SENG2200 – Assignment 1: Report

Isabella Andrews

31st March 2021

This report illustrates the design and coding approach taken to SENG2200 Assignment 1. Over the course of completing this assignment, the time taken to code and correct errors were tracked, and errors were logged as they occurred. The proportion of errors encountered is reported to help identify the best methods of approaching and executing a programming assignment. Thoughts on inheritance as to how it could relate to special cases such as squares and rectangles are discussed.

Designing

One hour was spent designing this assignment by working through the specifications and pulling together the relevant information and mandatory requirements. It is evident from the time spent coding versus the time spent correcting errors and debugging that not enough time was given to designing the structure and implementation of this assignment before the beginning of coding.

Coding

Approximately 12 hours were spent coding and testing this assignment. Of those 12 hours, only 4-5 hours were spent developing and coding the initial structure and implementations of the program, e.g., the creation of a circular doubly-linked list data structure, input and output formatting, calculations, etc.

Correcting

Once the initial structure and implementation were complete, approximately eight hours were spent attempting to correct an abundance of errors that predominantly affected the functionality of the program and its intended results.

Design Errors

Most errors came from the poor design of the assignment due to unforeseen time limitations. Coding began with only a rough idea of what needed to be done to achieve the required results. This approach caused the coding and implementation errors to be far more challenging to debug and resolve.

Coding or Implementation Errors

85% of errors were implementation errors, specifically NullPointerException. The remaining 15% of errors were code related, i.e., FileNotFound and InputMismatchException. The implementation errors could have been drastically reduced by taking the time to better plan and design the assignment in its initial stages.

In conclusion, planning and design are crucial to completing a programming assignment or project. Without adequate planning and design, a far more significant amount of time is spent on the assignment overall, specifically correcting human errors that could have been avoided.

How could you treat Rectangles and Squares as special cases for this assignment?

Both a square and a rectangle are quadrilaterals, two-dimensional shapes known to have four sides and four vertices or points. Using inheritance, a quadrilateral class could be designed to initialise and handle the data pertaining to the typical characteristics of quadrilaterals. A square or rectangle class could then extend the quadrilateral class, inheriting all the methods of the 'parent' class and adding methods that relate specifically to either a square or a rectangle, e.g., a square has four equal sides. In contrast, only the opposite sides are equal for a rectangle; this fact would affect the calculation of the area of the quadrilateral because it is distinct.