**Reflection and Final Program**

Jordon Paynter

Department of Computer Science, Colorado State University Global

CSC 372: Computer Programming II

Professor Chintan Thakkar

September 3, 2023

Contents

[**Reflection and Final Program** 3](#_Toc145152863)

[**Introduction** 3](#_Toc145152864)

[**Module 1: Object-Oriented Programming Principles** 3](#_Toc145152865)

[**Module 2: Building User Interfaces with Swing GUI Components** 3](#_Toc145152866)

[**Module 3: Exploring JavaFX for Modern UI Development** 4](#_Toc145152867)

[**Module 4: Advanced Object-Oriented Programming** 4](#_Toc145152868)

[**Module 5: Mastering Recursion** 4](#_Toc145152869)

[**Module 6: Efficient Searching and Sorting Algorithms** 5](#_Toc145152870)

[**Module 7: Java Collections and Map Frameworks** 5](#_Toc145152871)

[**Module 8: Exception Handling in Java** 5](#_Toc145152872)

[**Conclusion** 6](#_Toc145152873)

[**Figures Section** 7](#_Toc145152874)

[**Figure 1.** 7](#_Toc145152875)

[**Figure 2** 8](#_Toc145152876)

[**Figure 3** 9](#_Toc145152877)

[**Figure 4** 10](#_Toc145152878)

# **Reflection and Final Program**

## **Introduction**

While studying programming through its various modules, I have gained invaluable insights into the world of Java programming and its applications. As I journeyed through this course, I received a solid foundation for effective coding and software development practices. I will reflect on the lessons learned from five of the eight modules and discuss how these lessons can be applied to effective coding. Each module has enhanced my programming skills and understanding, ultimately shaping me into a more proficient coder.

## **Module 1: Object-Oriented Programming Principles**

In Module 1, I delved deeper into Java programming, focusing on advanced features and object-oriented programming principles. The concept of organizing classes hierarchically in a tree structure and inheriting variables and methods from higher-level classes was a key takeaway. This organization reduces code duplication and redundancy, a fundamental principle in effective coding. By applying these principles, I can create modular and efficient code that is easier to maintain and extend.

## **Module 2: Building User Interfaces with Swing GUI Components**

Module 2 introduced me to Swing GUI components, essential for developing graphical user interfaces in Java applications. I learned to create intuitive user interfaces with components like text fields, buttons, and sliders. Understanding layout managers for effective UI design is crucial. Moreover, attaching event handlers to UI controls enables user interaction, making Java applications more user-friendly. Applying these concepts ensures my software has an engaging and efficient user interface.

## **Module 3: Exploring JavaFX for Modern UI Development**

Module 3 shifted the focus to JavaFX, a contemporary Java graphical user interface library. While the development concepts remained similar, the syntax differed significantly from Swing. Learning to work with JavaFX layout "panes" and binding components to event handlers was essential. This module emphasized the importance of adapting to new technologies and exploring examples to grasp the nuances of JavaFX. Effective coding in modern Java applications requires keeping up with evolving libraries and frameworks.

## **Module 4: Advanced Object-Oriented Programming**

Module 4 is built upon a previous course's foundational knowledge of object-oriented programming. It introduced advanced OOP concepts, including the use of abstract classes and inheritance. I learned how to create classes encapsulating data and behavior, providing a blueprint for creating objects. Understanding inheritance allowed me to build subclasses based on existing classes, promoting code reuse and extensibility. Abstract classes guided the design of subclasses and emphasized the importance of method definition. These lessons in OOP enhance code organization and promote scalability, crucial aspects of effective coding.

## **Module 5: Mastering Recursion**

Module 5 introduced me to recursion, a fundamental programming technique in Java. I learned how to create methods that call themselves powerful tools for solving complex problems. Recursion often provides elegant solutions to tasks like calculating factorials. Crucially, it emphasizes the importance of a terminating condition, preventing infinite recursion. This control flow and problem-solving lesson is invaluable for writing efficient and error-free code.

## **Module 6: Efficient Searching and Sorting Algorithms**

Searching and sorting data are fundamental operations in software development. Module 6 taught me the importance of selecting efficient algorithms to minimize program run-time. The concept of "Big-Oh" analysis was introduced as a tool for evaluating algorithm performance. By choosing the suitable algorithm for the task, I can ensure that my code executes efficiently, preventing user frustration caused by slow-running programs.

## **Module 7: Java Collections and Map Frameworks**

Module 7 provided an overview of Java's collections and map frameworks, essential for managing and manipulating data. I learned about various data structures, such as ArrayList, linked lists, sets, hash maps, etc. Each data structure has unique functionalities, including adding, removing, sorting, searching, and iteration. Understanding when and how to use these data structures is vital for efficient coding. This module emphasized the importance of selecting the right data structure for specific programming problems, ensuring optimal performance in real-world applications.

## **Module 8: Exception Handling in Java**

Module 8, the course's final module, focused on Java exception handling. It introduced me to the techniques to control abnormal program termination and effectively handle errors. Exception handling is critical to coding, ensuring that programs gracefully handle unexpected situations. I learned to use try-catch blocks to catch and handle exceptions, preventing crashes and improving the reliability of Java applications. Mastering exception handling is essential for delivering robust and user-friendly software.

## **Conclusion**

These eight modules have given me a comprehensive understanding of Java programming and effective coding practices. Each module has contributed to my growth as a programmer, from object-oriented programming principles to user interface design, recursion, algorithm optimization, data structures, and exception handling. I am well-equipped to create code that is not only functional but also viable, competent, and vigorous. I am optimistic that these lessons will be a foundation for confronting complex challenges and bringing high-quality software solutions.

# **Figures Section**

## **Figure 1.**

Execution of StudentDataIO with no error entries.

A screenshot of a computer program

Description automatically generated

## **Figure 2**

Student Data is saved in the student\_data.txt file.

A screenshot of a computer

Description automatically generated

## **Figure 3**

Try-Catch handling of User Errors

A screenshot of a computer program

Description automatically generated

## **Figure 4**

Github Repo verificationA screenshot of a computer

Description automatically generated