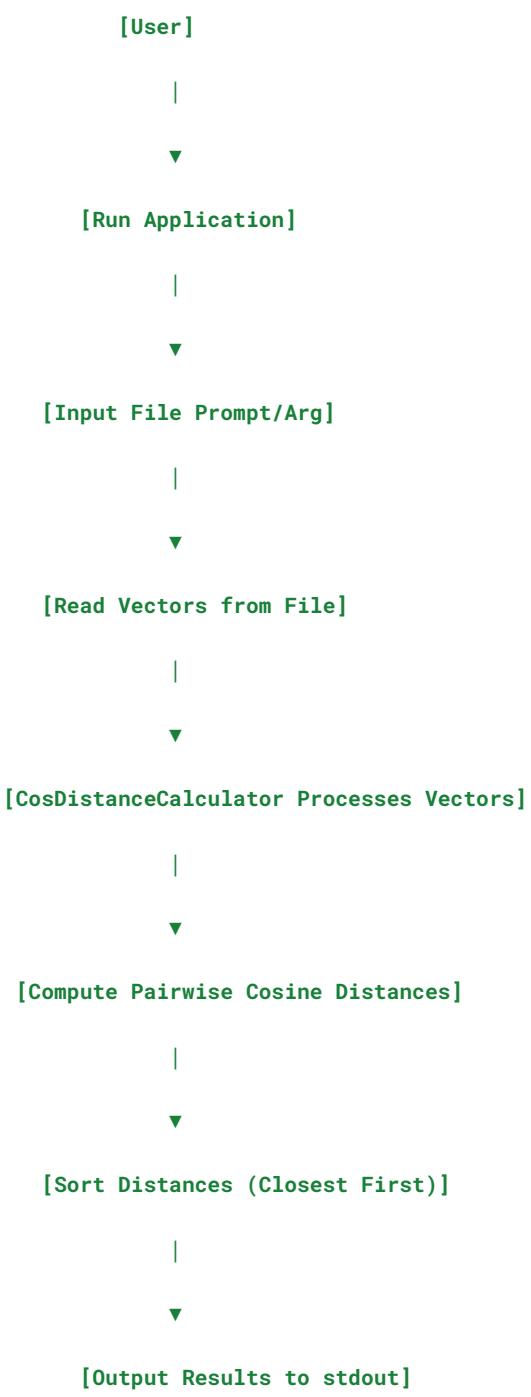
Lab Week 5: Cos-distance – Phillip G. Bradford

User-Flow Diagram



In this lab, we extend our C++ skills by creating an application that computes pairwise cosine distances between vectors in $\mathbb{R}^k$ (with $k \geq 2$). Each vector is assumed to originate from (0, 0, …, 0) and is represented by a line of space-separated double values in an input file. The cosine similarity is derived from the geometric dot product—the sum of the products of corresponding vector components—and the magnitude (or norm) of each vector is computed as the square root of the sum of the squares of its components. The cosine similarity is then calculated as the dot product divided by the product of the magnitudes, and the cosine distance is defined as 1 minus this similarity. This lab leverages these fundamental geometric concepts to quantify how "close" two vectors are based on their directional alignment.

The application is architected around a modular design with clear separation between computation and I/O. The core functionality resides in the `CosDistanceCalculator` class, which encapsulates methods for reading vectors from a file, validating that all vectors share the same dimensions, and performing the necessary computations to determine both the cosine similarity and the resulting cosine distance. By employing the Strategy Pattern, the design isolates the distance computation in a dedicated function, allowing for potential extensions such as alternate similarity metrics in the future. The project includes both header (`CosDistance.h`) and implementation (`CosDistance.cpp`) files, along with a `main.cpp` file that handles user interaction and output formatting. Unit tests have been implemented using the `doctest` framework in `test_cosdistance.cpp` to ensure the correctness of the distance calculations and ordering of results.

To run the application, compile the main source files together. For example, if you are using a Unix-like environment or GitHub Codespaces, navigate to your lab directory and execute:

```
g++ -std=c++17 -o cos_distance main.cpp CosDistance.cpp
```

Then, run the executable with an input file containing your vectors, for instance:

```
./cos_distance vectors.txt
```

The program will prompt, “Enter the filename containing vectors:” if no command-line argument is provided. It expects a text file where each line represents a vector, such as:

```
1.0 2.0 3.0
4.0 5.0 6.0
7.0 8.0 9.0
```

Upon successful execution, the application computes all pairwise cosine distances and prints the results in order from the smallest (closest vectors) to the largest, similar to:

```
Pairwise cosine distances (ordered by closeness):
Vector 0 and Vector 1 => Distance: 0.1234
Vector 0 and Vector 2 => Distance: 0.2345
Vector 1 and Vector 2 => Distance: 0.3456
```

This output confirms that the program correctly processes the file, computes the distances, sorts them, and displays the results to the user. The lab submission includes the complete GitHub repository (with all header and source files), a video demonstrating the application in action, and this one-page document that outlines the application architecture and user flow.