Written Response
Methods of Iterating
Draft 01

Creative coding inhabits a space where logic and imagination coexist, at the intersection of technology and art, allowing many possibilities: from sleek, high-fidelity imitations of reality to intentional distortions that exposes the workings of the machine behind it. I want to focus more on the intentional "imperfection" of creative coding: by making the computational process more visible, the 'code' also becomes a collaborator in the process, like a notebook laid open.

alidasun (2025)

For this assignment, I chose to replicate Alida Sun's artwork made through the medium C++ and AV synesthesia (alidasun, 2025) to try my hand at creative coding. Sun's work aligns creative coding with resistance against big tech. Her "distortion" and "low-poly" aesthetics function as a form of refusal of smoothness and efficiency, and instead favours expressive friction.

In my iterations moving forward, I want to experiment with visual and sonic disruption and interaction while learning how to code, as a way of exploring the methods of coding "fluency". While AI-assisted coding is a pragmatic support, I also want to echo Sun's resistance by adopting the early pedagogical approaches of coding. I want to dive into the iterative process with repetitions, exercises and incremental development and emphasize on learning through sustained effort by engaging in its community and literature.

Reference:

alidasun (2025) 'Curiosity + play = the most powerful forms of research' [Instagram] 01/12. Available at: https://www.instagram.com/p/DRua5lOiGlR/ (Accessed: Mar 2).

Written Response
Methods of Iterating
Draft 02

Somewhere in my childhood home are stacks of books on coding C++ and JavaScript that belonged to my mother. She could code computer games like the back of her hand, but ironically, I never saw her play any of them. She possessed the knowledge to create them, but like the virtue of code, she also remained on the backend of her work. Intrigued by this query, for this assignment I chose to assess the political off-screen of one of the most technocratic artistic expressions: creative coding.

When I started researching how to code, I came across several hurdles: firstly, the subject seemed infinite given my little to no understanding of coding, and secondly, the conundrum of using AI (my reference held a very anti-AI stance, which was partly why I chose it). It was Donna Haraway's *Situated Knowledges* that provided some clarity for me in this experiment. She advocates in her article a constructivist approach to epistemology (Haraway, 1988, p.576): no knowledge starts from scratch or exists outside of personal experience (much to the provocation of scientific thinking), but instead is built on social, historical, and even bodily contexts. Haraway (1988, p.596) concludes that there is no knowledge without a body-based mind to situate it.

This worked on a couple of layers in my case: it is also how code was created initially, and how it is now used. Coding began as a clerical process before it earned its capitalistic fame, and was layered into computers by women delegated to clerical work in a male-dominated workforce. It started with the inspirations of a typical "abstract masculinity," broken into binaries and dualities. Now coding has taken a new shape, especially for beginners such as myself: it builds on pre-existing libraries, optimised software, and also AI (which in itself is a constructivist product), as well as the help of many technicians who explain its myriad workings. With a collectivist form of knowledge supporting the process, the act of learning code has become constructivist, much to Haraway's appreciation.

Does that include AI? AI exists at a tricky intersection of tech-colonisation and accessibility. Its database has mostly been built with Eurocentric and Anglophonic information archived during an unequal world, and it also has a detrimental ecological impact. But it is a learning model—it still rebuilds and proposes a space to amend its tendencies. My workaround for this was referring more to videos and technicians who provided me with more esoteric and specific knowledge regarding my questions and minimising the use of AI to cleaning up

the code.

Haraway strongly proposes the presence of 'bodies' in the production of knowledge, and to echo that, I maintained the aspect of interaction in my iterations. In my project, the code (essentially a binary, dichotomous, and predisposed system) is only activated through participation. Bodies, then, become the force that subverts the coercively standardising gaze of technology. In conclusion, my iterations set out to create a dialogue between the ever-widening imperialistic tendencies of new technologies that run on invisible and uncredited knowledges of others, and the idea of needing the participation of subjective and randomised bodies to reclaim power from them.

Reference:

Haraway, D., 1988. Situated Knowledges: The Science Question in Feminism and the Privilege of Partial Perspective. *Feminist Studies*, 14(3), pp.575–599. doi:10.2307/3178066.

Written Response
Methods of Iterating
Draft 03

For methods of interating, I use Creative Coding to explore as my tool. In this document, I trace my process of understanding fluency in creative coding as a language, developed through play, mimicry and error within a deliberately low-tech, intentional approach. Rather than approaching coding as a skill to be mastered in my imitation of the reference, the process retrospectively resists the dominant "Learn to Code" paradigm (Lorusso, 2023, p.33), instead adopting a "Code to Learn" attitude that turns the act of coding into craft.

alidasun, 2025

I chose Alida Sun's work as a reference because her practice itself is an array of iterations: quick sketches written in C++ that resist the smooth, hyper-efficient aesthetics associated with technocratic art. Her work foregrounds distortion, friction and imperfection. Attempting to replicate aspects of her practice allowed me to frame my own learning as form of resistance - by prioritizing process, intentionality and experimentation over optimization.

My point of entry into creative coding was grounded in a rudimentary understanding of HTML. I began working with P5.js through YouTube Tutorials, copying programs with limited comprehension of how they functioned as a whole. This phase of "mimicry" became my first mode of learning. While it produced visual outputs, the process felt disingenuous, exposing a disconnect between outcome and understanding.


alidasun, 2025

This discomfort led me to reconsider coding as a linguistic system than purely technical one. When technicians in the Physical Computing Lab described my code as "mine" and not something that can be reverse-engineered, I realized I was in an early stage of language learning (similar to, say, babbling), piecing together grammatical structures without full fluency. With the support of technicians and AI, I produced a first initiation of the reference, accepting partial comprehension and error as intrinsic to learning.

In subsequent iterations, my process shifted from imitation to play. Working collaboratively with the technicians, I started to change parameters (like colour, length, sound, text) to develop my language "fluency" through interaction than linear instruction. "Syntax"

equated sentence structure, “functions” acted as verbs and “variables” became, a form of collective nouns. Fluency emerged increased “literacy” of the language.

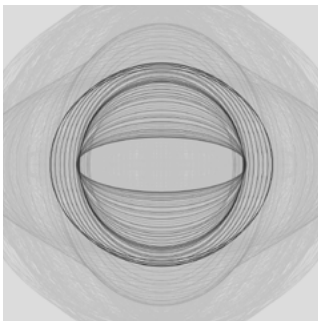


Fig. 1. Mouse-interaction based sketches

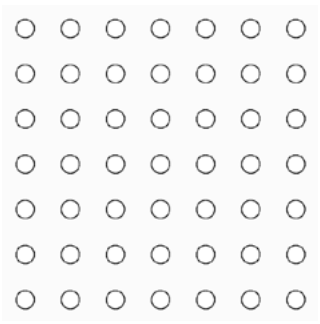


Fig. 2. Geometric Explorarations

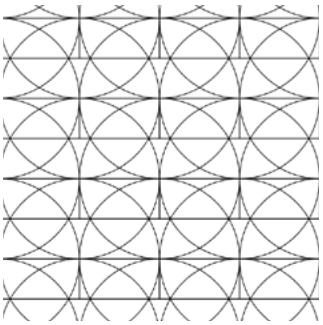


Fig. 3. Geometric Explorations 2

A workshop on Procedural Drawing further reframed creative coding as a communicative system rather than a simple input-output system. Exercises inspired from *Conditional Design Workbook* demonstrated how constraints could generate unexpected outcomes, just the way play functions within its own internal logic. This also made creative coding more about curation than execution - craft blooming within a set of rules. It was at this point I also flipped my view of interacting with code - a given set of rules became the “coder” and my execution of the rules helped me inhabit the role of the “compiler”.

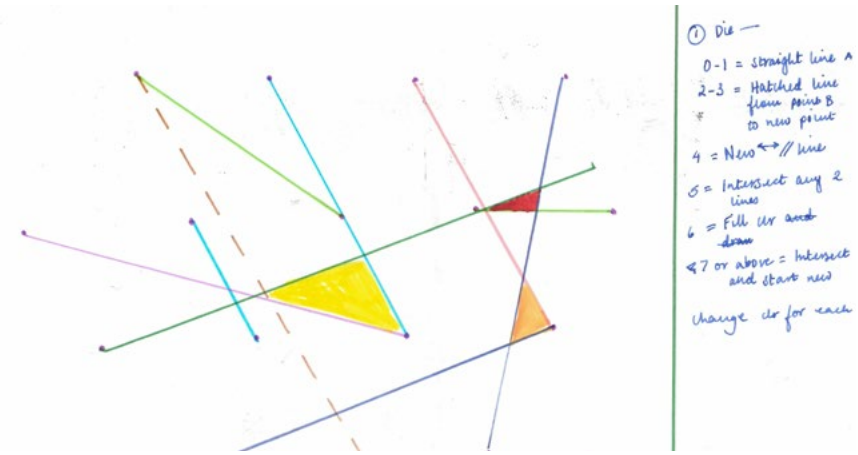


Fig. 4. Computational Design Handrawn Exercise

Throughout the process, interaction remained central to my interactions. Drawing on Donna Haraway’s concept of Situated Knowledge, the work requires bodily participation (Haraway, 1988, p.596) to activate the code - yet again emphasizing on play, and disrupting the impersonal logic of technology, situating knowledge as embodied and relational.

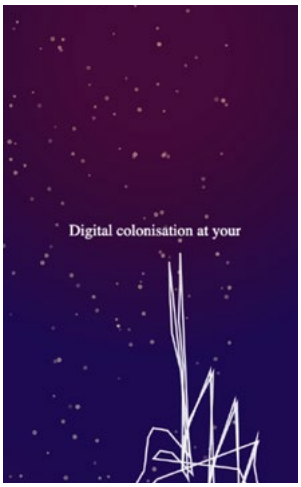


Fig. 5. 'Fingertip'- Recognition text reveal

Fig. 6. More details on iterations like adding colour

Fig. 7. A "rubberband" interaction with hands

Note-taking became a material component of my learning. I printed and annotated my code, marking syntax, recurring structures and errors. Errors - such as unfinished strings or undefined variables - became to read as gaps in translation of language instead of failures of running the code. Errors revealed the scaffolding of the language, exposing moments of hesitation similar to when learning to speak a new language. In this way, error functioned as a voice than a flaw.

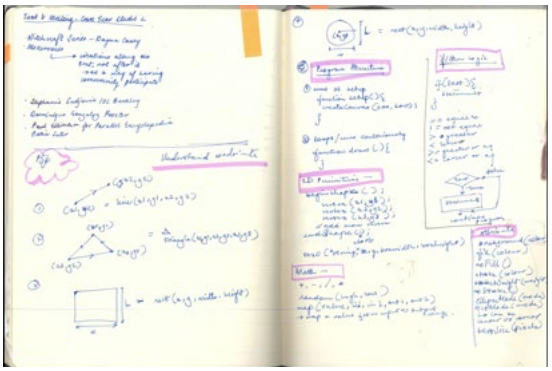
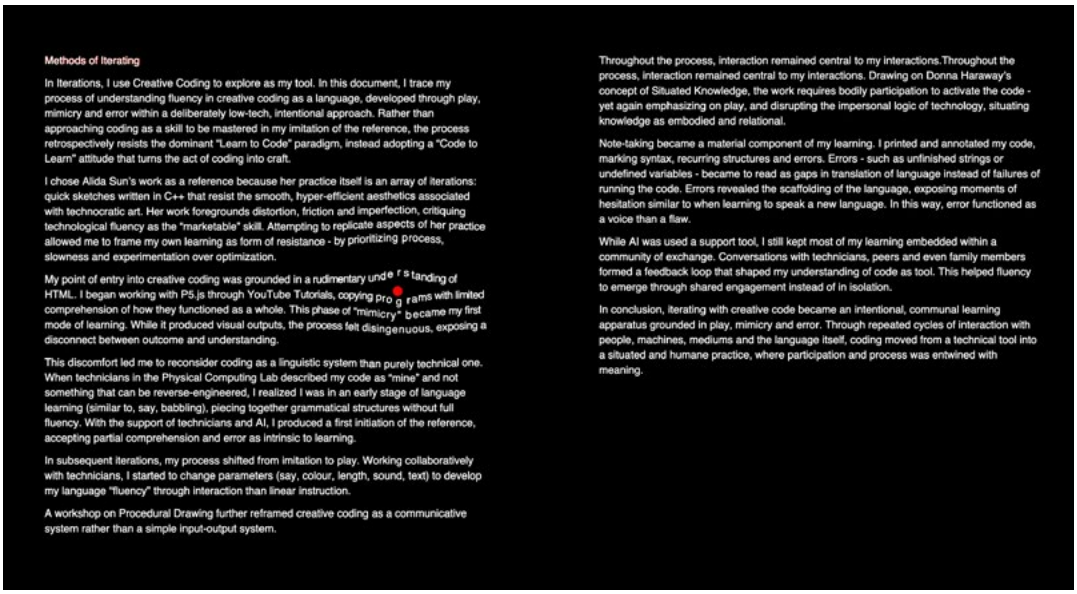


Fig. 8. Notebook full of handwritten notes and definitions

Fig. 9. An "decoding" of code through annotations

While AI was used a support tool, I still kept most of my learning embedded within a community of exchange. Conversations with technicians, peers and even family members formed a feedback loop that shaped my understanding of code as tool. This helped fluency to emerge through shared engagement instead of in isolation.

In conclusion, iterating with creative code became an intentional, communal learning apparatus grounded in play, mimicry and error. Through repeated cycles of interaction with people, machines, mediums and the language itself, coding moved from a technical tool into a situated and humane practice, where participation and process was entwined with meaning.



Fig, 10. Final written submission

(The web version of this text is hosted here)

References:

Conditional Design (2013) Conditional Design Workbook. Amsterdam: Valiz.

Haraway, D., 1988. Situated Knowledges: The Science Question in Feminism and the Privilege of Partial Perspective. Feminist Studies, 14(3), pp.575–599. doi:10.2307/3178066.

Lorusso, S. (2023) Introduction. In: Graphic Design in the Post-Digital Age: A Survey of Practices Fueled by Creative Coding. Set Margins, pp. 33–36