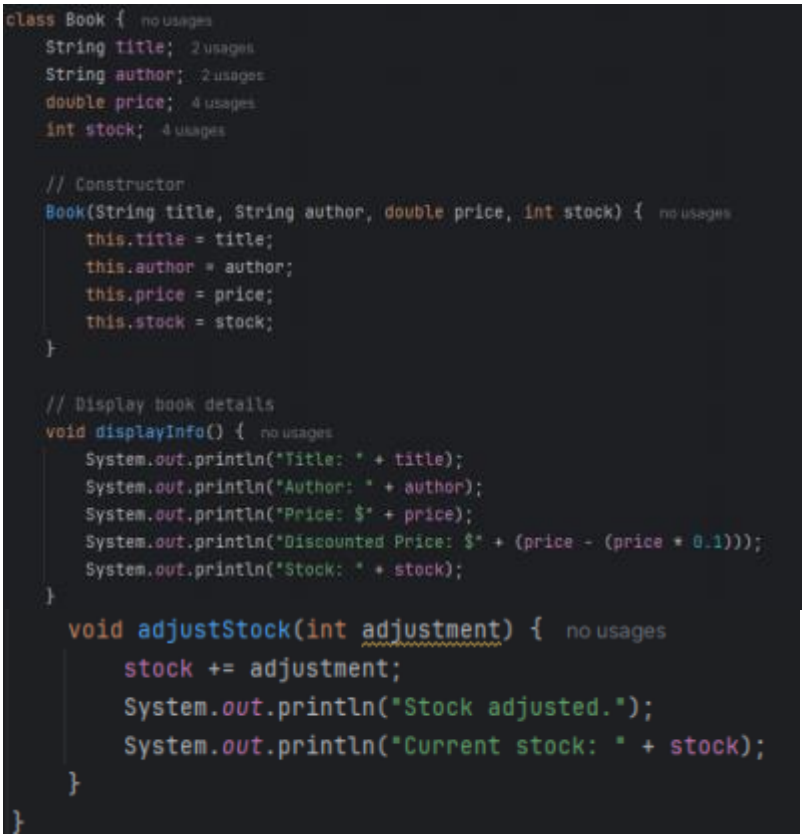
LAPORAN PRATIKUM CODELAB PEMORGRAMAN LANJUT 3H
MODUL 2

Nama : Yaasmin Ramadhani
NIM : 202410370110197
Kelas : Pemrograman lanjut 3H

CODELAB



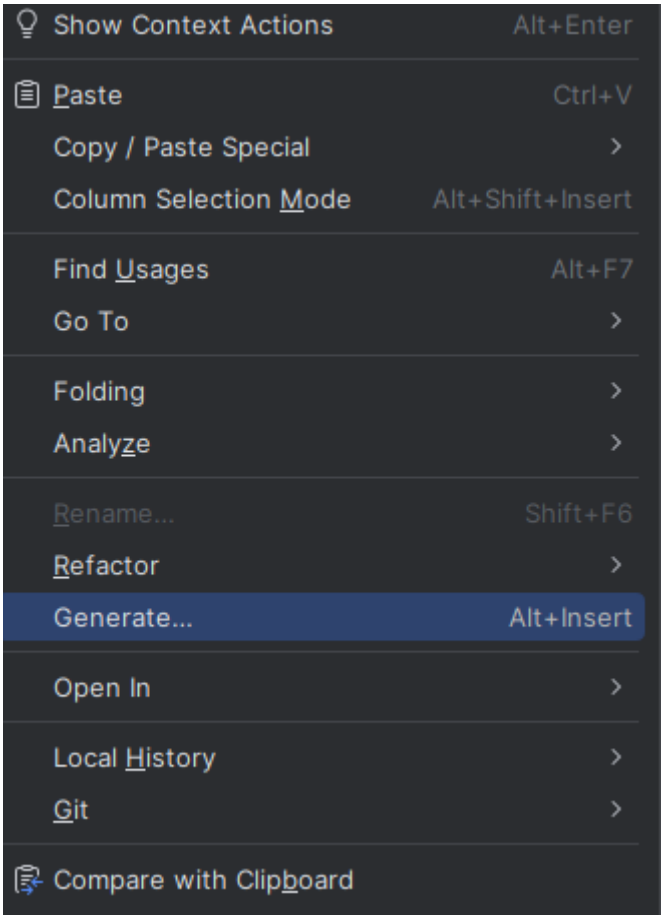
```
class Book {  no usages
    String title;  2 usages
    String author;  2 usages
    double price;  4 usages
    int stock;  4 usages

    // Constructor
    Book(String title, String author, double price, int stock) {  no usages
        this.title = title;
        this.author = author;
        this.price = price;
        this.stock = stock;
    }

    // Display book details
    void displayInfo() {  no usages
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Price: $" + price);
        System.out.println("Discounted Price: $" + (price - (price * 0.1)));
        System.out.println("Stock: " + stock);
    }
    void adjustStock(int adjustment) {  no usages
        stock += adjustment;
        System.out.println("Stock adjusted.");
        System.out.println("Current stock: " + stock);
    }
}
```

```java
class Library {  2 usages  new *
    public Book book;  2 usages
    public String location;  2 usages

    public Library(Book book, String location) {  1 usage  new *
        this.book = book;
        this.location = location;
    }

    // Display library and book information
    public void showLibraryInfo() {  2 usages  new *
        System.out.println("Library Location: " + location);
        book.displayInfo();
    }
}
```

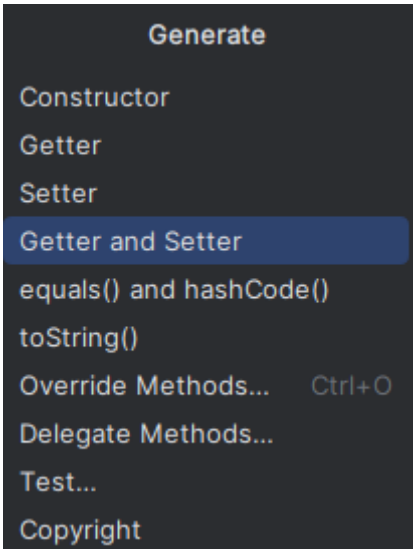
```java
class MainApp {  new *
    public static void main(String[] args) {  new *
        Book book1 = new Book( title: "Harry Potter",  author: "J.K. Rowling",  price: 10,  stock: 2);
        Library lib = new Library(book1,  location: "Perpustakaan Kota");

        // Display initial information
        lib.showLibraryInfo();

        // Add more stock
        book1.adjustStock( adjustment: 5);

        // Display updated information
        lib.showLibraryInfo();
    }
}
```

Program ini digunakan untuk mengelola data buku dan perpustakaan. Namun, ada beberapa bagian kode yang masih perlu diperbaiki supaya lebih jelas dan mudah dipahami. Berikut rencana refactoring yang akan dilakukan :

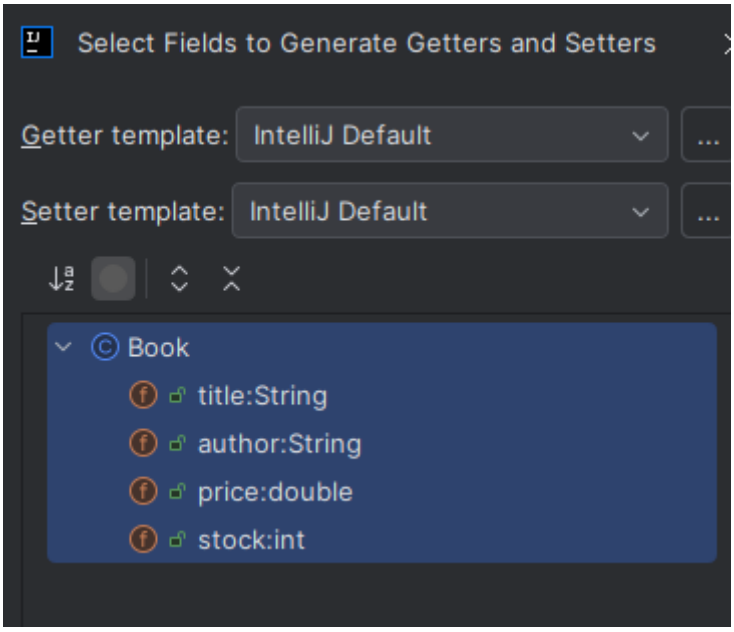
1. Pada class Book, tambahkan setter dan getter untuk field title, author, stock, dan price. Selain itu, buat juga setter untuk field book dan location pada Class Library. (**Clue: Encapsulate Field).**
   Langkah-langkah:
   a. Klik kanan pada salah satu baris, lalu pilih menu Generate.



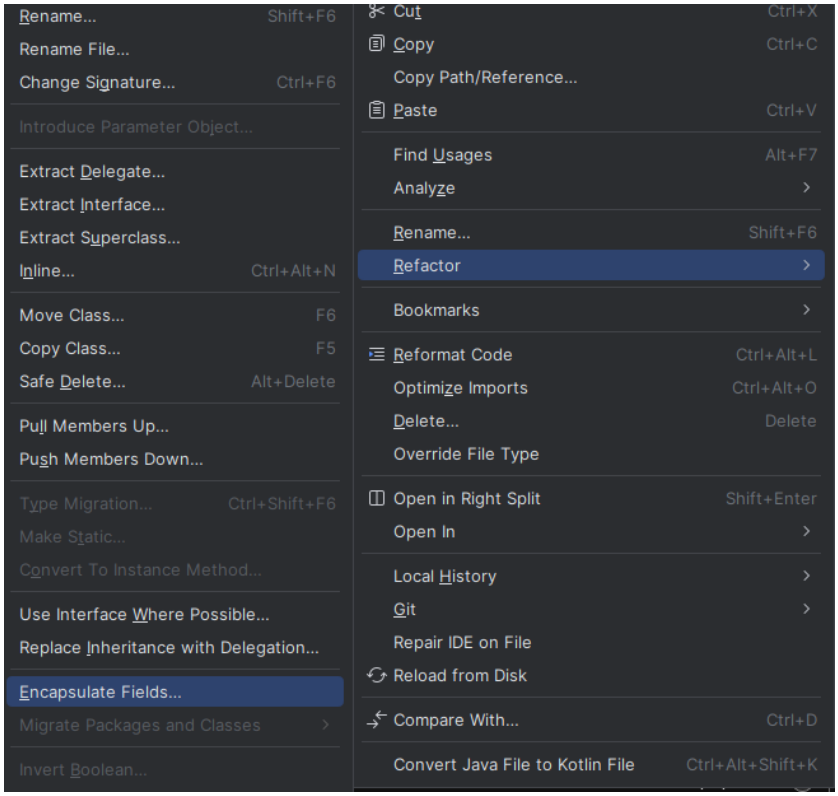
b. Kemudian pilih menu Getter dan Setter



c. Pilih variabel yang akan dibuat dengan metode Getter dan Setter, kemudian tekan OK



d. Pada class Book klik kanan,lalu pilih Refactor kemudian pilih lagi opsi Encapsulate Fields



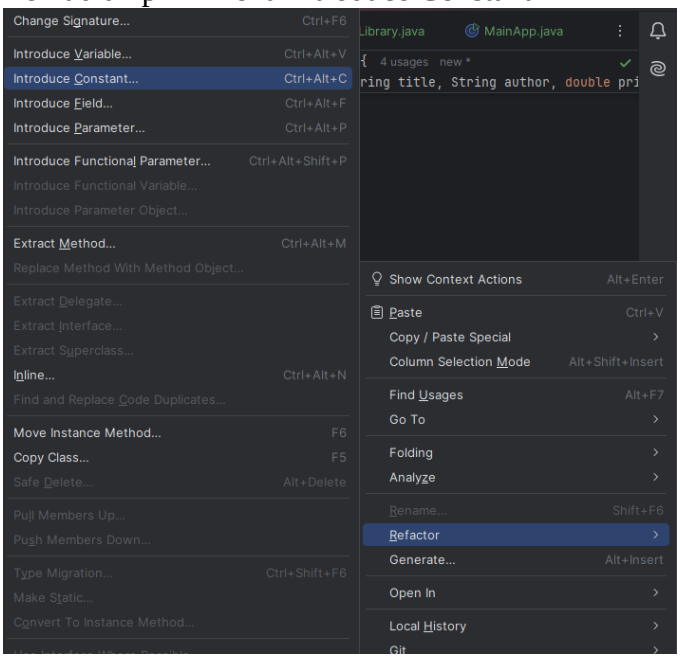
e. Pilih semua variable lalu pilih private pada menu Encapsulate Fields lalu klik Refactor untuk menjalankan

BEFORE

AFTER

```
private String title;  2 usages
private String author;  2 usages
private double price;  2 usages
private int stock;  2 usages
```

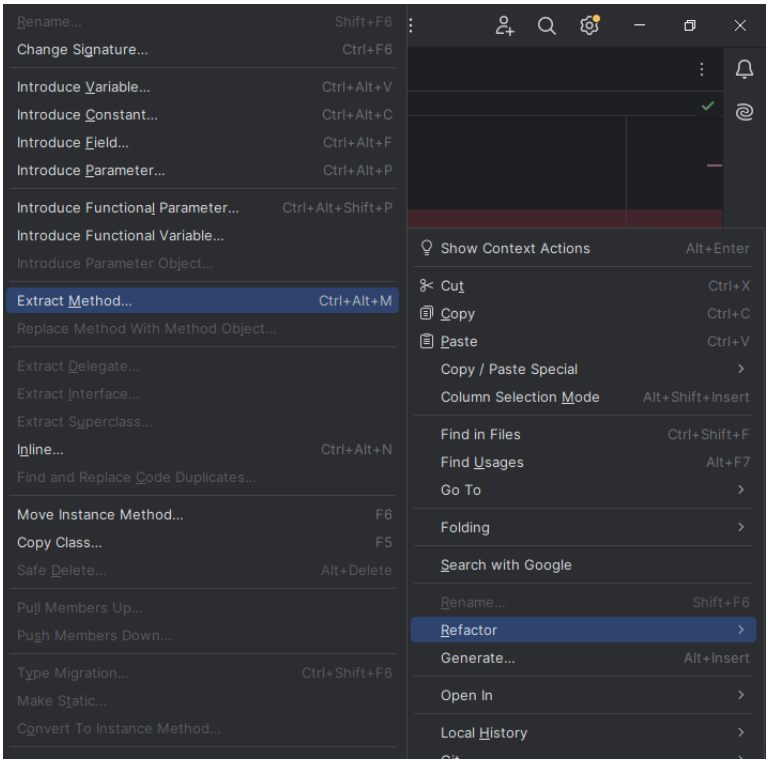
2. Perkenalkan sebuah konstanta baru di Class Book untuk menyimpan nilai diskon (misalnya DISCOUNT_RATE = 0.1). (Clue: **Introduce Constant**)
   Langkah- langkah :
   a. Klik kanan pada Discount price tepatnya pada nilai 0.1
   b. Pilih refactor kemudian pilih menu Introduce Constant



   c. Berikan nama Variableny DISCOUNT_RATE

```
public static final double DISCOUNT_RATE = 0.1;
private String title;  2 usages
```

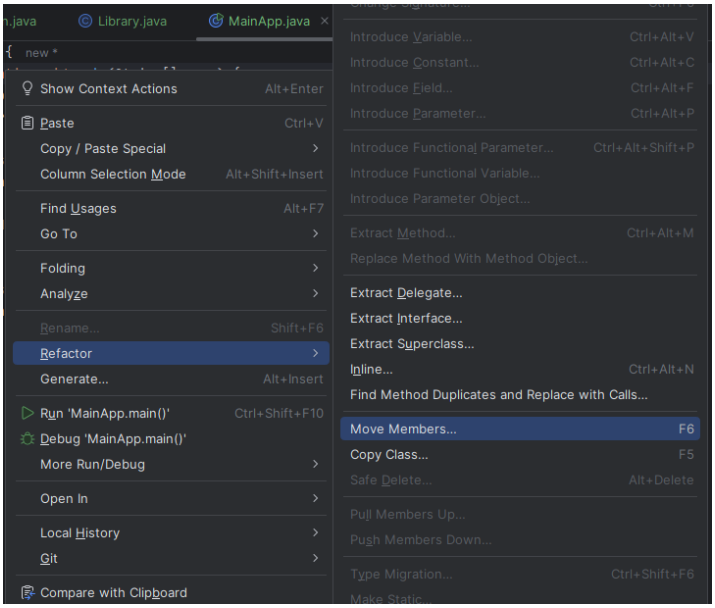
3. Pisahkan perhitungan harga diskon dari displayInfo () dan ubah bagian tersebut menjadi metode baru di dalam class Book dengan nama calculateDiscount (). **(clue: Extract Method).**
   Langkah – langkah:
   a. Pilih kode yang akan dipisahkan lalu blok kemudian klik kanan,lalu pilih Refactor kemudian pilih menu Extract Method



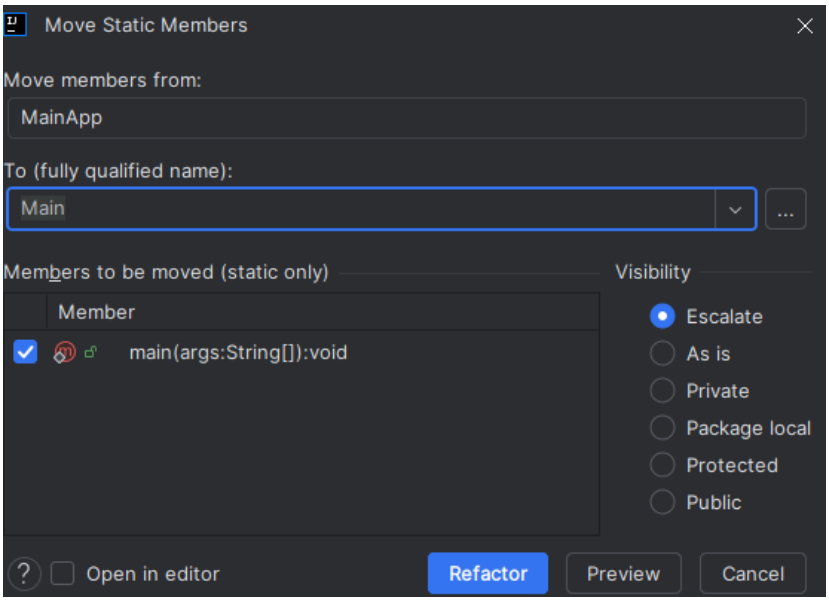
b. Beri nama methodnya calculateDiscount.

```
private double calculateDiscount() {  1 usage  new *
    return getPrice() - (getPrice() * DISCOUNT_RATE);
}
```

4. Pindahkan method main() dari class MainApp ke kelas baru bernama Main (yang telah kita buat), lalu hapus kelas MainApp setelah dipindahkan. Tujuannya untuk menata program agar method main() yang merupakan titik awal dari eksekusi sebuah program ditempatkan di class Main sehingga kode menjadi lebih terorganisir dan mudah dipahami.
   Langkah – langkah :
   a. Pilih method yang akan dipindahkan (class main()) lalu klik kanan stelah itu pilih Refactor lalu Move Members

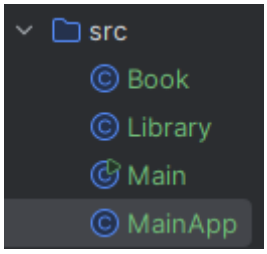


b. Tentukan kelas tujuan dimana kita akan memindahkan method lalu klik Refactor

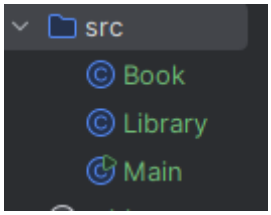


c. Yang terakhir hapus class MainApp

BEFORE



AFTER


src
Book
Library
Main