Concepts of C++ Programming
winter term 2022

Technical University of Munich
CIIP Group

Exercise sheet 5 — 2022-11-24

Due date: 2022-12-15 16:59

The goal of this exercise sheet, is to work with class, constructors, operator overloading and some of the standard library algorithms.

Exercise 1:

Complete the implementation of a mathematical vector class.

Mathematical vectors are expected to behave slightly different than `std::vector`, plus include a different set of operations. Compared to `std::vector`, one would expect to be able to add and subtract two vectors, add, subtract, multiply and divide a vector with a scalar. Other important operations are norm (e.g. the Euclidean norm), or other coefficient-wise transformations (e.g. log-transformations and such). On the other hand, operations like `push_back` are not expected for such a class. In general, a mathematical vector is not necessarily expected to grow dynamically.

For example, this would be simple column vector:

$$[1, 2, 3, 4, 5, 6]^T \in R^6$$

Or more general:

$$[x_1, x_2, \dots, x_n]^T \in R^n$$

where each $x_i, \forall i \in [1, n]$ is a coefficient of the vector. It can be scaled by a scalar:

$$[x_1, x_2, \dots, x_n]^T \cdot s = [x_1 \cdot s, x_2 \cdot s, \dots, x_n \cdot]^T \in R^n$$

The norm or length of a vector is defined as:

$$||[x_1, x_2, \dots, x_n]^T|| = \sum_1^n x_i \cdot x_i$$

A normalized vector is one, where the norm is equal to 1. To normalize a vector, you can divide it by its length:

$$\hat{x} = \frac{x}{||x||}$$

Finally, note that thou math starts counting at 1, the vector you implement starts counting at 0.

You need to: implement all constructors of `Vector`, its member function and the free functions declared in the `vector.h` file.

If you are interested to look into a proper linear algebra library, you can check out Eigen or blaze. Both are (mostly dense) linear algebra library, with Eigen being quite well known. Eigen has been around for quite some time, and you will the code is quite complex. But it is used extensively in e.g. Tensorflow. blaze on the other hand is more modern and the user facing code is rather elegant. There are many more sparse and dense linear algebra libraries out there. However, many of them are written in C, Fortran or are a very thing wrapper around the former two. Hence, they shall not be mentioned here :-)

Corrections:

Some comments in `vector.h` are not precise or wrongly copy and pasted. This list is the errata:

- `min`: This function throws, if the given vector is empty.
- `max`: This function throws, if the given vector is empty.
- `argmin`: This function throws, if the given vector is empty.
- `argmax`: This function throws, if the given vector is empty.