBuilder sb = new StringBuilder("Kode Buku: " + kodeBuku + ", Judul: " + judul + ", Tahun Terbit: " + tahunTer
29             for (Pengarang p : daftarPengarang) { // Loop untuk menampilkan setiap pengarang di daftar
30                 sb.append(" - " + p + "\n");
31             }
32             return sb.toString();
33         }
34     } //akhir kelas Buku
35
36

```

- Pengarang

```

8      *
9      * @author Calvin Pati
10     */
11     public class Pengarang { //awal kelas pengarang
12         String nik, nama, alamat; // Variabel untuk simpan data pengarang
13
14         public Pengarang(String nik, String nama, String alamat) { // Konstruktor untuk inisialisasi pengarang
15             this.nik = nik; // untuk set nik
16             this.nama = nama; // untuk set nama
17             this.alamat = alamat; // untuk set alamat
18         }
19
20         public String toString() { // Method ini untuk menampilkan data pengarang dalam bentuk string
21             return "NIK: " + nik + ", Nama: " + nama + ", Alamat: " + alamat;
22         }
23     } //akhir kelas pengarang
24

```

## ■ Kelas KelolaBuku

```
11  * @author Calvin Pati
12  */
13  public class KelolaBuku { //awal kelas kelolaBuku
14      public static void main(String[] args) {
15          Scanner scanner = new Scanner(System.in); // untuk buat objek Scanner untuk input pengguna
16
17          System.out.print("Masukkan jumlah buku yang mau disimpan: ");
18          int jumlahBuku = scanner.nextInt(); // baca jumlah buku dari input pengguna
19          scanner.nextLine(); // newline supaya input berikutnya tidak terlewat
20
21          Buku[] daftarBuku = new Buku[jumlahBuku]; //untuk buat array untuk simpan daftar buku
22
23          for (int i = 0; i < jumlahBuku; i++) { // Loop untuk memasukkan data setiap buku
24              System.out.println("\nMasukkan data buku ke-" + (i + 1));
25              System.out.print("Kode Buku: ");
26              String kodeBuku = scanner.nextLine(); // untuk baca kode buku
27              System.out.print("Judul: ");
28              String judul = scanner.nextLine(); // untuk baca judul buku
29              System.out.print("Tahun Terbit: ");
30              int tahunTerbit = scanner.nextInt(); // untuk baca tahun terbit buku
31              System.out.print("Jumlah Pengarang: ");
32              int jumlahPengarang = scanner.nextInt(); // untuk baca jumlah pengarang buku
33              scanner.nextLine(); // newline supaya input berikutnya tidak terlewat
34
35              Buku buku = new Buku(kodeBuku, judul, tahunTerbit, jumlahPengarang); // untuk buat objek buku baru
36
37              for (int j = 0; j < jumlahPengarang; j++) { // Loop untuk memasukkan data pengarang
38                  System.out.println("Masukkan data pengarang ke-" + (j + 1));
39                  System.out.print("NIK: ");
40                  String nik = scanner.nextLine(); // untuk baca NIK pengarang
41                  System.out.print("Nama: ");
42                  String nama = scanner.nextLine(); // untuk baca nama pengarang
43                  System.out.print("Alamat: ");
44                  String alamat = scanner.nextLine(); // untuk baca alamat pengarang
45                  buku.tambahPengarang(new Pengarang(nik, nama, alamat), j); // untuk menambah pengarang ke dalam buku
46              }
47              daftarBuku[i] = buku; // untuk simpan buku ke dalam daftar buku
48          }
49
50          System.out.println("\nDaftar Buku:");
51          for (Buku b : daftarBuku) { // Loop untuk menampilkan semua data buku
52              System.out.println(b);
53          }
54
55          System.out.print("\nMasukkan tahun untuk mencari jumlah buku yang terbit: ");
56          int tahunCari = scanner.nextInt(); // untuk baca tahun yang dicari dari input pengguna
57          int jumlah = 0; // Variabel untuk hitung jumlah buku yang terbit di tahun tertentu
58          for (Buku b : daftarBuku) { // Loop untuk cek setiap buku
59              if (b.tahunTerbit == tahunCari) jumlah++; // Jika tahun terbit sesuai, maka tambahkan jumlah
60          }
61          System.out.println("Jumlah buku yang terbit pada tahun " + tahunCari + ": " + jumlah);
62
63          Buku bukuTertua = daftarBuku[0], bukuTerbaru = daftarBuku[0]; // untuk inialisasi buku tertua dan terbaru
64          for (Buku b : daftarBuku) { // Loop untuk cari buku tertua dan terbaru
65              if (b.tahunTerbit < bukuTertua.tahunTerbit) bukuTertua = b; // Jika lebih tua, maka perbarui buku tertua
66              if (b.tahunTerbit > bukuTerbaru.tahunTerbit) bukuTerbaru = b; // Jika lebih baru, maka perbarui buku terbaru
67          }
68
69          System.out.println("\nBuku terbitan paling lama:\n" + bukuTertua); // untuk cetak buku tertua
70          System.out.println("Buku terbitan paling akhir:\n" + bukuTerbaru); // untuk cetak buku terbaru
71
72          scanner.close(); //tutup scanner
73      }
74  } //akhir kelas KelolaBuku
```

## ■ Output

```
--- exec:3.1.0:exec (default-cli) @ perbaikanPBOModul5 ---
Masukkan jumlah buku yang mau disimpan: 3

Masukkan data buku ke-1
Kode Buku: 207
Judul: Belajar Java
Tahun Terbit: 2015
Jumlah Pengarang: 2
Masukkan data pengarang ke-1
NIK: 207789
Nama: Max
Alamat: Yogyakarta
Masukkan data pengarang ke-2
NIK: 305673
Nama: Cantika
Alamat: Jakarta

Masukkan data buku ke-2
Kode Buku: 708
Judul: Belajar Python
Tahun Terbit: 2025

Jumlah Pengarang: 1
Masukkan data pengarang ke-1
NIK: 207758
Nama: Jonas
Alamat: Jakarta Selatan

Masukkan data buku ke-3
Kode Buku: 232
Judul: Belajar MQL
Tahun Terbit: 2021
Jumlah Pengarang: 3
Masukkan data pengarang ke-1
NIK: 403214
Alamat: Bali
Masukkan data pengarang ke-2
NIK: 567789
Nama: Samuel
Alamat: Kalimantan
Masukkan data pengarang ke-3
NIK: 567780
Nama: Safira
Alamat: Bengkulu

Daftar Buku:
Kode Buku: 207, Judul: Belajar Java, Tahun Terbit: 2015
Pengarang:
  - NIK: 207789, Nama: Max, Alamat: Yogyakarta
  - NIK: 305673, Nama: Cantika, Alamat: Jakarta

Kode Buku: 708, Judul: Belajar Python, Tahun Terbit: 2025
Pengarang:
  - NIK: 207758, Nama: Jonas, Alamat: Jakarta Selatan

Kode Buku: 232, Judul: Belajar MQL, Tahun Terbit: 2021
Pengarang:
  - NIK: 403214, Nama: Dewi, Alamat: Bali
  - NIK: 567789, Nama: Samuel, Alamat: Kalimantan
  - NIK: 567780, Nama: Safira, Alamat: Bengkulu

Masukkan tahun untuk mencari jumlah buku yang terbit: 2021
Jumlah buku yang terbit pada tahun 2021: 1

Buku terbitan paling lama:
Kode Buku: 207, Judul: Belajar Java, Tahun Terbit: 2015
Pengarang:
  - NIK: 207789, Nama: Max, Alamat: Yogyakarta
  - NIK: 305673, Nama: Cantika, Alamat: Jakarta

Buku terbitan paling akhir:
Kode Buku: 708, Judul: Belajar Python, Tahun Terbit: 2025
Pengarang:
  - NIK: 207758, Nama: Jonas, Alamat: Jakarta Selatan

-----
BUILD SUCCESS
-----

Total time: 03:54 min
Finished at: 2025-03-24T19:55:54+07:00
-----
```

d. Analisis perbandingan program saat praktikum dan perbaikan program:

▪ **Perbedaan kedua program**

- Struktur data pengarang:
  - Di program versi sebelum perbaikan data pengarang hanya berupa satu atribut namaPengarang di dalam kelas Buku.
  - Di program versi sesudah perbaikan data pengarang sudah dipisah menjadi kelas Pengarang, yang terdapat atribut nik, nama, dan alamat.
- Hubungan antara buku dan pengarang:
  - Di program pertama Setiap buku hanya punya satu pengarang.
  - Di program kedua setiap buku sudah punya lebih dari satu pengarang dengan memakai array daftarPengarang.
- Tambahan fitur pencarian:
  - Di program pertama hanya bisa mencari buku sesuai dengan tahun terbit dan nama pengarang.
  - Di program kedua bisa mencari jumlah buku sesuai dengan tahun tertentu dan menentukan buku tertua serta terbaru.
- Output:
  - Di program pertama output daftar buku dalam bentuk tabel tanpa format yang rapi
  - Di program kedua sudah memakai StringBuilder untuk menampilkan data buku dengan lebih terstruktur..

▪ **Kekurangan program pertama:**

1. Setiap buku hanya punya satu pengarang, jadi tidak fleksibel untuk buku dengan banyak pengarang.
2. Tidak ada pemisahan data pengarang sebagai entitas yang berbeda, jadi kurang terstruktur.
3. Output daftar buku tidak terformat dengan baik atau berantakan.
4. Tidak punya fitur pencarian buku tertua dan terbaru.

▪ **Kelebihan program versi perbaikan**

1. Struktur data lebih baik karena ada kelas Pengarang, jadi setiap buku memiliki lebih dari satu pengarang.
2. Data pengarang lebih lengkap karena memakai NIK dan alamat.
3. Output lebih terstruktur karena memakai toString().
4. Memakai fitur pencarian jumlah buku sesuai dengan tahun tertentu jadi lebih efisien.
5. Bisa menampilkan buku tertua dan terbaru.

- **Kesimpulan dari perbedaan program:**

Program kedua versi sesudah perbaikan mempunyai struktur yang lebih baik dan lebih fleksibel dalam menangani banyak pengarang dalam satu buku. Dan juga , ada fitur tambahan seperti pencarian jumlah buku dan buku tertua/terbaru jadi program lebih fungsional.



C. Referensi yang diacu saat membuat laporan, termasuk gen AI jika Anda menggunakannya.

- MODUL V PRAKTIKUM PBO ARRAY OF OBJECTS SEBAGAI PARAMETER DAN ARGUMEN DALANMETODE
- Poe AI
- <https://www.petanikode.com/java-oop/>

D. Refleksi proses belajar

- Apakah Anda semakin paham ATAU semakin tidak paham? : Ya....
- Mengapa? : karena saat saya dapat membandingkan kekurangan kelebihan program ang saya buat saat praktikum dengan program versi perbaikan sehingga saya bisah evalusasi dan upgrade pemahaman saya.
- Apa rencana Anda selanjutnya terkait topik yang sudah dibahas maupunnyangkan dibahas minggu berikutnya dalam matakuliah PBO? : rencana saya saya mau lebih mengoptimalkan persiapan saya sebelum masuk perkuliah agar dapat memberikan hasil maksimal terkait pree test dan saat mengerjakan program saat praktikum.