LAPORAN PRATIKUM CODELAB PBO 2E MODUL 4

****

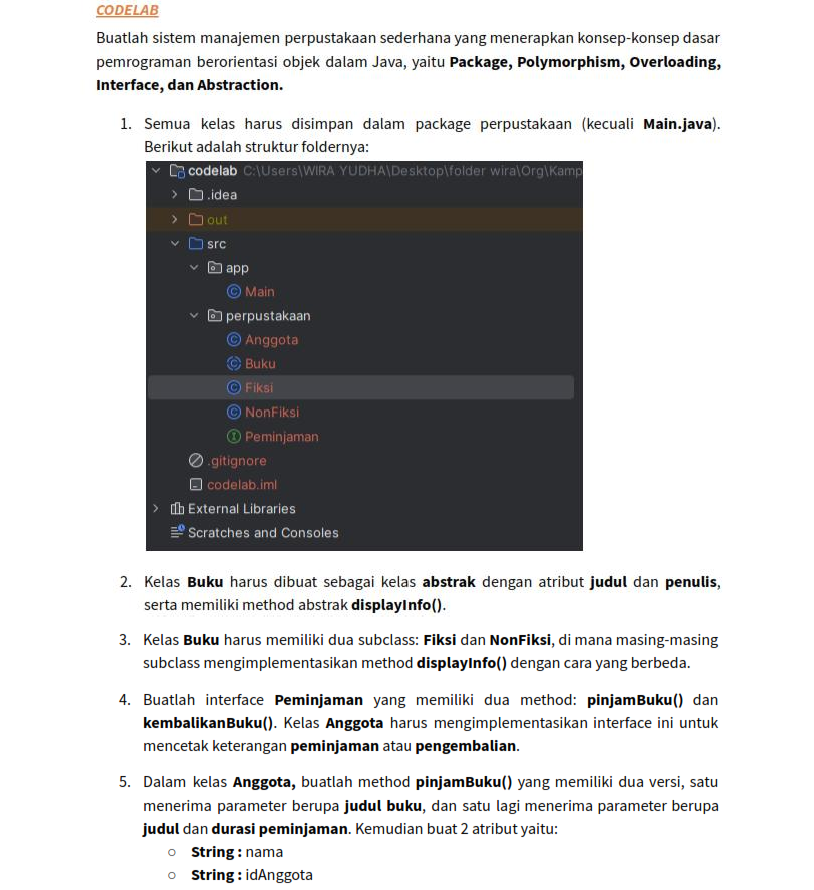
Nama: Julian Abi Mustofa Nabila

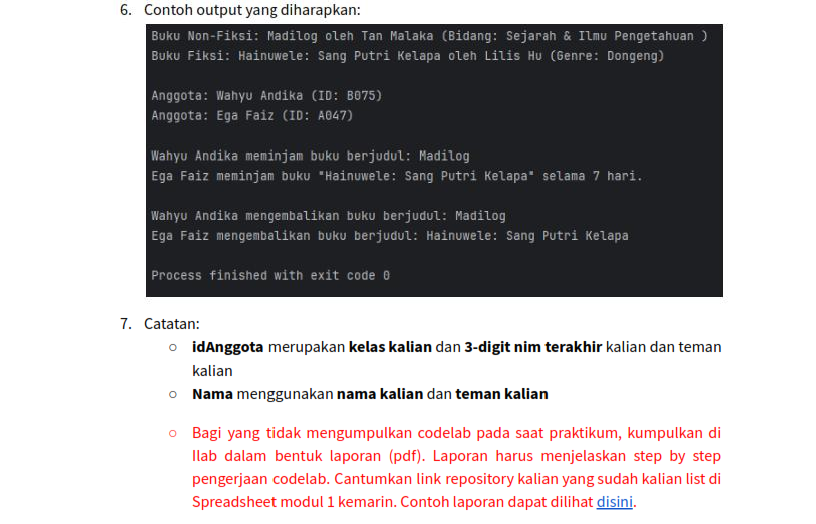
NIM: 202410370110175

Kelas: PBO 2E

Github: https://github.com/Grobiels/PBO\_E\_Julian-Abi-Mustofa\_175

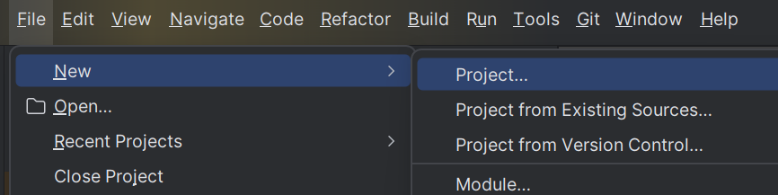
# CODELAB:



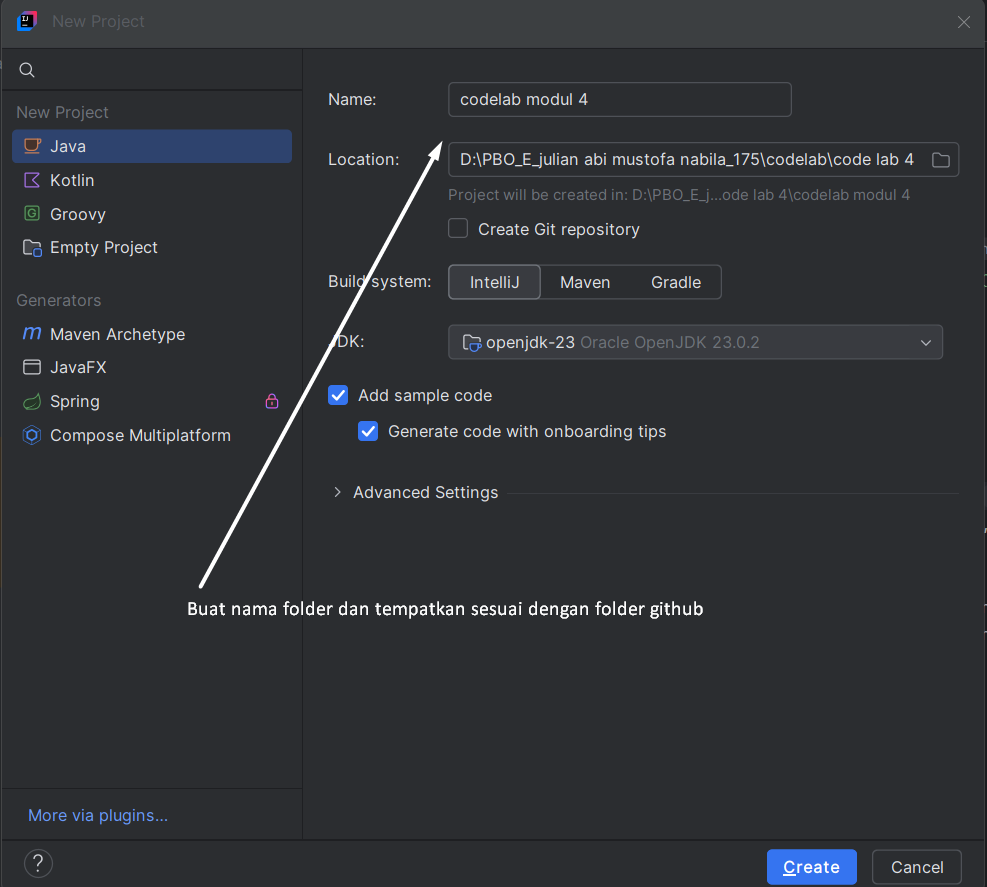


**LANGKAH-LANGKAH:**

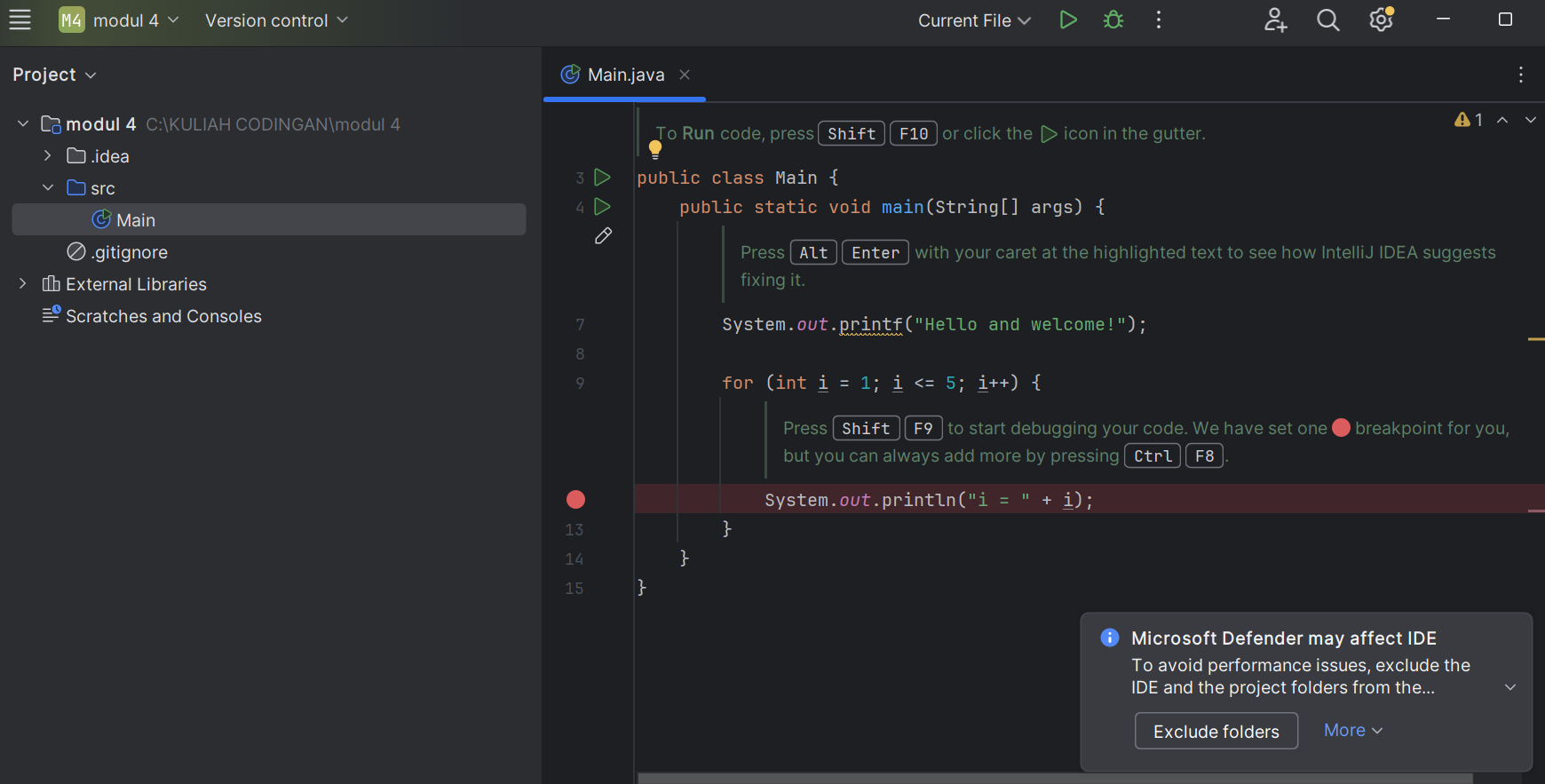
1. **Membuat *New Project***
   * Langkah pertama yang harus di lakukan ialah membuat *Project* Baru. Caranya dengan menekan tombol *New Project* seperti contoh gambar di bawah ini:



* + Setelah itu, muncul *Popup* silakan menamai *project* anda di **Name: [1]**, menyimpan project anda di **Location: [2]**, untuk penyimpanan project silakan ikuti contoh yang ada di Modul 1**,** lalu tekan **Create [3]**:

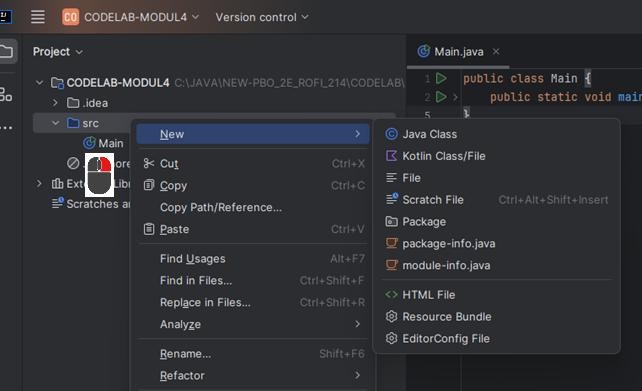


* + - Tampilan setelah membuat *project* baru:



# 2. Membuat Package dan Class

* + untuk membuat *Package* klik kanan pada *folder* **src [1]***,* klik **New [2],** lalu klik lagi **package [3]**. Nanti muncul *popup* untuk mengisi nama *package* tersebut. Silakan namai *package* tesebut sesuai yang ada di dalam modul:

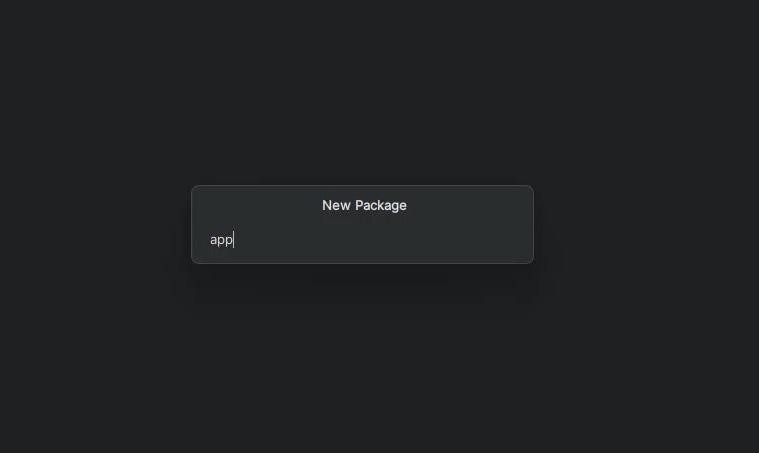


1

2

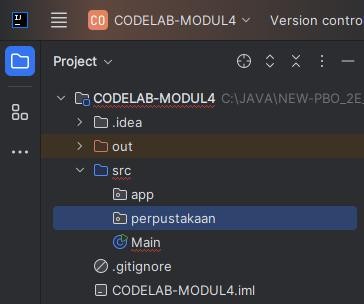
3

* + Namain *Package* tersebut



Untuk nama *Package*, Silakan sesuaikan yang ada di modul.

* + *Package* **app** telah dibuat, ulangi proses ini untuk membuat *Package* **Perpustakaan**

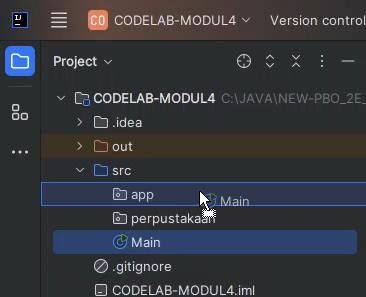


*Package* yang tadi di buat

Ulangi proses pembuatan *Package*

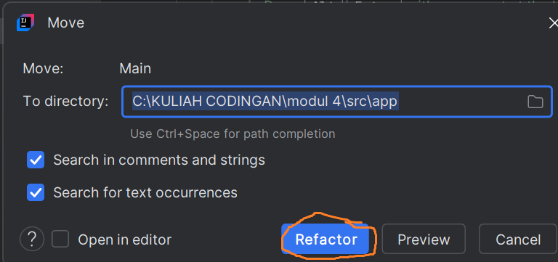
* + Sebelum membuat *Class*, *Class* **Main** yang sudah ada dari awal**,** dipindahkan dulu ke *Package* **app**, dengan cara *drag and drop Class* **Main** ke *Package* **app**, nanti muncul *popup* **Move**, lalu klik

**Refactor**, *Class* **Main** sudah di *Package* **app** seperti gambar di bawah ini :

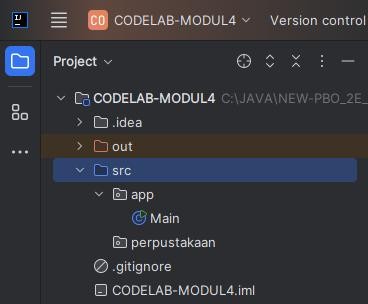


Drag **->**

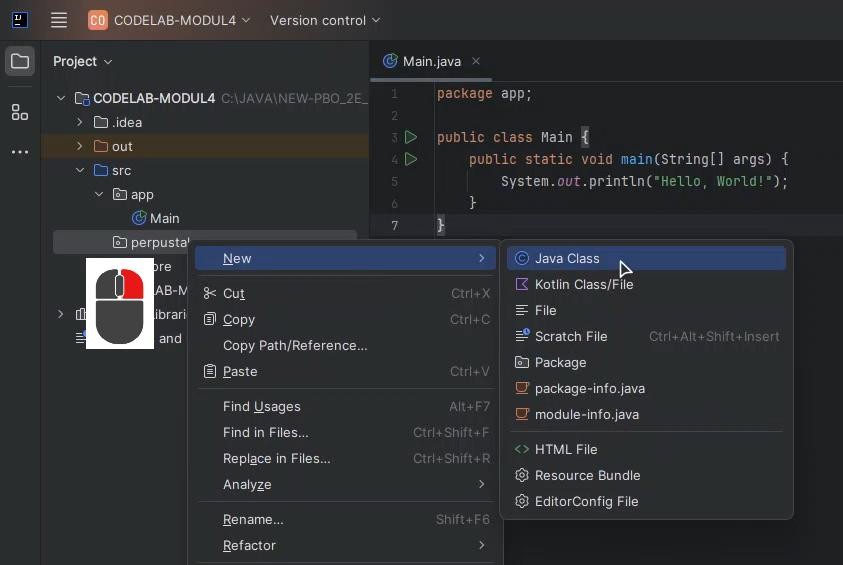
* + Klik Refactor



* + *Class* **Main** sudah di *package* **app**:



* + Untuk membuat *class*, **klik kanan** pada *package* yang mau di isi class **[1]**, misalnya pada *package* **perpustakaan**, klik **New [2]**, klik **Java Class [3],** muncul popup lalu pilih **Class [4]** dan namai *class* tersebut sesuai yang ada di modul, lalu tekan **Enter**:



1

3

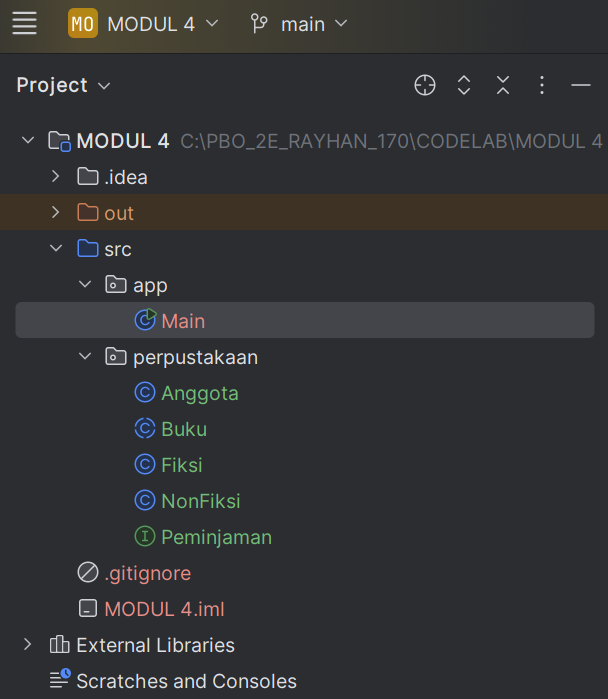
2

* + Lalu tekan **ENTER**

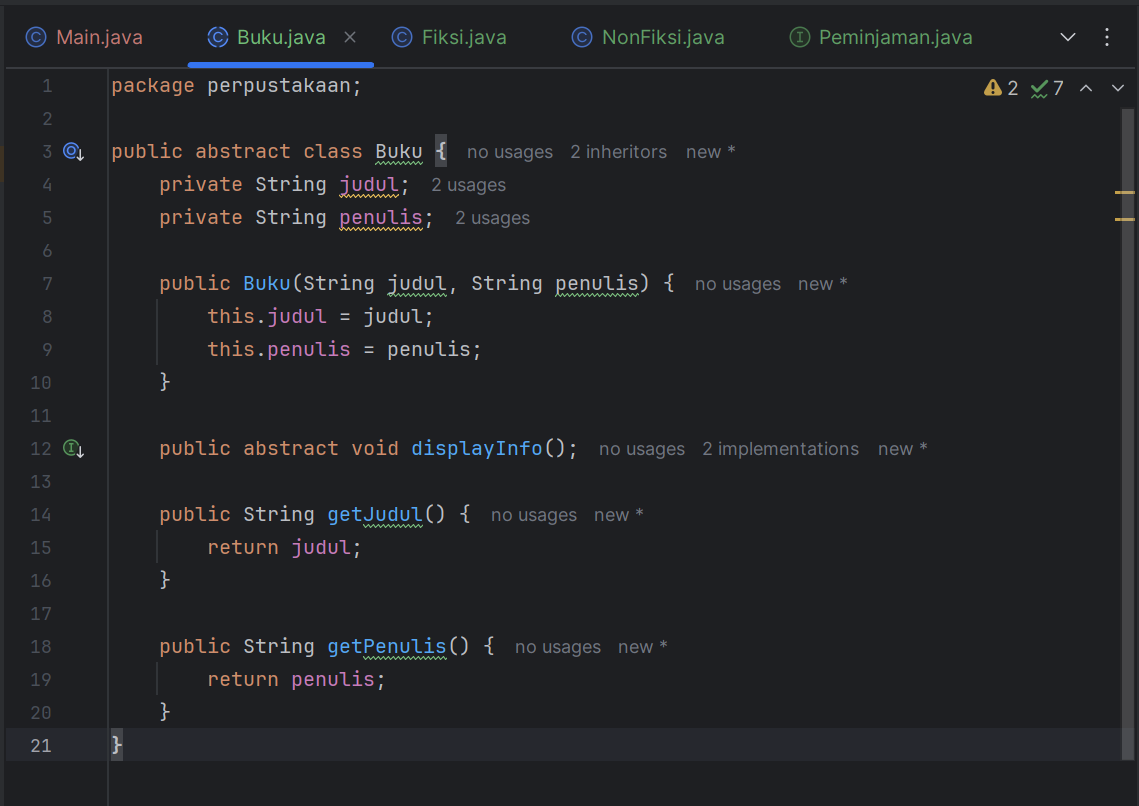


3

* + Ulangi Langkah tadi sampai seperti gambar dibawah ini:



1. Implementasi codelab
2. Class buku



* 1. Buat kelas Buku dengan dilengkapi atribut judul dan juga penulis.
  2. Tambahkan juga method abstrak displayInfo.

1. Subclass fiksi dan nonfiksi
   1. Buat kelas Fiksi dan NonFiksi yang mewarisi Buku.
   2. Tambahkan atribut khusus:

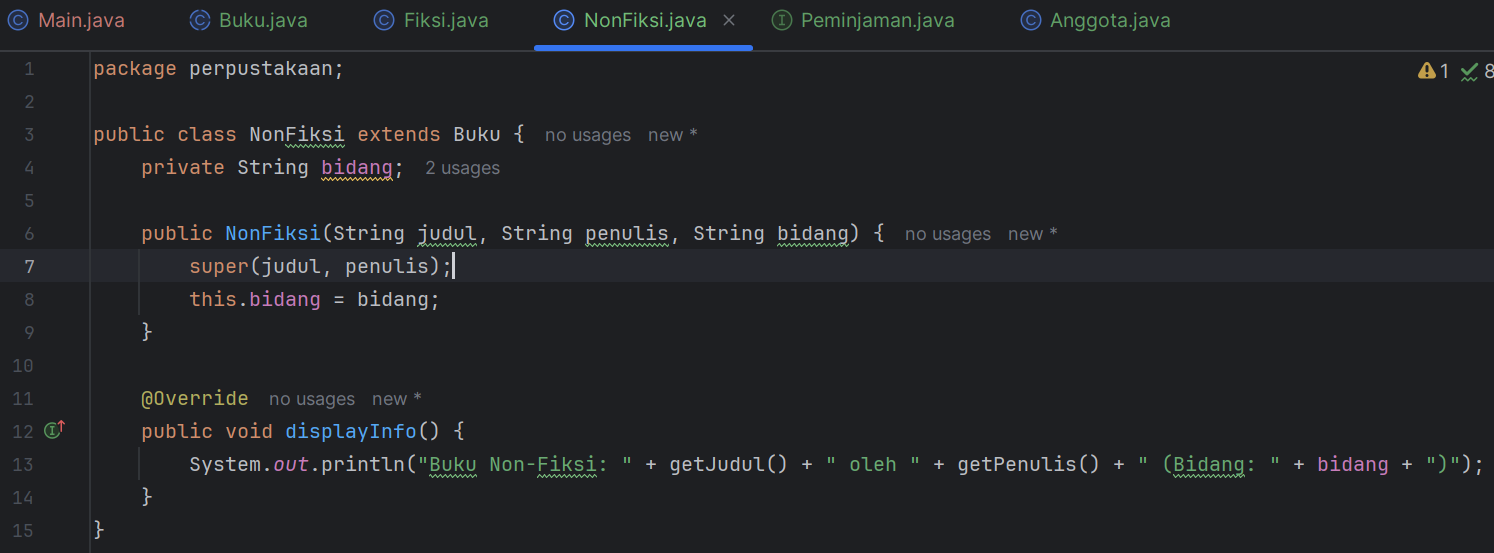
Genre dan NonFiksi

* 1. Override method displayInfo() guna menampilkan info spesifik.

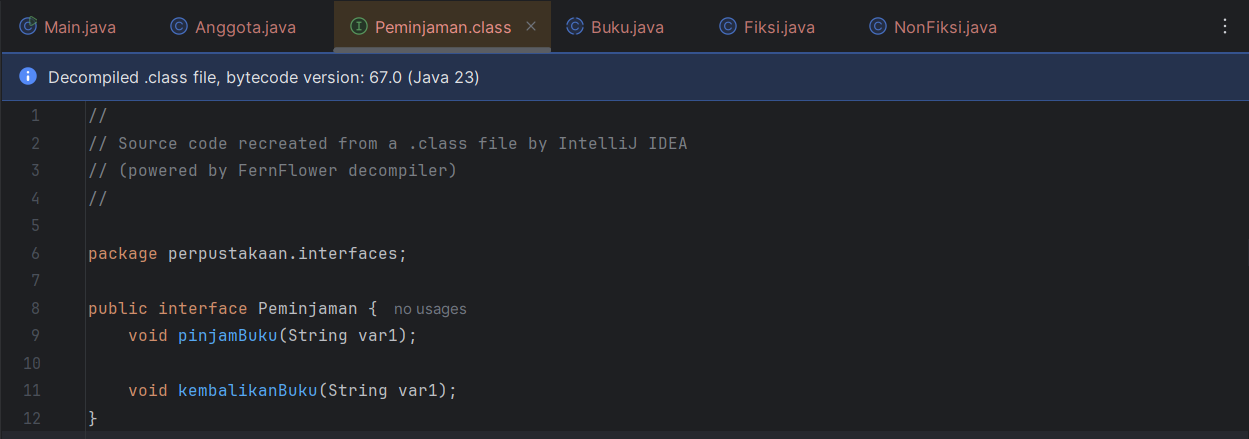
1. Fiksi

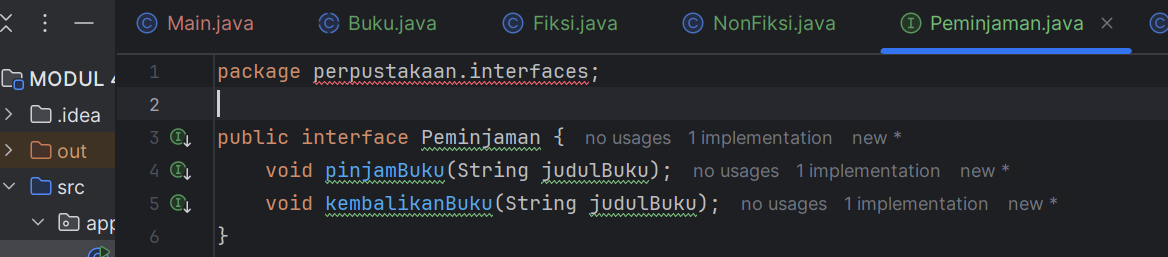


1. nonfiksi



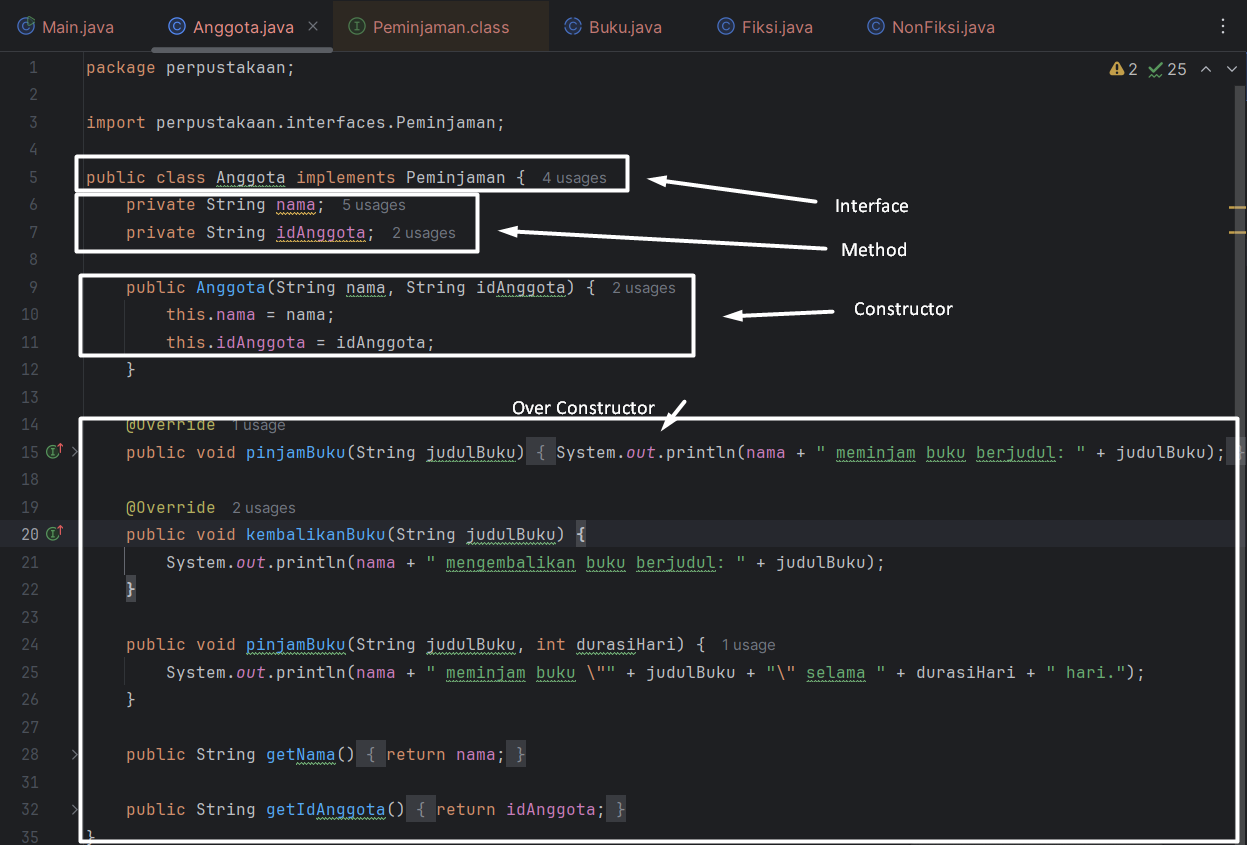
1. *Class* **peminjaman** jadikan *interface*



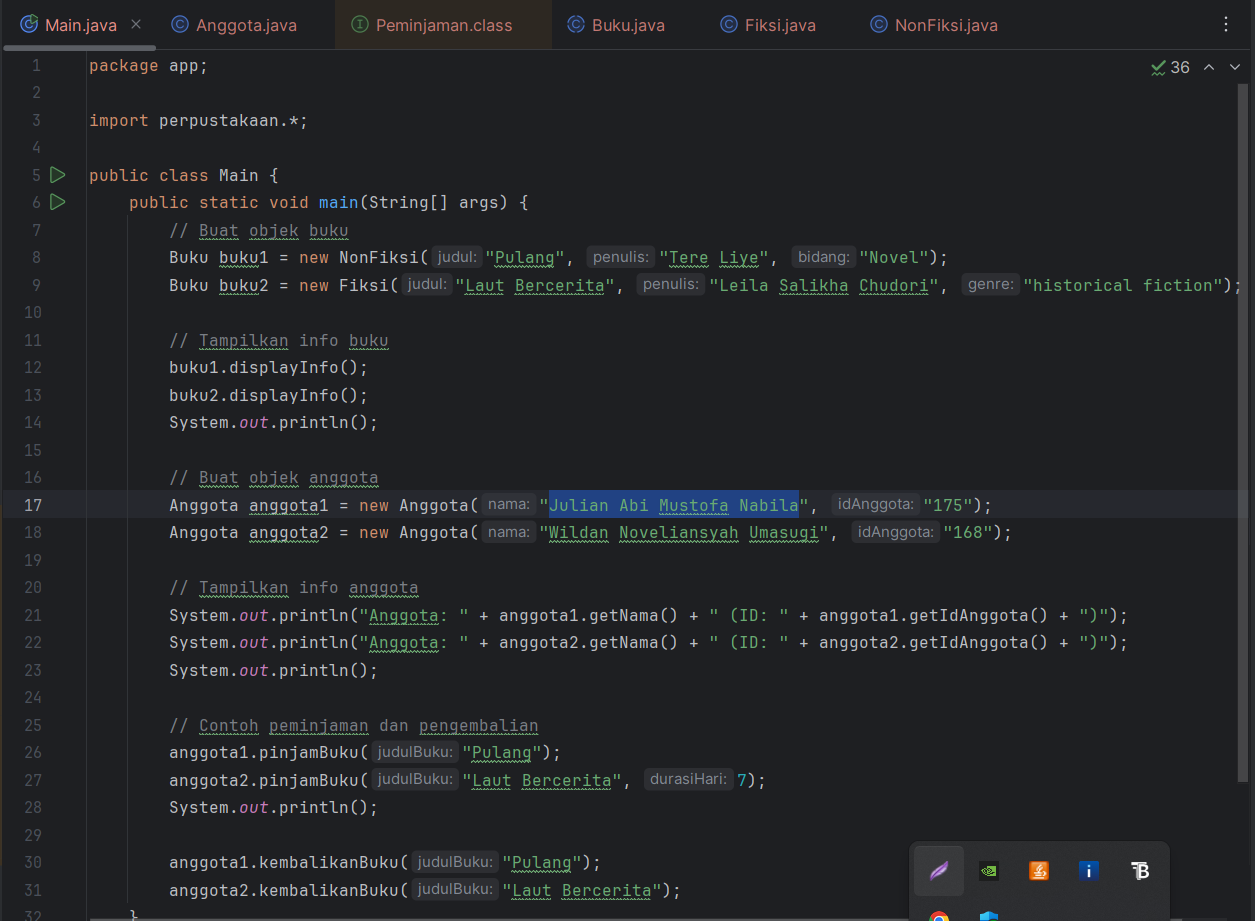
1.  Kata class disini, diganti menjadi kata interface. Gunanya untuk menentukan operasi wajib untuk proses di peminjaman buku.

Method->

1. Buat method **pinjamBuku()** dan **kembalikanBuku()**.
2. Class Anggota menggunakan interface Peminjaman
3. Gunakan interface **Peminjaman**
4. Tambahkan atribut **nama** dan **idAnggota**.
5. Buat **constructor** untuk mengisi nilai atribut
6. Buat ulang (override) method dari interface dengan logika meminjam dan mengembalikan buku.



1. Di dalam **package app**, buat **class Main** untuk menjalankan semua bagian dari **package perpustakaan**



1. Gunakan import perpustakaan.\*; untuk mengambil semua isi dari package **perpustakaan**.
2. Buat objek untuk **buku**, **anggota**, **peminjaman**, dan
3. pengembalian Panggil method untuk menampilkan hasil/output.

# PENJELASAN:

| **Kategori** | **Subjek** | **Penjelasan** | **Implementasi** |
| --- | --- | --- | --- |
| **Jenis Entitas** | Prime | Entitas mandiri, tidak tergantung entitas lain | class Buku (abstract) |
|  | Dependent | Bergantung pada entitas lain | Fiksi dan NonFiksi mewarisi Buku |
| **Jenis Atribut** | Tunggal | Tidak bisa dipecah lagi | judul, penulis di class Buku |
|  | Nilai Tunggal | Menyimpan satu nilai per entitas | genre (Fiksi), bidang (NonFiksi) |
|  | Mandatory | Wajib diisi | judul, penulis dalam constructor Buku |
| **Relasi OOP** | Inheritance | Pewarisan dari parent class ke subclass | Fiksi dan NonFiksi mewarisi Buku |
|  | Interface | Kontrak method wajib dibuat di class yang mengimplementasikan | Anggota mengimplementasikan Peminjaman |
|  | One-to-Many | Satu entitas terhubung ke banyak entitas lain | Buku ke Fiksi dan NonFiksi |
| **Atribut Spesifik** | Buku - judul | Judul buku, wajib diisi | String, Mandatory |
|  | Buku - penulis | Nama penulis, wajib diisi | String, Mandatory |
|  | Fiksi - genre | Jenis genre, wajib diisi | String, Mandatory |
|  | NonFiksi - bidang | Bidang buku, wajib diisi | String, Mandatory |
|  | Anggota - idAnggota | ID anggota (format: Huruf + 3 digit NIM), wajib diisi | String, Mandatory |
| **OOP Concepts** | Abstraction | Menyembunyikan detail dengan abstract class | abstract class Buku dengan method displayInfo() |
|  | Polymorphism | Method yang sama dengan perilaku berbeda (overloading dan overriding) | displayInfo() di Fiksi/NonFiksi, pinjamBuku() di Anggota |
|  | Interface | Method yang harus dibuat dalam class | Interface Peminjaman dengan pinjamBuku() dan kembalikanBuku() |
| **Relasi & Kardinalitas** | Buku - Fiksi | Pewarisan (Inheritance), 1 buku punya banyak jenis | One-to-Many |
|  | Anggota - Peminjaman | Interface, satu anggota gunakan satu interface | One-to-One |