

# **LAPORAN 6**

## **Pemrograman Berorientasi Objek**

**“PENGEMBANGAN  
METODE DALAM KELAS  
YANG MEMUAT ARRAY  
OF OBJECTS”**

**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  
2025**

# latihan nomer 1

## program Versi Array Biasa

### A. DIAGRAM CLASS

Penggambaran diagram kelas

Pegawai
- NIP: String - nama: String - gajiPokok: double
<<constructor>>(nama: String, NIP: String, gajiPokok: double) + getNIP(): String + getNama(): String + getGajiPokok(): double + setNIP(NIP: String): void + setNama(nama: String): void + setGajiPokok(gajiPokok: double): void + tampilkanInfo(): void

Perusahaan
- namaPerusahaan: String - alamat: String - pemilik: String - npwp: String - daftarPegawai: Pegawai[] - jumlahPegawai: int
<<constructor>>Perusahaan(namaPerusahaan: String, alamat: String, pemilik: String, npwp: String, kapasitasPegawai: int) + tambahPegawai(pegawaiBaru: Pegawai): void + tampilkanDaftarPegawai(): void + getPegawaiGajiTertinggi(): Pegawai + getPegawaiGajiTerendah(): Pegawai + getRataRataGajiPokok(): double + getKapasitasPegawai(): int + getJumlahPegawai(): int

KelolaPerusahaan
+ main(args: String[]): void

## B. LISTING PROGRAM

```
11 public class Pegawai { //awal kelas Pegawai
12 private String NIP; // Menyimpan Nomor Induk Pegawai (identifikasi unik)
13 private String nama; // Menyimpan nama lengkap pegawai
14 private double gajiPokok; // Menyimpan besaran gaji dasar pegawai
15
16 // Fungsi pembangun untuk membuat objek pegawai baru
17 public Pegawai(String nama, String NIP, double gajiPokok) {
18     this.nama = nama; // Mengisi nama pegawai
19     this.NIP = NIP; // Mengisi nomor induk pegawai
20     this.gajiPokok = gajiPokok; // Mengisi gaji pokok pegawai
21 }
22
23 // Fungsi untuk mengambil data pegawai
24 public String getNIP() {
25     return NIP; // Mengembalikan nomor induk pegawai
26 }
27
28 public String getName() {
29     return nama; // Mengembalikan nama pegawai
30 }
31
32 public double getGajiPokok() {
33     return gajiPokok; // Mengembalikan gaji pokok pegawai
34 }
35
36 // Fungsi untuk mengubah data pegawai
37 public void setNIP(String NIP) {
38     this.NIP = NIP; // Memperbarui nomor induk pegawai
39 }
40
41 public void setName(String nama) {
42     this.nama = nama; // Memperbarui nama pegawai
43 }
44
45 public void setGajiPokok(double gajiPokok) {
46     this.gajiPokok = gajiPokok; // Memperbarui gaji pokok pegawai
47 }
48
49 // Fungsi untuk menampilkan informasi pegawai ke layar
50 public void tampilkanInfo() {
51     System.out.println("NIP: " + NIP); // Menampilkan nomor induk
52     System.out.println("Nama: " + nama); // Menampilkan nama
53     System.out.println("Gaji Pokok: Rp " + gajiPokok); // Menampilkan gaji
54 }
55 }
56
```

```
11 public class Perusahaan { //awal kelas Perusahaan
12 // Data dasar perusahaan
13 private String namaPerusahaan; // Nama resmi perusahaan
14 private String alamat; // Alamat lengkap perusahaan
15 private String pemilik; // Nama pemilik/peegang saham utama
16 private String npwp; // Nomor NPWP perusahaan untuk pajak
17
18 // Sistem penyimpanan pegawai
19 private Pegawai[] daftarPegawai; // Tempat menyimpan data pegawai (array)
20 private int jumlahPegawai; // Jumlah pegawai yang aktif saat ini
21
22 // Membangun perusahaan baru
23 public Perusahaan(String namaPerusahaan, String alamat, String pemilik, String npwp, int kapasitasPegawai) {
24     this.namaPerusahaan = namaPerusahaan; // Set identitas perusahaan
25     this.alamat = alamat; // Set lokasi kantor
26     this.pemilik = pemilik; // Catat pemilik
27     this.npwp = npwp; // Simpan data pajak
28     this.daftarPegawai = new Pegawai[kapasitasPegawai]; // Siapkan tempat pegawai
29     this.jumlahPegawai = 0; // Belum ada pegawai awal
30 }
31
32 // Sistem rekrutmen pegawai
33 public void tambahPegawai(Pegawai pegawaiBaru) {
34     // Cek apakah masih ada lowongan
35     if (jumlahPegawai >= daftarPegawai.length) {
36         System.out.println("Kapasitas pegawai sudah penuh! Tidak bisa menambah pegawai baru.");
37         return;
38     }
39     // Tambahkan pegawai baru ke sistem
40     daftarPegawai[jumlahPegawai] = pegawaiBaru;
41     jumlahPegawai++;
42     System.out.println("Pegawai " + pegawaiBaru.getName() + " berhasil ditambahkan.");
43 }
44
45 // Menampilkan daftar karyawan
46 public void tampilkanDaftarPegawai() {
47     // Cek apakah ada pegawai
48     if (jumlahPegawai == 0) {
49         System.out.println("Belum ada pegawai di perusahaan ini.");
50         return;
51     }
52     // Tampilkan header
53     System.out.println("==== Daftar Pegawai " + namaPerusahaan + " =====");
54     // Loop untuk menampilkan semua pegawai
55     for (int i = 0; i < jumlahPegawai; i++) {
56         daftarPegawai[i].tampilkanInfo(); // Tampilkan info per pegawai
57         System.out.println("-----"); // Garis pemisah
58     }
59 }
60
61 // Mencari pegawai dengan gaji tertinggi
62 public Pegawai getPegawaiGajiTertinggi() {
63     // Jika tidak ada pegawai, kembalikan null
64     if (jumlahPegawai == 0) return null;
65
66     // Asumsikan pegawai pertama yang tertinggi
67     Pegawai tertinggi = daftarPegawai[0];
68
69     // Bandingkan dengan pegawai lain
70     for (int i = 1; i < jumlahPegawai; i++) {
71         if (daftarPegawai[i].getGajiPokok() > tertinggi.getGajiPokok()) {
72             tertinggi = daftarPegawai[i]; // Update jika ditemukan yang lebih tinggi
73         }
74     }
75     return tertinggi;
76 }
```

Activate Windows  
Go to Settings to activate

```

77
78 // Mencari pegawai dengan gaji terendah (mirip dengan tertinggi)
79 public Pegawai getPegawaiGajiTerendah() {
80     if (jumlahPegawai == 0) return null;
81     Pegawai terendah = daftarPegawai[0];
82     for (int i = 1; i < jumlahPegawai; i++) {
83         if (daftarPegawai[i].getGajiPokok() < terendah.getGajiPokok()) {
84             terendah = daftarPegawai[i]; // Update jika ditemukan yang lebih rendah
85         }
86     }
87     return terendah;
88 }
89
90 // Menghitung rata-rata gaji semua pegawai
91 public double getRataRataGajiPokok() {
92     // Jika tidak ada pegawai, rata-rata 0
93     if (jumlahPegawai == 0) return 0;
94
95     double total = 0;
96     // Jumlahkan semua gaji
97     for (int i = 0; i < jumlahPegawai; i++) {
98         total += daftarPegawai[i].getGajiPokok();
99     }
100     // Hitung rata-rata
101     return total / jumlahPegawai;
102 }
103
104 // Mengecek kapasitas maksimum pegawai
105 public int getKapasitasPegawai() {
106     return daftarPegawai.length; // Panjang array = kapasitas
107 }
108
109 // Mengecek jumlah pegawai saat ini
110 public int getJumlahPegawai() {
111     return jumlahPegawai; // Jumlah pegawai yang terdaftar
112 }
113 }
114

```

```

13 public class KelolaPerusahaan { //awal kelas KelolaPerusahaan
14     public static void main(String[] args) {
15         // Membuka scanner untuk menerima input dari user
16         Scanner scanner = new Scanner(System.in);
17
18         // Bagian pendirian perusahaan
19         System.out.println("=== Pendirian Perusahaan ===");
20         // Meminta data dasar perusahaan
21         System.out.print("Masukkan Nama Perusahaan : ");
22         String namaPerusahaan = scanner.nextLine(); // Membaca nama perusahaan
23
24         System.out.print("Masukkan Alamat : ");
25         String alamat = scanner.nextLine(); // Membaca alamat perusahaan
26
27         System.out.print("Masukkan Nama Pemilik : ");
28         String pemilik = scanner.nextLine(); // Membaca nama pemilik
29
30         System.out.print("Masukkan NPWP : ");
31         String npwp = scanner.nextLine(); // Membaca nomor NPWP
32
33         // Menentukan ukuran perusahaan
34         System.out.print("Masukkan kapasitas pegawai (maksimal): ");
35         int kapasitasPegawai = scanner.nextInt(); // Membaca kapasitas pegawai
36         scanner.nextLine(); // Membersihkan newline setelah input angka
37
38         // Membuat objek perusahaan baru dengan data yang dimasukkan
39         Perusahaan perusahaan = new Perusahaan(namaPerusahaan, alamat, pemilik, npwp, kapasitasPegawai);
40
41         // Proses rekrutmen pegawai awal
42         System.out.println("\n=== Rekrutmen Pegawai Awal ===");
43         System.out.print("Masukkan jumlah pegawai awal: ");
44         int jumlahAwal = scanner.nextInt(); // Membaca jumlah pegawai awal
45         scanner.nextLine(); // Membersihkan newline
46
47         // Loop untuk memasukkan data setiap pegawai
48         for (int i = 0; i < jumlahAwal; i++) {
49             System.out.println("\nData pegawai ke- " + (i + 1));
50
51             System.out.print("Nama Pegawai : ");
52             String namaPegawai = scanner.nextLine(); // Nama pegawai
53
54             System.out.print("NIP Pegawai : ");
55             String nip = scanner.nextLine(); // Nomor induk pegawai
56
57             System.out.print("Gaji Pokok : ");
58             double gajiPokok = scanner.nextDouble(); // Gaji pokok
59             scanner.nextLine(); // Membersihkan newline
60
61             // Membuat objek pegawai baru dan menambahkannya ke perusahaan
62             Pegawai pegawaiBaru = new Pegawai(namaPegawai, nip, gajiPokok);
63             perusahaan.tambahPegawai(pegawaiBaru);
64         }
65
66         // Menu interaktif utama
67         int pilihan;
68         do {
69             // Menampilkan pilihan menu
70             System.out.println("\n=== Menu Utama ===");
71             System.out.println("1. Tampilkan Daftar Pegawai");
72             System.out.println("2. Tampilkan Pegawai dengan Gaji Tertinggi");
73             System.out.println("3. Tampilkan Pegawai dengan Gaji Terendah");
74             System.out.println("4. Tampilkan Rata-rata Gaji");
75             System.out.println("5. Tambah Pegawai Baru");
76             System.out.println("0. Keluar");
77             System.out.print("Pilihan Anda: ");
78
79             // Membaca pilihan user
80             pilihan = scanner.nextInt();
81             scanner.nextLine(); // Membersihkan newline
82
83             // Pemrosesan pilihan menu
84             switch (pilihan) {
85                 case 1: // Menampilkan daftar pegawai
86                     perusahaan.tampilkanDaftarPegawai();
87                     break;
88
89                 case 2: // Mencari pegawai bergaji tertinggi
90                     Pegawai tertinggi = perusahaan.getPegawaiGajiTertinggi();
91                     System.out.println("\n=== Pegawai dengan Gaji Tertinggi ===");
92                     if (tertinggi != null) {
93                         tertinggi.tampilkanInfo(); // Menampilkan jika ditemukan
94                     } else {
95                         System.out.println("Belum ada pegawai."); // Pesan jika kosong
96                     }
97                     break;
98
99                 case 3: // Mencari pegawai bergaji terendah
100                     Pegawai terendah = perusahaan.getPegawaiGajiTerendah();
101                     System.out.println("\n=== Pegawai dengan Gaji Terendah ===");
102                     if (terendah != null) {
103                         terendah.tampilkanInfo(); // Menampilkan jika ditemukan
104                     } else {
105                         System.out.println("Belum ada pegawai."); // Pesan jika kosong
106                     }
107                     break;
108
109                 case 4: // Menghitung rata-rata gaji
110                     double rataRata = perusahaan.getRataRataGajiPokok();
111                     // Menampilkan dengan format mata uang

```

Activate  
Go to Setting

```

112 System.out.println("\nRata-rata Gaji Pegawai: Rp " + String.format("%.2f", rataRata));
113 break;
114
115 case 5: // Menambah pegawai baru
116 // Cek ketersediaan slot pegawai
117 if (perusahaan.getJumlahPegawai() >= perusahaan.getKapasitasPegawai()) {
118 System.out.println("\nKapasitas pegawai sudah penuh!");
119 break;
120 }
121
122 // Proses input data pegawai baru
123 System.out.println("\n== Tambah Pegawai Baru ==");
124 System.out.print("Nama Pegawai : ");
125 String namaPegawai = scanner.nextLine();
126
127 System.out.print("NIP Pegawai : ");
128 String nip = scanner.nextLine();
129
130 System.out.print("Gaji Pokok : ");
131 double gajiPokok = scanner.nextDouble();
132 scanner.nextLine(); // Membersihkan newline
133
134 // Membuat dan menambahkan pegawai baru
135 Pegawai pegawaiBaru = new Pegawai(namaPegawai, nip, gajiPokok);
136 perusahaan.tambahPegawai(pegawaiBaru);
137 break;
138
139 case 0: // Keluar dari program
140 System.out.println("Terima kasih telah menggunakan sistem ini.");
141 break;
142
143 default: // Pilihan tidak valid
144 System.out.println("Pilihan tidak valid!");
145 }
146 } while (pilihan != 0); // Loop sampai user memilih keluar
147
148 // Menutup scanner untuk menghindari memory leak
149 scanner.close();
150 }
151 }
152

```

Activate Windows  
Go to Settings to activate

## C. OUTPUT Unit Test nya

- Setelah program nya saya perbaiki dan saya test memakai package test unit maka hasilnya untuk program ini versi array biasa seperti berikut

```
--- resources:3.3.1:resources (default-resources) @ PBOModul6Fix ---
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\PBOModul6Fix\src\main\resources

--- compiler:3.13.0:compile (default-compile) @ PBOModul6Fix ---
Recompiling the module because of changed source code.
Compiling 3 source files with javac [debug release 23] to target\classes

--- exec:3.1.0:exec (default-cli) @ PBOModul6Fix ---
=== Pendirian Perusahaan ===
Masukkan Nama Perusahaan : PT.Forex Pratama
Masukkan Alamat : Waikabubak, Sumba Barat, NTT
Masukkan Nama Pemilik : Bapak Celvin Umbu Pati
Masukkan NPWP : 2078
Masukkan kapasitas pegawai (maksimal): 200000

=== Rekrutmen Pegawai Awal ===
Masukkan jumlah pegawai awal: 3

Data pegawai ke-1
Nama Pegawai : Celin
NIP Pegawai : 207788
Gaji Pokok : 200000000
Pegawai Celin berhasil ditambahkan.

Data pegawai ke-2
Nama Pegawai : Max
NIP Pegawai : 207889
Gaji Pokok : 300000000
Pegawai Max berhasil ditambahkan.

Data pegawai ke-3
Nama Pegawai : Jonathan
NIP Pegawai : 207899
Gaji Pokok : 400000000
Pegawai Jonathan berhasil ditambahkan.

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
0. Keluar
Pilihan Anda: 1
===== Daftar Pegawai PT.Forex Pratama =====
NIP: 207788
Nama: Celin
Gaji Pokok: Rp 2.0E8
-----
NIP: 207889
Nama: Max
Gaji Pokok: Rp 3.0E8
-----
NIP: 207899
Nama: Jonathan
Gaji Pokok: Rp 4.0E8
-----

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
0. Keluar
Pilihan Anda: 4
Rata-rata Gaji Pegawai: Rp 300.000.000,00

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
0. Keluar
Pilihan Anda: 2

=== Pegawai dengan Gaji Tertinggi ===
NIP: 207899
Nama: Jonathan
Gaji Pokok: Rp 4.0E8

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
0. Keluar
Pilihan Anda: 3

=== Pegawai dengan Gaji Terendah ===
NIP: 207788
Nama: Celin
Gaji Pokok: Rp 2.0E8

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
0. Keluar
Pilihan Anda: 0
Terima kasih telah menggunakan sistem ini.
-----
BUILD SUCCESS
-----
Total time: 04:40 min
Finished at: 2025-04-06T21:42:46+07:00
-----
```

## D. ANALISA

### Pegawai

No	Syntax	penjelasan
1	<pre>private String NIP; private String nama; private double gajiPokok;</pre>	// ini adalah atribut privat yang menyimpan data pegawai: NIP (nomor induk), nama, dan gaji pokok. Digunakan untuk enkapsulasi.
2	<pre>public Pegawai(String nama, String NIP, double gajiPokok) { ... }</pre>	//Konstruktor untuk membuat objek Pegawai baru. Parameter digunakan untuk mengisi nilai awal atribut.
3	<pre>public String getNIP() { return NIP; }</pre>	// Method getter untuk mengambil nilai NIP pegawai.
4	<pre>public void tabung(int jumlah) { saldo = saldo + jumlah; }</pre>	// metode untuk menambah saldo rekening sebesar jumlah yang diinput
5	<pre>public double getGajiPokok() { return gajiPokok; }</pre>	//Method getter untuk mengambil nilai gaji pokok pegawai.
6	<pre>public void setNIP(String NIP) { this.NIP = NIP; }</pre>	//Method setter untuk mengubah nilai NIP pegawai.
7	<pre>public void setName(String nama) { this.nama = nama; }</pre>	//Method setter untuk mengubah nama pegawai.
8	<pre>public void setGajiPokok(double gajiPokok) { this.gajiPokok = gajiPokok; }</pre>	//Method setter untuk mengubah nilai gaji pokok pegawai.
9	<pre>public void tampilkanInfo() { System.out.println(...); }</pre>	// Method untuk menampilkan seluruh informasi pegawai ke layar: NIP, nama, dan gaji pokok.

## Perusahaan

No	Syntax	penjelasan
1	private String namaPerusahaan; private String alamat; private String pemilik; private String npwp;	// Atribut privat untuk menyimpan data identitas perusahaan seperti nama, alamat, pemilik, dan NPWP.
2	private Pegawai[] daftarPegawai; private int jumlahPegawai;	// Atribut untuk menyimpan array pegawai dan jumlah pegawai aktif saat ini.
3	public Perusahaan(String namaPerusahaan, String alamat, String pemilik, String npwp, int kapasitasPegawai) { ... }	// Konstruktor untuk membuat objek perusahaan baru dengan data awal dan kapasitas jumlah pegawai.
4	public void tambahPegawai(Pegawai pegawaiBaru) { ... }	// Method untuk menambahkan pegawai baru ke dalam daftar pegawai perusahaan jika kapasitas belum penuh.
5	public void tampilkanDaftarPegawai() { ... }	// Method untuk menampilkan seluruh daftar pegawai di perusahaan.
6	public Pegawai getPegawaiGajiTertinggi() { ... }	// Method untuk mencari dan mengembalikan objek pegawai dengan gaji pokok tertinggi.
7	public Pegawai getPegawaiGajiTerendah() { ... }	// Method untuk mencari dan mengembalikan objek pegawai dengan gaji pokok terendah.
8	public double getRataRataGajiPokok() { ... }	// Method untuk menghitung dan mengembalikan rata-rata gaji pokok semua pegawai.
9	public int getKapasitasPegawai() { return daftarPegawai.length; }	// Method getter untuk mengambil kapasitas maksimum jumlah pegawai.
10	public int getJumlahPegawai() { return jumlahPegawai; }	// Method getter untuk mengambil jumlah pegawai yang aktif saat ini.

## KelolaPerusahaan

No	Syntax	penjelasan
1	<code>public static void main(String[] args) {</code>	// Fungsi utama Java, titik masuk program. Semua eksekusi dimulai dari sini.
2	<code>Scanner scanner = new Scanner(System.in);</code>	// Membuat objek Scanner untuk membaca input dari keyboard.
3	<code>System.out.println("=== Pendirian Perusahaan ==="); dsb.</code>	// Menampilkan teks pendirian perusahaan dan meminta data perusahaan (nama, alamat, pemilik, NPWP).
4	<code>int kapasitasPegawai = scanner.nextInt();</code>	// Membaca jumlah maksimum pegawai yang bisa ditampung perusahaan.
5	<code>Perusahaan perusahaan = new Perusahaan(...);</code>	// Membuat objek Perusahaan dengan data yang sudah diinputkan
6	<code>System.out.print("Masukkan jumlah pegawai awal: "); dsb.</code>	// Meminta jumlah pegawai awal dan memulai pengisian data pegawai satu per satu.
7	<code>for (int i = 0; i &lt; jumlahAwal; i++) { ... }</code>	// Loop input data setiap pegawai awal dan menambahkan ke dalam objek Perusahaan.
8	<code>int pilihan; do { ... } while (pilihan != 0);</code>	// Struktur looping menu utama yang terus berjalan sampai user memilih keluar (pilihan == 0).
9	<code>System.out.println("=== Menu Utama ==="); dsb.</code>	// Menampilkan daftar menu: lihat pegawai, cari gaji tertinggi/terendah, tambah pegawai, dll.
10	<code>switch (pilihan) { case 1: ... }</code>	// Menangani logika dari setiap pilihan menu menggunakan struktur switch-case.
11	<code>perusahaan.tampilkanDaftarPegawai();</code>	// Menampilkan semua pegawai yang sudah tersimpan di objek perusahaan.
12	<code>Pegawai tertinggi = perusahaan.getPegawaiGajiTertinggi();</code>	// Mengambil data pegawai dengan gaji tertinggi, lalu menampilkannya.
13	<code>Pegawai terendah = perusahaan.getPegawaiGajiTerendah();</code>	// Mengambil data pegawai dengan gaji terendah, lalu menampilkannya.

14	<code>double rataRata = perusahaan.getRataRataGajiPokok();</code>	//Menghitung rata-rata gaji seluruh pegawai dan menampilkannya.
15	<code>if (perusahaan.getJumlahPegawai() &gt;= perusahaan.getKapasitasPegawai())</code>	//Mengecek apakah kapasitas pegawai penuh sebelum menambah pegawai baru.
16	<code>Pegawai pegawaiBaru = new Pegawai(...); perusahaan.tambahPegawai(pegawaiBaru);</code>	//Membuat dan menambahkan objek Pegawai baru ke dalam daftar pegawai perusahaan.
17	<code>System.out.println("Terima kasih...") &amp; scanner.close();</code>	//Menampilkan pesan penutup dan menutup Scanner agar tidak terjadi kebocoran memori (memory leak).

## latihan kasus 1

### program versi array list

## A. DIAGRAM CLASS

Pegawai
- NIP: String - nama: String - gajiPokok: double
<<constructor>>Pegawai(NIP: String, nama: String, gajiPokok: double) + getNIP(): String + getNama(): String + getGajiPokok(): double + setNIP(NIP: String): void + setNama(nama: String): void + setGajiPokok(gajiPokok: double): void + tampilkanInfo(): void + equals(Object o): boolean + hashCode(): int

Perusahaan
- namaPerusahaan: String - alamat: String - pemilik: String - npwp: String - daftarPegawai: List<Pegawai>
<<constructor>>Perusahaan(namaPerusahaan: String, alamat: String, pemilik: String, npwp: String) + tambahPegawai(pegawaiBaru: Pegawai): void + tampilkanDaftarPegawai(): void + getPegawaiGajiTertinggi(): Pegawai + getPegawaiGajiTerendah(): Pegawai + getRataRataGaji(): double + hapusPegawai(NIP: String): boolean + getJumlahPegawai(): int + getNamaPerusahaan(): String + getAlamat(): String + getPemilik(): String + getNpwp(): String

KelolaPerusahaan
+ main(String[ ]): void

## B. LISTING PROGRAM

```

13 public class Pegawai { //awal kelas Pegawai
14     private String NIP; // Menyimpan nomor induk pegawai (unik)
15     private String nama; // Menyimpan nama lengkap pegawai
16     private double gajiPokok; // Menyimpan besaran gaji pokok
17
18     // Constructor untuk membuat objek Pegawai baru
19     public Pegawai(String NIP, String nama, double gajiPokok) {
20         this.NIP = NIP; // Mengisi NIP
21         this.nama = nama; // Mengisi nama
22         this.gajiPokok = gajiPokok; // Mengisi gaji pokok
23     }
24
25     // Method getter untuk mendapatkan NIP
26     public String getNIP() {
27         return NIP; // Mengembalikan nilai NIP
28     }
29
30     // Method getter untuk mendapatkan nama
31     public String getNama() {
32         return nama; // Mengembalikan nilai nama
33     }
34
35     // Method getter untuk mendapatkan gaji pokok
36     public double getGajiPokok() {
37         return gajiPokok; // Mengembalikan nilai gaji pokok
38     }
39
40     // Method setter untuk mengubah NIP
41     public void setNIP(String NIP) {
42         this.NIP = NIP; // Memperbarui nilai NIP
43     }
44
45     // Method setter untuk mengubah nama
46     public void setNama(String nama) {
47         this.nama = nama; // Memperbarui nilai nama
48     }
49
50     // Method setter untuk mengubah gaji pokok
51     public void setGajiPokok(double gajiPokok) {
52         this.gajiPokok = gajiPokok; // Memperbarui nilai gaji pokok
53     }
54
55     // Method untuk menampilkan informasi pegawai
56     public void tampilkanInfo() {
57         System.out.println("NIP: " + NIP);
58         System.out.println("Nama: " + nama);
59         // Menampilkan gaji dengan format mata uang
60         System.out.println("Gaji Pokok: Rp " + String.format("%.2f", gajiPokok));
61         System.out.println("-----");
62     }
63
64     // Override method equals untuk membandingkan kesamaan objek Pegawai
65     @Override
66     public boolean equals(Object o) {
67         if (this == o) return true; // Jika objek sama
68         if (o == null || !getClass().isAssignableFrom(getClass())) return false; // Jika objek null atau kelas berbeda
69         Pegawai pegawai = (Pegawai) o; // Casting objek
70         return Objects.equals(NIP, pegawai.NIP); // Membandingkan berdasarkan NIP
71     }
72
73     // Override method hashCode untuk konsistensi dengan equals
74     @Override
75     public int hashCode() {
76         return Objects.hash(NIP); // Generate hashCode berdasarkan NIP
77     }

```

Activate Window  
Go to Settings to activate

```

14 public class Perusahaan { //awal kelas Perusahaan
15     private String namaPerusahaan; // Menyimpan nama perusahaan
16     private String alamat; // Menyimpan alamat perusahaan
17     private String pemilik; // Menyimpan nama pemilik
18     private String npwp; // Menyimpan nomor NPWP
19
20     // Menggunakan ArrayList untuk menyimpan koleksi pegawai
21     private List<Pegawai> daftarPegawai;
22
23     // Constructor untuk membuat objek Perusahaan baru
24     public Perusahaan(String namaPerusahaan, String alamat, String pemilik, String npwp) {
25         this.namaPerusahaan = namaPerusahaan; // Set nama perusahaan
26         this.alamat = alamat; // Set alamat
27         this.pemilik = pemilik; // Set pemilik
28         this.npwp = npwp; // Set NPWP
29         this.daftarPegawai = new ArrayList<>(); // Inisialisasi ArrayList
30     }
31
32     // Method untuk menambahkan pegawai baru
33     public void tambahPegawai(Pegawai pegawaiBaru) {
34         // Cek apakah pegawai dengan NIP yang sama sudah ada
35         if (daftarPegawai.contains(pegawaiBaru)) {
36             System.out.println("Pegawai dengan NIP " + pegawaiBaru.getNIP() + " sudah ada!");
37             return;
38         }
39         daftarPegawai.add(pegawaiBaru); // Tambahkan pegawai ke ArrayList
40         System.out.println("Pegawai " + pegawaiBaru.getNama() + " berhasil ditambahkan.");
41     }
42
43     // Method untuk menampilkan daftar semua pegawai
44     public void tampilkanDaftarPegawai() {
45         if (daftarPegawai.isEmpty()) { // Cek jika tidak ada pegawai
46             System.out.println("Belum ada pegawai di perusahaan ini.");
47             return;
48         }
49         System.out.println("==== Daftar Pegawai " + namaPerusahaan + " =====");
50         // Loop melalui semua pegawai menggunakan for-each
51         for (Pegawai pegawai : daftarPegawai) {
52             pegawai.tampilkanInfo(); // Tampilkan info pegawai
53         }
54     }
55
56     // Method untuk mencari pegawai dengan gaji tertinggi
57     public Pegawai getPegawaiGajiTertinggi() {
58         if (daftarPegawai.isEmpty()) return null; // Return null jika tidak ada pegawai
59
60         Pegawai tertinggi = daftarPegawai.get(0); // Asumsikan pegawai pertama tertinggi
61         // Cari pegawai dengan gaji lebih tinggi
62         for (Pegawai pegawai : daftarPegawai) {
63             if (pegawai.getGajiPokok() > tertinggi.getGajiPokok()) {
64                 tertinggi = pegawai;
65             }
66         }
67         return tertinggi;
68     }
69
70     // Method untuk mencari pegawai dengan gaji terendah
71     public Pegawai getPegawaiGajiTerendah() {
72         if (daftarPegawai.isEmpty()) return null; // Return null jika tidak ada pegawai
73
74         Pegawai terendah = daftarPegawai.get(0); // Asumsikan pegawai pertama terendah
75         // Cari pegawai dengan gaji lebih rendah
76         for (Pegawai pegawai : daftarPegawai) {
77             if (pegawai.getGajiPokok() < terendah.getGajiPokok()) {
78                 terendah = pegawai;
79             }
80         }
81         return terendah;
82     }
83
84     // Method untuk menghitung rata-rata gaji semua pegawai
85     public double getRataRataGaji() {
86         if (daftarPegawai.isEmpty()) return 0; // Return 0 jika tidak ada pegawai
87
88         double total = 0;
89         // Hitung total gaji semua pegawai
90         for (Pegawai pegawai : daftarPegawai) {
91             total += pegawai.getGajiPokok();
92         }
93         return total / daftarPegawai.size(); // Hitung rata-rata
94     }
95
96     // Method untuk menghapus pegawai berdasarkan NIP
97     public boolean hapusPegawai(String NIP) {
98         // Buat objek dummy untuk pencarian
99         Pegawai pegawai = new Pegawai(NIP, "", 0);
100         return daftarPegawai.remove(pegawai); // Hapus pegawai jika ditemukan
101     }
102
103     // Method getter untuk mendapatkan jumlah pegawai
104     public int getJumlahPegawai() {
105         return daftarPegawai.size(); // Kembalikan ukuran ArrayList
106     }
107
108     // Method getter untuk nama perusahaan
109     public String getNamaPerusahaan() {
110         return namaPerusahaan;
111     }
112
113     // Method getter untuk alamat perusahaan
114     public String getAlamat() {
115         return alamat;
116     }
117
118     // Method getter untuk nama pemilik
119     public String getPemilik() {
120         return pemilik;
121     }
122
123     // Method getter untuk NPWP
124     public String getNpwp() {
125         return npwp;
126     }
127
128

```

Activat  
Go to Set

```

11  * @author Celvin Pati
12  */
13  public class KelolaPerusahaan {
14      public static void main(String[] args) {
15          // Membuat objek scanner untuk input dari user
16          Scanner scanner = new Scanner(System.in);
17
18          // Input data perusahaan
19          System.out.println("=== Pendirian Perusahaan ===");
20          System.out.print("Masukkan Nama Perusahaan : ");
21          String namaPerusahaan = scanner.nextLine(); // Membaca nama perusahaan
22
23          System.out.print("Masukkan Alamat : ");
24          String alamat = scanner.nextLine(); // Membaca alamat perusahaan
25
26          System.out.print("Masukkan Nama Pemilik : ");
27          String pemilik = scanner.nextLine(); // Membaca nama pemilik
28
29          System.out.print("Masukkan NPWP : ");
30          String npwp = scanner.nextLine(); // Membaca NPWP
31
32          // Membuat objek perusahaan baru
33          Perusahaan perusahaan = new Perusahaan(namaPerusahaan, alamat, pemilik, npwp);
34
35          // Input data pegawai awal
36          System.out.println("\n=== Rekrutmen Pegawai Awal ===");
37          System.out.print("Masukkan jumlah pegawai awal : ");
38          int jumlahAwal = scanner.nextInt(); // Membaca jumlah pegawai
39          scanner.nextLine(); // Membersihkan newline
40
41          // Loop untuk input data setiap pegawai
42          for (int i = 0; i < jumlahAwal; i++) {
43              System.out.println("\nData pegawai ke-" + (i + 1));
44
45              System.out.print("NIP Pegawai : ");
46              String nip = scanner.nextLine(); // Membaca NIP
47
48              System.out.print("Nama Pegawai : ");
49              String namaPegawai = scanner.nextLine(); // Membaca nama
50
51              System.out.print("Gaji Pokok : ");
52              double gajiPokok = scanner.nextDouble(); // Membaca gaji
53              scanner.nextLine(); // Membersihkan newline
54
55              // Membuat dan menambahkan pegawai baru
56              Pegawai pegawaiBaru = new Pegawai(nip, namaPegawai, gajiPokok);
57              perusahaan.tambahPegawai(pegawaiBaru);
58          }
59
60          // Menu interaktif
61          int pilihan;
62          do {
63              // Menampilkan menu pilihan
64              System.out.println("\n=== Menu Utama ===");
65              System.out.println("1. Tampilkan Daftar Pegawai");
66              System.out.println("2. Tampilkan Pegawai dengan Gaji Tertinggi");
67              System.out.println("3. Tampilkan Pegawai dengan Gaji Terendah");
68              System.out.println("4. Tampilkan Rata-rata Gaji");
69              System.out.println("5. Tambah Pegawai Baru");
70              System.out.println("6. Hapus Pegawai");
71              System.out.println("0. Keluar");
72              System.out.print("Pilihan Anda: ");
73
74              pilihan = scanner.nextInt(); // Membaca pilihan user
75              scanner.nextLine(); // Membersihkan newline
76
77              // Pemrosesan pilihan menu
78              switch (pilihan) {
79                  case 1: // Tampilkan daftar pegawai
80                      perusahaan.tampilkanDaftarPegawai();
81                      break;
82                  case 2: // Cari pegawai bergaji tertinggi
83                      Pegawai tertinggi = perusahaan.getPegawaiGajiTertinggi();
84                      System.out.println("\n=== Pegawai dengan Gaji Tertinggi ===");
85                      if (tertinggi != null) {
86                          tertinggi.tampilkanInfo();
87                      } else {
88                          System.out.println("Belum ada pegawai.");
89                      }
90                      break;
91                  case 3: // Cari pegawai bergaji terendah
92                      Pegawai terendah = perusahaan.getPegawaiGajiTerendah();
93                      System.out.println("\n=== Pegawai dengan Gaji Terendah ===");
94                      if (terendah != null) {
95                          terendah.tampilkanInfo();
96                      } else {
97                          System.out.println("Belum ada pegawai.");
98                      }
99                      break;
100                 case 4: // Hitung rata-rata gaji
101                     double rataRata = perusahaan.getRataRataGaji();
102                     System.out.printf("\nRata-rata Gaji Pegawai: Rp %.2f\n", rataRata);
103                     break;
104                 case 5: // Tambah pegawai baru
105                     System.out.println("\n=== Tambah Pegawai Baru ===");
106                     System.out.print("NIP Pegawai : ");
107                     String nip = scanner.nextLine();
108
109                     System.out.print("Nama Pegawai : ");
110                     String namaPegawai = scanner.nextLine();
111
112                     System.out.print("Gaji Pokok : ");
113                     double gajiPokok = scanner.nextDouble();
114                     scanner.nextLine();
115
116                     // Membuat dan menambahkan pegawai baru
117                     Pegawai pegawaiBaru = new Pegawai(nip, namaPegawai, gajiPokok);
118                     perusahaan.tambahPegawai(pegawaiBaru);
119                     break;
120                 case 6: // Hapus pegawai
121                     System.out.print("\nMasukkan NIP Pegawai yang akan dihapus: ");
122                     String nipHapus = scanner.nextLine();
123                     if (perusahaan.hapusPegawai(nipHapus)) {
124                         System.out.println("Pegawai berhasil dihapus");
125                     } else {
126                         System.out.println("Pegawai dengan NIP tersebut tidak ditemukan");
127                     }
128                     break;
129                 case 0: // Keluar dari program
130                     System.out.println("Terima kasih telah menggunakan sistem ini.");
131                     break;
132                 default: // Pilihan tidak valid
133                     System.out.println("Pilihan tidak valid!");
134             }
135             while (pilihan != 0); // Loop sampai user memilih keluar
136
137             scanner.close(); // Menutup scanner
138         }
139     }
140 }

```

## C. OUTPUT

- Setelah program nya saya modifikasi menggunakan array list maka hasilnya sebagai berikut.

```
--- resources:3.3.1:resources (default-resources) @ versiArrayList ---
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\versiArrayList\src\main\resources

--- compiler:3.13.0:compile (default-compile) @ versiArrayList ---
Recompiling the module because of changed source code.
Compiling 3 source files with javac [debug release 23] to target\classes

--- exec:3.1.0:exec (default-cli) @ versiArrayList ---
==== Pendirian Perusahaan ====
Masukkan Nama Perusahaan : TP.Forex Pratama
Masukkan Alamat : Sumba Barat
Masukkan Nama Pemilik : Calvin Umbu Pati
Masukkan NPPW : 207789

==== Rekrutmen Pegawai Awal ====
Masukkan jumlah pegawai awal: 2

Data pegawai ke-1
NIP Pegawai : 207789
Nama Pegawai : Catriona
Gaji Pokok : 500000000
Pegawai Catriona berhasil ditambahkan.

Data pegawai ke-2
NIP Pegawai : Max
Nama Pegawai : Max
Gaji Pokok : 600000000
Pegawai Max berhasil ditambahkan.

==== Menu Utama ====
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar

Pilihan Anda: 1
===== Daftar Pegawai TP.Forex Pratama =====
NIP: 207789
Nama: Catriona
Gaji Pokok: Rp 500.000.000,00
-----
NIP: Max
Nama: Max
Gaji Pokok: Rp 600.000.000,00
-----

==== Menu Utama ====
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 2

==== Pegawai dengan Gaji Tertinggi ====
NIP: Max
Nama: Max
Gaji Pokok: Rp 600.000.000,00
-----

==== Menu Utama ====
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 3

==== Pegawai dengan Gaji Terendah ====
NIP: 207789
Nama: Catriona
Gaji Pokok: Rp 500.000.000,00
-----

==== Menu Utama ====
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 4

Rata-rata Gaji Pegawai: Rp 550.000.000,00

==== Menu Utama ====
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 5

==== Tambah Pegawai Baru ====
NIP Pegawai : 207785
Nama Pegawai : Daniel
Gaji Pokok : 500000000
Pegawai Daniel berhasil ditambahkan.
```

Activate W  
Go to Settings

```
=====
=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 6

Masukkan NIP Pegawai yang akan dihapus: 207789
Pegawai berhasil dihapus

=== Menu Utama ===
1. Tampilkan Daftar Pegawai
2. Tampilkan Pegawai dengan Gaji Tertinggi
3. Tampilkan Pegawai dengan Gaji Terendah
4. Tampilkan Rata-rata Gaji
5. Tambah Pegawai Baru
6. Hapus Pegawai
0. Keluar
Pilihan Anda: 0
Terima kasih telah menggunakan sistem ini.
=====
BUILD SUCCESS
=====
Total time: 03:15 min
Finished at: 2025-04-07T01:26:41+07:00
=====
```

## D. ANALISA

### Pegawai

No	Syntax	penjelasan
1	java public Pegawai(String NIP, String nama, double gajiPokok) { this.NIP = NIP; this.nama = nama; this.gajiPokok = gajiPokok; }	// Membuat objek Pegawai baru dengan NIP, nama, dan gaji pokok.
2	java public String getNIP() { return NIP; }	//Mengembalikan nilai NIP pegawai.
3	java public String getNama() { return nama; }	//Mengembalikan nama pegawai.
4	java public double getGajiPokok() { return gajiPokok; }	// Mengembalikan gaji pokok pegawai.
5	java public void setNIP(String NIP) { this.NIP = NIP; }	// Mengubah NIP pegawai.
6	java public void setName(String nama) { this.nama = nama; }	// Mengubah nama pegawai.
7	java public void setGajiPokok(double gajiPokok) { this.gajiPokok = gajiPokok; }	Mengubah gaji pokok pegawai.
8	java public void tampilkanInfo() { System.out.println("NIP: " + NIP); System.out.println("Nama: " + nama); System.out.println("Gaji Pokok: Rp " + String.format("%.2f", gajiPokok)); System.out.println("-----"); }	//Menampilkan informasi lengkap pegawai di konsol.
9	java @Override public boolean equals(Object o) { if (this == o) return true; if (o == null	//untuk memastikan bahwa objek yang dibandingkan (o) bukan null dan memiliki class yang sama dengan objek saat ini (this).
10	java @Override public int hashCode() { return Objects.hash(NIP); }	//Menghasilkan kode hash berdasarkan NIP agar sesuai dengan equals().

### Perusahaan

No	Syntax	penjelasan
1	private String namaPerusahaan; private String alamat; private String pemilik; private String npwp;	// Menyimpan data identitas perusahaan: nama, alamat, nama pemilik, dan NPWP.

2	private List<Pegawai> daftarPegawai;	// Menyimpan daftar objek Pegawai yang bekerja di perusahaan, pakai ArrayList.
3	public Perusahaan(String namaPerusahaan, String alamat, String pemilik, String npwp)	// Constructor untuk membuat objek Perusahaan baru dan menginisialisasi atribut serta ArrayList untuk pegawai.
4	public void tambahPegawai(Pegawai pegawaiBaru)	// Menambahkan pegawai ke daftar. Dicek dulu apakah NIP-nya sudah ada (pakai contains yang tergantung method equals() di kelas Pegawai).
5	public void tampilkanDaftarPegawai()	// Menampilkan semua informasi pegawai yang ada di daftarPegawai. Jika kosong, tampilkan pesan "belum ada pegawai".
6	public Pegawai getPegawaiGajiTertinggi()	// Mengembalikan objek Pegawai dengan gaji pokok paling tinggi. Cek satu per satu menggunakan perbandingan.
7	public Pegawai getPegawaiGajiTerendah()	// Sama seperti sebelumnya, tapi mencari gaji pokok paling rendah.
8	public double getRataRataGaji()	// Menghitung rata-rata dari seluruh gaji pokok pegawai yang ada.
9	public boolean hapusPegawai(String NIP)	// Menghapus pegawai berdasarkan NIP. Dibuat objek dummy Pegawai dengan NIP yang dicari, lalu dihapus dari daftar (dengan bantuan equals()).
10	public int getJumlahPegawai()	// Mengembalikan jumlah total pegawai yang sedang terdaftar di perusahaan.
11	getNamaPerusahaan(), getAlamat(), getPemilik(), getNpwp()	// Getter standar untuk mengambil informasi identitas perusahaan.

#### KelolaPerusahaan

No	Syntax	penjelasan
----	--------	------------

1	Scanner scanner = new Scanner(System.in);	// Membuat objek Scanner untuk membaca input dari keyboard.
2	System.out.println("=== Pendirian Perusahaan ===");	// Menampilkan judul bagian pendirian perusahaan.
3	scanner.nextLine()	// Membaca input berupa teks dari user.
4	Perusahaan perusahaan = new Perusahaan(...);	// Membuat objek Perusahaan dengan input nama, alamat, pemilik, dan NPWP.
5	int jumlahAwal = scanner.nextInt();	// Membaca jumlah pegawai awal dari user.
6	for (int i = 0; i < jumlahAwal; i++) { ... }	// Melakukan loop sebanyak jumlahAwal untuk mengisi data pegawai satu per satu.
7	Pegawai pegawaiBaru = new Pegawai(...);	// Membuat objek Pegawai baru dari input user.
8	perusahaan.tambahPegawai(pegawaiBaru);	// Menambahkan pegawai baru ke perusahaan.
9	do { ... } while (pilihan != 0);	// Melakukan pengulangan menu sampai user memilih keluar.
10	switch (pilihan)	// Menjalankan fitur sesuai pilihan user
11	tampilkanDaftarPegawai()	// Menampilkan daftar semua pegawai menggunakan
12	getPegawaiGajiTertinggi()	// Menampilkan pegawai dengan gaji tertinggi menggunakan
13	getPegawaiGajiTerendah()	// Menampilkan pegawai dengan gaji terendah menggunakan
14	getRataRataGaji()	// Menghitung dan menampilkan rata-rata gaji dengan

15	hapusPegawai(nip)	//Menghapus pegawai berdasarkan NIP dengan
16	default	// Menampilkan pesan jika pilihan tidak valid.
17	scanner.close();	// Menutup resource Scanner untuk mencegah kebocoran memori



## Latihan kasus 2

### A. DIAGRAM CLASS

Buku
- kodeBuku: String - judul: String - tahunTerbit: int - daftarPengarang: Pengarang[] - jumlahPengarangSekarang: int
<<constructor>> Buku(kodeBuku: String, judul: String, tahunTerbit: int, jumlahPengarang: int) + tambahPengarang(p: Pengarang): boolean + getTahunTerbit(): int + toString(): String

Pengarang
- nik: String - nama: String - alamat: String
+ Pengarang(nik, nama, alamat) + toString(): String

KelolaBuku
+ main(String[ ]): void

## B. LISTING PROGRAM

```

9  * @author Calvin Pati
10 */
11 public class Buku {
12     private String kodeBuku; // Menyimpan kode unik buku
13     private String judul; // Menyimpan judul buku
14     private int tahunTerbit; // Menyimpan tahun terbit buku
15     private Pengarang[] daftarPengarang; // Array untuk menyimpan daftar pengarang
16     private int jumlahPengarangSekarang = 0; // Penghitung jumlah pengarang yang sudah ditambahkan
17
18     // Konstruktor untuk inisialisasi objek Buku
19     public Buku(String kodeBuku, String judul, int tahunTerbit, int jumlahPengarang) {
20         this.kodeBuku = kodeBuku; // Set kode buku
21         this.judul = judul; // Set judul buku
22         this.tahunTerbit = tahunTerbit; // Set tahun terbit
23         // Inisialisasi array daftarPengarang dengan kapasitas tertentu
24         this.daftarPengarang = new Pengarang[jumlahPengarang];
25     }
26
27     public boolean tambahPengarang(Pengarang pengarang) {
28         // Cek apakah masih ada slot kosong
29         if (jumlahPengarangSekarang < daftarPengarang.length) {
30             daftarPengarang[jumlahPengarangSekarang] = pengarang; // Tambahkan pengarang
31             jumlahPengarangSekarang++; // Increment counter
32             return true;
33         }
34         return false; // Return false jika array sudah penuh
35     }
36
37     public int getTahunTerbit() {
38         return tahunTerbit;
39     }
40
41     @Override
42     public String toString() {
43         // Menggunakan StringBuilder untuk efisiensi
44         StringBuilder sb = new StringBuilder("Kode Buku: " + kodeBuku +
45             ", Judul: " + judul +
46             ", Tahun Terbit: " + tahunTerbit +
47             "\nPengarang:\n");
48         // Loop melalui semua pengarang yang sudah ditambahkan
49         for (int i = 0; i < jumlahPengarangSekarang; i++) {
50             sb.append("    - " + daftarPengarang[i].toString() + "\n");
51         }
52         return sb.toString();
53     }
54 }

```

```

8  *
9  * @author Calvin Pati
10 */
11 public class Pengarang {
12     private String nik; // Nomor Induk Kependudukan pengarang
13     private String nama; // Nama lengkap pengarang
14     private String alamat; // Alamat pengarang
15
16     public Pengarang(String nik, String nama, String alamat) {
17         this.nik = nik; // Set NIK
18         this.nama = nama; // Set nama
19         this.alamat = alamat; // Set alamat
20     }
21
22     @Override
23     public String toString() {
24         return "NIK: " + nik + ", Nama: " + nama + ", Alamat: " + alamat;
25     }
26 }
27
28

```

```

11  * @author Calvin Pati
12 */
13 public class KelolaBuku {
14     private Buku[] daftarBuku; // Array untuk menyimpan daftar buku
15     private int jumlahBukuSekarang = 0; // Penghitung jumlah buku yang sudah ditambahkan
16     public KelolaBuku(int kapasitas) {
17         this.daftarBuku = new Buku[kapasitas]; // Inisialisasi array dengan kapasitas tertentu
18     }
19
20     public void tambahBuku(Buku buku) {
21         // Cek apakah masih ada slot kosong
22         if (jumlahBukuSekarang < daftarBuku.length) {
23             daftarBuku[jumlahBukuSekarang] = buku; // Tambahkan buku
24             jumlahBukuSekarang++; // Increment counter
25         }
26     }
27
28     public int hitungBukuPerTahun(int tahun) {
29         int jumlah = 0;
30         // Loop melalui semua buku
31         for (int i = 0; i < jumlahBukuSekarang; i++) {
32             // Jika tahun terbit sama, increment counter
33             if (daftarBuku[i].getTahunTerbit() == tahun) {
34                 jumlah++;
35             }
36         }
37         return jumlah;
38     }
39
40     public Buku cariBukuTertua() {
41         if (jumlahBukuSekarang == 0) return null; // Jika tidak ada buku, return null
42         Buku tertua = daftarBuku[0]; // Asumsikan buku pertama adalah yang tertua
43         // Loop melalui sisa buku
44         for (int i = 1; i < jumlahBukuSekarang; i++) {
45
46

```

```

44         // Jika ditemukan buku dengan tahun lebih lama, update referensi
45         if (daftarBuku[i].getTahunTerbit() < tertua.getTahunTerbit()) {
46             tertua = daftarBuku[i];
47         }
48     }
49     return tertua;
50 }
51
52 public Buku cariBukuTerbaru() {
53     if (jumlahBukuSekarang == 0) return null; // Jika tidak ada buku, return null
54
55     Buku terbaru = daftarBuku[0]; // Asumsikan buku pertama adalah yang terbaru
56     // Loop melalui sisa buku
57     for (int i = 1; i < jumlahBukuSekarang; i++) {
58         // Jika ditemukan buku dengan tahun lebih baru, update referensi
59         if (daftarBuku[i].getTahunTerbit() > terbaru.getTahunTerbit()) {
60             terbaru = daftarBuku[i];
61         }
62     }
63     return terbaru;
64 }
65
66 public void tampilkanSemuaBuku() {
67     // Loop melalui semua buku yang sudah ditambahkan
68     for (int i = 0; i < jumlahBukuSekarang; i++) {
69         System.out.println(daftarBuku[i]); // Gunakan method toString()
70     }
71 }
72
73 // Method main sebagai entry point program
74 public static void main(String[] args) {
75     Scanner scanner = new Scanner(System.in); // Membuat objek Scanner untuk input
76
77     System.out.print("Masukkan jumlah buku yang mau disimpan: ");
78     int jumlahBuku = scanner.nextInt(); // Membaca jumlah buku
79     scanner.nextLine(); // Membersihkan newline
80
81     // Membuat objek KelolaBuku dengan kapasitas tertentu
82     KelolaBuku kelolaBuku = new KelolaBuku(jumlahBuku);
83
84     // Loop untuk input data buku
85     for (int i = 0; i < jumlahBuku; i++) {
86         System.out.println("\nMasukkan data buku ke-" + (i + 1));
87         System.out.print("Kode Buku: ");
88         String kodeBuku = scanner.nextLine(); // Input kode buku
89         System.out.print("Judul: ");
90         String judul = scanner.nextLine(); // Input judul
91         System.out.print("Tahun Terbit: ");
92         int tahunTerbit = scanner.nextInt(); // Input tahun terbit
93         System.out.print("Jumlah Pengarang: ");
94         int jumlahPengarang = scanner.nextInt(); // Input jumlah pengarang
95         scanner.nextLine(); // Membersihkan newline
96
97         // Membuat objek Buku baru
98         Buku buku = new Buku(kodeBuku, judul, tahunTerbit, jumlahPengarang);
99
100        // Loop untuk input data pengarang
101        for (int j = 0; j < jumlahPengarang; j++) {
102            System.out.println("Masukkan data pengarang ke-" + (j + 1));
103            System.out.print("NIK: ");
104            String nik = scanner.nextLine(); // Input NIK
105            System.out.print("Nama: ");
106            String nama = scanner.nextLine(); // Input nama
107            System.out.print("Alamat: ");
108            String alamat = scanner.nextLine(); // Input alamat
109
110            // Menambahkan pengarang ke buku
111            buku.tambahPengarang(new Pengarang(nik, nama, alamat));
112        }
113
114        // Menambahkan buku ke sistem
115        kelolaBuku.tambahBuku(buku);
116    }
117
118    // Menampilkan semua buku
119    System.out.println("\nDaftar Buku:");
120    kelolaBuku.tampilkanSemuaBuku();
121
122    // Mencari buku berdasarkan tahun terbit
123    System.out.print("\nMasukkan tahun untuk mencari jumlah buku yang terbit: ");
124    int tahunCari = scanner.nextInt();
125    int jumlah = kelolaBuku.hitungBukuPerTahun(tahunCari);
126    System.out.println("Jumlah buku yang terbit pada tahun " + tahunCari + ": " + jumlah);
127
128    // Mencari buku tertua dan terbaru
129    Buku bukuTertua = kelolaBuku.cariBukuTertua();
130    Buku bukuTerbaru = kelolaBuku.cariBukuTerbaru();
131
132    System.out.println("\nBuku terbitan paling lama:\n" + bukuTertua);
133    System.out.println("Buku terbitan paling akhir:\n" + bukuTerbaru);
134
135    scanner.close(); // Menutup scanner
136 }

```

## C. OUTPUT

- Setelah program nya saya perbaiki dan saya test maka tidak ada lagi yang error di program.

```
--- exec:3.1.0:exec (default-cli) @ PBOModul6Fix ---
Masukkan jumlah buku yang mau disimpan: 2

Masukkan data buku ke-1
Kode Buku: 200
Judul: belajar java
Tahun Terbit: 2015
Jumlah Pengarang: 1
Masukkan data pengarang ke-1
NIK: 207890
Nama: Catriona
Alamat: philipines

Masukkan data buku ke-2
Kode Buku: 809
Judul: belajar python
Tahun Terbit: 2023
Jumlah Pengarang: 2
Masukkan data pengarang ke-1
NIK: 207980
Nama: max
Alamat: amerika serikat
Masukkan data pengarang ke-2
NIK: 307980
Nama: derek
Alamat: uni soviet

Daftar Buku:
Kode Buku: 200, Judul: belajar java, Tahun Terbit: 2015
Pengarang:
  - NIK: 207890, Nama: Catriona, Alamat: philipines

Kode Buku: 809, Judul: belajar python, Tahun Terbit: 2023
Pengarang:
  - NIK: 207980, Nama: max, Alamat: amerika serikat
  - NIK: 307980, Nama: derek, Alamat: uni soviet

Masukkan tahun untuk mencari jumlah buku yang terbit: 2023
Jumlah buku yang terbit pada tahun 2023: 1

Buku terbitan paling lama:
Kode Buku: 200, Judul: belajar java, Tahun Terbit: 2015
Pengarang:
  - NIK: 207890, Nama: Catriona, Alamat: philipines

Buku terbitan paling akhir:
Kode Buku: 809, Judul: belajar python, Tahun Terbit: 2023
Pengarang:
  - NIK: 207980, Nama: max, Alamat: amerika serikat
  - NIK: 307980, Nama: derek, Alamat: uni soviet

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

Total time: 02:17 min
Finished at: 2025-04-07T03:22:42+07:00
-----
```

## D. ANALISA

### Buku

No	Syntax	penjelasan
1	public Buku(String kodeBuku, String judul, int tahunTerbit, int jumlahPengarang	// Konstruktor yang dipanggil saat membuat objek Buku. Inisialisasi data dan array pengarang.
2	public boolean tambahPengarang(Pengarang pengarang)	// deklarasikan atribut x1 dan y1 sebagai titik pertama segitiga.
3	public int getTahunTerbit()	// Method getter untuk mengambil nilai dari properti tahunTerbit
4	@Override public String toString()	// Override dari method toString() untuk menampilkan informasi lengkap buku dan pengarangnya.

### Pengarang

No	Syntax	penjelasan
1	public Pengarang(String nik, String nama, String alamat)	// Konstruktor untuk membuat objek Pengarang dengan data lengkap
2	public String toString()	//Override method toString() dari class Object.

### KelolaBuku

No	Syntax	penjelasan
1	KelolaBuku(int kapasitas)	//Konstruktor untuk inisialisasi array daftarBuku
2	void tambahBuku(Buku buku)	//Menambahkan objek Buku ke array
3	int hitungBukuPerTahun(int tahun)	//Menghitung jumlah buku berdasarkan tahun terbit

4	Buku cariBukuTertua()	//Mencari buku dengan tahun terbit paling lama
5	Buku cariBukuTerbaru()	//Mencari buku dengan tahun terbit paling baru
6	void tampilkanSemuaBuku()	//Menampilkan semua buku yang tersimpan
7	main(String[] args)	//Entry point program

## Praktikum kasus 3

### A. DIAGRAM CLASS

Segitiga
- x1: double - y1: double - x2: double - y2: double - x3: double - y3: double
<<constructor>> hitungSisi(xa: double, ya: double, xb: double, yb: double): double + hitungKeliling(): double + hitungLuas(): double

KumpulanSegitiga
- jumlahSegitiga: int - daftarSegitiga: ArrayList<Segitiga>
+ KumpulanSegitiga() + tambahSegitiga(segitiga: Segitiga): void + hitungRataRataLuas(): double + cariSegitigaTerluas(): Segitiga + cariSegitigaTerkecil(): Segitiga + getJumlahSegitiga(): int

SegitigaMain
+ main(String[ ]): void

## B. LISTING PROGRAM

```
11 public class Segitiga { //awal kelas segitiga
12     // Deklarasi variabel koordinat titik 1 (x1, y1)
13     public double x1, y1;
14     // Deklarasi variabel koordinat titik 2 (x2, y2)
15     public double x2, y2;
16     // Deklarasi variabel koordinat titik 3 (x3, y3)
17     public double x3, y3;
18
19     // Method untuk menghitung panjang sisi antara dua titik
20     // Parameter: koordinat titik A (xa, ya) dan titik B (xb, yb)
21     public double hitungSisi(double xa, double ya, double xb, double yb) {
22         // Menghitung jarak Euclidean antara dua titik menggunakan rumus Pythagoras
23         return Math.sqrt(Math.pow(xa - xb, 2) + Math.pow(ya - yb, 2));
24     }
25
26     // Method untuk menghitung keliling segitiga
27     public double hitungKeliling() {
28         // Menjumlahkan panjang ketiga sisi segitiga dengan memanggil method hitungSisi()
29         return hitungSisi(x1, y1, x2, y2) +
30             hitungSisi(x2, y2, x3, y3) +
31             hitungSisi(x3, y3, x1, y1);
32     }
33
34     // Method untuk menghitung luas segitiga menggunakan rumus Heron
35     public double hitungLuas() {
36         // Menghitung semi-perimeter (setengah dari keliling)
37         double s = hitungKeliling() / 2;
38         // Mengaplikasikan rumus Heron untuk menghitung luas
39         return Math.sqrt(s * (s - hitungSisi(x1, y1, x2, y2)) *
40             (s - hitungSisi(x2, y2, x3, y3)) *
41             (s - hitungSisi(x3, y3, x1, y1)));
42     }
43 }
```

```
13 public class KumpulanSegitiga {
14     // Atribut untuk menyimpan jumlah segitiga dalam kumpulan
15     private int jumlahSegitiga;
16     // Atribut untuk menyimpan daftar segitiga menggunakan ArrayList
17     private ArrayList<Segitiga> daftarSegitiga;
18
19     // Konstruktor untuk inisialisasi awal
20     public KumpulanSegitiga() {
21         this.jumlahSegitiga = 0; // Inisialisasi jumlah segitiga = 0
22         this.daftarSegitiga = new ArrayList<>(); // Inisialisasi ArrayList kosong
23     }
24
25     // Method untuk menambahkan segitiga ke dalam kumpulan
26     // Parameter: objek Segitiga yang akan ditambahkan
27     public void tambahSegitiga(Segitiga segitiga) {
28         this.daftarSegitiga.add(segitiga); // Menambahkan ke ArrayList
29         this.jumlahSegitiga++; // Menambah counter jumlah segitiga
30     }
31
32     // Method untuk menghitung rata-rata luas semua segitiga
33     public double hitungRataRataLuas() {
34         // Penanganan kasus ketika tidak ada segitiga
35         if (jumlahSegitiga == 0) {
36             return 0; // Mengembalikan 0 jika tidak ada segitiga
37         }
38
39         double totalLuas = 0;
40         // Loop melalui semua segitiga dan akumulasi total luas
41         for (Segitiga segitiga : daftarSegitiga) {
42             totalLuas += segitiga.hitungLuas();
43         }
44         // Hitung rata-rata dengan membagi total dengan jumlah segitiga
45         return totalLuas / jumlahSegitiga;
46     }
47
48     // Method untuk mencari segitiga dengan luas terbesar
49     public Segitiga cariSegitigaTerluas() {
50         // Penanganan kasus ketika tidak ada segitiga
51         if (jumlahSegitiga == 0) {
52             return null; // Mengembalikan null jika tidak ada segitiga
53         }
54
55         // Asumsikan segitiga pertama sebagai yang terluas
56         Segitiga terluas = daftarSegitiga.get(0);
57         // Loop untuk membandingkan dengan segitiga lainnya
58         for (Segitiga segitiga : daftarSegitiga) {
59             if (segitiga.hitungLuas() > terluas.hitungLuas()) {
60                 terluas = segitiga; // Update jika ditemukan yang lebih luas
61             }
62         }
63         return terluas;
64     }
65
66     // Method untuk mencari segitiga dengan keliling terkecil
67     public Segitiga cariSegitigaTerkecil() {
68         // Penanganan kasus ketika tidak ada segitiga
69         if (jumlahSegitiga == 0) {
70             return null; // Mengembalikan null jika tidak ada segitiga
71         }
72
73         // Asumsikan segitiga pertama sebagai yang terkecil
74         Segitiga terkecil = daftarSegitiga.get(0);
75         // Loop untuk membandingkan dengan segitiga lainnya
76         for (Segitiga segitiga : daftarSegitiga) {
77             if (segitiga.hitungKeliling() < terkecil.hitungKeliling()) {
78                 terkecil = segitiga; // Update jika ditemukan yang lebih kecil
79             }
80         }
81         return terkecil;
82     }
83
84     // Getter method untuk mendapatkan jumlah segitiga
85     public int getJumlahSegitiga() {
86         return jumlahSegitiga;
87     }
88 }
89
90
91
```

```

11 public class SegitigaMain {
12     // Method utama sebagai entry point program
13     public static void main(String[] args) {
14         // Membuat objek segitiga pertama (seg1)
15         Segitiga seg1 = new Segitiga();
16         // Mengisi koordinat titik-titik segitiga pertama
17         seg1.x1 = 1; seg1.y1 = 2;
18         seg1.x2 = 7; seg1.y2 = 2;
19         seg1.x3 = 7; seg1.y3 = 9;
20
21         // Membuat objek segitiga kedua (seg2)
22         Segitiga seg2 = new Segitiga();
23         // Mengisi koordinat titik-titik segitiga kedua
24         seg2.x1 = 0; seg2.y1 = 0;
25         seg2.x2 = 5; seg2.y2 = 4;
26         seg2.x3 = 8; seg2.y3 = 2;
27
28         // Membuat objek segitiga ketiga (seg3)
29         Segitiga seg3 = new Segitiga();
30         // Mengisi koordinat titik-titik segitiga ketiga
31         seg3.x1 = 3; seg3.y1 = 1;
32         seg3.x2 = 6; seg3.y2 = 1;
33         seg3.x3 = 3; seg3.y3 = 4;
34
35         // Membuat objek KumpulanSegitiga
36         KumpulanSegitiga kumpulan = new KumpulanSegitiga();
37         // Menambahkan segitiga-segitiga ke dalam kumpulan
38         kumpulan.tambahSegitiga(seg1);
39         kumpulan.tambahSegitiga(seg2);
40         kumpulan.tambahSegitiga(seg3);
41
42         // Menampilkan informasi tentang kumpulan segitiga
43         System.out.println("Jumlah segitiga dalam kumpulan: " + kumpulan.getJumlahSegitiga());
44         System.out.println("Rata-rata luas segitiga: " + kumpulan.hitungRataRataLuas());
45
46         // Mencari dan menampilkan segitiga terluas
47         Segitiga terluas = kumpulan.cariSegitigaTerluas();
48         System.out.println("Segitiga terluas memiliki luas: " + terluas.hitungLuas());
49
50         // Mencari dan menampilkan segitiga terkecil
51         Segitiga terkecil = kumpulan.cariSegitigaTerkecil();
52         System.out.println("Segitiga terkecil memiliki keliling: " + terkecil.hitungKeliling());
53     }
54 }

```

## C. Output

- Setelah program nya saya perbaiki dan dimodifikasi maka outputnya sebagai berikut i

```

-----< com.mycompany:PBOModul6Fix >-----
Building PBOModul6Fix 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----

--- resources:3.3.1:resources (default-resources) @ PBOModul6Fix ---
skip non existing resourceDirectory C:\Users\Windows\OneDrive\Documents\NetBeansProjects\PBOModul6Fix\src\main\resources

--- compiler:3.13.0:compile (default-compile) @ PBOModul6Fix ---
Recompiling the module because of changed source code.
Compiling 9 source files with javac [debug release 23] to target\classes

--- exec:3.1.0:exec (default-cli) @ PBOModul6Fix ---
Jumlah segitiga dalam kumpulan: 3
Rata-rata luas segitiga: 12.166666666666666
Segitiga terluas memiliki luas: 21.0
Segitiga terkecil memiliki keliling: 10.242640687119284

BUILD SUCCESS

Total time: 3.520 s
Finished at: 2025-04-07T20:06:00+07:00

```

## D. ANALISA

### Segitiga

No	Syntax	penjelasan
1	public class Segitiga {	// Mendeklarasikan kelas Segitiga sebagai kelas publik.
2	public double x1, y1;	//Koordinat titik pertama segitiga.
3	public double x2, y2;	//Koordinat titik kedua segitiga
4	public double x3, y3;	//Koordinat titik ketiga segitiga.
5	public double hitungSisi(double xa, double ya, double xb, double yb)	// Method untuk menghitung jarak (sisi) antara dua titik menggunakan rumus Pythagoras
6	Math.pow(xa - xb, 2) + Math.pow(ya - yb, 2)	//Mengkuadratkan selisih masing-masing koordinat.
7	Math.sqrt(...)	// Mengambil akar kuadrat dari hasil penjumlahan kuadrat selisih koordinat.
8	public double hitungKeliling()	//Method untuk menghitung total panjang tiga sisi segitiga.
9	hitungSisi(x1, y1, x2, y2) + hitungSisi(...)	//Menjumlahkan panjang sisi-sisi segitiga.
10	public double hitungLuas()	//Method untuk menghitung luas segitiga dengan rumus Heron.
11	double s = hitungKeliling() / 2;	//Menghitung semi-perimeter (setengah dari keliling).
12	Math.sqrt(s * (s - a) * (s - b) * (s - c)) }	Rumus Heron: menghitung luas berdasarkan panjang sisi dan semi-perimeter.

### KumpulanSegitiga

No	Syntax	penjelasan
1	public class KumpulanSegitiga {	// Mendefinisikan kelas KumpulanSegitiga.
2	private int jumlahSegitiga;	//Atribut untuk menyimpan jumlah segitiga yang ditambahkan.
3	private ArrayList<Segitiga> daftarSegitiga;	//Menyimpan daftar objek Segitiga menggunakan ArrayList
4	public KumpulanSegitiga() {	//Konstruktor untuk inialisasi objek KumpulanSegitiga.
5	this.jumlahSegitiga = 0;	//Inialisasi nilai awal jumlahSegitiga ke 0.
6	this.daftarSegitiga = new ArrayList<>();	//Membuat list kosong untuk menampung segitiga.
7	public void tambahSegitiga(Segitiga segitiga)	// Method untuk menambahkan objek Segitiga ke dalam list.
8	daftarSegitiga.add(segitiga);	//Menambahkan segitiga ke dalam list
9	jumlahSegitiga++;	//Menambah counter setiap kali ada segitiga baru ditambahkan.
10	public double hitungRataRataLuas()	//Method untuk menghitung rata-rata luas semua segitiga.
11	if (jumlahSegitiga == 0) return 0;	//Menghindari pembagian nol jika list kosong.
12	totalLuas += segitiga.hitungLuas();	//Menjumlahkan semua luas segitiga.
13	return totalLuas / jumlahSegitiga;	//Mengembalikan rata-rata luas.
14	public Segitiga cariSegitigaTerluas()	//Mencari segitiga dengan luas terbesar.
15	Segitiga terluas = daftarSegitiga.get(0);	//Inialisasi segitiga pertama sebagai yang terluas.
16	if (segitiga.hitungLuas() > terluas.hitungLuas())	//Cek apakah luas lebih besar, jika ya, update terluas.
17	public Segitiga cariSegitigaTerkecil()	//Mencari segitiga dengan keliling terkecil.

18	Segitiga terkecil = daftarSegitiga.get(0);	//Inisialisasi segitiga pertama sebagai yang terkecil.
19	if (segitiga.hitungKeliling() < terkecil.hitungKeliling())	//Cek apakah keliling lebih kecil, jika ya, update terkecil.
20	public int getJumlahSegitiga()	//Getter untuk mengembalikan jumlah segitiga yang tersimpan.

### SegitigaMain

No	Syntax	penjelasan
1	public class SegitigaMain {	// Mendefinisikan kelas utama untuk menjalankan program.
2	public static void main(String[] args) {	//Method main, sebagai titik masuk (entry point) eksekusi program Java.
3	Segitiga seg1 = new Segitiga();	//Membuat objek Segitiga pertama.
4	seg1.x1 = 1; seg1.y1 = 2; <i>(dan seterusnya)</i>	//Mengatur koordinat titik-titik segitiga pertama.
5	Segitiga seg2 = new Segitiga();	//Membuat objek Segitiga kedua.
6	Segitiga seg3 = new Segitiga();	//Membuat objek Segitiga ketiga.
7	KumpulanSegitiga kumpulan = new KumpulanSegitiga();	// Membuat objek KumpulanSegitiga untuk menyimpan beberapa segitiga.
8	kumpulan.tambahSegitiga(seg1); <i>(dst.</i>	//Menambahkan segitiga ke dalam list
9	System.out.println("Jumlah segitiga...");	// Menampilkan jumlah segitiga dalam kumpulan ke layar.
10	System.out.println("Rata-rata luas...");	// Menampilkan rata-rata luas dari semua segitiga dalam kumpulan.
11	Segitiga terluas = kumpulan.cariSegitigaTerluas();	// Mencari segitiga dengan luas terbesar.
12	System.out.println("Segitiga terluas...");	// Menampilkan luas dari segitiga yang paling besar.
13	Segitiga terkecil = kumpulan.cariSegitigaTerkecil();	// Mencari segitiga dengan keliling terkecil.
14	System.out.println("Segitiga terkecil...");	// Menampilkan keliling dari segitiga yang paling kecil.