ULAB
UNIVERSITY OF LIBERAL ARTS BANGLADESH

Course Code: CSE 2104
Course Title: Object-oriented Programming Lab
Lab Work 3

SUBMITTED TO :

Shakib Mahamud Dipto
CSE
lecturer
University of Liberal Arts

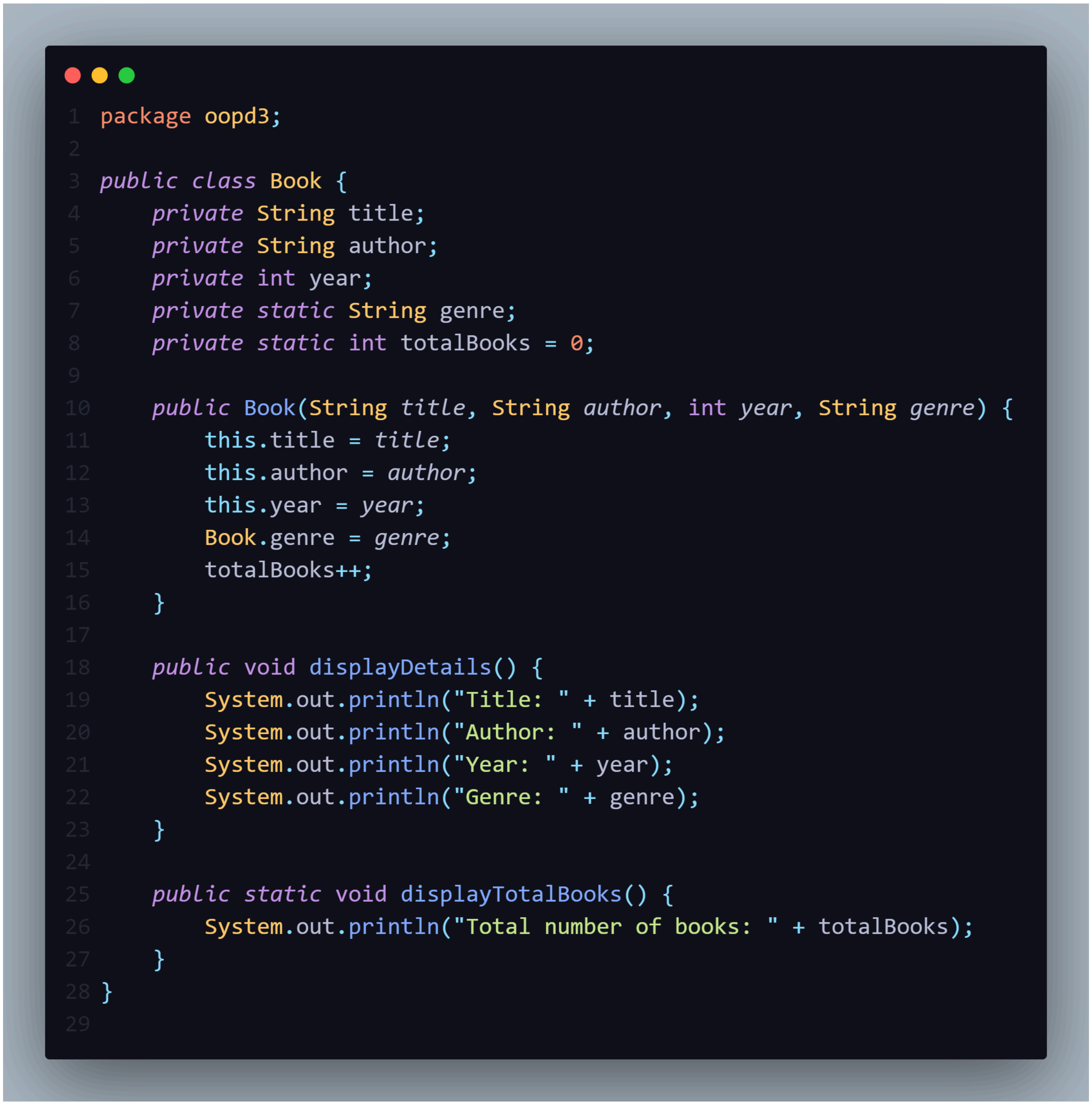
SUBMITTED BY :

Piyas sarkar
223014024
Section 4

Question1:

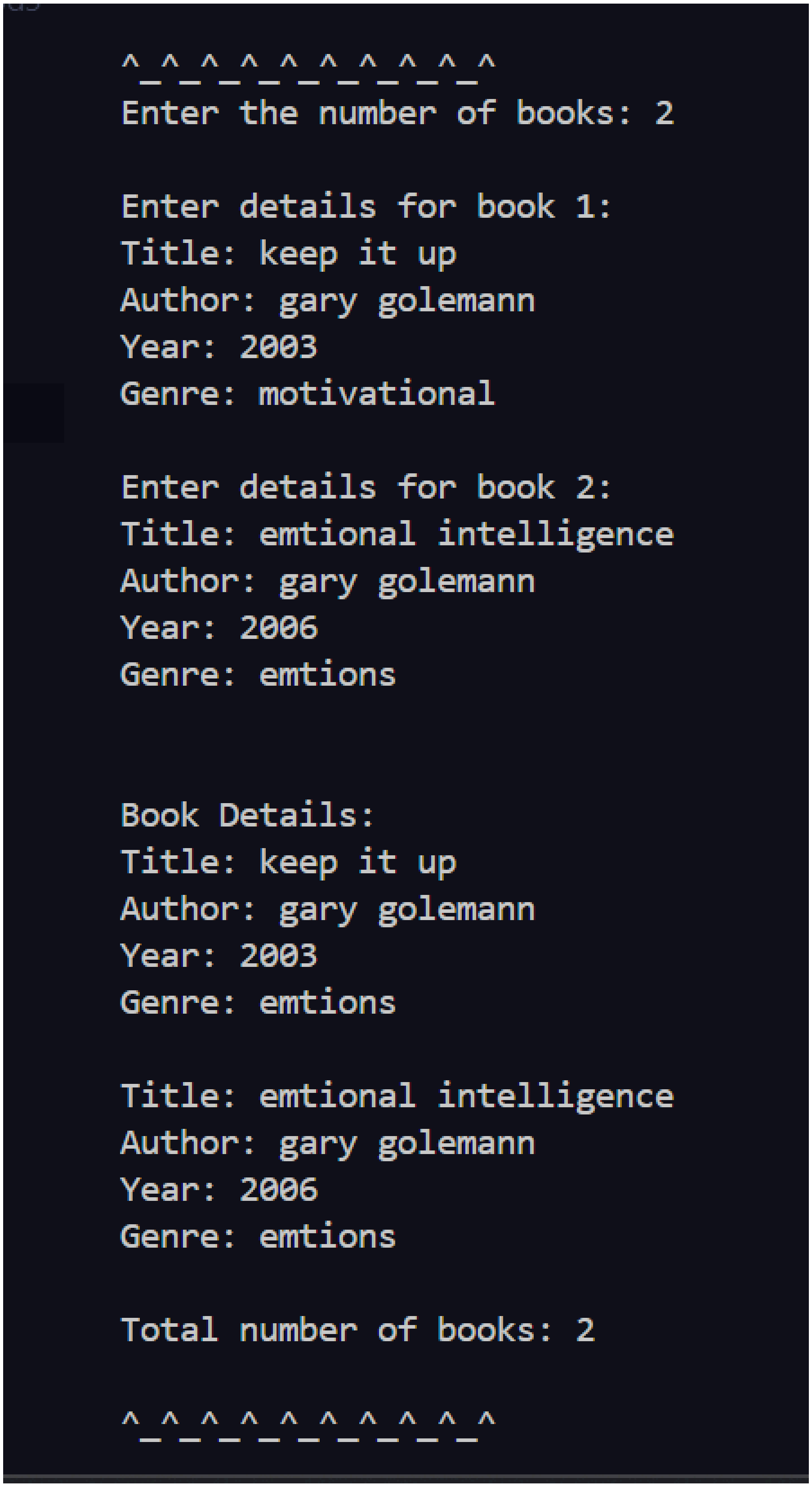
Create a Book class with properties such as title, author, year (object variables) and genre (class variable). Implement a parameterized constructor to initialize 3 objects (Get the object variables as input from the user). Include an object method to display the book details and a class method to display the total number of books.

Solution Code:



```java
package oopd3;

public class Book {
    private String title;
    private String author;
    private int year;
    private static String genre;
    private static int totalBooks = 0;

    public Book(String title, String author, int year, String genre) {
        this.title = title;
        this.author = author;
        this.year = year;
        Book.genre = genre;
        totalBooks++;
    }

    public void displayDetails() {
        System.out.println("Title: " + title);
        System.out.println("Author: " + author);
        System.out.println("Year: " + year);
        System.out.println("Genre: " + genre);
    }

    public static void displayTotalBooks() {
        System.out.println("Total number of books: " + totalBooks);
    }
}

```

Output:



```
^_^_^_^_^_^_^_^_^_^
Enter the number of books: 2

Enter details for book 1:
Title: keep it up
Author: gary golemann
Year: 2003
Genre: motivational

Enter details for book 2:
Title: emtional intelligence
Author: gary golemann
Year: 2006
Genre: emtions


Book Details:
Title: keep it up
Author: gary golemann
Year: 2003
Genre: emtions

Title: emtional intelligence
Author: gary golemann
Year: 2006
Genre: emtions

Total number of books: 2

^_^_^_^_^_^_^_^_^_^
```

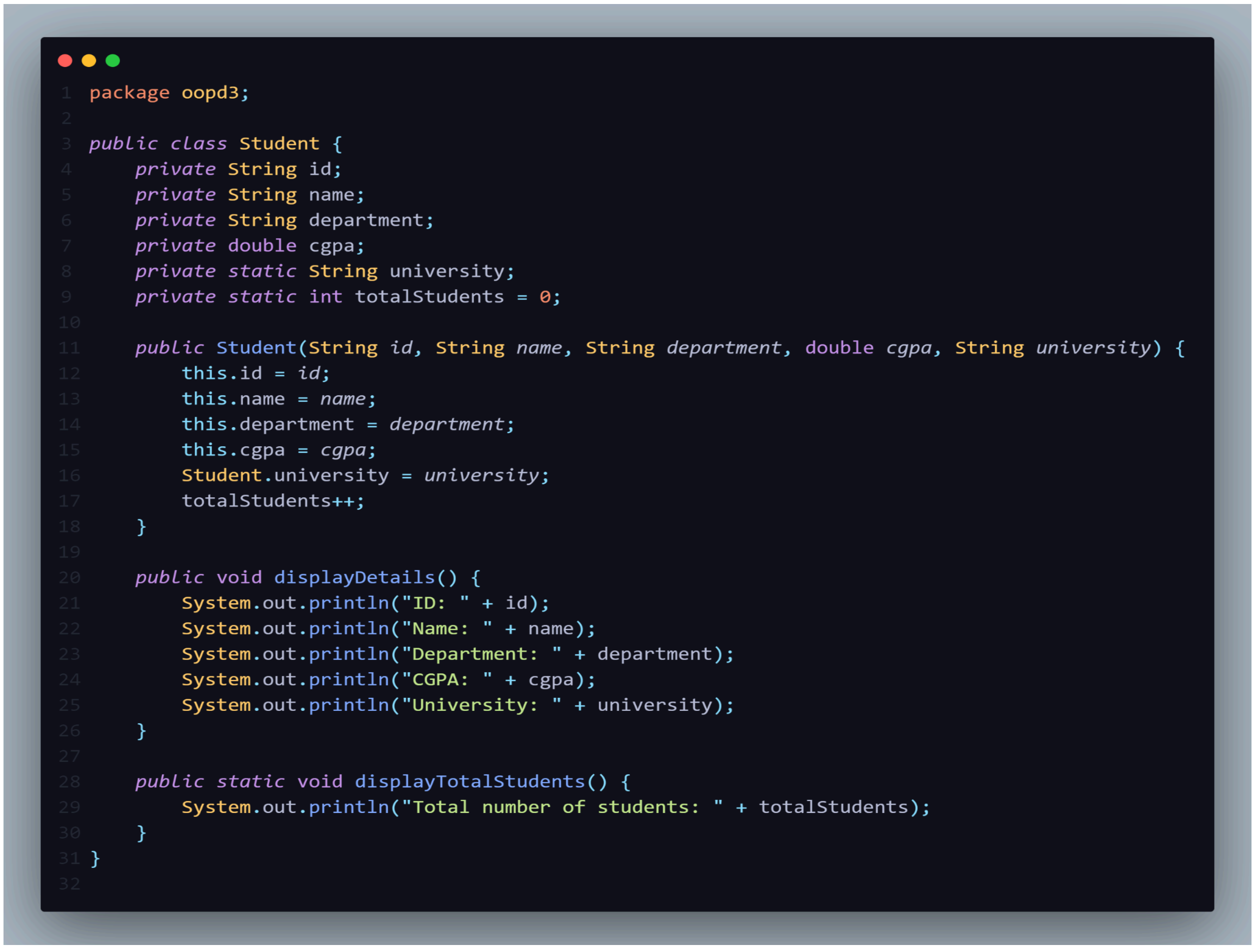
Explanation:

The Book class represents a book containing attributes like title, author, year, and genre. A parameterized constructor of the class is used to initialize these attributes for every book object. It is possible to build an array of Book objects and populate the details of each book with user input. The class offers two methods: displayTotalBooks, a static function that shows the total number of books generated, and displayDetails, which prints the information of a single book. Thanks to this framework, it is simple to monitor and show facts for a library or collection of books, which enables the effective and structured management of book information.

Question2:

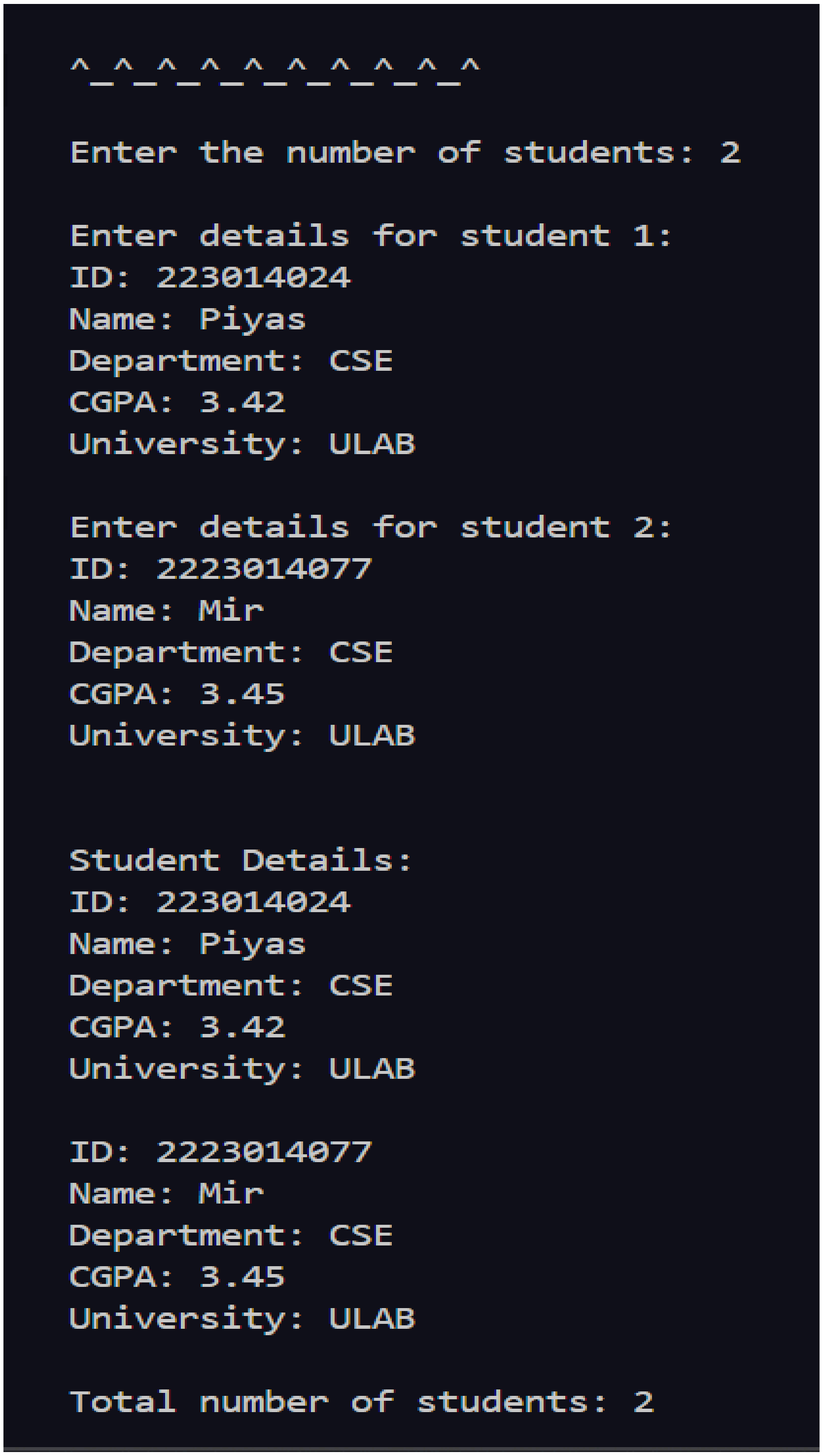
Create a Student class with properties such as id, name, department, CGPA (object variables), and university (class variable). Implement a parameterized constructor to initialize 3 objects (Get the object variables as input from the user). Include an object method to display the student details and a class method to display the total number of students.

Solution Code:



```java
package oopd3;

public class Student {
    private String id;
    private String name;
    private String department;
    private double cgpa;
    private static String university;
    private static int totalStudents = 0;

    public Student(String id, String name, String department, double cgpa, String university) {
        this.id = id;
        this.name = name;
        this.department = department;
        this.cgpa = cgpa;
        Student.university = university;
        totalStudents++;
    }

    public void displayDetails() {
        System.out.println("ID: " + id);
        System.out.println("Name: " + name);
        System.out.println("Department: " + department);
        System.out.println("CGPA: " + cgpa);
        System.out.println("University: " + university);
    }

    public static void displayTotalStudents() {
        System.out.println("Total number of students: " + totalStudents);
    }
}

```

Output:

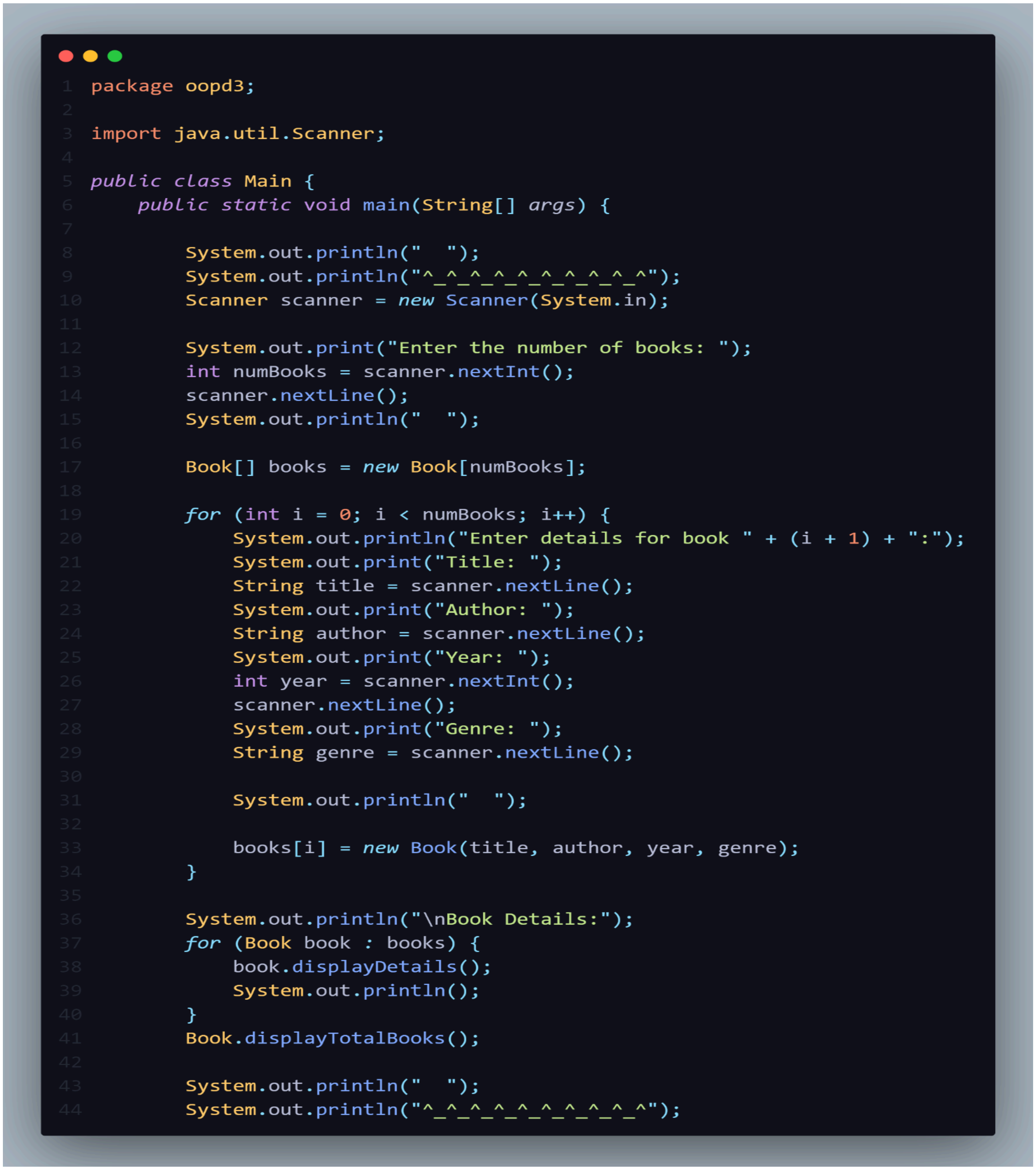


```
^_^_^_^_^_^_^_^_^_^

Enter the number of students: 2

Enter details for student 1:
ID: 223014024
Name: Piyas
Department: CSE
CGPA: 3.42
University: ULAB

Enter details for student 2:
ID: 2223014077
Name: Mir
Department: CSE
CGPA: 3.45
University: ULAB


Student Details:
ID: 223014024
Name: Piyas
Department: CSE
CGPA: 3.42
University: ULAB

ID: 2223014077
Name: Mir
Department: CSE
CGPA: 3.45
University: ULAB

Total number of students: 2
```

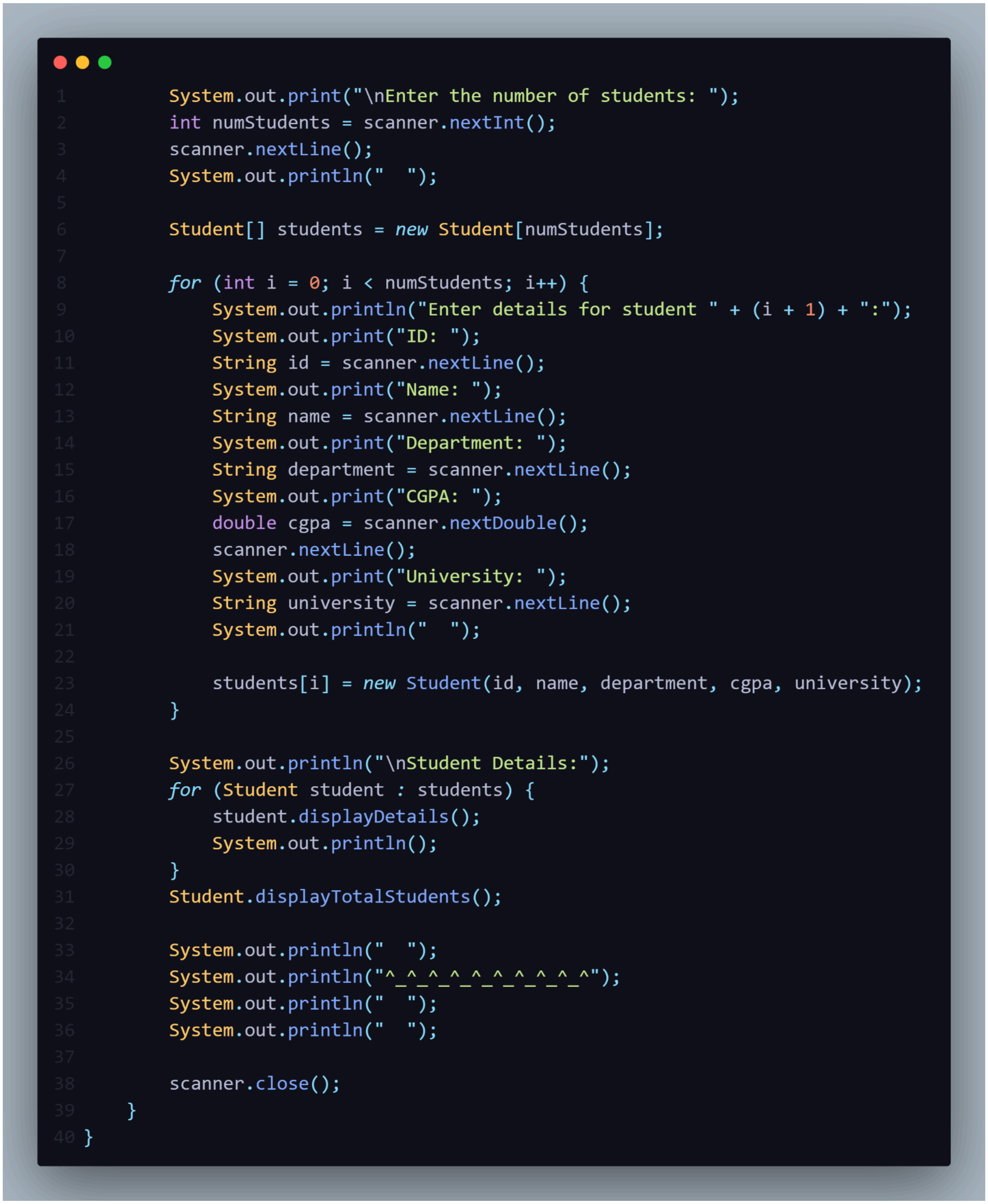
Explanation:

A student with attributes like name, department, CGPA, and university is modeled by the Student class. It has a parameterized constructor to initialize these attributes based on user input, just like the Book class. With the ability to store students in an array, the application can efficiently manage many student records. The class additionally has a static method displayTotalStudents to display the total number of student objects produced. Each student object has a displayDetails method to print its information. This configuration makes handling student data easier, making it appropriate for use in academic records or institution databases.

Main File:



```java
package oopd3;

import java.util.Scanner;

public class Main {
    public static void main(String[] args) {

        System.out.println("  ");
        System.out.println("^_^_^_^_^_^_^_^_^_^");
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of books: ");
        int numBooks = scanner.nextInt();
        scanner.nextLine();
        System.out.println("  ");

        Book[] books = new Book[numBooks];

        for (int i = 0; i < numBooks; i++) {
            System.out.println("Enter details for book " + (i + 1) + ":");
            System.out.print("Title: ");
            String title = scanner.nextLine();
            System.out.print("Author: ");
            String author = scanner.nextLine();
            System.out.print("Year: ");
            int year = scanner.nextInt();
            scanner.nextLine();
            System.out.print("Genre: ");
            String genre = scanner.nextLine();

            System.out.println("  ");

            books[i] = new Book(title, author, year, genre);
        }

        System.out.println("\nBook Details:");
        for (Book book : books) {
            book.displayDetails();
            System.out.println();
        }
        Book.displayTotalBooks();

        System.out.println("  ");
        System.out.println("^_^_^_^_^_^_^_^_^_^");
```



```
        System.out.print("\nEnter the number of students: ");
        int numStudents = scanner.nextInt();
        scanner.nextLine();
        System.out.println("  ");

        Student[] students = new Student[numStudents];

        for (int i = 0; i < numStudents; i++) {
            System.out.println("Enter details for student " + (i + 1) + ":");
            System.out.print("ID: ");
            String id = scanner.nextLine();
            System.out.print("Name: ");
            String name = scanner.nextLine();
            System.out.print("Department: ");
            String department = scanner.nextLine();
            System.out.print("CGPA: ");
            double cgpa = scanner.nextDouble();
            scanner.nextLine();
            System.out.print("University: ");
            String university = scanner.nextLine();
            System.out.println("  ");

            students[i] = new Student(id, name, department, cgpa, university);
        }

        System.out.println("\nStudent Details:");
        for (Student student : students) {
            student.displayDetails();
            System.out.println();
        }
        Student.displayTotalStudents();

        System.out.println("  ");
        System.out.println("^_^_^_^_^_^_^_^_^_^");
        System.out.println("  ");
        System.out.println("  ");

        scanner.close();
    }
}
```

To rapidly build and iterate over Book and Student objects, I've built both for and for each loop in the main code that is given. To provide flexibility and scalability, the application first asks the user for input to decide how many objects to construct. The for-each loop gracefully iterates across the arrays to show the information of each item, while the for loop gathers user inputs and initializes the objects. Using this method guarantees a smooth and well-organized process for handling numerous student and book records.