



# K.R. Mangalam University

## School of Engineering & Technology

### Fundamentals Of Java Programming Lab (ENCA203) Assignment 4 Student Management System

Submitted by:

Name: RAKESH G  
Kumar

Roll No: 2401201064

Class: BCA (AI & DS)

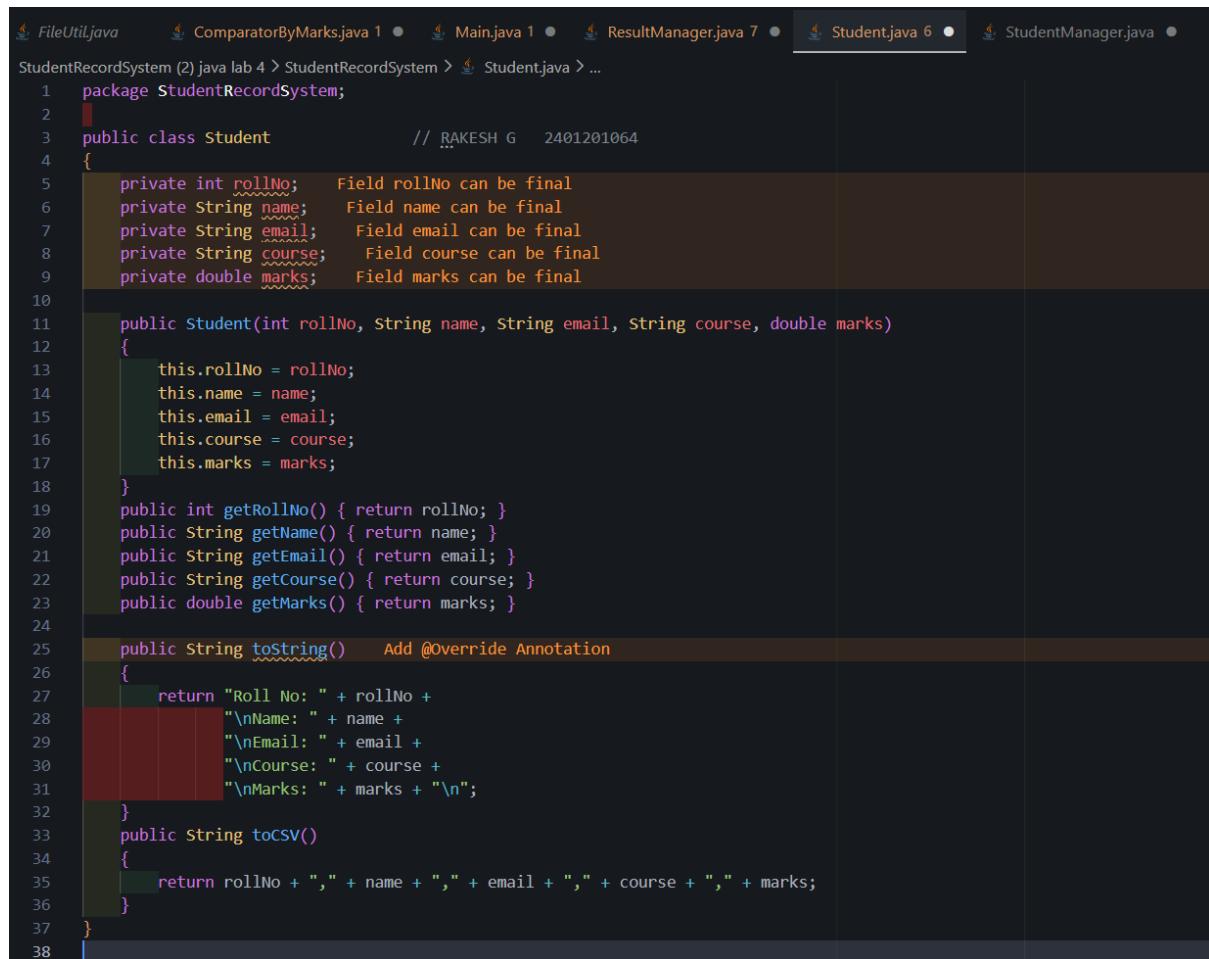
Submitted to:

Dr. Manish

GitHub Repository:

[https://github.com/rakesh4407/Java\\_Assignment\\_Rakesh](https://github.com/rakesh4407/Java_Assignment_Rakesh)

## 1. Student.java

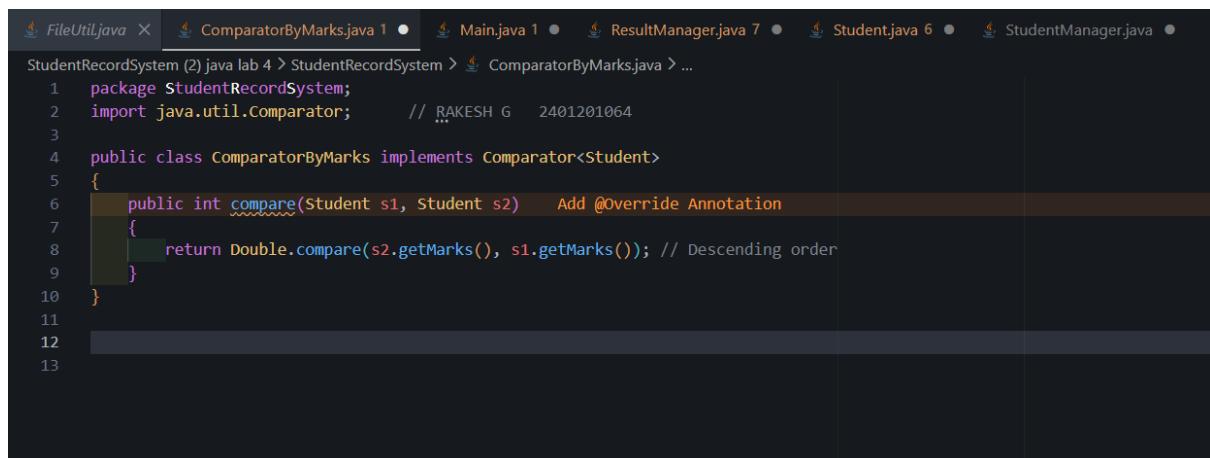


```
FileUtil.java ComparatorByMarks.java 1 Main.java 1 ResultManager.java 7 Student.java 6 StudentManager.java

StudentRecordSystem (2) java lab 4 > StudentRecordSystem > Student.java > ...

1 package StudentRecordSystem;
2
3 public class Student           // RAKESH G  2401201064
4 {
5     private int rollNo;    Field rollNo can be final
6     private String name;   Field name can be final
7     private String email;  Field email can be final
8     private String course; Field course can be final
9     private double marks;  Field marks can be final
10
11    public Student(int rollNo, String name, String email, String course, double marks)
12    {
13        this.rollNo = rollNo;
14        this.name = name;
15        this.email = email;
16        this.course = course;
17        this.marks = marks;
18    }
19    public int getRollNo() { return rollNo; }
20    public String getName() { return name; }
21    public String getEmail() { return email; }
22    public String getCourse() { return course; }
23    public double getMarks() { return marks; }
24
25    public String toString()      Add @Override Annotation
26    {
27        return "Roll No: " + rollNo +
28               "\nName: " + name +
29               "\nEmail: " + email +
30               "\nCourse: " + course +
31               "\nMarks: " + marks + "\n";
32    }
33    public String toCSV()
34    {
35        return rollNo + "," + name + "," + email + "," + course + "," + marks;
36    }
37 }
38
```

## 2. ComparatorByMarks.java



```
FileUtil.java × ComparatorByMarks.java 1 Main.java 1 ResultManager.java 7 Student.java 6 StudentManager.java

StudentRecordSystem (2) java lab 4 > StudentRecordSystem > ComparatorByMarks.java > ...

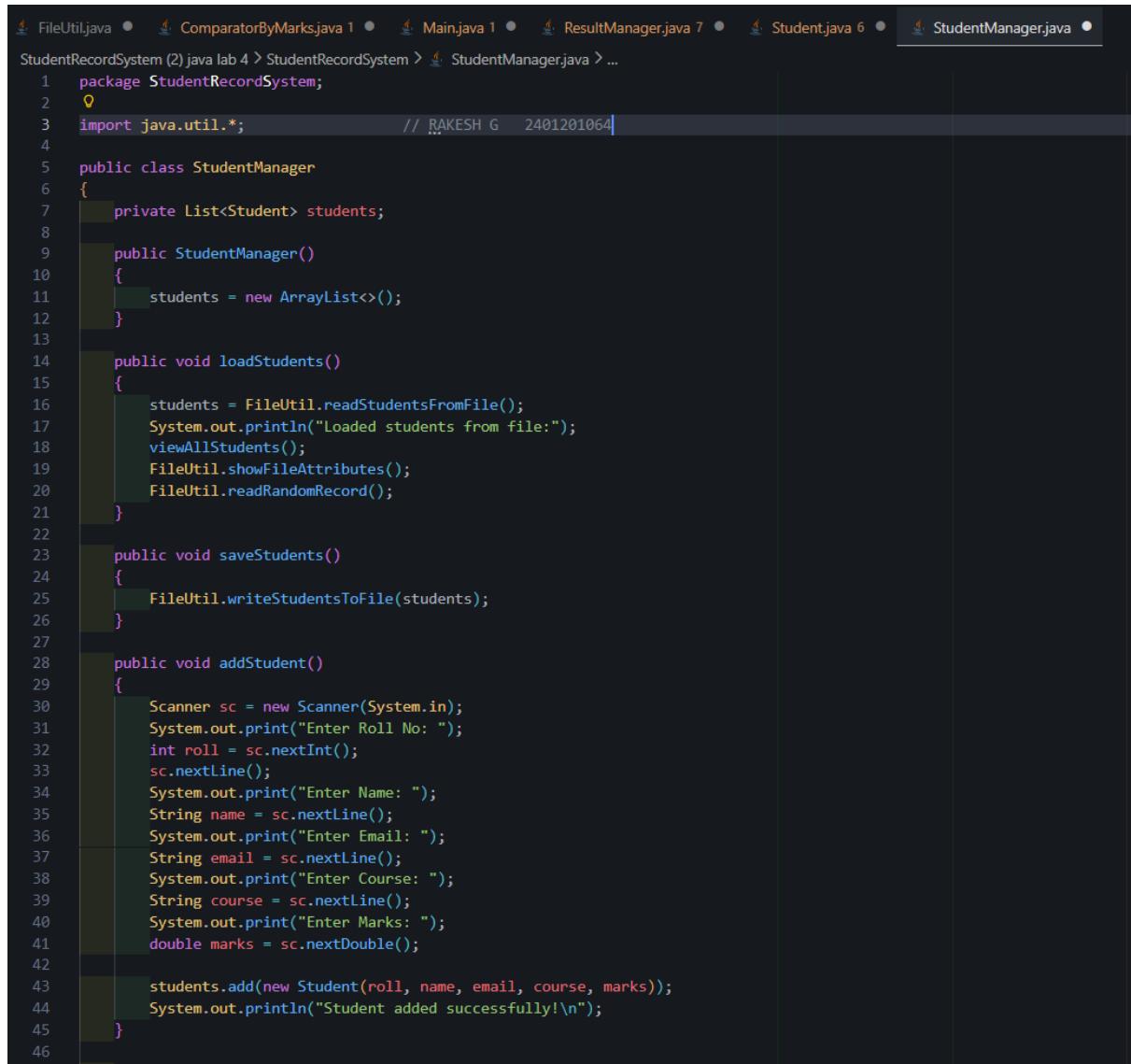
1 package StudentRecordSystem;
2 import java.util.Comparator;      // RAKESH G  2401201064
3
4 public class ComparatorByMarks implements Comparator<Student>
5 {
6     public int compare(Student s1, Student s2)      Add @Override Annotation
7     {
8         return Double.compare(s2.getMarks(), s1.getMarks()); // Descending order
9     }
10 }
11
12
13
```

### 3. FileUtil.java

```
FileUtil.java ● ComparatorByMarks.java 1 ● Main.java 1 ● ResultManager.java 7 ● Student.java 6 ● StudentManager.java ●
StudentRecordSystem (2) java lab 4 > StudentRecordsSystem > FileUtil.java > Java > FileUtil
1 package StudentRecordSystem;
2 import java.io.*;
3 import java.util.*;
4 public class FileUtil // RAKESH G 2401201064
5 {
6     private static final String FILE_NAME = "students.txt";
7     public static List<Student> readStudentsFromFile()
8     {
9         List<Student> list = new ArrayList<>();
10        try (BufferedReader br = new BufferedReader(new FileReader(FILE_NAME)))
11        {
12            String line;
13            while ((line = br.readLine()) != null)
14            {
15                String[] parts = line.split(",");
16                if (parts.length == 5)
17                {
18                    int rollNo = Integer.parseInt(parts[0]);
19                    String name = parts[1];
20                    String email = parts[2];
21                    String course = parts[3];
22                    double marks = Double.parseDouble(parts[4]);
23                    list.add(new Student(rollNo, name, email, course, marks));
24                }
25            }
26        } catch (IOException e)
27        {
28            System.out.println("File not found or error reading file. A new file will be created on save.");
29        }
30        return list;
31    }
32    public static void writeStudentsToFile(List<Student> students)
33    {
34        try (BufferedWriter bw = new BufferedWriter(new FileWriter(FILE_NAME)))
35        {
36            for (Student s : students)
37            {
38                bw.write(s.toCSV() + "\n");
39            }
40        } catch (IOException e)
41        {
42            System.out.println("Data saved successfully to " + FILE_NAME);
43        }
44    }
45    public static void showFileAttributes()
46    {
47        File file = new File(FILE_NAME);
48        System.out.println("\n--- File Attributes ---");
49        System.out.println("File Name: " + file.getName());
50        System.out.println("Path: " + file.getAbsolutePath());
51        System.out.println("Size: " + file.length() + " bytes");
52        System.out.println("Readable: " + file.canRead());
53        System.out.println("Writable: " + file.canWrite());
54        System.out.println("-----\n");
55    }
56    public static void readRandomRecord()
57    {
58        try (RandomAccessFile raf = new RandomAccessFile("students.txt", "r"))
59        {
60            long length = raf.length();
61            long position = length / 2;
62            raf.seek(position);
63            String line = raf.readLine();
64            System.out.println("Random Record Read:\n" + line);
65        } catch (IOException e)
66        {
67            System.out.println("Error reading random record: " + e.getMessage());
68        }
69    }
70}
71}
72}
73}
```

```
FileUtil.java ● ComparatorByMarks.java 1 ● Main.java 1 ● ResultManager.java 7 ● Student.java 6 ● StudentManager.java ●
StudentRecordSystem (2) java lab 4 > StudentRecordsSystem > FileUtil.java > Java > FileUtil
4 public class FileUtil // RAKESH G 2401201064
33     public static void writeStudentsToFile(List<Student> students)
34     {
35         bw.write(s.toCSV() + "\n");
36     }
37     System.out.println("Data saved successfully to " + FILE_NAME);
38 }
39 }
40 catch (IOException e)
41 {
42     System.out.println("Error writing to file: " + e.getMessage());
43 }
44 }
45 }
46 public static void showFileAttributes()
47 {
48     File file = new File(FILE_NAME);
49     System.out.println("\n--- File Attributes ---");
50     System.out.println("File Name: " + file.getName());
51     System.out.println("Path: " + file.getAbsolutePath());
52     System.out.println("Size: " + file.length() + " bytes");
53     System.out.println("Readable: " + file.canRead());
54     System.out.println("Writable: " + file.canWrite());
55     System.out.println("-----\n");
56 }
57 public static void readRandomRecord()
58 {
59     try (RandomAccessFile raf = new RandomAccessFile("students.txt", "r"))
60     {
61         long length = raf.length();
62         long position = length / 2;
63         raf.seek(position);
64         String line = raf.readLine();
65         System.out.println("Random Record Read:\n" + line);
66     } catch (IOException e)
67     {
68         System.out.println("Error reading random record: " + e.getMessage());
69     }
70 }
71 }
72 }
```

#### 4. StudentManager.java



The screenshot shows a Java code editor with the file `StudentManager.java` open. The code implements a `StudentManager` class that interacts with a list of `Student` objects. It includes methods for loading students from a file, saving them to a file, and adding new students. The code uses `Scanner` to read input from the console.

```
1 package StudentRecordSystem;
2
3 import java.util.*; // RAKESH G 2401201064
4
5 public class StudentManager
6 {
7     private List<Student> students;
8
9     public StudentManager()
10    {
11        students = new ArrayList<>();
12    }
13
14     public void loadStudents()
15    {
16        students = FileUtil.readStudentsFromFile();
17        System.out.println("Loaded students from file:");
18        viewAllStudents();
19        FileUtil.showFileAttributes();
20        FileUtil.readRandomRecord();
21    }
22
23     public void saveStudents()
24    {
25        FileUtil.writeStudentsToFile(students);
26    }
27
28     public void addStudent()
29    {
30        Scanner sc = new Scanner(System.in);
31        System.out.print("Enter Roll No: ");
32        int roll = sc.nextInt();
33        sc.nextLine();
34        System.out.print("Enter Name: ");
35        String name = sc.nextLine();
36        System.out.print("Enter Email: ");
37        String email = sc.nextLine();
38        System.out.print("Enter Course: ");
39        String course = sc.nextLine();
40        System.out.print("Enter Marks: ");
41        double marks = sc.nextDouble();
42
43        students.add(new Student(roll, name, email, course, marks));
44        System.out.println("Student added successfully!\n");
45    }
46}
```

```
FileUtil.java ● ComparatorByMarks.java 1 ● Main.java 1 ● ResultManager.java 7 ● Student.java 6 ● StudentManager.java ●
StudentRecordSystem (2) java lab 4 > StudentRecordSystem > StudentManager.java > Java > StudentManager > deleteByName(String name)
5  public class StudentManager
6
7      public void viewAllStudents()
8      {
9          Iterator<Student> it = students.iterator();
10         while (it.hasNext())
11         {
12             System.out.println(it.next());
13         }
14     }
15     public void searchByName(String name)
16     {
17         for (Student s : students)
18         {
19             if (s.getName().equalsIgnoreCase(name))
20             {
21                 System.out.println("Student Found:\n" + s);
22                 return;
23             }
24         }
25         System.out.println("No student found with name: " + name);
26     }
27     public void deleteByName(String name)
28     {
29         Iterator<Student> it = students.iterator();
30         boolean removed = false;
31         while (it.hasNext())
32         {
33             if (it.next().getName().equalsIgnoreCase(name))
34             {
35                 it.remove();
36                 removed = true;
37             }
38             if (removed)
39                 System.out.println("Student deleted successfully!");
40             else
41                 System.out.println("No student found with that name.");
42         }
43     }
44     public void sortByMarks()
45     {
46         students.sort(new ComparatorByMarks());
47         System.out.println("Sorted Student List by Marks:");
48         viewAllStudents();
49     }
50 }
```

## 5. Main.java

```
FileUtil.java ● ComparatorByMarks.java 1 ● Main.java 1 ● ResultManager.java 7 ● Student.java 6 ● StudentManager.java ●
StudentRecordSystem (2) java lab 4 > StudentRecordSystem > Main.java > Java > Main > main(String[] args)
1 package StudentRecordSystem;
2 import java.util.*;
3 public class Main // RAKESH G 2401201064
4 {
5     Run main | Debug main | Run | Debug
6     public static void main(String[] args)
7     {
8         StudentManager manager = new StudentManager();
9         manager.loadStudents();
10
11        Scanner sc = new Scanner(System.in);
12        int choice;
13
14        do
15        {
16            System.out.println("===== Capstone Student Menu =====");
17            System.out.println("1. Add Student");
18            System.out.println("2. View All Students");
19            System.out.println("3. Search by Name");
20            System.out.println("4. Delete by Name");
21            System.out.println("5. Sort by Marks");
22            System.out.println("6. Save and Exit");
23            System.out.print("Enter choice: ");
24            choice = sc.nextInt();
25            sc.nextLine();
26
27            switch (choice) Convert switch to rule switch
28            {
29                case 1:
30                    manager.addStudent();
31                    break;
32
33                case 2:
34                    manager.viewAllStudents();
35                    break;
36
37                case 3:
38                    System.out.print("Enter name to search: ");
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 }
```

```
FileUtil.java ● ComparatorByMarks.java 1 ● Main.java 1 ● ResultManager.java 7 ● Student.java 6 ● StudentManager.java ●
StudentRecordSystem (2) java lab 4 > StudentRecordSystem > Main.java > Java > Main > main(String[] args)
3 public class Main // RAKESH G 2401201064
5 public static void main(String[] args)
6 {
7     case 1:
8         manager.addstudent();
9         break;
10
11    case 2:
12        manager.viewAllstudents();
13        break;
14
15    case 3:
16        System.out.print("Enter name to search: ");
17        String searchName = sc.nextLine();
18        manager.searchByName(searchName);
19        break;
20
21    case 4:
22        System.out.print("Enter name to delete: ");
23        String delName = sc.nextLine();
24        manager.deleteByName(delName);
25        break;
26
27    case 5:
28        manager.sortByMarks();
29        break;
30
31    case 6:
32        manager.saveStudents();
33        System.out.println("Records saved. Exiting...");
34        break;
35
36
37    default:
38        System.out.println("Invalid choice! Try again.");
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64 }
```

## 6. Output

```
PROBLEMS 25 OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE SPELL CHECKER 6

Active code page: 65001

D:\RAKESH\VSC>cd "d:\RAKESH\VSC\StudentRecordSystem (2)" && java lab 4\StudentRecordSystem\" && javac ResultManager.java && java ResultManager
Picked up JAVA_TOOL_OPTIONS: -Dstdout.encoding=UTF-8 -Dstderr.encoding=UTF-8
Picked up JAVA_TOOL_OPTIONS: -Dstdout.encoding=UTF-8 -Dstderr.encoding=UTF-8

===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 1
Enter Roll Number: 64
Enter Student Name: Rakesh
Enter marks for subject 1: 98
Enter marks for subject 2: 97
Enter marks for subject 3: 95
Student added successfully. Returning to main menu...

===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 2
Enter Roll Number to search: 64
Roll Number: 64
Student Name: Rakesh
Marks: 98 97 95
Average: 96.66666666666666
Result: Pass
Search completed.

===== Student Result Management System =====
1. Add Student
2. Show Student Details
3. Exit
Enter your choice: 3
Exiting program. Thank you!
Scanner closed. Program terminated.

d:\RAKESH\VSC\StudentRecordSystem (2) java lab 4\StudentRecordSystem>
```

## **Explanation:**

The **Student Record Management System** is a Java-based application developed to manage student details efficiently using **file handling** and the **Java Collections Framework**. The main objective of this program is to store, retrieve, and manipulate student information such as roll number, name, email, course, and marks in a structured and organized way.

At the beginning of the program, student records are **loaded from an external file (students.txt)** using **BufferedReader**, ensuring data persistence even after the program ends. The program allows users to perform various operations such as **adding a new student, viewing all students, searching by name, deleting a record, and sorting students by marks**. When the user exits, all updated data is **saved back to the file** using **BufferedWriter**.

The system uses an **ArrayList** to store multiple student objects dynamically. It applies the **Comparator** interface to sort student records based on marks and uses the **Iterator** interface to traverse and display data efficiently. Additionally, file attributes are accessed through the **File** class, and random record reading is demonstrated using the **RandomAccessFile** class.

The project is divided into several Java classes:

- **Student.java**: Defines the student entity and its attributes.
- **ComparatorByMarks.java**: Sorts students in descending order of marks.
- **FileUtil.java**: Handles reading, writing, and file-related operations.
- **StudentManager.java**: Manages operations such as add, search, delete, and sort.
- **Main.java**: Provides a menu-driven interface for user interaction.

This assignment helped in understanding how **Object-Oriented Programming (OOP)**, **Collections**, and **File Handling** can be integrated to build a complete data management application. It demonstrates the use of **Encapsulation, Iterators, Comparators, and Persistent Storage**, making it a practical example of how Java can be used for real-world data management systems.