Laporan Pengerjaan Praktikum Teknik Pemrogaman
Pertemuan 7

Oleh:

Nama : Muhammad Nabil Syauqi Rasyiq
NIM : 241524018
Kelas : 1A

Sarjana Terapan Program Studi Teknik Informatika
Jurusan Teknik Komputer dan Informatika
Politeknik Negeri Bandung
2025

DAFTAR ISI

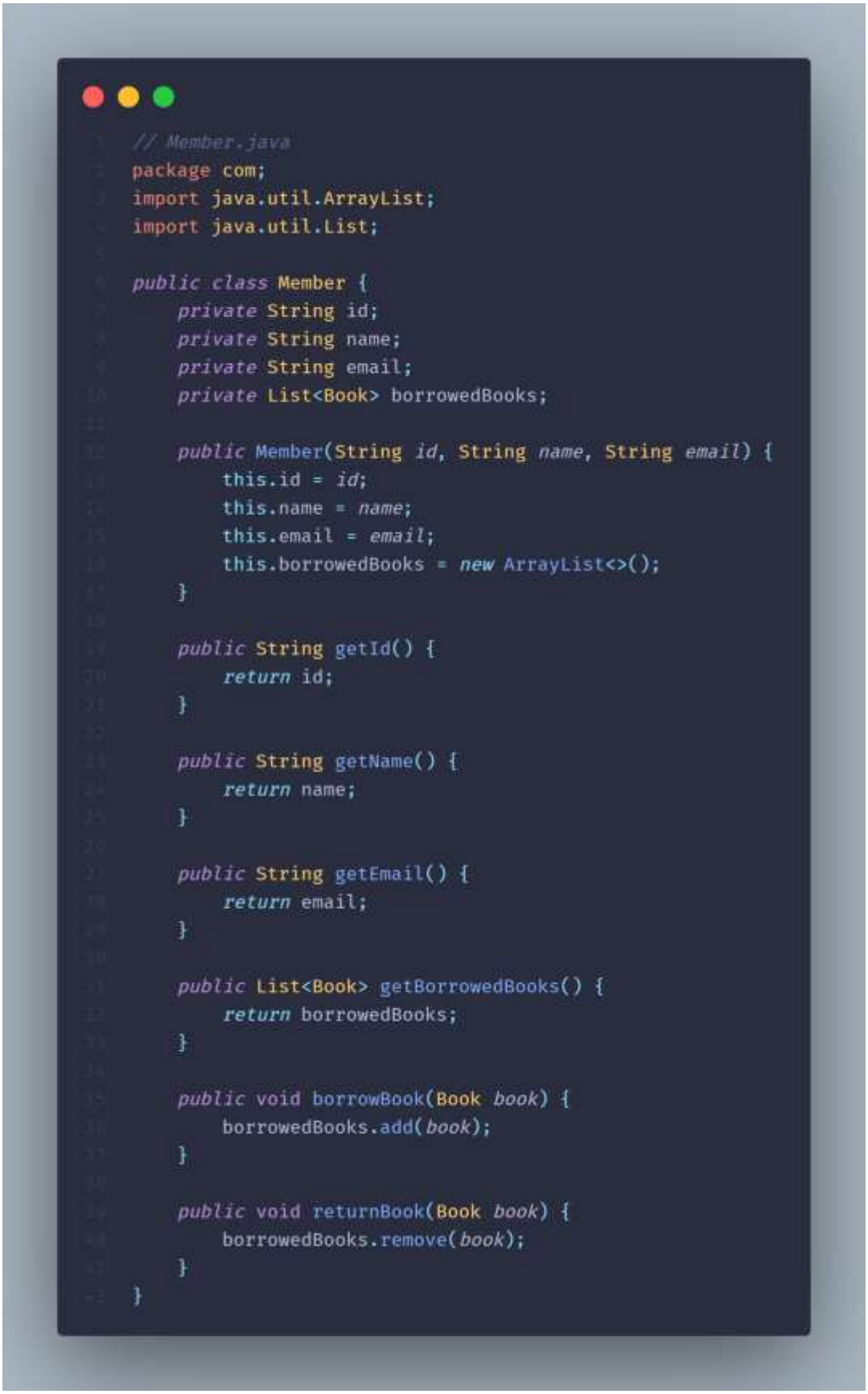
SOURCE CODE

Source code : https://github.com/Rasyiq603011/Uji-Coba-Penggunaan-Matriks

```java
// Book.java
package com;
public class Book {
    private String id;
    private String title;
    private String author;
    private boolean available;

    public Book(String id, String title, String author) {
        this.id = id;
        this.title = title;
        this.author = author;
        this.available = true;
    }

    public String getId() {
        return id;
    }

    public String getTitle() {
        return title;
    }

    public String getAuthor() {
        return author;
    }

    public boolean isAvailable() {
        return available;
    }

    public void setAvailable(boolean available) {
        this.available = available;
    }
}
```

```java
// Library.java
package com;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class Library {
    private List<Book> books;
    private Map<String, Member> members;

    public Library() {
        books = new ArrayList<>();
        members = new HashMap<>();
    }

    public void addBook(Book book) {
        books.add(book);
    }

    public void registerMember(Member member) {
        members.put(member.getId(), member);
    }

    public boolean borrowBook(String bookId, String memberId) {
        Book book = findBookById(bookId);
        Member member = members.get(memberId);

        if (book != null && member != null && book.isAvailable()) {
            book.setAvailable(false);
            member.borrowBook(book);
            return true;
        }
        return false;
    }

    public boolean returnBook(String bookId, String memberId) {
        Book book = findBookById(bookId);
        Member member = members.get(memberId);

        if (book != null && member != null && !book.isAvailable()) {
            book.setAvailable(true);
            member.returnBook(book);
            return true;
        }
        return false;
    }

    public Book findBookById(String id) {
        for (Book book : books) {
            if (book.getId().equals(id)) {
                return book;
            }
        }
        return null;
    }

    public List<Book> getAllBooks() {
        return books;
    }

    public Map<String, Member> getAllMembers() {
        return members;
    }
}
```

```java
// LibraryReport.java
package com;

public class LibraryReport {
    private Library library;

    public LibraryReport(Library library) {
        this.library = library;
    }

    public void generateBorrowingReport() {
        System.out.println("=== BORROWING REPORT ===");
        for (Member member : library.getAllMembers().values()) {
            System.out.println("Member: " + member.getName());
            System.out.println("Borrowed books: " + member.getBorrowedBooks().size());
            for (Book book : member.getBorrowedBooks()) {
                System.out.println("- " + book.getTitle() + " by " + book.getAuthor());
            }
            System.out.println();
        }
    }

    public void generateBookInventoryReport() {
        int availableCount = 0;
        int borrowedCount = 0;

        System.out.println("=== BOOK INVENTORY REPORT ===");
        for (Book book : library.getAllBooks()) {
            if (book.isAvailable()) {
                availableCount++;
            } else {
                borrowedCount++;
            }
            System.out.println("- " + book.getTitle() + " by " + book.getAuthor() +
                               " [" + (book.isAvailable() ? "Available" : "Borrowed") + "]");
        }

        System.out.println("\nTotal books: " + (availableCount + borrowedCount));
        System.out.println("Available books: " + availableCount);
        System.out.println("Borrowed books: " + borrowedCount);
    }
}
```



```java
// Member.java
package com;
import java.util.ArrayList;
import java.util.List;

public class Member {
    private String id;
    private String name;
    private String email;
    private List<Book> borrowedBooks;

    public Member(String id, String name, String email) {
        this.id = id;
        this.name = name;
        this.email = email;
        this.borrowedBooks = new ArrayList<>();
    }

    public String getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public String getEmail() {
        return email;
    }

    public List<Book> getBorrowedBooks() {
        return borrowedBooks;
    }

    public void borrowBook(Book book) {
        borrowedBooks.add(book);
    }

    public void returnBook(Book book) {
        borrowedBooks.remove(book);
    }
}
```

```java
// Main.java
package com;

public class Main {
    public static void main(String[] args) {
        // Membuat buku
        Book book1 = new Book("B001", "Java Programming", "John Doe");
        Book book2 = new Book("B002", "Python Basics", "Jane Smith");

        // Membuat anggota
        Member member = new Member("M001", "Alice", "alice@example.com");

        // Membuat perpustakaan
        Library library = new Library();

        // Menambahkan buku ke perpustakaan
        library.addBook(book1);
        library.addBook(book2);

        // Mendaftarkan anggota
        library.registerMember(member);

        // Meminjam buku
        boolean borrowed = library.borrowBook("B001", "M001");
        System.out.println("Book borrowed: " + borrowed);

        // Mencari buku
        Book foundBook = library.findBookById("B002");
        if (foundBook != null) {
            System.out.println("Found book: " + foundBook.getTitle());
        }

        // Membuat laporan
        LibraryReport report = new LibraryReport(library);
        report.generateBorrowingReport();
    }
}
```

PERHITUNGAN CBO

Manual:

Perhitungan CBO dapat dilakukan dengan melihat berapa banyak suatu class bergantung pada class lain

Kelas	Class yang diakses	Tempat Akses	Perhitungan
Book.java	-	-	CBO = 0
Member.java	Book	Sebagai Objek dan parameter	CBO = 1
Library.java	Book	Sebagai Objek dan akses method	1 + 1 = 2 CBO =2
	Member	Sebagai Objek dan akses method	
LibraryReport.java	Library	Sebagai Objek dan akses method	1 + 1 + 1 = 3 CBO = 3
	Book	Sebagai Objek dan akses method	
	Member	Sebagai Objek dan akses method	
Main.java	Library	Sebagai Objek dan akses method	1 + 1 + 1 + 1 = 4 CBO = 4
	Book	Sebagai Objek dan akses method	
	Member	Sebagai Objek dan akses method	
	LibararyReport	Sebagai Objek dan akses method	

Nilai CBO:

Class Book: 0

Class Member: 1

Class Library: 2

Class LibraryReport: 3

Main: 4

Menggunakan CK metriks :

class	cbo	cboModified
com.Main	4	4
com.LibraryReport	3	4
com.Book	0	4
com.Library	3	6
com.Member	1	4

Penjelasan dan pemahaman

Pada Perhitungan CBO diatas terdapat perbedaan antara perhitungan manual dan perhitungan menggunakan CK metrics. Jika kita membandingkan bagaian colom CBO dengan perhitungan CBO secara manual terdapat perbedaan di bagian Library. Pada perhitungan manual library harusnya memiliki nilai CBO 2 sedangkan pada perhitungan menggunakan CK metrics CBOnya bernilai 3. Hal yang menurut saya dapat menyebabkan hal ini adalah karena library mengimport Map dari java util yang tidak dilakukan oleh class lain.

CBO modified kemungkinan besar merupakan perhitungan CBO yang lebih ketat dengan menambahkan kriteria tertentu, tetapi sejauh ini saya belum bisa mengetahui apa yang menjadi pertimbangan lebih.

PERHITUNGAN LCOM

Manual:

Perhitungan LCOM dapat dilakukan dengan menghitung method yang tidak mengakses atribut yang sama. Berikut perhitungan manual yang saya lakukan:

Kelas	Nama Method	Atribut yang diakses	Perhitungan
Book.java	getId()	Id	Total relation = 5 x 4 / 2 = 10 Jumlah relation denagn akses sama = 2 x 1 / 2 = 1 Jumlah relation dengan akses berbeda = 10 – 1 = 9 LCOM = 9 -1 =8
	getTitle()	Title	
	getAuthor()	Author	
	isAvailable()	Available	
	setAvailable()	Available	
Member.java	getId()	Id	Total relation = 6 x 5 / 2 = 15 Jumlah relation dengan akses sama = 3 x 2 / 2 = 3 Jumlah relation denagn akses berbeda = 15 – 3 = 12 LCOM = 12 – 3 = 9
	getName()	Name	
	getEmail()	Email	
	getBorrowedBook()	BorrowedBooks	
	borrowBook()	BorrowedBooks	
	returnBook()	BorrowedBooks	
Library.java	addBook()	Books	Total relation = 7 x 6 / 2 = 21 Jumlah relation dengan akses sama A = 5 x 4 / 2 = 10 B = 4 x 3 / 2 = 6 C = 2 x 1 / 2 = 1 A + B - C = 15 ***Penjelasan***
	registerMember()	Members	

	borrorwBook()	Books, Members	A adalah jumlah relation yang sama dari yang mengakses attribute books, borrow dan return masuk perhitungan B adalah jumlah relation yang sama dari yang mengakses attribute Members, borrow dan return masuk perhitungan C adalah jumlah relation antar return dan borrow untuk mencegah relation double pada perhitungan sebelumnya ***Penjelasan*** Jumlah relation dengan akses berbeda = 21 – 15 = 6 LCOM = 6 – 15 = -9
	returnBook()	Books, Members	
	findBookById()	Books	
	getAllBook()	Books	
	getAllMembers()	Members	
LibraryReport.java	generateBorrowingReport()	Library	Total relation = 2 x 1 / 2 = 1 Jumlah relation dengan akses sama = 2 x 1 / 2 = 1

	generateInventoryReport()	Library	Jumlah relation denagn akses berbeda = 1 – 1 = 0 LCOM = 0 – 1 = -1
Main.java	-	-	LCOM = 0

Hasil Perhitungan Manual LCOM:

Class Book: 8 (Low Cohesion)

Class Member: 9 (Low Cohesion)

Class Library: -9 (High Cohesion)

Class LibraryReport: -1 (High Cohesion)

Main: 0 (High Cohesion)

Menggunakan CK metriks:

class	lcom	lcom*
com.Main	0	0
com.LibraryReport	0	0
com.Book	3	0.62500006
com.Library	0	0.4375
com.Member	3	0.6428572

Penjelasan dan Pemahaman

Pada perhitungan LCOM diatas kita dapat melihat hasil yang kita dapat dari perhitunagn manual dan perhitungan menggunakan CK metrics memiliki perbedaan yang sangat jauh. Hal ini dapat terjadi karena bisa jadi terdapat perbedaan indicator perhitungan, pembulatau, bahkan sampai perbedaan metode perhitungan.

Jika kita menganggap hasil 0 dari hasil perhitungan menggunakan ck metrics sebagai jenis high cohesion maka hasilnya sesuai dengan manual yang juga memberikan hasil high cohesion, begitupun juga member dan book yang menunjukan low cohesion. Dari hal tersebut kita dapat

mengambil Kesimpulan bahwa apapun metode yang digunakan menghitung seharusnnya tidak memberikan perbedaan dari segi hasil Kesimpulan

LESSON LEARN

Dari praktikum kali ini saya belajar bagaimana cara menghitung sebuah indicator yang dapat menjadi sebuah pertimbangan dalam pembuatan program berorientasi objek. Aku mempelajari bagaimana menghitung Coupling Between Objects (CBO) dan Lack of Cohesion of Methods (LCOM). Saya mempelajari secara dasar bagaimana cara menghitungnya secara manual dan mengeksplorasi bagaiman cara menghitungnya menggunakan tools.

Pada praktikum kali ini juga aku mengeksplorasi banyak ha terkait perhitungan CBO dan LCOM menggunakan tolls. Pertama tama aku mencoba menggunakan sonar qube namun, dengan menggunakan versi community aku tidak mendapatkan hasil untuk CBO dan LCOM. Setelah tidak berhasil menggunakan sonarqube aku mencoba menggunakan plugins yang ada di eclips namun tidak membuahkan hasil yang baik. Terakhir aku mencoba menggunakan CK matrics, pada saat mencoba menggunakan CK metricks aku perlu banyak mengatur banyak hal. Seperti environment variabel, penyesuaian versi java dan sebagainya. Walaupun cukup ribet tapi [ada akhirnya aku berhasil menggunakan CK metrics.

Pelajaran yang aku dapat dari praktikum kali ini adalah pentinganya rapih di segala hal terutama pada bagian data. Manajemen data yang baik akan memudahkan kita untuk dapat mengolah informasi dengan lebih baik. Selain itu, ketenangan dalam menghadapi tantangan dan masalah juga menjadi hal yang penting dalam mendukung hasil pekerjaan kita.

Terakhir saya ingin mengucapkan terima kasih banyak kepada bapak Zulkifli Arsyad yang telah membrikan pengajaran kurang lebih 8 pertemuan kebelakang. Saya mohon maaf apabila pernah mengatakan atau berbuat hal yang tidak mengenakan, saya mohon keridhoan bapak untuk keberkahan ilmu saya. Minal aizin wal fa izin mohon maaf lahir dan batin.