# Monday

## 7:00-8:30 Reading 3D math primer

* Freshen up on vector notations and meanings.
* Learn about the application of the dot product, according to the book the dot operation for vectors is essential for 3D maths. Understanding the dot product should help me understand other 3D math operations.

## 9:30-10:45 Listen in on project start presentation

I am expected to make a renderer. I am not sure what a renderer is supposed to do and how to make one. I should first research into what ray tracing is and how it is applied.

## 11:00-13:00 Research what ray tracing is

I was reading [this](https://www.scratchapixel.com/lessons/3d-basic-rendering/introduction-to-ray-tracing/raytracing-algorithm-in-a-nutshell) but do not understand the ray tracing solutions that they discuss. I should research more into general 3D graphics before researching into ray tracing. I’ll try to study the material a bit more focused.

I have studied this lesson which I’ve learned the following things from.

Ray tracing is a method that tries to simulate real-world light behavior. We cast a ray through a pixel from our image plane. The end of the ray is the color that the pixel will take.

Ray tracing happens as follows.

We cast a primary ray through the center of a pixel on the image plane.

If it collides with an object create light ray towards the closest light source.

If light ray collides with other object make it shadow ray.

If shadow ray give black color (or dark depending on material)

If light ray return normal color (or do something depending on material)

The only things that I’m not really sure about are what happens when there are multiple light sources. Is tracing a ray to the closest light source really the appropriate solution? Should I create light rays for all light sources and use an average of the results.

Also, What happens when the primary ray doesn’t touch an object (it goes straight to the sky). Should I make the sky an object so that it can reflect the color?

## 13:00-15:15 Create a ray tracer program following a tutorial

I will read more about ray tracing and existing algorithms.

The teachers have recommended me to look through “[Ray Tracing in a Weekend](https://drive.google.com/open?id=1_MZBMUSO25pg1gyeHa_D8WXu7r37CvC6)” while also coding besides it. So I will set up a simple solution to get some programming started.

I will be using [SFML](https://www.sfml-dev.org/) because it supplies me with a window that I can work with. It also has an image class in which I can change each pixel with ease. This should help with getting the correct colors for the ray tracer. Before I will create a ray tracer model, I will follow the tutorial mentioned above to get a correct ray tracer running.

I found a way to modify individual pixels with SFML. With this, I can start following the tutorial.

## 16:00-17:00 Fill out the MathLibrary source file

## 17:00-19:00 Research a bit about libraries

# Tuesday

## 12:00-13:30 Research GLFW

I want to try and use something different than SFML. To open a window, I wanted to look at GLFW. GLFW is a library that initializes and managed windows for OpenGL. I was reading about OpenGL contexts and windows but it seems that I need way more than that. I also need some library that can load all the OpenGL functions, something like GLEW and I need something that can load images and textures for OpenGL. Although I’m not too sure about the last one because I don’t have to load any textures.

Either way, while I’m interested in learning all this, it isn’t what this project is about. I am supposed to write a Ray Tracer, that is the focus of this project. Researching all these OpenGL libraries is not necessarily so I will focus on the Ray Tracer itself.

## 13:30-14:00 Gloss over CppCoreGuidelines

This is a huge text file that I will probably have to read. Because I have to invest time in it one day, I want to know how I can effectively read the file. I will gloss over it and try to understand the structure of the text file.

## 14:30-17:00 Read Ray Tracer in a Weekend

I’m having a bit of trouble understanding a formula:

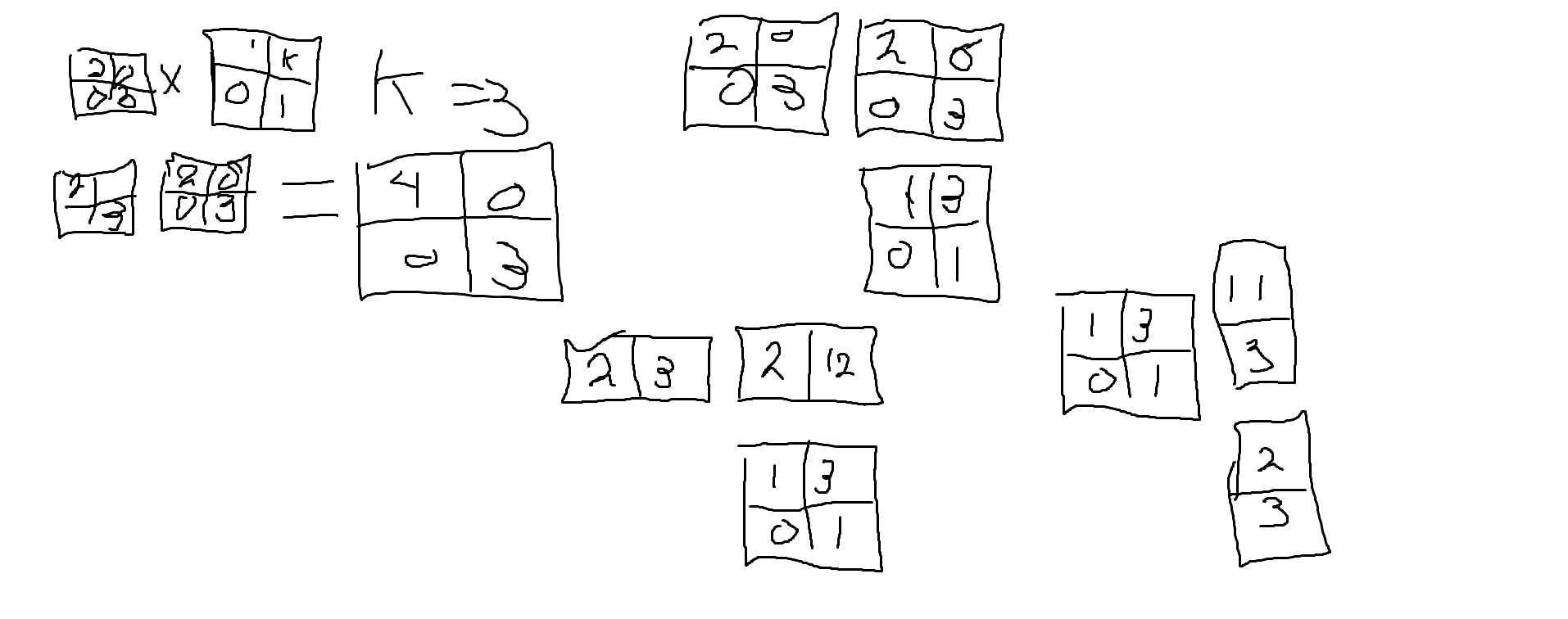
**dot(**(p​ - C)​ ,(p​ - C)​**)** = (x-cx)\*(x-cx) + (y-cy)\*(y-cy) + (z-cz)\*(z-cz).

After taking a rest, I will look at it again. Maybe venting my head will help a bit.

# Wednesday

## 10:00-12:00 Vector Algebra

I learned about matrices but had a hard time understanding it. The presentation was a bit fast. I will look into the sheet in my free time and ask questions if I don’t understand things because I don’t really know what to ask at this moment.

I do understand how to multiply matrices now. That was something that I struggle with quite a bit because I couldn’t understand why the dot product has to be used. Vector multiplication is fundamental knowledge for more advanced techniques. I will study the sheets again with that knowledge.

**I will be buying a new notepad soon**. But the problem was that I didn’t know how to apply these transformation matrices. If you read this from left up to right down you can see how I tried to apply different methods to apply this shear transform matrix. In the end, I got the desired result when I used a column vector. I will discuss this with Bojan because he gave this lecture. I definitely need to understand transform matrices because they are used extensively in 3D programming.

## 13:00-14:00 Analytic geometry

Analytic geometry has many things in common with vector algebra. It is about the shapes that appear when objects intersect with each other. I'm not so sure how I would apply this in my studies.

## 15:00-16:00 C++ Guidelines

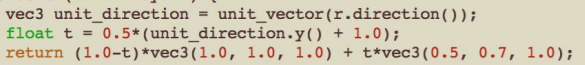
It was a discussion of the core guidelines. I have already read that myself.

## 16:00-17:00 Graphics expert group

This week is about ray tracing which is very relevant to this project. It was a bit beyond me though. Instead of talking about how to make a ray tracer, it discussed how to use a ray tracer with DirectX 12.

## 18:00-20:00 Continuing with Ray Tracer in a Weekend



I finally understand why the tutorial uses all those magic numbers. For example, this expression:

Is because we have to make a gradient. By using the y-direction and dividing by two we get a value between 0 and 1. The tutorial uses a range of [1,-1] so we add 1 to make sure it is positive and divide it by 2. The two vectors that are returned are the two colors to blend between for the gradient.

# Thursday

## 7:15-8:15 Reading math fundamentals book

## 9:00-13:30 Filling out previous days in work and learning log

## 15:00-16:15 Fill up MathLibrary

## 16:30-17:00 Continue with the tutorial

“or expanding the full form of the ray p(​ t) :

**dot((A​ + t\*B ​- C)​ ,(A​ + t\*B​ - C​)) = R\*R**

The rules of vector algebra are all that we would want here, and if we expand that equation and move all the terms to the left hand side we get:

**t\*t\*dot(B,​ B)​ + 2\*t\*dot(B,A​-C​) + dot(A-C,A​-C​) - R\*R = 0**”

I don’t get how the author doe this step. But I do understand what he is trying to achieve. I will ask the teachers about this.

# Friday

14:00-15:00 Change format of work log