CS 499 Milestone Three: Algorithms and Data Structure Enhancement

# Artifact Description

The artifact I selected for the algorithms and data structure enhancement is my Arrow Grain Calculator App. Originally created as part of my software development coursework, the application calculates the total grain weight of an arrow based on its components. In its earlier form, users manually entered the weight of each part, including the shaft, and pressed a button to compute the total. The recent enhancements introduced algorithmic improvements, dynamic calculations, and real-time data flow.

# Justification for Inclusion in ePortfolio

I chose this artifact because it demonstrates my ability to apply algorithms and manage data dynamically in a front-end application. I implemented a real-time shaft weight calculation based on Grains Per Inch (GPI) and arrow length. This replaced manual entry, introducing an essential use of mathematical logic and live calculation. For example, shaftGrains = GPI × Arrow Length. This formula is automatically evaluated as the user types, using JavaScript’s `useEffect` hook.  
  
In addition, I implemented a dynamic calculation for the FOC (Front of Center) percentage, an advanced archery metric. The FOC is calculated using the formula:  
  
balancePoint = [(Tip × L) + (Insert × (L−1)) + (Shaft × (L⁄2)) + (Fletching × (L−2)) + (Knock × 0)] ÷ TotalWeight  
FOC% = ((BalancePoint − (L⁄2)) ÷ L) × 100  
  
These formulas allow the app to provide FOC feedback instantly. I also added color-coded feedback for the FOC result, providing visual guidance based on archery standards.

# Course Outcomes Addressed

This enhancement demonstrates significant progress toward the following program outcomes:

* Designing computing solutions using algorithmic principles and managing trade-offs in design.
* Demonstrating innovative use of data and logic in real-time UI components.
* Delivering coherent, technically sound, and user-focused solutions through visual feedback and live computation.

The original outcome I targeted was improvement through structured logic and real-time responsiveness. With the addition of GPI-based shaft computation and FOC analysis, I have exceeded those expectations and provided a stronger showcase of applied algorithm design in a modern application.

# Reflection on the Enhancement Process

Through this enhancement, I deepened my understanding of algorithmic thinking in UI-driven applications. The main challenge was structuring `useEffect` dependencies to ensure calculations only updated when relevant values changed, preventing performance issues and infinite loops. I also had to carefully manage the order of calculations so FOC would not be computed until all required parts had valid numeric input. Building this real-time feature reinforced how state-driven logic and algorithm design intersect in React.

A screenshot of a computer

AI-generated content may be incorrect.