Project Report:

Portfolio Project – Students Manager

Alejandro Ricciardi

Colorado State University Global

CSC372: Programming 2

Professor: Dr. Vanessa Cooper

August 4, 2024

Contents

Project Report:

Portfolio Project – Students Manager

This documentation is part of the Portfolio Project Assignment from CSC372: Programming 2 at Colorado State University Global. This Project Report is an overview of the assignment and the Students Manager program's functionality, testing scenarios, and outputs. The program is coded in Java JDK-21.

The Assignment Directions:

Option #1: **Lessons Learned and Final Program**

Milestones

Milestone 1 (due at the end of Week 4): Java source code (with corrections as required) for programs created in Modules 1-3.

Milestone 2 (due at the end of Week 7): Java source code (with corrections as required) for programs created in Modules 5-6.

Final Portfolio Project

In Week 8, the components you must complete for your Portfolio Project are the Lessons Learned Reflection and the Final Program. Carefully review the requirements below.

Lessons Learned Reflection

Write a summary outlining the lessons you have learned in this programming course. Reflect on how these lessons can be applied to effective coding.

This essay portion of your Portfolio must:

- Be 2-3 pages in length (not including the required title and references pages)
- Be formatted according to APA guidelines in the CSU Global Writing Center.
- Include at least three outside references from the course readings, formatted according to APA requirements.

Final Program

Write a Java program that incorporates a loop that prompts the user for student data. Student data are private fields in a student class including:

- String name
- String address
- double GPA

Each student object is stored in a linked list.

After the user completes the data entry, output the contents of the linked list in ascending sorted order by name to a regular text file that can be opened and viewed using a simple plain-text editor such as notepad.

Validate numeric data for Grade Point Average (GPA).

Compile your Lessons Learned Reflection, source code, screenshots of the application executing, and results into a single document.

Format your document in MS Word, according to APA guidelines in the CSU Global Writing Center, particularly in developing your Lessons Learned Reflection. Support your reflection with a minimum of three references, as noted above. Include both a cover page and a reference page with your Portfolio Project.

⚠️ **Program Notes:**

- I got permission from Dr. Cooper to use the JavaFX library to display the program outputs.
- Added my own icon to the window frame – logo.png.
- Added search functionality (not an assignment requirement).
- Added read file functionality (not an assignment requirement).
- Added the option to add fake data to the file for troubleshooting purposes (not an assignment requirement).
- Implemented my own selection sort and binary search algorithms.
- Created a UML class diagram.
- **For the source code please see the following: Student.java, StudentManager.java, NameComparator.java, GpaComparator.java, SortSearchUtil.java, StudentManagerApp.java**

Project Map:

- **Project Report.pdf**: A pdf file (this file) containing an overview of the assignment and the Students Manager program.
- **README.md**: A markdown file containing information about the project, intended to be viewed on GitHub.
- **Lessons Learned and Reflection.doc:** A Word document containing a summary and reflections on the lessons I have learned in this programming course.
- **Milestone-1:** Directory containing the Portfolio Milestone assignment from Module 4.
- **Milestone-2:** Directory containing the Portfolio Milestone assignment from Module 7.
- **Application:** A folder containing the source code, Java code files for the Students Manager program.

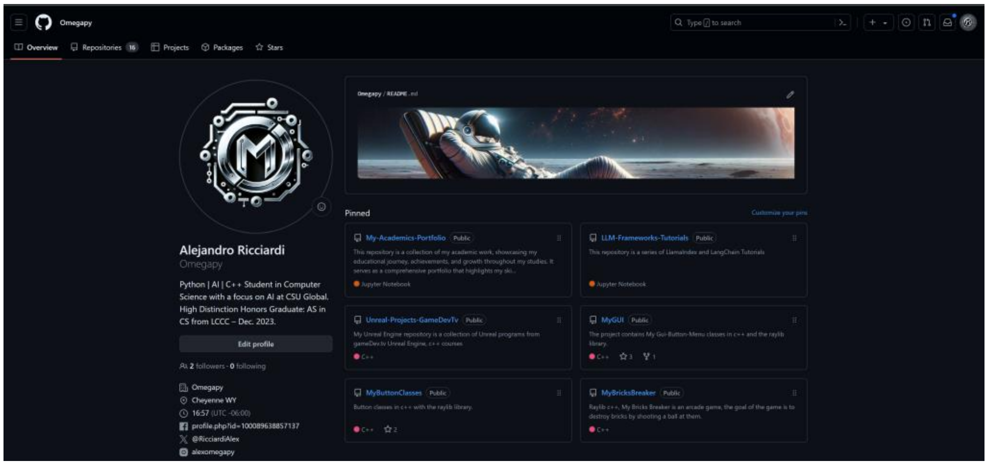
Program Description:

The Students Manager is a small Java application that utilizes JavaFX GUI allowing the user to add, view, search, and sort students data:

- Student data management (name, address, GPA)
- File-based storage
- Sorting by name or GPA
- Search functionality
- Basic data validation

Git Repository

This is a picture of my GitHub page:



Alejandro Ricciardi
Omegapy

Python | AI | C++ Student in Computer Science with a focus on AI at CSU Global. High Distinction Honors Graduate: AS in CS from LCCC – Dec. 2023.

2 followers · 0 following

Omegapy
Cheyenne WY
16:57 (UTC -06:00)
profile.php?id=100089638857137
@RicciardiAlex
alexomegapy

Omegapy / README.md

My-Academics-Portfolio Public
This repository is a collection of my academic work, showcasing my educational journey, achievements, and growth throughout my studies. It serves as a comprehensive portfolio that highlights my ski...
Jupyter Notebook

LLM-Frameworks-Tutorials Public
This repository is a series of LlamaIndex and LangChain Tutorials
Jupyter Notebook

Unreal-Projects-GameDevTv Public
My Unreal Engine repository is a collection of Unreal programs from gameDev.tv Unreal Engine, c++ courses
C++

MyGUI Public
The project contains My Gui-Button-Menu classes in c++ and the raylib library.
C++ 3 1

MyButtonClasses Public
Button classes in c++ with the raylib library.
C++ 2

MyBricksBreaker Public
Raylib c++, My Bricks Breaker is an arcade game, the goal of the game is to destroy bricks by shooting a ball at them.
C++

I use GitHub as my Distributed Version Control System (DVCS), the following is a link to my GitHub, Omegapy.

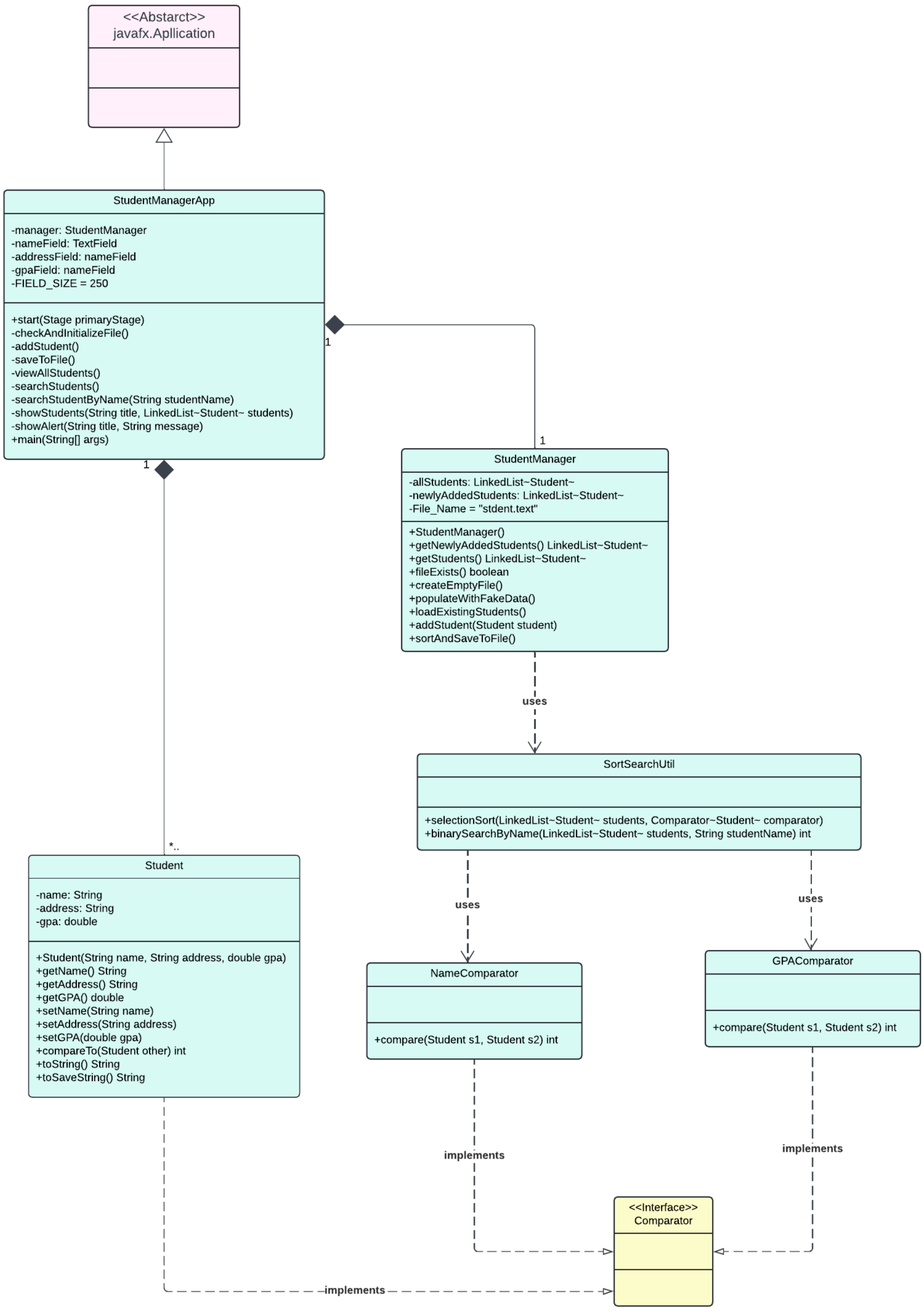
My GitHub repository that is used to store this assignment is named My-Academics-Portfolio and the link to this specific assignment is:

Classes Description:

- **Student Class**
  Creates student objects. This class stores student data and provides methods for data access and validation. It implements the Comparable interface to be used by the SortSearchUtil class to sort and search students
- **StudentManager class**
  Manages student objects and handles file operations. This class is responsible for adding, storing, and retrieving student data, as well as reading from and writing to a file.
- **NameComparator Class**
  Comparator for sorting Student objects based on their names. Implements an ascending order comparison (A to Z), used by the SortSearchUtil class. This class implements the Comparator interface.
- **GpaComparator class**
  Comparator for sorting Student objects based on their names. Implements a descending order comparison (highest to lowest GPA), used by the SortSearchUtil class. This class implements the Comparator interface.
- **SortSearchUtil class**
  Utility class for sorting and searching students. Provides methods for selection sort and binary search on a LinkedList of Student objects.

- **StudentManagerApp class**
  Main application class for the Student Manager program. This class creates the user interface (GUI).

UML Class Diagram


<<Abstarct>>
javafx.Apllication
StudentManagerApp
-manager: StudentManager
-nameField: TextField
-addressField: nameField
-gpaField: nameField
-FIELD_SIZE = 250
+start(Stage primaryStage)
-checkAndInitializeFile()
-addStudent()
-saveToFile()
-viewAllStudents()
-searchStudents()
-searchStudentByName(String studentName)
-showStudents(String title, LinkedList~Student~ students)
-showAlert(String title, String message)
+main(String[] args)
1
1
StudentManager
-allStudents: LinkedList~Student~
-newlyAddedStudents: LinkedList~Student~
-File_Name = "stdent.text"
+StudentManager()
+getNewlyAddedStudents() LinkedList~Student~
+getStudents() LinkedList~Student~
+fileExists() boolean
+createEmptyFile()
+populateWithFakeData()
+loadExistingStudents()
+addStudent(Student student)
+sortAndSaveToFile()
1
uses
SortSearchUtil
+selectionSort(LinkedList~Student~ students, Comparator~Student~ comparator)
+binarySearchByName(LinkedList~Student~ students, String studentName) int
*..
Student
-name: String
-address: String
-gpa: double
+Student(String name, String address, double gpa)
+getName() String
+getAddress() String
+getGPA() double
+setName(String name)
+setAddress(String address)
+setGPA(double gpa)
+compareTo(Student other) int
+toString() String
+toSaveString() String
uses
uses
NameComparator
+compare(Student s1, Student s2) int
GPAComparator
+compare(Student s1, Student s2) int
implements
implements
<<Interface>>
Comparator
implements

Screenshots

Program Functionality

Figure 1
Welcome Window

Note:

Figure 2

Note:

Figure 3

Figure 4
Students

Note:

Figure 5
Console Students Sorted by Roll Number Selection Sort Steps

As shown in Figure 1 through Figure 5 the program runs without any issues displaying the correct outputs as expected.