Thematic content overview:

- The joy of making things.
- Starting impressions - simple interactions and basic concepts - layered complexity - wide range of applicability - translating examples into specific use case scenarios.
- The transition to OOP was hard conceptually - layers of indirection.
- Purely analytical VS experimental approach to software design/ engagement.
- Debugging - stepping beyond theoretical analysis and experiencing the complete process.
- Methodology adoption and personalization; how a sense of structure helps focus the learning experience and how you end up adapting that methodology to your personal understanding.
- Greater creative freedom and personal purpose alignment status satisfactory (very).

Reflection:
Making things happen on screen is just so fulfilling. The experience of writing code with a result in mind, then seeing that result made me so happy from the very first ellipse that moved across the screen. Programming, I've concluded, is just like writing fiction except the things you write actually become true. It's funny how a language that seems so obscure and technical at first glance turns out to feel so natural, immediate, and even more direct than some graphical interfaces once you start becoming fluent in it. The only way I can account for this experience is by realizing that everything that can be done in a computer is fundamentally an abstraction of a lower-level concept or process. Programming languages are in that sense a more natural way of communicating with the computer.

Every little introduction of a new concept opened up my mind to the evidently endless possibilities. The exercises allowed me to immediately apply my ideas and test my understanding of the bigger structural concepts and the small behavioral operations. By peaking ahead into future lectures I found I could come back to previous exercises and apply my newly acquired skills to explore beyond what I had previously conceived of. It was a delight to figure out how to adequately translate my basic learnings to a 3D coordinate system using webGL, gradually expanding the complexity of my programs. The first 3D cubes movement exercise was an essential step in being able to tackle the 2x2 rubik's cube simulation for example.

OOP was harder to internalize than purely functional programming. I must admit that I struggled a bit at this point of the course. But "getting it" was correspondingly rewarding. And it placed past experiences trying to learn it on my own into perspective.

The fact that all game code bases in engines use OOP, in all it’s indirect interconnected complexity, has been the biggest obstacle for me when examining prefabs, Schematics, and references (in Frostbite, Blueprints in Unreal) when attempting to comprehend even the simplest behaviour. I’m now considerably more confident that I could navigate those environments with greater ease. And it’s striking to realize how something that seems utterly unapproachable from a purely analytical, reverse-engineering, point of view becomes clearer, in a general, holistic way with a little bit of guidance and the space for forward experimentation.

As a result of building my own programs from the ground up at a small scale and having complete authorship of every mistake and necessary debugging, I now feel like I have acquired a more symmetrical view of the development process. What I understood mostly in theory about how videogames worked is now considerably enriched by a fundamental base of practical experience. A base of techniques that I've been able to adapt to my own way of working and reasoning through problems that, although perhaps not representative of the best way of doing things, I can reflect and improve on.

I feel like I have a lot to learn, and that I've only scratched the surface of the potential of my new-found powers. And I feel like I am exponentially increasing my ability to express myself creatively. I am getting closer to my goal of understanding the underlying processes of artistic production software and creating both evocative experiences and tools to create those experiences.