This Java code implements a simple book store management system. It includes classes for representing books ("Book") and a book store ("BookStore"), as well as a main class ("BookStoreManagementSystem") to interact with the user. Let's break down each part of the code:

1. Book Class

The Book class represents a book with four private attributes: 'title', 'author', 'price', and 'quantity'.

```
class Book {
    private String title;
    private String author;
    private double price;
    private int quantity;

    public Book(String title, String author, double price, int quantity) {
        this.title = title;
        this.author = author;
        this.price = price;
```

```java
        this.quantity = quantity;
    }

    public String getTitle() {
        return title;
    }

    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }

    public int getQuantity() {
        return quantity;
    }

    @Override
    public String toString() {
        return "Title: " + title + ", Author: " + author + ", Price: " + price
+ ", Quantity: " + quantity;
    }
}
```

> Constructor: Initializes a new book with the given title, author, price, and quantity.

>Getter and Setter: Methods for getting the title and quantity, and setting the quantity.

>toString() Method: Returns a string representation of the book's details.

2. BookStore Class

The BookStore class manages a collection of books using an ArrayList.

```
class BookStore {
    private ArrayList<Book> books;
```

```
public BookStore() {
    books = new ArrayList<>();
}


public void addBook(Book book) {
    books.add(book);
    System.out.println("Book added successfully.");
}


public Book searchBook(String title) {
    for (Book book : books) {
        if (book.getTitle().equalsIgnoreCase(title)) {
            return book;
        }
    }
    return null;
}


public void listBooks() {
    if (books.isEmpty()) {
        System.out.println("No books available.");
    } else {
        for (Book book : books) {
                System.out.println(book);
            }
        }
    }


    public void sellBook(String title, int quantity) {
        Book book = searchBook(title);
        if (book != null) {
            if (book.getQuantity() >= quantity) {
                book.setQuantity(book.getQuantity() - quantity);
                System.out.println("Sold " + quantity + " copies of " +
title);
            } else {
                System.out.println("Not enough copies available.");
            }
        } else {
            System.out.println("Book not found.");
        }
    }
}
```

>Constructor: Initializes the books ArrayList.

>addBook() Method: Adds a new book to the collection.

>searchBook() Method: Searches for a book by title, ignoring case, and returns the book if found.

>listBooks() Method: Lists all books in the collection or indicates if no books are available.

>sellBook() Method: Sells a specified quantity of a book if it exists and has sufficient stock.

3. BookStoreManagementSystem Class

This class contains the main method and interacts with the user through a console interface.

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

```
Scanner scanner = new Scanner(System.in);
BookStore store = new BookStore();
int choice;

do {
    System.out.println("Book Store Management System");
    System.out.println("1. Add Book");
    System.out.println("2. Search Book");
    System.out.println("3. List All Books");
    System.out.println("4. Sell Book");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    choice = scanner.nextInt();
    scanner.nextLine();  // Consume newline

    switch (choice) {
        case 1:
            System.out.print("Enter title: ");
            String title = scanner.nextLine();
            System.out.print("Enter author: ");
            String author = scanner.nextLine();
            System.out.print("Enter price: ");
            double price = scanner.nextDouble();
```

```
  System.out.print("Enter quantity: ");
  int quantity = scanner.nextInt();
  scanner.nextLine();  // Consume newline
  Book book = new Book(title, author, price, quantity);
  store.addBook(book);
  break;

case 2:
  System.out.print("Enter title to search: ");
  title = scanner.nextLine();
  Book foundBook = store.searchBook(title);
  if (foundBook != null) {
    System.out.println("Book found: " + foundBook);
  } else {
    System.out.println("Book not found.");
  }
  break;

case 3:
  store.listBooks();
  break;

case 4:
```

```
                    System.out.print("Enter title to sell: ");
                    title = scanner.nextLine();
                    System.out.print("Enter quantity to sell: ");
                    quantity = scanner.nextInt();
                    scanner.nextLine();  // Consume newline
                    store.sellBook(title, quantity);
                    break;

                case 5:
                    System.out.println("Exiting the system. Goodbye!");
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
                    break;
            }
        } while (choice != 5);

        scanner.close();
    }
}
```

```
}
```

ChatGPT

```
import java.util.ArrayList;
import java.util.Scanner;

class Book {
    private String title;
    private String author;
    private double price;
    private int quantity;

    public Book(String title, String author, double price, int
quantity) {
        this.title = title;
        this.author = author;
        this.price = price;
        this.quantity = quantity;
    }

    public String getTitle() {
        return title;
    }
    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }

    public int getQuantity() {
        return quantity;
    }

    @Override
    public String toString() {
        return "Title: " + title + ", Author: " + author + ", Price: " + price
+ ", Quantity: " + quantity;
    }
}

class BookStore {
    private ArrayList<Book> books;

    public BookStore() {
        books = new ArrayList<>();
    }
```

```
public void addBook(Book book) {
    books.add(book);
    System.out.println("Book added successfully.");
}

public Book searchBook(String title) {
    for (Book book : books) {
        if (book.getTitle().equalsIgnoreCase(title)) {
            return book;
        }
    }
    return null;
}

public void listBooks() {
    if (books.isEmpty()) {
        System.out.println("No books available.");
    } else {
        for (Book book : books) {
            System.out.println(book);
        }
    }
    }


    public void sellBook(String title, int quantity) {
        Book book = searchBook(title);
        if (book != null) {
            if (book.getQuantity() >= quantity) {
                book.setQuantity(book.getQuantity() - quantity);
                System.out.println("Sold " + quantity + " copies of " +
title);
            } else {
                System.out.println("Not enough copies available.");
            }
        } else {
            System.out.println("Book not found.");
        }
    }
}


public class BookStoreManagementSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        BookStore store = new BookStore();
        int choice;
```

```
do {
    System.out.println("Book Store Management System");
    System.out.println("1. Add Book");
    System.out.println("2. Search Book");
    System.out.println("3. List All Books");
    System.out.println("4. Sell Book");
    System.out.println("5. Exit");
    System.out.print("Enter your choice: ");
    choice = scanner.nextInt();
    scanner.nextLine();  // Consume newline


    switch (choice) {
        case 1:
            System.out.print("Enter title: ");
            String title = scanner.nextLine();
            System.out.print("Enter author: ");
            String author = scanner.nextLine();
            System.out.print("Enter price: ");
            double price = scanner.nextDouble();
            System.out.print("Enter quantity: ");
            int quantity = scanner.nextInt();
            scanner.nextLine();  // Consume newline
```

```
  Book book = new Book(title, author, price, quantity);
  store.addBook(book);
  break;


case 2:
  System.out.print("Enter title to search: ");
  title = scanner.nextLine();
  Book foundBook = store.searchBook(title);
  if (foundBook != null) {
    System.out.println("Book found: " + foundBook);
  } else {
    System.out.println("Book not found.");
  }
  break;


case 3:
  store.listBooks();
  break;


case 4:
  System.out.print("Enter title to sell: ");
  title = scanner.nextLine();
  System.out.print("Enter quantity to sell: ");
```

```
                    quantity = scanner.nextInt();
                    scanner.nextLine();  // Consume newline
                    store.sellBook(title, quantity);
                    break;

                case 5:
                    System.out.println("Exiting the system. Goodbye!");
                    break;

                default:
                    System.out.println("Invalid choice. Please try again.");
                    break;
            }
        } while (choice != 5);

        scanner.close();
    }
}
```

Explain this code in detail

ChatGPT

This Java code implements a simple book store management system. It includes classes for representing books (Book) and a book store (BookStore), as well as a main class (BookStoreManagementSystem) to interact with the user. Let's break down each part of the code:

1. Book Class

The Book class represents a book with four private attributes: title, author, price, and quantity.

java

Copy code

```java
class Book {
   private String title;
   private String author;
   private double price;
   private int quantity;

   public Book(String title, String author, double price, int quantity) {
       this.title = title;
       this.author = author;
       this.price = price;
       this.quantity = quantity;
```

```
    }

    public String getTitle() {
        return title;
    }

    public void setQuantity(int quantity) {
        this.quantity = quantity;
    }

    public int getQuantity() {
        return quantity;
    }

    @Override
    public String toString() {
        return "Title: " + title + ", Author: " + author + ", Price: " + price
+ ", Quantity: " + quantity;
    }
}
```

Constructor: Initializes a new book with the given title, author, price, and quantity.

Getter and Setter: Methods for getting the title and quantity, and setting the quantity.

toString() Method: Returns a string representation of the book's details.

2. BookStore Class

The BookStore class manages a collection of books using an ArrayList.

java

Copy code

```
class BookStore {
    private ArrayList<Book> books;


    public BookStore() {
        books = new ArrayList<>();
    }


    public void addBook(Book book) {
        books.add(book);
        System.out.println("Book added successfully.");
    }
```

```
public Book searchBook(String title) {
    for (Book book : books) {
        if (book.getTitle().equalsIgnoreCase(title)) {
            return book;
        }
    }
    return null;
}


public void listBooks() {
    if (books.isEmpty()) {
        System.out.println("No books available.");
    } else {
        for (Book book : books) {
            System.out.println(book);
        }
    }
}


public void sellBook(String title, int quantity) {
    Book book = searchBook(title);
    if (book != null) {
        if (book.getQuantity() >= quantity) {
                book.setQuantity(book.getQuantity() - quantity);
                System.out.println("Sold " + quantity + " copies of " +
title);
            } else {
                System.out.println("Not enough copies available.");
            }
        } else {
            System.out.println("Book not found.");
        }
    }
}
```

Constructor: Initializes the books ArrayList.

addBook() Method: Adds a new book to the collection.

searchBook() Method: Searches for a book by title, ignoring case, and returns the book if found.

listBooks() Method: Lists all books in the collection or indicates if no books are available.

sellBook() Method: Sells a specified quantity of a book if it exists and has sufficient stock.

3. BookStoreManagementSystem Class

This class contains the main method and interacts with the user through a console interface.

java

Copy code

```
public class BookStoreManagementSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        BookStore store = new BookStore();
        int choice;

        do {
            System.out.println("Book Store Management System");
            System.out.println("1. Add Book");
            System.out.println("2. Search Book");
            System.out.println("3. List All Books");
            System.out.println("4. Sell Book");
            System.out.println("5. Exit");
            System.out.print("Enter your choice: ");
            choice = scanner.nextInt();
            scanner.nextLine();  // Consume newline

            switch (choice) {
                case 1:
                    System.out.print("Enter title: ");
                    String title = scanner.nextLine();
                    System.out.print("Enter author: ");
                    String author = scanner.nextLine();
                    System.out.print("Enter price: ");
                    double price = scanner.nextDouble();
                    System.out.print("Enter quantity: ");
                    int quantity = scanner.nextInt();
                    scanner.nextLine();  // Consume newline
                    Book book = new Book(title, author, price, quantity);
                    store.addBook(book);
                    break;

                case 2:
                    System.out.print("Enter title to search: ");
                    title = scanner.nextLine();
                    Book foundBook = store.searchBook(title);
                    if (foundBook != null) {
                        System.out.println("Book found: " + foundBook);
                    } else {
                        System.out.println("Book not found.");
                    }
                    break;

                case 3:
                    store.listBooks();
```

```
      break;

    case 4:
      System.out.print("Enter title to sell: ");
      title = scanner.nextLine();
      System.out.print("Enter quantity to sell: ");
      quantity = scanner.nextInt();
      scanner.nextLine();  // Consume newline
      store.sellBook(title, quantity);
      break;

    case 5:
      System.out.println("Exiting the system. Goodbye!");
      break;

    default:
      System.out.println("Invalid choice. Please try again.");
      break;
  }
} while (choice != 5);

scanner.close();
```

```
    }
}
```

>Scanner: Used to read user input from the console.

>BookStore Instance: Created to manage the books.

>Menu Loop: Displays a menu with options to add, search, list, sell books, or exit. Continues to prompt the user until they choose to exit.

>Switch Statement: Handles each menu option:

>Add Book: Prompts for book details and adds the book to the store.

>Search Book: Prompts for a title and searches for the book.

>List All Books: Lists all books in the store.

>Sell Book: Prompts for a title and quantity, and sells the specified quantity if possible.

>Exit: Exits the program.

This system provides basic functionalities for managing a book store through a command-line interface.