Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Computer Science for Digital Media

Teaching Queer Code: Educating Computer Science under Social Aesthetics

Master's Thesis

Artur Solomonik
Born Jan 10, 1997 in Tscherepowez

Matriculation Number 115715

1. Referee: Prof. Dr. Eva Hornecker

Submission date: June 8, 2023

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, Germany, June 8, 2023

. . . . . .
Artur Solomonik

Abstract

Creating software is closely tied to overcoming knowledge gaps between peers, automating busy work and improving code bases. The underlying processes function under a collective effort to maximize profit. With a multitude of ever-growing research tracks, the creative possibilities and social discourses seem diverse and fruitful. This promise, however, is challenging to upkeep as the industrial complex highly relies on the labor of writing code in the most effective ways possible. Technical evolution is rarely discussed on its ethics and politics during education, while the people affected by it do not get the time and resources to build digital fluency. The need for software developers and educational imbalance paint a narrative of the powerful and future-shaping developer. Learning resources closely relate to learning specific programming languages or frameworks, understanding the principles of code delivery, and working effectively in corporate teams. Swiftly, the potential of an unexplored medium such as code becomes an instrument for the industry, rather than subject to critical analysis and playing. What seems to be an opportunity to shape ideas in ways unimagined, becomes a commodity that is ought to be taught by the complex power dynamics of the cisheteropatriarchy.

Computer science research groups, art collectives and (cyber-)feminist institutions embrace the idea of queering coding and standing up for the notions of creative processes in technology. Understanding the prevalent power of soft- and hardware and the ones who shape it, individuals can be educated on the focus of computer science by introducing feminist ideas on approaching the medium. Forming a collective consciousness on the impact of the medium code and applying it in technology even outside the generative context enables novel interactions with technology and the people influenced by it. With that in mind, while this thinking saw its implementation in coding projects with children, university courses, teaching resources, public art installations, and video games already, digital, independent learning platforms and respective online infrastructure remains mostly unexplored.

In my thesis, I present an online education platform for exploring the principles of computer science, its societal implications and implementations in art, culture and prose. It is to show that by teaching creative code through a feminist gaze, people can critically comprehend the ideas of computer science. Based on the principles of Aesthetic Programming, alongside a corpus of resources from leading research on creative programming and curated guides, I

provide an editor environment to build pseudocode through the possibilities of digital internet art. Over the course of two weeks, individuals work on their own programs while reflecting on their works and ideas. Through a probe study, and an interview, the individuals' assessment on their personal growth in the field reveal the potential on teaching queer code.

With a data set consisting of digital collages, questionnaires, and interview transcripts by each participant, the quality of the course is assessed, and individual learning processes identified. Using Thematic Analysis, prominent themes reveal key aspects that need to be well-defined for conducting future programming courses leaning towards a post-digital paradigm while reflecting on the effectiveness of communicating social, and political problems adjacent to the technological complex. By evaluating the experiences of every learner, we can establish emerging needs, and abilities of a course participant who is *learning to code* effectively, the Peri-Digital Learner.

Contents

Glossary

Pseudocode: Semantic representation of algorithms and programming processes through custom syntax. Usually defined by scientific convention, its purpose is to explain the code that is executed elsewhere.

Pseudoprogram: Free interpretation of pseudocode through the addition of custom styling, unconventional syntax and simulated execution. By running the code, choices in both style and program logic influence the visual feedback. Visual feedback is provided through graphical animation. Lines of pseudocode are tagged in their abstract purpose, e.g., they generate, execute, print, assign, or unassign.

Pseudoprogram Editor: Visual editor inspired by graphical programs and IDEs. They create collage-like animations that feature pseudocode and simulate its execution through visual feedback.

Programming: Sequence of commands abstracting flows of action, used mostly to describe the working of computers. By mending human language to a form that machines will understand, we define programming languages by how and how far we approach machine language syntactically.

Aesthetic Programming: Teaching of coding fundamentals through accompanying discussion of the sociopolitical aspects thereof. Learning different aspects of programming includes a supplementary analysis of their impact on society and their creative potential.

Code: Syntax used for programming, largely used synonymously with programming. When programming is a term more related to the abstract concept of designing requests for a machine to execute, code is related to the explicit syntax used.

Queer Code: Queering in a technical context means rethinking concepts, shifting the inherent paradigms and actively disengaging from its value in the

industrial context. Queer code describes machine logic focused not on performance and consistency, but on creative expression and abstraction of concepts that are not necessarily to be executed.

Cyberfeminism: Umbrella term for describing the feminist movement based on experience of individuals in the digital space. It underlines the value of the internet as a political space that is largely influenced by the industry. The goal is to reclaim the internet as a creative medium, providing room for marginalized voices in new emerging medial spaces.

Post-Digital: A paradigm to approach digital media through a lens where technology is no longer a part of society. It enforces the raw, and tangible confrontation with a medium to invoke critical thoughts, understand the medium's essential nature, and approach technology less biased from the industrial complex.

Materiality: Introduced by material feminism, materiality refers to the individual's ability to embody the medium by tangible experiences. Learners are perceived as actively engaging with the non-human medium, and their environment during learning processes. It resembles a basis to identify power dynamics within a context such as technology.

Zine: A zine is a publication medium used to convey information without the need of dedicated printing. Usually provocative, with a focus on self-expression, or aiming to inform on a special topic. As part of a kit, a zine can be used to inform participants and provide a physical representation of their progress during a study.

Tag: Tags specifically used for thematic analysis that try to attach context to an artifact of the study. During the analysis, the code collection is revised iteratively through summary and addition.

Theme: Final finding of the Thematic Analysis that forms from iterations of the entirety of Tags. Establishing themes of the study results provides concrete concepts from the data of a qualitative study.

Peri-Digital: The transition between the Digital, and Post-Digital way of designing a course on digital issues. This style relies on an adaptable course form that is able to educate learners through both, digital, and post-digital concepts. The individual is autonomous and has full control over what style adheres to their current goals and needs.

Acknowledgements

I thank my supervisor, Britta Schulte for their insightful support to make me not lose track, and focus on my first considerable contribution to the field of queer computer science. I thank Prof. Eva Hornecker for accepting me as a thesis candidate, and standing up for me, so I could write on a topic close to my heart, yet far from the curriculum of Computer Science for Digital Media at the Bauhaus-University of Weimar. I thank Katta Spiel for their willingness to be an external second referee, and their impulses at the early stages of my work. My thanks go to the Faculty of Media, and Prof. Benno Stein for accepting the thesis, and the application for an external referee. Thank you, to every participant of the study, who managed to provide this work with creative insight, and moral support to continue my work on this unexplored topic.
I thank Winnie Soon and Geoff Cox for their great course on Aesthetic Programming that inspired the design of this research. Any topical writing in the LYRE zine, including the course material hyperlinks, must be attributed to them. Mindy Seu's Cyberfeminism Index helped me greatly source the knowledge necessary to discuss the topic of cyberfeminism in a meaningful way. My gratitude to my last seminar at Bauhaus-University, the Bauhaus.Module *Technik und Gender* by Christin Sirtl and Britta Schulte which inspired me to pursue and enable this topic. I thank the Media faculty for shaping me as a student to be able to speak on this topic from a perspective of a computer scientist. I also thank the authors of the webisthesis template for their work. Further gratitude goes to the HCI department, which helped me evaluate my work in open discussions, and publically address the research field respectfully. All my love to friends and family, who helped me overcome doubts, and difficulties.

This thesis was written to educate on unjust power dynamics appropriated through technology. To honor the work of the feminist movement, and the minorities affected by discriminatory systems, I strive to help people navigate technology by themselves.

Chapter 1

Introduction

Teaching computer programming, in its prominence, is a vital process to familiarize people with creating, maintaining, and improving software. Despite its seemingly complex and mathematical nature, Computer Science remains to be growing in attractiveness for young people to pursue.[Besart, 2023, Zahidi, 2023] The Digest of Education Statistics for 2021 reports a steady increase of annual Bachelor graduation of computer scientists. From the years 2015 to 2021, computer, information sciences and support services graduations increased by 33.7% implying a significant popularity of computer science related degrees in general. With a high demand of programming labor within the industry[Masterson, 2021], companies release their own educational programs[AFE, Fac, Goo] to enable a swift, cheap and engaging experience to welcome young, able and motivated staff. Coding is perceived as lucrative and exclusive to people who understand and memorize the intricacies of modern programming frameworks, while areas outside the commercial scope are rarely explored. Moreover, the increasing influence of the industry on programming education is reflected in code that is written to be efficient and marketable, as well as releasing products that appeal to as many people as possible. This trend not only limits the potential of code, but may be exploited by power dynamics where certain demographics are unfamiliar with the underlying processes. Digital literacy becomes essential for navigating through the infinite virtual landscape, yet its education becomes increasingly difficult when the first resources online will teach the technology of companies. These popular learning tools focus on successive, gamified, online interfaces to teach fundamental coding principles, yet relevant social implications, expressive potential, and power dynamics remain ignored. Acknowledging these factors during learning processes may teach the fun and alternative way of thinking of computer science, but enables critical reflection of new and old technologies.

In this thesis, I present a learning platform that focuses on removing executable code from programming education, making full use of the concept of pseudocode, a coding convention that focuses on a program's concept rather than its execution. Approaching this coding style during first programming experiences eases individuals into the topic field without relying on learning biased syntax, or administrative tasks during setup. While most learning platforms already addressed this problem with embedded coding environments, they remain dependent on industry standards that exclude self-expression from initial learning experiences. By reading about current discourse on technology, exploring art of various media and reflecting artistically on the newly learned concepts, individuals may find their own path in the field of discussions. Through an alternative representation of a computer program, the platform shall spark interest and bridge the gaps between people and code without relying on industry-driven courses. With the goal to shed light on the creative possibilities and potential threats of this medium, I try to improve digital literacy among people of all age groups. This effort is supported by established resources that adhere to queerfeminist research and a conscious effort to communicate power dynamics in the learned medium. What remains to show is that a programming course on the social and mathematical fundamentals of computer science conducted over creative pseudocode will produce effective learning experiences.

To identify such learning experiences, a probe study was conducted to provide an environment that encourages play with a medium that is usually not presented in a tangible fashion. A kit consisting of a digital playground alongside an analogue zine invites participants to overcome the boundaries of programming as an abstract concept that takes place on computers only. Based on the work by Soon and Cox and their referenced learning material, I designed an easily digestible zine with excerpts and graphics. Designed to give a brief introduction to the topic, embedded codes lead participants to a graphical editor environment giving access to further digital resources as well as the ability to create pseudocode based on open, thought-provoking questions. Through animations, graphics and prose, code can be enhanced to become a form of expression that doesn't require people to be familiar with neither such graphical interfaces, nor programming syntax. With submissions for every revised chapter, the study simulates a course experience that is in favor of the participants' freedom to maneuver through the complex themes of programming while pursuing projects based on their individual stance and opinion on every topic. A successful course completion comes in the form of a final interview that captures the participants' motivation behind their work as well as their personal experiences with the study.

A final thematic analysis of the interview transcripts under examination of the submitted pseudocode collages, and their inspiration, revealed the findings supporting this form of coding education. By examining the artifacts to identify traces of possible leanings alongside evaluations of the course form, multiple iterations of reflexive analysis shall reveal strengths and weaknesses of the course form at hand. Patterns emerging from the set of findings, will give insight on the quality of the course form, and how it was reflected in the participants' results.

Chapter 2

Background

With computing as an increasingly vital part of society, discussions on the technologies involved have been widespread. Its value as a powerful tool to automate tasks, organize social structures, or schedule our daily activities has been helpful not only for companies, but for many people with access to the internet. This infinite space, that would be the home for interactions between these parties with diverging interest in the medium, has created power structures that have to be explored to enforce a virtual life for people online without risking their safety. Access to computing is the ability to create program code, and the education thereof remains to be understood on itself.

Teaching how to code has evolved over the steady evolution of online structures. With books on programming[Matthes, 2019, Stroustrup, 2014] making room for online courses[Harvard, Imbrizi, 2021, Portilla, 2021], and corporations providing the skillets they are seeking for free[AFE, Fac, Goo], the resources seem to be plentiful. Online learning platforms have made a great effort in providing live coding platforms for users to feel more involved in the coding process by solving pre-defined problems, running validity checks, and discussing problems with other learners. With a course curriculum tied to the programming languages learned which feature prominent market-specific programming languages, the journey of a programmer seems to be set in stone by industry-funded learning platforms[Campbell, 2021, Lunden, 2021] and constantly updated frameworks by other developers. What leaves to be explored is a discussion of these technologies on a social layer, that remains ignored throughout such a journey.

In an exemplary design on ethics and social responsibility in the CS curriculum, Martin and Weltz argue the value of discussing inherent issues with undergraduate students. The guidelines introduce critical questions early on to

progressively build social thinking when designing software. Main concerns about topics such as user privacy, system security, artificial intelligence, or algorithm performance to provide a curriculum can only be realized under supporting infrastructure. Under pedagogy-driven principles and proposals on time-management, the motivation for a more social computer science curriculum is eminent. Its realization, however, places a great deal of responsibility and agency upon students of differing goals and values. With an exhaustive and diverse program such as CS, students may feel overwhelmed or irritated with tackling problems that form unanswered questions to their interest in the field. That *learning to code* has been systemically ingrained into study curricula where code is exclusively functional and a skill popular on the job market[Williamson, 2016] is a promise for unreflective practice for developing future-shaping, social technology. Programming remains a field that pushes away anyone but men in an effort to shape the perfect programmer who can both communicate with the masculine machine, and the feminine human - even to this day.[Tassabehji et al., 2020] One way to shift this paradigm is presented by feminist research, which advocates for shaping technology to embrace creativity and self-expression.[Peppler and Kafai, 2005] Showcasing that potential and inviting students to open discussions on the field creates a synthesis of technological and social aspects of CS without putting either in disregard. Re-thinking the established systems that infer unjust dynamics of power that remained untouched over the past decades means to queer such systems.

Queer computer science is a young research track footing on fundamental feminist research and the related cyberfeminism movement. The term was introduced in 1991 by the artist collective VNS Matrix in their provocative piece on gender and identity in a world of new emerging media.[da Rimini et al., 1991] In this manifesto, the collective advocates for women's empowerment and dismantling of hierarchies in technology. Their work criticizes a cis male dominated software industry ("big daddy mainframe"), embraces the cyborg identity and proclaims programming to be a process of liberation, and self-expression. This movement brought light to the cyberfeminism term and invites discussing and implement it for the sake of an intersection of technology, feminism, and the digital world. The tackled issues remain persistent throughout years of technical progress, which requires further analysis of the inherent power structures. This thesis aims to highlight and educate these ideas to an audience mostly unfamiliar with the systemic imbalance in the technological complex. Inspired by the raw, artistic nature, the study encourages creative and unconventional interpretations of coding. In opposition to most learning experiences of programming, I aim to strongly advocate for feminist resources throughout the study.

Feminism, with its goal to form social consciousness and the willingness to learn from marginalized groups to assess power structures, forms a fitting interface to design a curriculum that can educate ethics in a sensitive and dynamic fashion. A study by Vora et al. examined the appeal of feminist research to STEAM students. Their positively-acclaimed year-long course to communicate and educate power structures within the industry proved to be difficult to translate to students without prior knowledge in activism and feminist critique. Overcoming such a hurdle proves to be a common problem that exists in many disciplines of study, where complex information must be conveyed to people unfamiliar with the field. Stoll et al. conducted a summary of plain language models present in empirical research to help laypeople understand the difficult topics at hand. The Commons Social Change Library provides guidelines to designing activist media "to support activists in making their spaces, events, meetings and communications more accessible".[Antje, 2023] Applying that thinking to learners and laypeople may prove beneficial to avoid initial misunderstanding of feminist ideas and invite new voices to the discussion.

To further examine the role code plays within computer science education, shifting away the view from strictly functional use opens up ways that may be more effective in translating complex cyberfeminist discourse. Creative code, as an oxymoron, as Knochel and Patton describe, moves away from programming as a solely functional work of labor towards expression and artistic freedom. The paper features a set of guidelines that will help educators introduce individuals to embracing code as creative. Art students would experience what their daily-use technology is made of, be playful with visual editors producing simple algorithms, and indulge in critical discourse. This thinking is promising in the context of artmaking, yet further investigation remains necessary within the computer science curriculum, or the public. Another unconventional attempt at reconsidering our perception of code was proposed by Dufva in their series on Digital Compost, where code is reinterpreted as a raw and unexplored field that is not relying on digital media - on the Post-Digital. The courses clarify a strong stance on sustainability and materiality in programming education, where to understand the fundamental problems of computer science, the existence of advanced technology is not a necessity.

A concept to formulate creative coding practices in STEAM education stems from the research area of Material Feminism, described by Alaimo and Hekman as a framework to include materiality in the discourse on understanding gender identity and power imbalance. When we observe an individual's agency, we try to identify the environments, complex entities and relationships shaping the

material world. Accordingly, learning in STEM can be analyzed in its materiality, where the bodies of learners actively engage with non-human computer science concepts and learning environments. In their research, de Freitas and Sinclair adapt this thinking to shift the perception on teaching mathematics. When we originally supplied fixed, sequential curricula to students, we now consider their bodies actively engaging with their environment and materials of learning. This idea is especially suitable when applied in the context of a probe study where the active play with a kit of resources is intended. We can further translate this concept to learning to code, where individuals engage with computer science concepts and programming environments in an interconnected fashion. By examining this process, we establish tools to teach programming and enable learning processes that atop of the scientific basis invite to reflect on the topic in a sociopolitical way. To take inspiration from the teachings of traditional media, teaching code may not only rely on processing pre-defined syntax. pseudocode as an already established representation of code for teaching and publication may become a first step into expressing yourself artistically through your own syntax fit under the scope of computing. By formatting, animating and customizing pseudocode, multi-medial opportunities arise that may appeal to an audience that is unfamiliar with code.

Soon and Cox provide an interface adhering to the concept of teaching code in the context of materiality accompanied by active engagement with sociopolitical discourse. The course structure engages with the core beliefs of cyberfeminist research to understand current technological paradigms while exploring their potential to be rebuilt in the scope of "class and capitalism, gender and sexuality, as well as race and the legacies of colonialism". The course is designed around the p5.js framework, a prominent JavaScript tool applied in commercial Creative Coding courses[Rodenbröker, 2019, Xiao, 2021]. Aesthetic Programming offers an interface open to "modification and reversioning" which I aim to accept by evaluating its contents in a lower-fidelity course. Without explicit functional programming and a considerably smaller time frame, a reiteration of their work will approach *learning code* post-digitally as described by Dufva. Here, educating principles that find application in mostly technology is conducted as if technology does not exist in the first place. It's an approach that tries to convey fields such as computer science without the need of computers. Learners ought to immerse themselves by physically engage with problems posed by the discipline through play, exploration, and collaboration. One goal of the post-digital approach is to communicate that a technical skill such as programming is not defined by its tools, like IDEs, programming languages, and empirical algorithms, but the individual itself, just like with any traditional medium.

The design of respective software follows the principle of the universal user that is able to overcome technical boundaries for their artistic vision while embracing internet art as a new medium. It aims to define the user as intelligent, and busy; moreover, any step they take during learning is characterized by artistic self-expression. Recognizing art within the ubiquitous processes in modern media serves as orientation to evaluate a modified course of Aesthetic Programming. Communicating this very concept is a step towards "informing users about themselves".[Lialina, 2012]

Chapter 3

Methodology

In order to assess the effectiveness of the course format and explore alternative teaching methods for coding, a probe study was conducted. The probe study aimed to create an environment that fostered a playful and individualized approach to engaging with the topic matter, while also generating meaningful learning experiences. This chapter refers to the work of Soon and Cox as the primary source of information for the participants to study. Their writings on computing fundamentals and sociopolitical interpretations formed the basis of the study, supplemented by additional explanations, resources, and media.

The qualitative study consisted of three phases, which functioned as sub-steps of the programming course, incorporating constant reflection on the technical and social aspects of coding. At the conclusion of the course, an in-depth analysis and evaluation were conducted to examine the effectiveness of the course format and to gather insights from participants' experiences.

The methodical steps involved in the study were as follows:
1. Probe Study: A kit was provided to the participants, which included an educational zine, supplementary materials, and a code editor. This probe study served as the initial phase of the programming course, encouraging participants to explore the materials and engage with the content in a playful and individual manner. An exemplary course progression with the probe can be observed in Figure 3.1.

2. Semi-Structured Interviews: Following the completion of the probe study, semi-structured interviews were conducted with the participants. These interviews aimed to explore the results and learning effects that emerged from their engagement with the provided resources. Participants were encouraged to share their thoughts, experiences, and reflections on the course material.

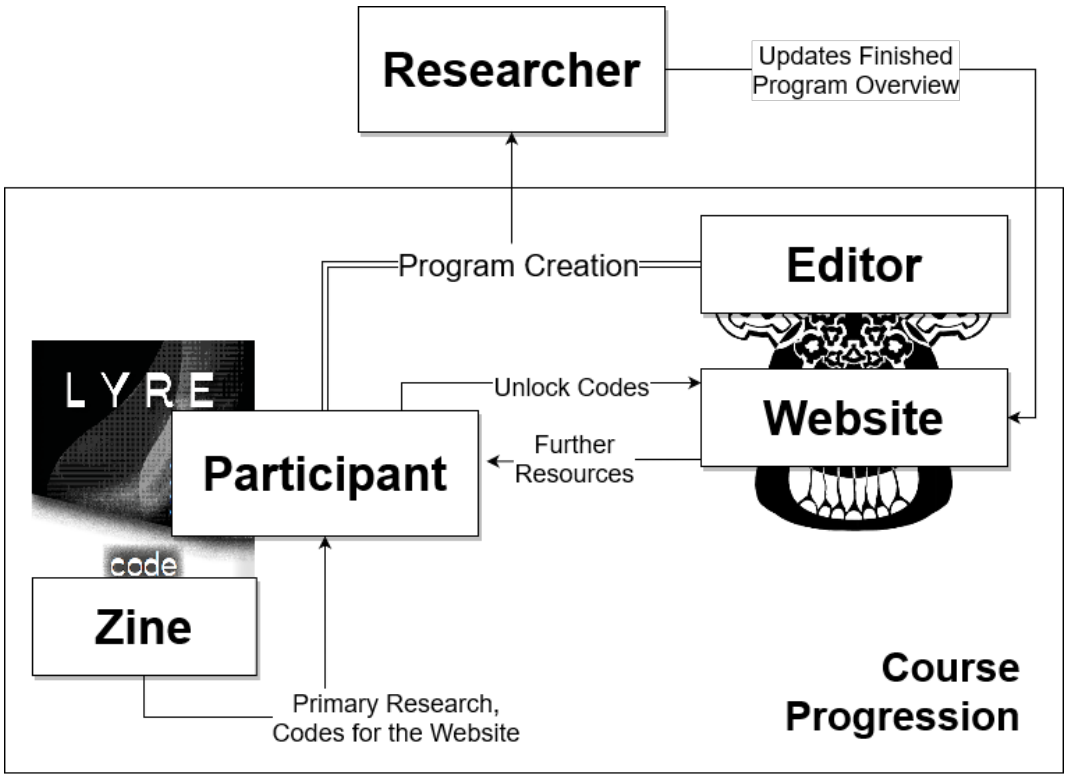


Figure 3.1: Study Design in the form of Course Progression

3. Thematic Analysis (TA): The qualitative data collected from the interviews was subjected to a Thematic Analysis (TA) approach, as described by Clarke and Braun. This analysis involved identifying and developing themes based on the initial design of the programming course. By examining patterns and recurring ideas within the data set, the prominent themes of participants' experiences were established.

By deducing insights from the prominent themes of the data set, the study aimed to evaluate whether teaching programming to non-experts, while incorporating social and political aspects, resulted in successful learning experiences. The analysis of the qualitative data aimed to uncover the impact of the course format on participants' understanding, commitment, and overall satisfaction with the learning process.

3.1 Study Design

The central part of the study comes in the form of an inspirational probe. A probe kit includes an informational zine that directly refers participants to a website for further educational material. Tasked to create creative collages regarding the topics presented, individuals can realize their ideas in a personalized editor environment resembling a graphical interface. Said tool is largely inspired by graphic programs while simulating conventional coding environments. Submissions are gathered through a supplementary questionnaire where, alongside the code, reasoning and inspirations are submitted. With this in mind, the second phase of the study commences in the form of a semi-structured interview. Here, participants discuss their results while engaging with their personal creations, experiences and opinions on their topics of interest. The goal is to get a better understanding of their artistic choices while establishing common ground on the adherent learning effect. Results form through multiple iterations of Thematic Analysis where we find codes that describe successful learning effects, positive experiences, and involvement with queerfeminist principles in programming.

Despite being the focus of the thesis, participants are not informed on their involvement with queerfeminist principles in our study design. While briefly referenced, feminist media is not communicated as such to avoid prejudice towards the topic. Despite its multifaceted goal, feminism is largely communicated as a radical movement to empower women. Goldberg [2014] Lind and Salo [2002] Albeit a vital part to the history of feminism, it resembles the sole identity of the movement. This notion, however, is mostly unrelated to the media presented to the participants and should not be overshadowed by common social bias.

3.1.1 Preliminary Questionnaire

To examine the participant's relation to the field of programming, a questionnaire assesses prior coding experience, and personal opinions on the matter. Aside from obtaining a basis for learning process discussion of each individual, initial confrontation with the topic field is established. Programming is not in everyone's consciousness, so softly easing one into the subject is crucial to not overwhelm participants. The specific questions and their intent are described in Section 3.3.2. With the submission of the preliminary questionnaire, the main user hub is unlocked for the submitter to explore.

3.1.2 Probe Study

As described by [Mattelmäki, 2005], my probes are of inspirational type and are meant to provoke experiences that may enable open reflection on the sensitive, social topics of computing in society. For this research, probes are defined as such by its fundamental properties. A kit consisting of a zine, a website with a personal space reserved for the participant, and an editor give room for exploratory opportunities that encourage individual and ambivalent confrontation with the topic. For the probes are meant to showcase the creative possibilities for both coders and educators, the study sets an exploratory goal of rethinking the role of computing outside the scope of the industry. By moving the study into learning environments chosen autonomously by the participants, the probes become a part of the users' subjective world as the zine may be taken anywhere, and programming tasks are encouraged to be personal and undefined. The resulting program collages, called pseudocodes, are artifacts of the self-documentation process of reading, note-taking, designing, and reflecting. The computer as a medium for creative self-documentation still remains overshadowed in probe studies by pen and paper, tablets, or mobile input technology. While the zine provides a tangible experience to gather information and identify oneself with the study, personal webspaces overcome the finiteness of zines and promise new playful exploration through familiar digital media. Where the amount of information presented in a zine is limited by its physicality, digital worlds promised by the internet in general provide infinite spaces for participants to indulge in.

Choosing this form of a free, yet tangible concept is motivated by the active involvement of the users with a complex topic area. Giving participants the building blocks to explore supplementary material, like a zine alongside a digital canvas, may enforce more intimate and playful experiences compared to a sequential completion of artificial problems as present in conventional online learning platforms. The open nature of probes encourages strategizing a personal workflow and treats a programming course not as a lecture of topics, but an exploration. As an opportunity to move away from the linear style that sees its implementation in modern day programming courses, probe studies leave room for creative involvement that may even give participants a sense of pride in the creative work they produced. Elevating the probe from a solely analogue form to a digital workspace, may provide further insight on the creative potential of digital media in a playful context and the definition of probe studies. Where digital programming courses lack in tangibility of the code work, the probe can be seen as a gift and real representation of their efforts. The focus of the probe is to facilitate the confrontation of people with programming

without prior knowledge of administrating development environments. Participants may skip entry boundaries by experiencing coding as a creative craft from the beginning. An explorative hands-on experience becomes complementary to the material feminist notion as described by de Freitas and Sinclair where the participants' bodies come together with programming, the concepts of computer science, and the probe as materials of learning.

As noted by Mattelmäki, probe studies come with their own difficulties, as their ambiguity may frustrate and confuse participants. Providing users with numerous ways of interacting with the probes, like note-taking in the zine, clicking through resources, using different building blocks for the collages, and finding ways to overcome its limitations, may give questioning users' momentum when trying out the different interactions. Furthermore, removing pressure in the form of strict deadlines may alleviate stress. Providing inspirational aid at all times to varying extent while having initial rapport with participants may also help users to overcome the hurdle to create freely. Nonetheless, such concerns may never be fully erased from a study with such open-ended tasks and underline the necessity to actively engage with learners when problems arise.

In the spirit of net art and creative programming, establishing web pages as a creative medium is crucial when we deal with new and emerging media as credible sources for self-expression and active discourse. Re-thinking its possibilities outside the scope of analytical tasks and corporate desires may open the way to new paths in research using novel and multi-faceted techniques such as probe studies.

3.1.3 Semi-structured Interview

After a preliminary analysis of the submitted digital collages and reflections, participants provide further insight on their experiences throughout course progression, as well as their learning effect from the viewed resources. In a semi-structured fashion, both focal topics are discussed over 30–60 minutes under calm circumstances. Conversations are recorded over smartphone microphones and immediately transferred to a laptop for storage and transcription. While every interview includes the same questions, depending on the participant's response, some may be emitted to not hinder the flow of the conversation. Shifting the discussion towards topics that participants may feel more comfortable discussing is necessary to highlight aspects of the study that participants felt strongly about. In the case of unique course submissions, individual questions that pose potentially interesting ideas may enable

intersectional thinking during analysis. These ideas can reflect in processes of self-study, references to resources unrelated to the course, and special interest in certain zine chapters. The goal of the interviews is to assess the thoughts on the study from perspectives of various backgrounds to generate meaningful and diverse data for the following analysis.

3.2 Participants

With seven hand-picked participants, the probe study remained in a more friendly and familiar context. To be sure of an initial rapport for enforcing a successful probe study, seven close friends of mine were chosen. This decision was driven by multiple factors. Participants should not be in regular contact with each other, they should be able to read, write on a keyboard, comprehend the English language, have a working computer and have little to no prior knowledge in the field of computing. That includes not having a degree in any computer science related study course, and not having pursued programming in corporate or private projects in the long run. A few samples were chosen to have a better picture of the individual work and qualitative data, as well as facilitate reacting to problems and overseeing the study in the scope of the thesis. I chose people who are of varying backgrounds in occupation, gender, and academic levels. This opens more possibilities of artistic expression as all these factors influence different life experiences. Table 3.1 shows the overall demographic of the participants involved. Ages of participants were not part of the picking process and coincidental. Throughout the study, participants obtained codes representing their identification and will be referred as such during analysis and evaluation.

Participant ID	**Birth year**	**Gender**	**Education**
jackfruit	1996	woman	Bachelor's degree
apple	2001	man	Bachelor's degree
pineapple	1997	man	Master's degree or above
peach	1996	woman	Bachelor's degree
lemon	1998	man	University entrance qualification
maracuja	1994	woman	Master's degree or above
grape	1998	man	University entrance qualification

Table 3.1: Research participant demographic

Every participant filled out a questionnaire regarding their personal view on programming, as well as their encounters with coding in daily life. The questions form a baseline to assess a learning effect as well as a shift in perception after course participation. They provide impulses to reflect on personal beliefs, as well as suitable follow-up questions in the final review.

3.3 Materials

Central resources for the study come in the form of an analogue zine, a digital learning platform, and supplementary material for conducting a probe study. Designed to inform participants of the topics of interest, the zine is designed especially to captivate and feel meaningful when in the hands of the reader. Additionally, it refers to the digital learning platform which offers a variety of resources to engage with, as well as an editor to interpret the chapter-specific questions. The motivation to shift between analogue and digital comes from the reference of codes throughout the zine. User codes remain unique and serve as a way to identify the work and their submitted task, while the codes of the courses are the same since there are no variations of the respective chapters. To give the kit a homely feeling, the design of the probe is cohesive throughout and is given character through custom graphics, icons and style. The title Assemblyre is a direct reference to the concept of Assembler, a programming language that directly transforms assembly language to computer language, and a lyre, which puts emphasis on the effort of highlighting the inherent creative potential of computing and its acceptance.

3.3.1 Analogue Zine

The zine provided during the study is the very first probe participants come in contact with. Designed in Adobe Illustrator and printed as an DIN A6 brochure, it is supposed to be portable and interesting to look at. Its design resembles the editor and takes inspiration from zines in the cyberfeminist sphere [refs]. Over 17 pages, participants can explore the topics in any order, with the possibility to write annotations after each segment. With the help of a personal introduction to the study, the booklet prompts to make use of the codes provided. Details about the user's task are explained and invite to directly jump into the first topic.

Topics were chosen related to the course structure introduced by Soon and Cox with several modifications. Most adjustments were taken to enable a concise zine that would touch on every topic without going into details that require one-on-one consultation. In that regard, full reference to original chapters was provided regardless of being obligatory to read. Figure [ref] shows the amount of content mirrored in the zine, as well the reasoning for not incorporating excerpts. Incorporating graphics from the digital resources tries to invoke reminiscent thoughts while playing with the probe. To manifest the learning process of every participant, the zine can be kept as a present and may leave room for further discussion when addressed during the interview.

3.3.2 Preliminary Questionnaire

A digital form presents the preliminary questionnaire to everyone participating in the study. Before the central tasks can be tackled, the questionnaire serves as an estimate for the learning progression baseline. In the following, individual questions are presented alongside their reasoning to be included to benefit the research question at hand. The whole form is semantically separated in two parts.

Demographics In the first part, we gather general demographic data that is relevant to the study without collecting exhaustive data that is not a focal point of it. Participants are asked about their birth year, their gender, as well as their highest level of education. The respective data helps to establish suitable participant profiles. For I am not attempting to find correlations between user demographics, and their learning experience, gathering a surplus of personal data on each participant is not a necessity. The three variables form a suitable estimate for their prior experience with technology and code, without infringing on their privacy to not distract from the main interest of the questionnaire - establishing a baseline for the participants' opinion and experience with programming.

Programming Experience During the second half of the questionnaire, individuals elaborate on their view on technology and computer science as a whole. Their overarching theme is the deduction of prior programming experience, and individual opinion on programming in its broadest context.

What have been your experiences with programming platforms and graphical software?

- I know of online learning platforms for programming.
- I have used such platforms privately.
- I have completed at least one course on such a platform.
- I was taught programming in school.
- I attempted to learn programming through a personal computer.
- I have used graphical software privately.

By easing the way into thinking about personal involvement with code, participants must reflect on when they were confronted with it. They may have had programming lessons in school, or online programming courses. Understanding where the first entry point to coding was, establishes a basis for tracking the learning experience throughout course duration. Since the actual task of the probe study is to work with a graphical editor, the users' experience with software for canvas manipulation is insightful. Assessing to what degree they are familiar with such an environment, helps to identify causes of error and frustration in the future. The explicit questioning of whether their contact with coding and graphical software has been for private use, implies that it was driven by individual interest, and not monetary needs in the context of industry requirements.

Programming Opinion *What is your personal opinion on programming? List adjectives separated by a comma (,) that would describe programming to you.*

By letting people describe the field in short terms, we obtain vocabulary that can be used in a later interview to deduce any potential change of heart. Establishing whether the inaccessibility of programming as described in related work reflects in the group of participants supports the morale of the course. Individuals are invited to reflect on their personal stance, and with their choice of words, may reveal their general approach to the course. It sets the tone for future submissions, and may be evidence for learning processes induced by revelations from the course material.

Would you agree/disagree with the following statements?

The question serves as a more precise deduction of the participants' opinion on programming, and introduces everyone to the tone of the course, as well as possible focal points. By using a Likert scale, participants can make responses quickly while creating the personal opinion profile. It provides further data that may reflect in later creative work as the questions revolve around access to programming in a broad context, and personal philosophy on art and technology.

General programming resources are difficult to find.

While not an attempt to evaluate programming resource accessibility, the response may reveal possibilities as to why the participants may have been

involved with computer programming in the past. While the resources are far from scarce, they are difficult for many to comprehend because of their essential industry bias. In case participants believe guides, and courses to be plentiful, there is reason to assume prior interest in the field.

I think dedicated online courses could uphold my interest in programming.

In accordance to the previous claim, if individuals have participated in programming courses before, inferring what has failed to keep them engaged is relevant when establishing a course form of your own. Understanding whether courses in general are interesting for people of different backgrounds, and establishing themes that encourage the exploration of the coding medium, require the participant's stance on their opinion on the form of education itself.

I have an easy time working with new commercial program interfaces (Photoshop, Microsoft Office, Social Media Platforms, Google Suite).

While the course is predominantly aimed at new learners of programming, resources on the connections between digital literacy and coding [Vee, 2017] prove to be tempting reasons to establish a baseline for learning progress. The claim is a rough estimate to assume whether participants may have an easier time adjusting to code syntax and the Assemblyre tool if they already have experience with popular state-of-the-art commercial systems. The ability to navigate industry software may have a saying in who finds programming more accessible.

I try to avoid using code when working with programs of my daily life. (Photoshop Plugins, One Drive Office, Google Sheets Scripts, Blender Add-ons, Website Builders, Video game Mods).

While the ability to modify your experience even in closed-source software opens many possibilities to use the tools to their fullest, they rarely play any role in the daily use of many. By questioning the participants' experiences in that regard, I aim to approach them as general purpose users, as described by Lialina. In finding the individuals' ability to examine every-day tools' boundaries, their creative work throughout the study may reveal novel strategies to overcome the limitations of code, shifting the paradigm of the editor.

School education has sparked an interest in programming for me.

Aside from the computer science lessons taught in school, the school as a

shared experience among all participants has many opportunities to spark interest in coding. While literature, art, math, and science are fields that benefit from computer science concepts, questioning if anything of that sort occurred to the individuals. School-taught programming, while rudimentary, provides valuable lessons on basic coding principles that may reflect in submissions, and their understanding of each topic may be influenced by the information remembered from that time period.

I would use program code for private projects rather than my professional career.

To further investigate the participants' view on programming, this claim serves as a transition into more specific questions. By asking where on the spectrum between private, and corporate projects they see themselves creating code, a clearer bias can be estimated. Questioning whether this bias has changed after course completion will help understand the participants' definitions of creative coding, and where they see themselves creating with the medium.

I like to express myself creatively (traditional/digital artwork, crafts, prose, design, dance, etc.)

Following the questioning on creativity as it plays a crucial role during probe exploration, documenting the participants' personal involvement with other creative hobbies provides a further way to interpret their future work. People who are comfortable expressing themselves with different kinds of media may have less of an entry barrier to the loosely posed tasks. It may also be possible to observe creations that feature other media in an interdisciplinary fashion. How the individuals define what a creative medium is, is solely up to them.

A creative process is defined by its result.

While the statement may be ambiguous and difficult to answer without context, its purpose is to communicate the tone of the course material. Again, inspired by the notions of Lialina, technology showcases that not only the final product is the eventual art work, but the many processes involved. Participants' submissions are not meant to be the only thing regarded as artistic; moreover, every step is observed to conduct an analysis that takes everything into consideration. Responses to the question may reveal the learner's attitude towards code as a creative medium, art as a concept, and how it reflects in their pseudocodes. *I understand what open-source and transparent code is.*

With the goal to build bridges between laypeople and the industry-shaped vocabulary of programming, the idea of open-source development is tempting to bring up to participants. Establishing whether individuals are familiar with such a concept, may match with previously established baseline responses implying technical knowledge. When the participants are unsure of a clear definition, it is up to the course to fill those gaps, which could be indicative of learning processes supporting the effectiveness of the course. *When using apps and programs, I question how data about me is captured and used.*

Despite the course covering a variety of topics, data remains to be the most complex, and content-heavy one. Giving the user a first opportunity to reflect on their view on sharing their data further immerses the participants in the course's goals. Aside from the importance of valuing personal data, the claim is exemplary for a topic you wouldn't find in conventional programming courses. Showcasing that partaking in the study involves programming, as well as the discussion of modern dangers of it may produce responses revealing the role these questions play in the participants' consciousness.

With some questions being specific, and possibly difficult to answer, a "No Opinion" checkbox, and a free text input for further elaboration remains as a valid response.

Since experiences and opinions are difficult to grasp through sole metrics and scales, participants are encouraged to provide context to their selections, which are valuable for further analysis. Like any free text prompt, they provide room for potential discussion on every topic, and an opportunity for individuals to explain themselves when they feel like the unary scale may feel limiting or ambivalent.

3.3.3 Digital Learning Platform

When participants visit the website mentioned in the booklet[1], they are greeted with a minimalist design and comprehensive texts that provide further details about the study, similar to Figure 3.2. The website is designed to be user-friendly, yet engaging, allowing participants to explore the site freely without frustration, which could negatively impact their motivation to continue. The process of entering the found codes is quick and responsive, creating a sense of mystery and intrigue.

[1] Artur Solomonik: Assemblyre (`http://assemblyre.glitch.me/`) Last access: 07.06.2023

Figure 3.2: Starting Page of the Assemblyre probe

The website serves as the central hub for the study, featuring all the learning materials and study information. Participants can watch videos, read articles, and play games while navigating through spaces that are both playful and informative. These experiences aim to emphasize the importance of digital literacy and promote content that is often overlooked in traditional code learning contexts.

To manage the multitude of media and ensure a smooth study experience, the website includes personalized space for participants to engage with the course. Participants can track their progress through the chapters and see their library of submitted pseudocodes fill up over time.

In addition to the study materials, the website also features five main navigation points that serve as sources of information about the site itself. The "Home" link directs users to general information about their tasks and provides access to the user hub. The "Editor" link allows participants to explore the tool independently of the study's chapters, providing an opportunity for playful experimentation. The "About" page references the resources used to create the website, ensuring full transparency. The "LYRE" section provides download links for a digital PDF version of the zine in case participants lose their booklet or prefer a digital format. Finally, the "More" link offers additional information outside the scope of the study, allowing participants to delve deeper into the world of computing if they found specific topics intriguing. These additional resources are objective tips based on my personal experiences as a developer and artist and are not directly related to the main study. Providing this information helps participants refresh their memory after a longer period without working on a task and alleviates any uncertainties they may have when using the tool in the experimental nature of the study.

By creating a user-friendly and informative platform, the website enhances participants' learning experience and encourages their continued engagement with the study materials.

Course Overview The overall course encompasses the text material, thought-provoking questions, and the curated list of online resources. The extent of the text was chosen with its educational value in mind, without overwhelming participants with new information for the chosen duration of the study. Every chapter consists of the excerpt from the guide on Aesthetic Programming by Soon and Cox [2020], the respective course ID code, open questions about the topic discussed, and a list of links to digital resources with short summaries presented in Figure 3.3. Driven by the multi-medial nature, I chose print media

Figure 3.3: Assemblyre Personal User Hub

for the reading portion to prevent long screen times, and digital websites for further resources to bring online sources to people's attention that only work when viewed through a Browser. This notion underlines the medial nature of computing and the web, as it enables expression that couldn't exist anywhere else. The material approach to programming is supposed to be reflected in the way participants interact and get involved with the media outside the necessity to code conventionally using a computer with an IDE.

Personal User Hub The main overview of the course progression comes in the form of a virtual hub as seen in Figure 3.4. Here, participants will unlock digital resources with every chapter. To take inspiration from the progress in common digital coding platforms, the study abstracts the progress inform of blocks that are visualized as unlocked when an entry is submitted. After each chapter title, a short summary of its content is presented alongside a link to the resources, as well as the editor.

Aside from the integral environments for the study, the website provides further inspiration for participants to privately explore the broader context of programming with resources that were curated under personal bias. The zine's base version can also be downloaded in case it is lost or more suitable in digital form. In case of bugs, a simple board about bug fixes notifies users about current technical difficulties. To lead participants into the study, all information on their tasks is displayed at the home page.

Task Progression Participants are tasked to create a *pseudocode* inspired by the contents of every chapter. An exhaustive overview over the digital course materials can be found in Chapter A. pseudocodes are defined and discussed in the very beginning and create a way for participants to express themselves creatively. With submissions, the collection of unlocked blocks rises and creates a place that users have power over. Instead of providing all the courses and what is to come from the very beginning, participants build up their hub progressively without thinking about the chapters that remain untouched. For the study only requires five submissions in contrast to the eight chapters, participants are not prompted to realize tasks for chapters they might not be interested in. This enforces exploratory thinking during the course without sticking to a streamlined narrative with controlled outcome. The fluctuating extent of the chapters alongside the varying resource media support this notion of a non-linear, person-driven course progression.

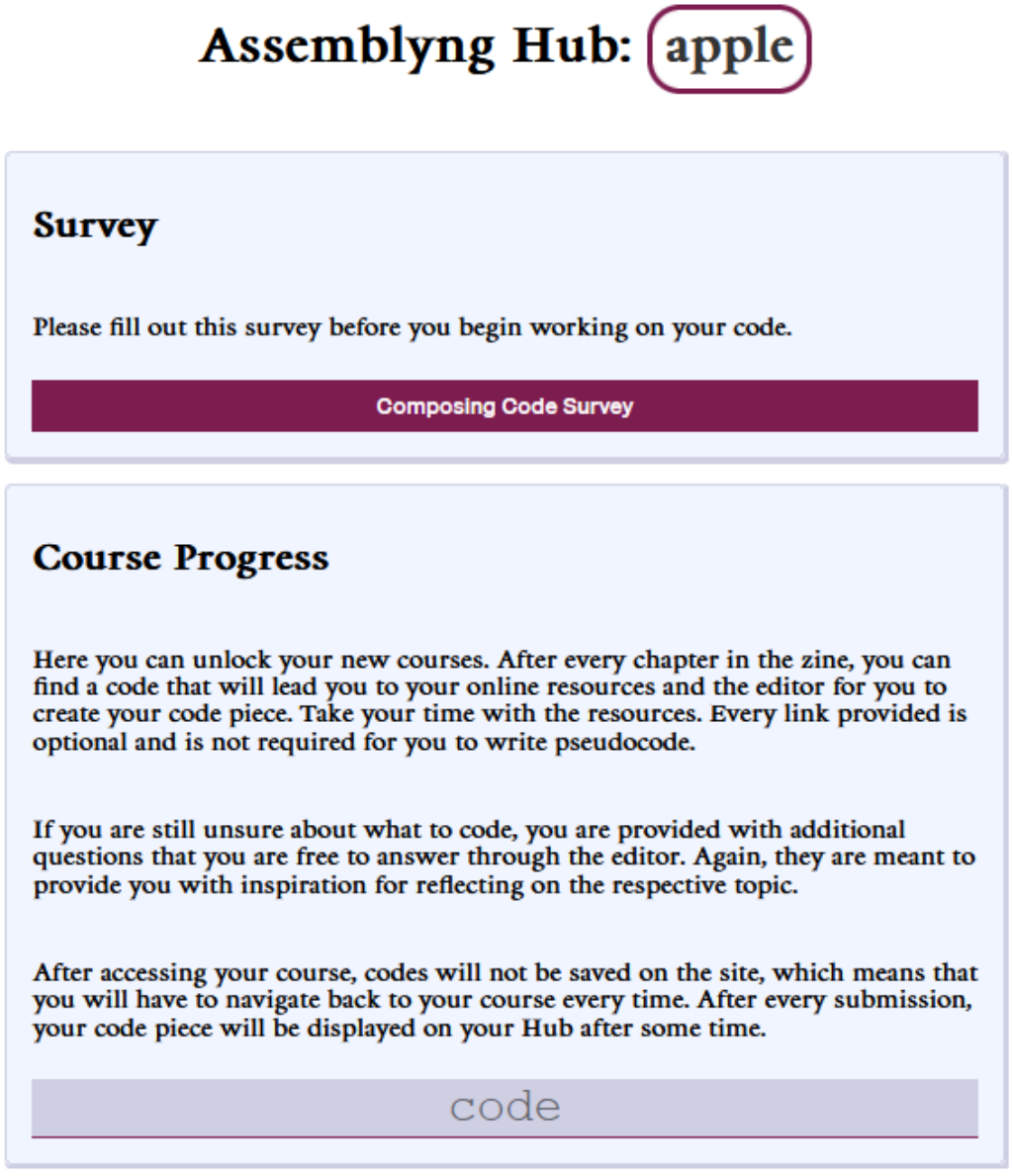


Figure 3.4: Assemblyre Personal User Hub

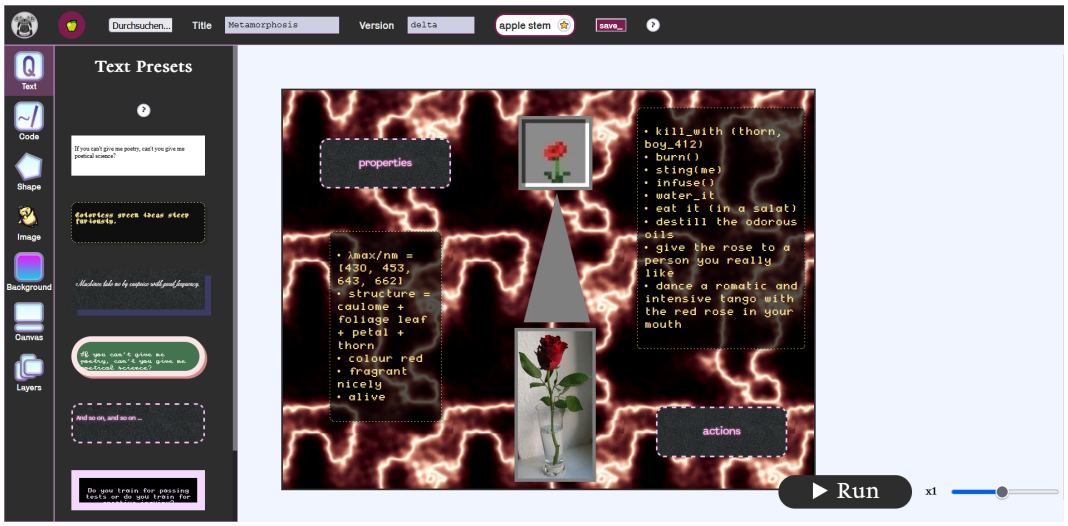


Figure 3.5: Assemblyre Editor Environment

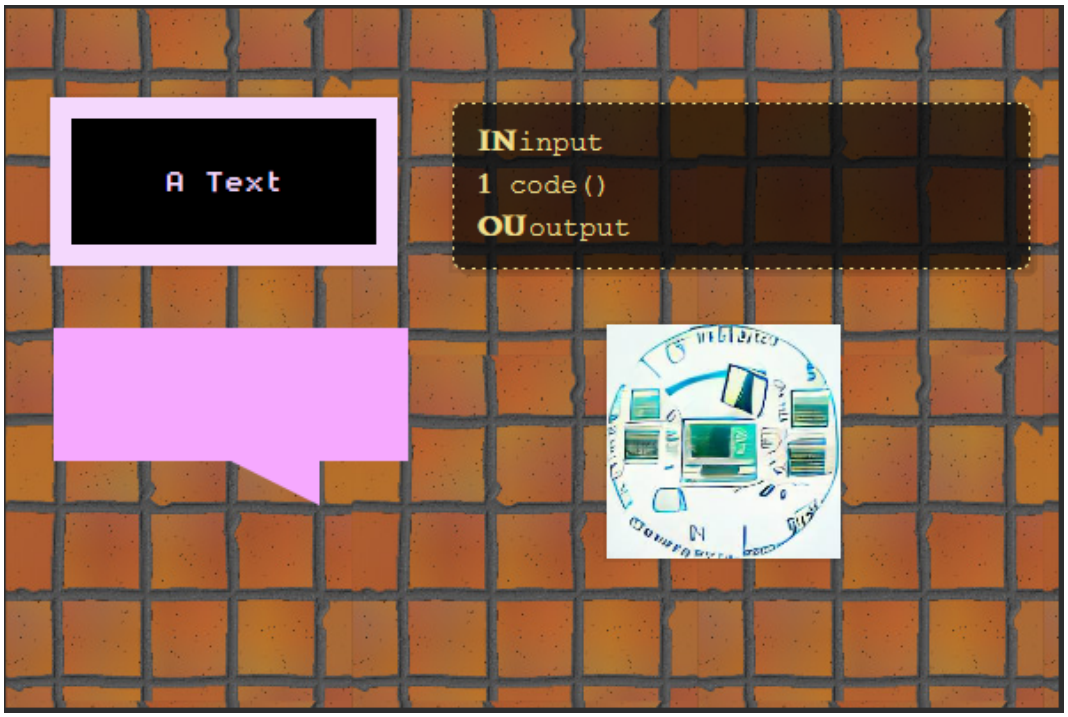


Figure 3.6: Editor building blocks: Text (upper left), Code (upper right), Shape (lower left), Image (lower right)

pseudocode Editor

The main programming takes place in a graphical editor shown in Figure 3.5 that functions as a combination of a coding and a designing tool. Its implementation is the main canvas for participants to express themselves to shift the perception from coding as a service to coding as an art form. The graphical interface is very similar to state-of-the-art online editors like Canva[2] or Plotno[3]. By creating building blocks like texts and images, users may place them on a canvas while editing them to their taste. Figure 3.6 showcases the four basic components as generated with preset styling. To improve and adjust this functionality to coding practices, coding blocks and animations to simulate the unique feeling of writing code. Coding blocks include tabbing, input/output pairs, line numbers, and different types of expression tags. Animations can be attached to each block to bring them to life when running the code. As soon as you push a button, the places blocks will emerge in the order of their layering, play their animations, and visualize the expressions set in the coding blocks. This is supposed to give users feedback on their creations through visual cues instead of returning error messages or console prompts that are prominent in coding courses. The goal is to paint pseudocode as a medium of self-expression under varying forms of abstraction. The combination of artistic capabilities of web browsers combined with theory on Computer Science brings new interactions that may influence learning experiences positively. With every piece, participants may provide a title, and a version number to uphold the abstraction of writing executable code, while creating the illusion of official releases.

After creating a piece and labeling it, participants can download their file for future edits or their submission. The exported file comes in the form of a JSON file with all information required to recreate the pseudocode in the editor, showcase it on the participant's hub, and analyze the interactions and tools used graphically. The structure gives insight on aesthetic preferences, which building blocks they felt most comfortable using, and further details on what conveyed their creative intent appropriately.

The technology for the realization of the tool makes use of the capabilities of browsers to render HTML, CSS, and JavaScript with the help of dedicated frontend frameworks. The implementation makes clear that it is written by someone who was not educated on the topics presented in the course from the beginning of their programming education. With the help of the open-source, state-of-the art Vue.js framework [ref], the interface is fully client-side

[2]Canva, `https://www.canva.com`. Last access: 07.06.2023

[3]Plotno Studio, `https://studio.polotno.com/`. Last access: 07.06.2023

to overcome server costs and privacy issues of host providers. Furthermore, server-based interactions may leave interactions hidden, and not transparent to the user. By using frameworks solely for visualizing the course and editing the canvas, it is clear that submissions are not added and processed automatically.

The canvas features draggable blocks that abstract the components of a program, and may function as decorations and background media. The prominent DOM-manipulating framework InteractJS facilitates the implementation and brings features that can be applied to the creation of the editor interface. Despite its simple nature, the inferred base code is opaque to the use; however, and creates a layer between the user and the program. For the duration of the study, these interactions remain implemented for the sake of convenience. The canvas functionalities are a central part of bringing the participants' code to life with the potential of layout and style.

Canvas Interaction The canvas is separated into three sections that try to mimic the experiences users might have with modern browser-based graphical interfaces. Sticking to this form of software may give participants an easier time adapting to the environment to not alienate them with technicalities.
An upper toolbar features links back to the hub, the ability to import files, name them, and export the results to a JSON file. A constantly available task bar remains like a heading to the current work and may remind users to name and export their progress throughout their tasks. The ability to navigate back to their hub is necessary to not frustrate users with the limitations of an exclusively client-side software, as an automatic save and submission are not possible.
The tools provided to users are presented on a singular sidebar, so users are at all times reminded of the possibilities they have to shape their ideas. For every tool, presets function as a way to overcome choice paralysis in the customization of fields. Instead of relying on users to know how their fields have to look, the editor provides multiple, distinct examples. Instead of focusing on optics, one can realize their idea more swiftly. This is supposed to overcome frustration and overthinking the processes that will be of less impact for the study results.
Text fields are fundamental building blocks for every graphical interface, and they here function as leverage to bring participants closer to expressing their ideas through pseudocode. Code fields provide an aesthetic basis for writing pseudocode with their representation of lines, input and output parameters, and dedicated styling. The possibility to import images gives room for further inclusion of inter-medial resources that are fully up to participants, and flexible

shape presets support this notion. These building blocks are all composed on a canvas that can be adjusted in size and background. All background images provided by the tool are handpicked with diversity, range of colors, and fun of use in mind. A layering section governs the stacking of building blocks on the canvas while bringing a sense of organization during editing.

The majority of interactions take place in the central section, which represents the active project worked on. Here, the selected presets and project preferences manifest graphically on a canvas and building blocks that can be transformed freely through scaling and translation. Every field can be edited by clicking on them and navigating a direct style control. Here, further adjustments to the provided presets can be made alongside animations. These, however, can only be previewed by entering the Run mode, which enables the final representation of the running program.

pseudocode Execution Executing a graphical program ties into providing participants with feedback that reflects on their project. Where programming associates code execution with parsing program code, translating it for the computer, and executing the corresponding requests while gathering possible errors, the editor abstracts this process while fully ignoring the existence of errors. With a button click, the programmed animations and transformed building blocks will emerge in sequential order by their layer stacking. The resulting feedback may introduce new ideas and adjustment possibilities to the user, which is supposed to mimic the interaction loop of programming with fixed syntax. The results are supposed to showcase the user's efforts while leaving a small factor of surprise and satisfaction from the final product. The final export into a functional file only amplifies the feeling of finishing such a project.

Submission and Reflection With a minimum of five submissions per participant, not only do I prompt the produced pseudocodes, but a short reflection on the creative work produced. This includes acknowledging inspirations, and questioning how the topic influences oneself or others. The vague nature of the questions may provide information on the participants' perception of their creative work. Not only is this an opportunity to consciously confront oneself with the topic for a last time, but also to develop strategies of formulating one's vision. Screening the pseudocodes alongside their reasoning and sources may aid the analysis as it shifts the main focus from the coding results back to the more complex creative choices participants have made.

Participants with starting problems receive the following information, with

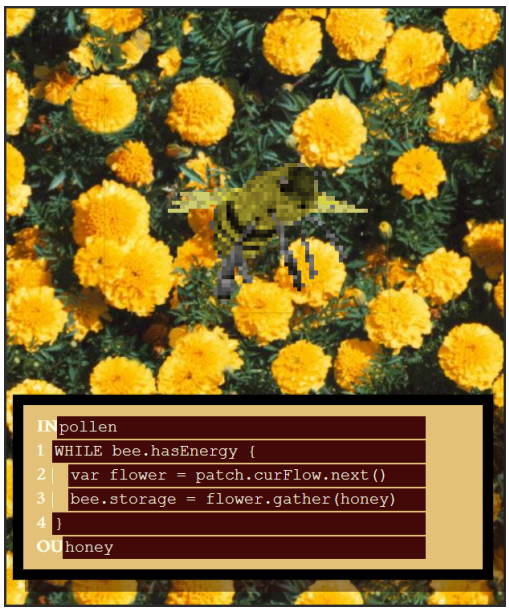


Figure 3.7: Example pseudocode on participant's request

variations in tone depending on their specific concerns:

pseudocodes are basically collages that you can pseudo-execute by triggering their animations with the Run button. What you create in the editor is up to you, and should be inspired by the chapter you read. You are not obligated to use any syntax from any programming language, but I encourage you to experiment with how you would put your thoughts into code, considering the principles you learned in the chapter. Not all chapters are of the same length, so consider looking into resources you personally find interesting, or inspiring. Your finesse with the tool is not important here, and I urge you to see it as a fun exercise that will remain unmarked. In case you keep struggling with inspiration, I can send you an example of a pseudocode.

The pseudocode in question 3.7 is a simple code fragment that was designed to give a brief overview over how different components can be stylized to approach the creative vision, as well as rudimentary pseudocode that tries to capture the process of bees gathering pollen to produce honey. While loops, variables, and algorithmic structures are introduced in the supporting material, their synthesis may become problematic for some.

3.3.4 Concluding Interview Material

Course completion comes in the form of a final interview, which includes numerous resources that are essential for conduct. With a formal privacy disclosure and an interview guide, as well as a summary of course results, the study is designed to build further context on participants' experiences and thoughts on the course topics.

The interview guide presented in Chapter A.5 prioritizes the evaluation of the users' overall experience with the course progression, further context on their pseudocodes, as well as the respective learning effect from relevant resources. After final submissions, we reserved time to gather possibly interesting questions for each interview to bring up. Individualized interview questions give leeway to uphold participants' interest in the topic, and may provide opportunities to determine successful learning processes. The guide serves a supportive role to lead the semi-structured interview in the direction most promising, depending on the participants' responses. With a broad and individual field of topics, not all questions are ought to be answered. As a tangible artifact, the individuals' submissions provide opportunities to specify details visually on a tablet. Final audio data is transferred to a laptop for the evaluation process.

3.4 Procedure

3.4.1 Text Collection

The excerpts were chosen according to the scope of this study. To avoid confusion, programming tasks with P5js, reflections on the original Aesthetic Programming course, and paragraphs relying on the involvement in it have been removed to serve the purpose of being publishable in a zine. With that in mind, I cited their work without obfuscating meaning and provided context where it may have been unclear. The goal of the zine would not be the active confrontation with the complex details, but the acknowledgment of their existence, to spark interest and leave the investigation up to the participants. [ref] visualizes the changes and adjustments I made to use the course for the scope of the study as described above. [1]

3.4.2 Resource Collection

Gathering resources to share with the participants required curating of commented references in the Aesthetic Programming course while exploring their references and sources. With the help of the Cyberfeminism Index [Seu] and the video game repository of Itch.io [Corcoran, 2013], the repository grew to offer a multi-faceted experience. Alongside short and long documentations, participants may look at art galleries and play video games supporting the topics of the course chapters. This easily digestible content is meant to narrow the gap between participants and the topic. Participants may enjoy the content, recognize the creative work of others in the field, and get accustomed to computing as something not inherently out of their reach.

3.4.3 Course Structure

To fit the course to the duration of the study while including every topic relevant to the problem, topic selection remained close to the source material. Since not all details of every chapter are a part of the study, brief references through online resources reference the broad context to not dismiss it. The topics discussed in their chapters are explained in [ref]. [2]

Each participant is meant to create five collages based on the chapters they found most interesting. While there are eight chapters to read about, only five of them are required to submit entries for. It's apparent that not every topic will appeal to everyone, and in the context of a probe, participants are meant to explore the topics and give their insight on the ones most intriguing to them. To provide an experience distant to conventional sequential task

progression, the course encourages exploration for the sake of unique thought processes for their selection. It is to say that chapters are not identical in their length and may give participants not comfortable with a topic the ability to pursue a different one. The study strives to bring people closer to the field of computing by building bridges closer to the technical field without choice paralysis or creative block. That is why a course structure with a clear line of sight while giving control to the user seemed helpful for the cause.

3.4.4 Participant Introduction

The study is communicated both in person and through online communication over e-mail. In a personal meeting, each participant is handed their probe kit and is instructed in the details of the study. In a collective message, the details are again described, and the website is shared. During the study, participants can message me personally when technical or topical difficulties arise. Message exchange is conducted over accessible platforms such as WhatsApp and Telegram, which requires consent on the privacy risks that are out of reach of the researcher. Personal support is provided in case of sensitive and complex problems. In case of creative hindrances, I provide further aid in form of pseudocode examples and further explanations on the ideas and scope of the tasks. Every reported difficulty was tracked and discussed personally later on to get further insight on potential shortcomings in the course design. In case of time constraints, the individual study deadline of two weeks are postponed for one further week. The complex and demanding nature of the study requires flexibility to enable participants to focus on the topical and creative aspects without worrying about deadlines that serve a purely motivational purpose.

3.4.5 Initial Questionnaire

To provide context to the following course progress, a short assessment of their experience and opinion with programming is conducted in the form of a survey. Here, the overall demographic, their societal perception of programming, and current knowledge on the topic is asked to aid the analysis of the probe study. Establishing a basis is crucial for estimating a learning progress in the final stages of the study. Participants are directly confronted with the general topic and may familiarize themselves with key interests of the course contents. It may also reveal unique traits of each participant that may reflect in the practical tasks. Table [ref] showcases the questions and their reasoning for being posed in the questionnaire.

3.4.6 Course Familiarization

Participants are presented with eight topics of computer programming, and will create at least five art pieces through the graphical interface on the website. Over the course of a minimum of two weeks, they can read the information in the zine, unlock the digital resources, and create a code piece of whatever topic they wish. For additional inspiration, questions are provided as thought impulses. If this does not suffice, I prepare example pseudocodes to share when participants struggle with the free nature of probe studies. Users require the confidence to not stress about the quality of their result, as it is not the focus of the research. Any doubts about the validity of submissions I try to approach with words of encouragement and further explanations on what participants are expected to create throughout the course. The writing throughout the course itself facilitates the idea to separate coding education from static syntax and creative limitations.

The zine provides an entry point to the topic and serves as a personal companion to participants. It showcases the participants' exploration of the topics. After reading the theoretical components of each module, the practical portion begins.

3.4.7 Resource Exploration

New content on the website is unlocked by the users as they progress the reading of the zine. By typing a new code on the platform, digital resources and open questions of interpretation are unlocked that are supposed to spark inspiration to the participants. Largely inspired by the resources provided in the Aesthetic Programming course, the references are realized in simple button clicks alongside a short preface. Instead of researching links from a long list of literature, the more visual and easily digestible content provides participants with an easier experience. Accordingly, they may invest their time to their liking and explore the things they find the most interesting instead of reading through long articles without personal connection.

The additional resources are used as inspiration for their programming task and will then navigate to the respective editor to start implementing. User may submit their work over their hub, and will see the progress in the form of their finished work. Every submission comes with a short description of the participant's vision, their artistic choices, and a reflection on the hypothetical power dynamics of the program. The submissions are then showcased to the participants over their hub.

Coding Task Progression After traversing the prose and resources of each chapter, participants are tasked to produce a pseudocode in a respective editor environment. The components that may help bring their idea to life are annotated and emulate the experience of programming conventional scripts. While the final result remains fascinating and engaging to look at, the main focus of the study is to make participants engage with the topic in a hands-on fashion devoid of confusing development environment setups. An in-depth description of every system component can be found in Section 3.3.3.

Users assemble and fill the building blocks with their ideas and run the pseudocode which sets everything into motion. Every block is fully customizable with the intent to lead to visual representation of ideas as quickly as possible. Iterating the process simulates the workflow of simple programming tasks and is finalized in a file that is attached with each task submission.

In the spirit of material feminism, understanding a medium requires individuals to engage with it physically and playfully. To approach this idea, participants are invited to explore every corner of the canvas and make use of it in whatever way possible. The multitude of building blocks may reveal preferences in customization, which may reflect in the final submission.

Submitting pseudocodes comes in the form of another questionnaire, which gives participants an opportunity to elaborate on their reasoning and inspirations. The goal is not to fully understand the topic or learn syntax, but to enforce creative thinking and play with the capabilities of the editor. All questions are optional to not leave the impression that every submission must have complex reasoning and skill, as any creative confrontation with the topic is meaningful.

3.4.8 Final Interview

A supporting interview will be conducted in a semi-structured form, where participants will be asked about their experience with the learning platform. If they are comfortable, participants may talk about the code pieces they created to assess their thought processes, and whether their ideas were realized on the learning platform accordingly. Talking to participants directly will also provide valuable information on whether the course was able to communicate the topics appropriately. This measurement is subjective and relies on my personal ability to deduce whether participants can talk freely about the topics of interest. My reasoning revolves around the participants' ability to discuss the topics they found interesting, how they translate the new knowledge to their

own life experience, and possible changes of heart regarding their stance on programming. Asking for their own opinion on their growth may be of help for the final deduction.

Interviews are conducted in a personal meeting in a familiar setting like my or their home. The interview is recorded on audio and is then transcribed to analyze the participants' opinion on the information provided, and their learning efforts.

3.4.9 Result Organization

For the analysis, the graphical interface's import functionality is used to view the code pieces created by the participants. The system counts the used components, whether they were customized, and the prominence of course chapters is counted. Questionnaires are exported into suitable tabular data to create profiles for every participant. pseudocodes are printed with annotations on animations alongside their reasoning from their submission questionnaire. Transcripts are spliced into excerpts and prepared for the thematic analysis iterations.

3.5 Analysis

With conducting the final interviews, the set of qualitative data provides a starting point for an inspection in the form of Thematic Analysis, as described by Clarke and Braun. The reflexive approach on understanding the experimental set of pseudocodes alongside their reasoning and the findings on learn effect and experience, provides a suitable framework for establishing meaning. Thorough confrontation by coding the result data and establishing themes throughout aids the analysis of creative artwork in this form of study. The reflexive nature underlines the feminist notion of identifying personal opinions and thoughts when familiarizing with the self-expression, the pseudocodes, of others. Themes may provide further insight on how participants collectively perceive the concept of creative coding in an educational framing, and what processes may have been involved in implementing ideas, overcoming difficulties, and learning to code. To avoid confusion when talking about coding in the context of Thematic Analysis and programming, identifying key descriptions for annotating data points will be referred to as tagging.

For the initial familiarization with the data set, an exhaustive summary of all submitted pseudocodes was created using the technology that formerly visualized participants' progress throughout their course. For the screening, full submissions in the form of the program alongside the participant's reflection were prepared and printed. Interview transcripts, for the sake of tagging, were partitioned into excerpts of dialogue on a topic. Because of the multi-medial nature of the collected data, I chose a digital method to proceed with the tagging and establishing of themes. For convenience, and accessibility, I used the open-source program Obsidian[4] to quickly arrange, and visualize the whole data set. The graph representation of every data point was especially useful to make assessments on the research questions after the last theme iteration.

The process included two days of dedicated analysis where tags and themes were iteratively revised and established. When performing thematic analysis, the integral requirement for successful execution remains in the reflexivity of the researcher. Consequential self-reflection serves as a tool to understand the researcher's role in the process, and enables confrontations with qualitative data that are meaningful yet respectful. In support, a TA diary was designed that would help me identify personal biases towards the topic, as well as apparent relations within the data set. After every session, distinctive ideas were put into a short paragraph alongside a personal reflection on every relevant topic. Finding similarities between my personal belief system and the

[4]Obsidian: `https://obsidian.md/`. Last access: 07.06.2023

responses of participants proved vital to the analysis as, after appropriate bias reflection, those were the key issues most relevant for deducing learning effects from the course. My findings are presented in chapter 4.2 with the respective phenomena apparent from the analysis.

Chapter 4

Results and Discussion

In the following, I will reference the data produced during the study in three stages: All six participants partook in one preliminary questionnaire on their programming opinion and experience, at least five course submissions to the chapters of their choice, and a final interview. The final data was analyzed for every part of the study respectively, and in comparison to each other. Accordingly, responses were considered in cross-examination, as well as individually to find meaningful themes and patterns. With the help of Thematic Analysis Synthesis, the entirety of the findings provides arguments for the effectiveness, potential, and overall success of a programming course tackling social issues under free and creative coding tasks.

4.1 Results

Over the course of the study, of eight participants, six reported successful completion. Causes for study discontinuations were external time restraints in both cases, which led to new participants being chosen under equal requirements. Every data point was formatted for screening purposes and translated from German to English with an effort to not alter their meaning.

4.1.1 Preliminary Questionnaire

With a questionnaire introducing participants to the study, its results provide verification on the programming knowledge of the participants alongside their personal judgement of the topics they were confronted about. Not only served the questionnaire as an aid to establish a baseline to assess their learning journey from, but it enables possible speculations on future submissions by each participant. All figures regarding the preliminary questionnaire can be viewed in chapter A.1.

Most participants verified the assumption of having few connections to the field of programming. While only one of them was familiar with online learning platforms on coding online, he hasn't made an attempt at mastering any of their courses. In addition, only half of the participants were taught programming in school, of which neither pursued any further qualifications. While only one participant ever attempted to learn how to program privately, four people were versed in graphical software, which is promising as the editor environment is designed as one. The data provides evidence for the lack of programming experience within the test group, while implying that a graphical program may not be difficult to navigate for them. Ensuring that participants are not versed in coding makes steps of improvements during course progression noticeable; moreover, it entails fresh view points during course exploration, which are necessary for evaluating novel programming course concepts.

A.1.2 showcases the ways every participant described programming as they see it personally. Most of the adjectives characterize programming as difficult and complex. Words like *algorithmic*, *mathematical*, *tricky*, or *structured* infer that people may perceive it as daunting or hard to get into. In specific, Peach, Jackfruit, Grape, and Lemon posed strong descriptions of their social perception of code, while the other remaining two chose descriptive and technical vocabulary for the characterization. This becomes more interesting when we observe the responses to the following questions regarding the coding experiences of every individual.

Figure A.1.3a and A.1.3b picture the text responses on questions regarding experiences and learning ambitions in programming. Participants regarding programming as complex and mathematical see their confrontation with it mostly in relation with general software tasks such as "game making, creating small programs [...], analyzing data", "excel", or "social media platforms". On the contrary, Apple and Grape had specific responses about programming courses they participated in, or projects they attempted. In general, all participants voiced interest in practical applications of code, as well as specific fields fit to their personal interests. The responses imply that the social view on programming is centered around the production of large software systems for participants who have no strong involvement with the topic matter. People who have had a glimpse at programming tutorials provide more specific goals for a dedicated course to solve problems for their private projects. The answers show that coding may feel too large of a topic for laypeople to get involved with.

4.1.2 Course Submissions

The most eye-catching artifacts produced during the study were the creative digital collages called pseudocodes. Section A.3.8 encompasses the entirety of course submissions by each individual. Alongside a JSON file that is needed to render the respective collage, it was possible to observe the graphical interpretation alongside the reasoning and interpretation. Further elaboration on possible misunderstandings or room for further discussion were conducted in the final interview, and its results will be presented during chapter 4.2. The results were observed under aspects regardless of skill. The goal was to deduce whether the topics of the chapter were realized in the creative piece, how they were interpreted, and what new perspective on the topic it may provide. Further investigation involves the prominence of course chapters, what materials seemed the most inspiring across participants, and what stylistic devices were used the most within the ASSEMBLYRE tool.

4.1.3 Final Interviews

The main goal was to find evidence for newly learned knowledge on the topic of courses, and establishing whether it can be applied in an open discussion. Every participant proved considerable and individual confrontations with certain course topics and were eager to elaborate on their creative collages. All six people voiced similar initial difficulties and provided feedback on the strategies to overcome them. When asked about the inspirations for their creative collages, most participants drew from personal experiences, their hobbies, intrusive thoughts while connecting them to the topics at hand. When directly confronted with elaborate ideas presented in the course, participants were equipped to form opinions supported by their pseudocodes. Direct responses to the course and individual experiences revealed possibilities to improve the overall course structure.

Initial Difficulties During the initial stages of the study, it became apparent that five out of six participants encountered difficulties, which were expected due to the nature of the probe study. While all participants had a solid understanding of the material and did not face any comprehension obstacles, they struggled with the open-ended task of creating a pseudocode. The concept of designing a program that was not meant to be executed immediately confused them. They expressed concerns about not being able to determine the correctness of their code, which added to their confusion.

Addressing this issue required open and transparent discussions among the participants and researchers. It was crucial to provide examples of possible submissions upon specific request to clarify expectations. It became evident that clarifying the course's emphasis on self-expression rather than strict program validity was essential right from the beginning. To ensure participants fully grasped the values the course aimed to convey, the implementation of an infrastructure that involved physical attendance was considered beneficial.
After thorough consultation and support, all the participants were able to overcome their initial hesitations and complete their submissions within the designated time frame. Granting participants the freedom to work independently, despite their initial concerns, proved to be feasible and opened up possibilities for future course designs that would not necessitate constant consultation and attendance for learning computer science fundamentals.

Moreover, the fact that non-tech-savvy learners, who would typically be unlikely to participate in similar programs, successfully completed the course and engaged with the subject is significant. Their accomplishments and active involvement support the assertion that this programming course provided a meaningful learning experience.

In summary, the study initially encountered challenges related to participant confusion and uncertainty in a probe study on programming. However, through effective communication, clarification of objectives, and support, all participants were able to overcome these obstacles and successfully complete their submissions. This experience highlights the feasibility of designing courses that prioritize self-expression over strict program validity and encourages the participation of individuals who may not typically engage with computer science education.

Overall Experience To assess whether the course form proved to be successful, participants reported on their experience with the course material, the platform, and the editor. While all participants reported positively on their enjoyment and interest in the course, these responses must be balanced against the effectiveness of conveying information in this style. The following evaluates the overall consensus of the participants in regard to their experiences with the course. I aim to deduce whether they improved, or hindered the course's ability to educate in the hopes of reproducing findings from similar courses. [Dufva, 2021, Peppler and Kafai, 2005, Vora et al., 2022]

The material presented to the participants remained to be the main attrac-

tion of the course. Participants appreciated the freedom to choose whatever media they found most interesting. Despite the varying amount of resources throughout the chapters, there was no correlation between the chapter complexity and size to the amount and quality of submissions. Contrary to the assumption that participants would prefer the video games and short videos over the longer texts and discussions, participants were open to every resource despite the long scientific journals. While the video game material gave many the inspiration for their pseudocodes, it were the talks by Lepore and Lippman, and Yuval Noah Harari and Ohlhaver that sparked the most ambitious creations by the participants. Contrary to the belief that complex feminist research may overwhelm readers unfamiliar to the discourse, specifically in the area of technology, participants who consumed the theoretical material did not characterize it as overwhelming. While it was perceived as extensive, participant Grape admitted to mostly skimming through it, and not giving it more attention than possible for his mental load. This does not infer that learning material must be easily digestible; more so, it would benefit from proper guidance, and discussion. Of all participants, four interpreted the more difficult material in their submissions [A.4.1:8, A.4.5:8, A.4.3:7, A.4.6:2] and requires further investigation on the effects of material of varying complexity. In general, the length and topic of learning materials did not correlate to them being less impressive to the learners.

All participants reported a mostly positive experience, where occasional technical difficulties would arise. For an online programming course to work on an online hosted platform, server space must suffice. Since data was not stored on a dedicated server administrated by myself, visual feedback for course submissions were not automatic and immediately apparent. This lead to confusion regarding data transfer, where user hubs wouldn't update before manually added. "I didn't know if my submission was actually working, so I was left out in the blue whether everything worked out. It would be cool to see your hub change as soon as you upload something" remained to be a prominent concern, and makes dedicated server structures attractive for future studies and general course organization. When asked about the feel of course progression through the hub, two participants felt encouraged, and liked seeing their own page getting additions with every submission. Providing feedback for course progression without falling into design habits that online coding platforms provide, e.g. achievement systems, linear completion, or progressive task unlocking, can be realized with a visual representation of work that the participant is able to manipulate. By actively engaging with the course, participants can influence their visual representation of their progression, implying that their engagement produces a unique playground in the form of their hub.

The editor as the main sandbox environment to produce pseudocodes was perceived as fun and engaging. Users noted that the graphics and backgrounds were especially engaging, which also influenced their choice of their favorite submission. The addition of unique, and playful imagery to the editor proved to be a fitting touch, as it invites users unfamiliar with both coding, and graphical interfaces to express themselves regardless of their skill. Both Apple, and Jackfruit pointed out that the graphics were fun, and their favorite part of the creative work.

Both participants and Peach stood out with their attempt to produce geometric artwork [A.4.1:3, A.4.3:1, A.4.2:2] despite the tool's lack of dedicated functionality. Observations as such support the definition of a user acting autonomously when facing interfaces they see themselves limited by; in particular, they are universal users as described by Lialina. Examining the autonomy participants embody when they circumvent technical limitations implies materiality. Seeing participants push the boundaries of the tool in the simplest of ways (like trying out Markdown functions [A.4.1:1] or embedding external links [A.4.1:3]) proves the learners' motivation and ability to express themselves through code.

Inspiration Sourcing Inspirations were the crucial part for designing the course in the first place, so examining their success in sparking peoples' creativity was crucial to determine their potential. It's apparent that all participants were especially drawn to the digital resources [A.4.1:1,3,4,8, refpseudoprograms:peach:4, A.4.5:4,8, A.4.3:1,5, A.4.4:4, A.4.6:1,2,6]. While the games seemed to bring joy to the participants, the longer, more serious discussions served as valuable material for elaborate discussions on the topic. For instance, [A.4.5:4] showcases an attempt at generative artistry while interpreting the concept in his own way by individualizing it through values that have more personal value. This is a trend in pseudocodes stemming from course resources, where their ideas are combined with topics the participants are familiar with. There is a fluctuating balance between the influence of the material, and the creator's hobbies and interests on the creative piece. Where participant Apple was taking great inspiration from the resources with less influence from personal interests, participant Peach was predominantly driven by her hobbies, namely cooking and knitting. Both are equally rich and give insight on their learning efforts, which are discussed in Chapter 4.2.3. Acknowledging the difference between the types of inspiration is crucial to determine the course quality, as it serves as a baseline to assess learning experiences. Further variables will be introduced in Chapter 4.2.1 during the central analysis.

The synthesis of empirical resources with individual factors is supported by the references to external resources that participants made use of [A.4.1:1,4,8] [A.4.6:7]. Especially, participant Apple was very clear about his use of AI images in his work to underline the referenced social inequality inferred by the work of Yuval Noah Harari and Ohlhaver and Benjamin. External research outside the scope of the course material leaves to assume great interest in the topic, and a desire to support the gravity of creative work, which provides further estimate on the course's quality. Apple's research, albeit invisible in the final result, becomes the more meaningful as it supports the idea of the "general-purpose user" described by Lialina. Regardless of the tool's capabilities, Apple engages with his world according to his motivation to make his case on racial inequality. To create images that support his claim, the evidence presented in his pseudocode are not the sole art, but the system he used to generate them. The heavy-handed generation of AI images that project their biased training data carry the meaning of the art piece, and in combination with pseudocode and graphics, highlight the ability to produce critical technological discourse in a *learning to program* context.

Course Improvements The course form, with its goal to educate, entertain, and inspire laypeople to approach programming, was positively acclaimed with remarks concerning infrastructure, and introduction. When asked about their overall experiences, users described it as fun, novel, and informational. All participants pointed out that the topics discussed were new, and engaging, and exploring the material was unfamiliar, yet intriguing. Participant Lemon pointed out that these types of courses don't really exist, and may be worth exploring for other fields of interest [A.6.5]. The multi-medial approach to autonomous learning felt refreshing [A.6], and the additional graphics resonated surprisingly well with the participants [A.6.1]. While the course proved to be an overall enjoyable experience for most, participants voiced valid and crucial concerns for future iterations.

While the texts in the zine were considered interesting and informative [A.6.5], participants felt like they were thrown into very difficult discourse without being equipped with the tools to make a real stance on it. [A.6.4]. The participant elaborated that, especially when you have no knowledge of programming, it is difficult to approach an explorative task with open questions. He was not the only one who felt frustrated with the creative task, as there was no clear feedback on whether their progress was right or wrong. While the course tries to distance itself from marking submissions as valid or not, not all individuals feel comfortable with this style of education. Participants noted that a brief

introduction to what one awaits, and what pseudocodes even are, may alleviate some of the starting issues.

All participants with initial difficulties provided their thoughts on the additional aid they were provided with. The four participants Peach, Pineapple, Jackfruit, and Lemon faced similar problems with their purpose in the study, and were received similar responses by myself to help them get a clear view of their task.

4.2 Discussion

In the following chapter, I will examine the final results of the qualitative thematic analysis process. The word *codes* from the original literature by Clarke and Braun was substituted with the word *tags* to avoid misunderstandings when addressing the act of coding. Alongside the TA discussion, every participant will be examined respectively to clearly distinguish between (1) the effectiveness, and quality of the course form, and (2) the learning experiences of every participant. Only when both questions are addressed separately, relations between the two can be deduced.

4.2.1 Thematic Analysis

To fully grasp the impact of the course, and whether it managed to fulfill the goals of communicating computer science basics alongside its social aesthetics, all data points, including course submissions, individual reasoning, and interview responses, produced a promising set of themes which can be observed in Table 4.1. The detailed description, highlighting, and file structure of the analysis can be openly accessed in the public repository on GitHub.[1]

Tag Description

From the 116 data points, I was able to deduce 63 codes that have their respective descriptions in the open repository. Many of the tags follow certain patterns that can be used to group them, and to point out their meaning to the study.

Appreciative Feedback Participants in many ways responded positively when asked about what they enjoyed about the study. While, obviously, games

[1] Arduqq: Assemblyre, `https://github.com/Arduqq/assemblyre`. Last access: 07.06.2023

and appealing visuals would be considered obvious stand-outs, especially the variety of media presented was equally interesting. The positive feelings the appreciation induced was surprisingly prevalent, as many noted that especially the different kinds of graphics that one may choose for a background left them engaged and longing for more. That leaves to assume that regardless of the actual content of the course, it won over the participants with its visuals.

Inspirations The mostly discussed topic throughout interviews, and submissions were the sources of inspirations the participants were able to gather. While most creative pieces were due to the appeal of the resources, personal interest, hobbies, and even the editor itself were valuable sources to ideas. My intent to bring participants a variety of resources to choose from proved successful as most individuals felt compelled to interpret the digital, and analogue media.

External Media While the set of material exhaustive, and diverse, as participants noted, many were led to the internet to widen their research. In search of programming tutorials, technical tools, and related topics, individuals were able to find help online in cases of inspirational blocks, and could even make a stronger point in their submission by including resources from elsewhere.

Personal Feelings The participants were frequently asked to elaborate on their personal stance on programming. In order to observe whether they were moved to re-think their stance, participants pointed out various observations regarding programming. Those ranged from person experiences with code, and the frustrations they faced with it, but also how it must feel to be a professional programmer, or how it oneself is affected by it. Especially personal reflection proved to be valuable moments, as they showcase reflexive moments that were probably induced by the course itself.

Problems With many flaws emerging from this experimental type of programming course, it was crucial to find the weak points to improve future iterations. Aside from the infamous programming bugs that were caused by my own oversights, it was mainly a question of the participants' compatibility with the course. While it's unfair to say that certain participants are not able to work on such a course because they just were not used to it, their reflections are invaluable to adjust the course form and its way of communicating.

Program Situations Individual feelings about coding inferred statements on where programs are, where they belong, and where we as people have a say on how we use them. Participants were able to form clear opinions on where they seem themselves using programming, where it currently is used or misused, and how the real world embeds them in our social framework.

Style and Strategy Despite the great variety of styling, interpretation, and overall message, participants produced pseudocodes that featured their very distinct hand-writing. Their syntax choices, color compositions, or ways to work on the assignments were fascinating as they also revealed their perception on how code looks like to them, and how it is comfortable for them to read, and write.

Theme Description

After numerous iterations of tagging, and grouping, I was able to establish 10 themes that capture the essence of the course progress well. They form principles that influence the outcome of the study in regard to the learning effect, and ability to communicate the problems of computer science. In the following, I will describe the principles, and how they attribute to my research question.

Concept Clearance The unusual way the study was presented in the form of a semi-digital probe study with its aim to encourage free play in a visual editor struck certain doubts in the participants. Initial problems were obvious, yet the ultimate error doesn't lie in the individuals' capabilities. The study uncovered that communicating a clear concept, and easing participants into the study is important for them to not feel overwhelmed, and frustrated. While all participants were able to finish the course, most of them only encountered problems at the very beginning because of uncertainty regarding the role in the study. Discussing the main frame of their tasks alongside dynamic consultations may help participants feel comfortable to express themselves, regardless of the open nature of a probe study.

Time and Guidance Economy When people offer their attention to you in any way, they invest energy, and time. The study was considered time-intensive; although, most participants took their time to explore the probe. When we design courses that are supposed to educate, the idea of the busy user is obvious. When designing a course, one has to attribute for how much time it would take for the participant to gather valuable information for them to apply in their tasks. In the case of an explorative study, the amount of guidance we

provide is the key to remaining observant while the individual explores the playfield. While it was not always successful to keep participants engaged with certain topics, or give them the guidance they sought, it's clear that these parameters have to be balanced out dynamically for every participant.

Course Enjoyment While the study was designed with visual appeal in mind, participants reassured that an attractive learning environment proved to be motivating, and enjoyable. Moreover, the tone of the materials must be fluctuating to enable experiences that are both fun, and informative. Even though the study aimed to move past gamification, and linear progression systems in favor of free exploration by an all-purpose user, games are not inherently bad study design. They contribute to experiences that stick to individuals, as their unique nature is a proof-of-concept for anecdotes. When designing a programming course, fun, and enjoyment play a considerable role in the participant's motivation.

Programming on the Web While at first obvious, the internet is a unique medium that is affected by programming very differently. Basic programming tasks rarely touch on the fundamentals of the internet, yet its existence greatly impacted the programs, and experiences of the participants. External resources played a valuable role in filling in the gaps for individuals when the course tutor wouldn't be able to. Its vast possibilities of providing endless knowledge should not be taken on the light shoulder, and most-likely must find application during *learning to code* processes. Not only was the actual programming taking place on an online website, but its absence would not have made any of the resources available. Examining these processes opens up a whole new research track that must be examined with a future that relies on the internet, and harbors numerous power structures to analyze.

Inspiration Network When we observe the different way participants felt inspired to create, to consume, or to engage with the discussion at hand, it becomes noticeable that all the factors intertwine sooner or later. While the initial motivation to create was induced by the material provided by the course, participants predominantly used their own experiences as inspirations to comment on the individual topics. It leaves to assume that the intersection of these influences is a common event during design processes that are meant to be explorative, and provoking. Accepting the participant as fully capable to get inspired by themselves is important to not take away from the exploration by themselves. They are aware of their surroundings, and the purpose of the course is to not control them, but to present information in diverse, and meaningful ways. What kind of creative work emerges is fully up to the learner.

Learning in Materiality Materiality, the embodiment of the individual in the learned medium, as well as the acknowledgement of the medium being transformative, played an active role in discussing the learning experience of every participant. Whether they found playful ways to overcome the editor's capabilities, the materializing of coding strategies, as well as the mere act of taking notes on a notebook fit well into the very theory the course was based on. The approach of the post-digital programmer proves successful when participants consider themselves programmers even though they did not produce a single line of working code. Their thoughts, emotions, and values intercept with the coding medium, and form unique learning experiences in which they themselves embodied the medium.

Aesthetics of Programming As previously stated, aesthetics are not solely visual. They are shaped by individual, and societal norms, and inherently transformative. Throughout the study, participants were asked to change their perspective, and to think about groups, minorities, and objects that would be affected by programming in numerous ways. Identifying these instances, where the individual is able to see themselves as an active player while acknowledging their own actions in the system, are crucial to make a statement on their social consciousness in regard to programming.

(Post-)Digital Learner A course form that leans towards the post-digital paradigm of *learning to code* will likely produce artifacts that can attributed to its nature. To my surprise, many learners with their individual perspectives voiced opinions, created algorithms, and experienced the course in ways that can be attributed to either the digital, or the post-digital. To be precise, conventional syntax, homogeneous recipes for course submissions, or a mostly technical perception of code are patterns that resemble the need for a more technically focussed course form such as the ones predominantly used in the industry. In contrary, participants thrived on the overall design of the course, and felt the most comfortable designing their pseudocodes without the need to tackle practical tasks. These inconsistencies among participants reveal weaknesses in the course form that can be avoided by embracing the strengths of existing concepts. Navigating between the two paradigms poses the question, whether hybrid forms of digital and post-digital courses could benefit from each other, and how we should avoid binary thinking when deciding on educational paradigms.

Summary

The thematic analysis revealed multiple key factors that have to be accounted for when we try to convey social aesthetics alongside programming fundamentals. Course forms that lean towards a post-digital approach require attentive care to not overwhelm participants with vaguely communicated concepts, and unreasonable time constraints. Especially when exploration, and play are central themes of a course, treating the participant as an autonomous, complex learner is essential. People choose interesting material by themselves, the research with the help of the internet, they overcome boundaries, and do not require academic intervention to be inspired. Individuals will discover the capabilities of a new medium on their, as they are the ones who understand their own flaws and how to cater to them.

To understand the participant's reasons to create, we ought to examine their inspirational network to deconstruct their ultimate motive. We can follow their trail of inspirations to pin down the information most valuable to them, in order to understand the impulses that formed every learning process. These impulses may come from personal revelations, changes in society, simple visual triggers, and exploring those possibilities may provide fruitful discussions on the future of social technology.

When we try to attribute certain characteristics to every learner, assessing their attitude towards the course paradigm seems tempting. While a digital learner is more likely to produce complex algorithms, provide technical commentary, and perform well in direct programming tasks, post-digital learners excel at defining unique syntax conventions, identifying systemic injustice in technology, and embracing the artistic potential of code. As simple as this assumption is, it certainly won't change the way we educate programming. The ideal learner is one that can traverse between these paradigms, supported by the course designer. They are able to explore the medium as a universal user, and can produce learning artifacts that reflect their learning needs, biases, interests, or capabilities well. Their ability to move on a spectrum between the digital, and post-digital world makes them the Peri-Digital Learner. One who embodies the medium by approaching it both, technically in its socially embedded form, and in an analogue way that captures its essence.

Table 4.1: Thematic Analysis: Themes, Tags, and tag occurrence Δ

themes	**tags**	Δ
Time Economy	problem of intense resources	5
Time Economy	problem of lack of information	1
Time Economy	problem of mental capacity requirements	2
Time Economy	problem of inspiration block	6
Time Economy	saved for later	1
Programming on the Web	external resource discussion	5
Programming on the Web	external resource help	2
Programming on the Web	external resource	11
Post-Digital Learner	programs and the real world	7
Post-Digital Learner	programs and me	4
Post-Digital Learner	programs and time	5
Post-Digital Learner	strategy of meta discussion	9
Post-Digital Learner	style of breaking editor capabilities	4
Post-Digital Learner	style of graphics over code	3
Post-Digital Learner	style of rudimentary code	7
Post-Digital Learner	mind over matter	2
Post-Digital Learner	strategy of traditional note-taking	6
Post-Digital Learner	Strategy of recipes	4
Post-Digital Learner	appreciation of open questions	2
Post-Digital Learner	strategy of distance from resource	3
Learning in Materiality	mind over matter	2
Learning in Materiality	strategy of traditional note-taking	6
Learning in Materiality	interdisciplinary and intermedia	3
Learning in Materiality	matter over mind	4
Learning in Materiality	playful	5
Learning in Materiality	personal positioning	4
Learning in Materiality	playing with colors	2
Learning in Materiality	programs and the real world	7
Learning in Materiality	playing with the editor	9
Inspiration Network	problem of inspiration block	6
Inspiration Network	inspired by anecdote	4
Inspiration Network	inspired by editor	7
Inspiration Network	inspired by hobby	6
Inspiration Network	inspired by interest	8
Inspiration Network	interdisciplinary and intermedia	3
Inspiration Network	inspired by resource	18
Guidance Economy	help needed	4
Guidance Economy	help unsuccessful	1
Guidance Economy	problem of inspiration block	6
Guidance Economy	help was successful	2
Digital Learner	Strategy of recipes	4

Digital Learner	not like other programming courses	5
Digital Learner	problem of editor bugs	4
Digital Learner	problem of no feeling of programming	2
Digital Learner	programming background	3
Digital Learner	strategy of conventional algorithm	10
Digital Learner	strategy of interface design	3
Digital Learner	strategy of iteration of resource	3
Digital Learner	style of long code	4
Digital Learner	matter over mind	4
Digital Learner	problem of out-of-place code	2
Digital Learner	problem of too much freedom	1
Digital Learner	problem of tool uncertainty	1
Digital Learner	style of technical focus	9
Course Enjoyment	appreciation of open questions	2
Course Enjoyment	appreciation of discourse	4
Course Enjoyment	appreciation of games	5
Course Enjoyment	appreciation of the editor	1
Course Enjoyment	appreciation of variety	5
Course Enjoyment	playful	5
Course Enjoyment	appreciation of visuals	6
Course Enjoyment	personal enjoyment	9
Concept Clearance	problem of out-of-place code	2
Concept Clearance	problem of too much freedom	1
Concept Clearance	problem of tool uncertainty	1
Concept Clearance	problem of demanding tasks	1
Concept Clearance	problem of initial confusion	1
Aesthetics of Programming	personal learnings	4
Aesthetics of Programming	personal perception of programming	7
Aesthetics of Programming	personal reflection	11
Aesthetics of Programming	personal revelations	10
Aesthetics of Programming	programs and emotions	3
Aesthetics of Programming	programs in art	2
Aesthetics of Programming	programs in consumerism	5
Aesthetics of Programming	reference to generative artistry	2
Aesthetics of Programming	programs in profession	4

4.2.2 Individual Participant Perspectives

In the pursuit of clear profiles on the participants' growth, and learning experiences, every individual is examined on their own on the basis of their interview. By analyzing their study artifacts in the form of their questionnaire, submissions, and interview transcript, we can get a clear idea of what possible course outcomes can look like.

Participant GRAPE

Participant Grape [A.6.2] had a positive experience with the programming basics course and found value in its content. Grape actively engaged with the course by exploring the available resources, even bookmarking the ones he found interesting. Not only does it imply that he had fun with the concept in general, but leaves to assume an interest in the topic beyond course completion. His interest in specialized topics adhering to programming like creating RPG Maker games proves an interest in environments that are inherently graphical, but offer a way to be more precise in your creative ideas through custom coding. This is supported by his personal opinion on programming, where he stated that he wouldn't avoid using code in programs that are not primarily focussing on it. [A.1]

Accordingly, the participant brought prior programming experience to the course, including skills in data analysis, creating polar plots in Python, and working with analysis algorithms. This background likely contributed to his ability to engage with the course material and grasp programming concepts more easily. His openness to programming resources reflects in his questionnaire responses as he is familiar with their availability, and is eager to learn from an online platform. That opens whether basic programming courses should be exclusively aimed at people who had no trace of programming in their lifetime.

During the course, participant Grape developed his own personal coding style and naming conventions. While this may have deviated from standard coding conventions [A.4.5:2,4,7], it showcased his adaptability and problem-solving approach. Grape's personalized style allowed him to better comprehend and troubleshoot his code, demonstrating his ability to navigate programming challenges effectively. The eagerness to transform syntax in a way he finds it easier to work with while maintaining a standard that is readable to others, he showcases a common goal of programmers to create programs that are maintainable, yet biased on the programmer's style: "When I create variables, I don't see their purpose, like the solutions on Stack Overflow. I don't see myself coding with them successfully. That's why when I try to do something difficult, I do something like ArrayWithThisValue1.". While industry-standard may avoid this approach, it's what by the definition of Grape is what keeps him creating, and revising his code. Embracing this individual approach where participants are able to develop strategies in code is crucial to let them remain reflective in the future when programming.

One aspect of the course that Grape particularly appreciated was its focus on social problems connected to programming. As he identifies: "if you are not educated, terrible things will happen". He found value in discussions on marginalized groups being ignored and issues of sexism. Through the course, Grape gained insights into the potential of software implementations in addressing social issues and the importance of considering the broader societal impact of coding decisions.

Grape also expressed confidence in discussing these social problems with other laypeople, citing a successful experience in resolving an argument related to a representative of a black rights organization. This indicates that the course not only enhanced his technical understanding but also facilitated the development of communication skills and critical thinking, enabling Grape to engage in constructive discussions on these topics. While this success can not be attributed to the course completion, it highlights his belief system that will greatly benefit from his digital literacy to create mindfully.

Furthermore, Grape exhibited a positive sentiment toward his course submissions, expressing an increasing appreciation for his work over time: "The more I look at them, the more I love them". This sense of accomplishment and confidence in his abilities suggests that the course effectively motivated and inspired Grape to delve deeper into programming concepts, which is supported by his interest in using it creatively, and professionally.

In terms of course delivery, Grape initially encountered some challenges in navigating the course website, particularly regarding code input. However, he ultimately found the experience enjoyable and praised the quality of the website and the provided digital resources. The design and functionality of the website contributed positively to his overall engagement with the course.

Based on Grape's feedback, it can be inferred that the programming basics course effectively communicated programming fundamentals and raised awareness of social problems connected to programming. Grape found value in the available resources, actively engaged with the coding exercises, and developed a deeper understanding of societal issues in the context of programming. His positive attitude towards programming was upheld by the social problems that proved to be transformative, and underline how reflecting on social topics during technical programming is a feasible path to pursue. However, Grape also provided suggestions for improvement, such as incorporating more complex thematic content and introducing incentives like a leveling system or hidden leaderboards to enhance motivation and motivation among participants. These suggestions aim to foster a more immersive learning experience in future iterations of the course.

Participant APPLE

Participant Apple [A.6] found the course surprising in terms of its programming content. Apple mentioned feeling pushed to explore creativity and self-expression, which suggests that the course encouraged a more open and non-structured approach to programming. This indicates that the course form successfully challenged participants to think beyond conventional programming paradigms, and was reflected in the submissions he produced.

Apple showed interest in various resources throughout the course, particularly those focused on creative aspects and practical application [A.4.1:1,3,4]. They mentioned being inspired by an artist who creates programmed art and engaging with digital workshops to try programming themselves. This indicates that the course effectively introduced practical programming skills and encouraged participants to explore creative avenues within the field. Generative artistry in its form remains to be inspiring to laypeople, which legitimizes its use in processes of *learning to code*.

Moreover, Apple's active participation in the course, including his attempts at programming a circle in a JavaScript-based environment, demonstrates a hands-on learning approach. While the course didn't active encourage participants to program in a hands-on style, the participant's actions imply that it would benefit the impact of it when guided through appropriately.

In terms of the course's communication of social problems connected to programming, the participant highlighted his surprise and deep reflection on the topic of AI adapting intrinsic racism in the data it is fed [A.4.1:8]. This demonstrates that the course effectively introduced and raised awareness of critical social issues, and was supported by real experiences in the web. The participant was motivated to prove the very claims of activists such as Benjamin, and used conventional AI image generation to create images of criminals and employees which both supported the claim. By using this as a case in a pseudocode submission, the participant created a program based on his own reflection, curiosity, and opinion that came together through active engagement with the biased technology.

Additionally, participant Apple's use of an image AI tool and his connection of the course material to a medical museum exhibition and discussions on race theory showcases the interdisciplinary lens through which they approached the topics. This suggests that the course form successfully encouraged participants to consider the broader societal implications of programming and its connections to various fields, which is supported by an interest in applying programming in his professional field.

Participant Apple learned from the course and found value in its approach to communicating programming basics and social problems. His active engagement, practical application of programming skills, and reflection on social issues indicate a meaningful learning experience. The course form's emphasis on creativity, practicality, and interdisciplinary exploration appears to be a valid approach for effectively conveying programming fundamentals and promoting critical understanding of social implications.

Participant PEACH

Participant Peach [A.6.1], despite having a fundamental computer science course in school, rarely encountered programming tasks in her daily life. She perceives coding as confusing and difficult, mostly associating it with spreadsheet workflows in Excel. This lack of creative experience led to difficulties in understanding the significance of pseudocode during the course. However, over the course of time, the participant gained a noticeable interest in the theoretical discourse of modern technology, establishing an understanding for ubiquitous computing processes behind appealing software UI.

The fact that programs weren't executable, contrary to common convention, immediately raised insecurities, which she voiced. With supplementary material and further explanations, she received guidance and started creating pseudocodes inspired by her personal hobbies and interests. Her course submissions, such as a lighthearted pie recipe, knitting instructions, and a flower store website, took a graphical approach while attempting to depict the programs running in the background.

Reflecting on her approach, she explains, "I imagined if I put it [the variable] in like this, now how would that play out, and then what's going to happen to it? [...] I thought, you can't really see that, and I just imagined that it would just run in the background." She acknowledges that any interface, even "just Pinterest pages," has an underlying code base that is often beyond comprehension. Accepting the editor "as a design element" and "a little ideas page" was crucial for completing the course. She states, "once I understood that I can just program anything, and not to use a language, [...] it was quite cool because [...] you could think through how you would implement it."

During discussions, Peach found social and political content related to programming the most appealing, as she wanted to explore complex topics without necessarily involving programming efforts. She shared her opinions on AI discourse, saying, "If that's the way it is, then it will take forever until artificial intelligence can really understand feelings. [...] [Programming is] somehow totally simplified, because the real world is more complicated than that and there are ten thousand other things involved that you can't represent. Maybe you could, but you'd spend ages on one little thing to represent all the influences alone."

Peach connects the consciousness of programs operating in the background to the reductive abstraction present in computer science, indicating that the course made a lasting impression on her. She also mentions the tremendous work required for such systems, which is reflected in her own work during the course when she asks observers to "imagine that [she] fed [the data] in one by one," adding "dot dot dot." This custom syntax reflects generative processes inspired by the written word and implies a material process where she physically confronts the concept and learns as

a result. By identifying and avoiding busy work, she developed an understanding of the significant amount of work put into ubiquitous algorithms.

Participant LEMON

Participant Lemon [A.6.5], who had no prior knowledge of programming, initially struggled with the open-ended nature of the study. They found the unguided tasks and non-linear progression to be particularly daunting, especially in an unfamiliar area. However, through detailed discussions on the essence of the course and examples of submissions, he was able to develop unique ideas and perspectives on the topics at hand. Reflecting on his submissions, the participant remarked, "I used to find it very abstract and didn't have a clear idea about it. But now I just find it very funny. Looking back on what I submitted, I didn't really think too much about it and just tried to make something that matched my thoughts." Their submissions had a lighthearted and comedic tone, often not relying on many components, but rather expressing specific thoughts and concepts that emerged during their research. Despite their simplicity, all the pseudocodes they created could be formalized as short algorithms that processed ideas and concepts, with clear inputs and outputs symbolized by images. What stood out about his approach was their consistent development of a unique style throughout the course, gradually increasing the complexity of their expressed algorithms. This growth in complexity indicates their active effort to improve and learn as they progressed through the study.

The participant did not shy away from tackling difficult questions, but they also did not feel that every creative piece he submitted needed to be equally complex. Instead, he embraced the freedom to explore and express his ideas. As the participant explained, "I found it very refreshing to have these big questions, but it was overwhelming at first when you had all these problems and thoughts posed. [...] Looking at the examples you sent me, I felt more free to submit whatever and suddenly, I had a lot more fun with the course." While the primary focus of the study was not to provide a purely entertaining experience, the participant's enjoyment and engagement with the course demonstrated how creating an engaging and enjoyable learning environment can greatly benefit the learning process.

Despite the simplicity of their submissions, the participant's interest in more complex topics did not wane. They showed a particular fascination with the social aspects of programming and were eager to explore discussions and concepts that went beyond the technical aspects of coding. For example, they were captivated by the game "Echo Beach"[2] which provided a showcase of the effects of Desktopization as de-

[2]Tim Sheinman: Echo Beach (https://tim-sheinman.itch.io/echo-beach) Last access: 04.06.2023

scribed by Soon and Cox and Lialina. Through in-depth discussions, the participant was able to independently identify the core issue being addressed by the game. They expressed, "It was so shocking when you realized that you would be able to endanger these people with just one click when I actually really liked them." This example highlights how incorporating divisive or thought-provoking concepts into new digital media can lead to meaningful insights and conclusions for participants without imposing a predetermined narrative or viewpoint.

In addition to their inclination towards the social aspect of programming, the participant also demonstrated a strong interest in sorting algorithms, indicating a desire for more specialized content and resources on the topic. This interest suggests that a broader course experience, covering a wider range of subjects and concepts within the field of programming, would be well-received and appreciated by the participant.

Participant JACKFRUIT

Participant Jackfruit's [A.6.3] learning effect in programming and its social implications can be observed in several key aspects. Firstly, despite initially feeling apprehensive and lacking knowledge in programming, Jackfruit was able to overcome her concerns and engage with the course material. She demonstrated a willingness to learn and adapt to new concepts, indicating a growth mindset: "I [...] was afraid of what I got myself into here! [laughs] I can't recall anything from my computer science course back in school. [...] But no, it wasn't like that at all, and I found that super nice to just dabble in it and get creative.".

Jackfruit's integration of her personal interests and hobbies into her programming projects reflects her learning effect. Her enjoyment of generative artistry and her ability to incorporate her own creative ideas into her work demonstrate a personal connection to the subject. This engagement indicates that she not only learned technical skills, but also developed an appreciation for the creative and expressive aspects of this style of coding. While she may not make use of the computer science theory as she doesn't consider it to be an interest of hers, the graphical design aspect was appreciated as she later felt more comfortable working with programs like Canva, and Gimp. Learning to program in this context is not entirely aimed at programming systems, but to appreciate and find confidence in existing software.

Furthermore, the participant's practical application of visual skills in her internship indicates a tangible learning outcome. Her being able to navigate new software by play is evidence for an improvement of digital literacy. This suggests that the course provided her with valuable skills that she could immediately apply in a real-world context.
In addition to technical proficiency, the course effectively conveyed the social implications of programming to participant Jackfruit. She was inspired by resources that highlighted social injustices and the impact of technology, such as the video she men-

tioned discriminatory practices based on skin color. This indicates that the course successfully communicated the broader ethical and societal aspects of technology, prompting her to consider the social implications of programming.

Moreover, participant Jackfruit showcased an awareness of the link between consumerism and technology. She recognized the role of advertisements and emotional design in tempting individuals to make purchases: "I immediately think of advertisements, marketing. But also the emotional technology that is designed in a way that we are tempted to buy their products.". This awareness suggests an understanding of the broader societal impact of technology and its influence on consumer behavior, highlighting a nuanced understanding of the intersection between capitalism, consumerism, and programming.

While the course does not provide explicit statements about the participant's comprehensive understanding of programming and its social implications, her ability to apply programming knowledge in her internship, her engagement with the social aspects of technology, and her integration of personal interests all indicate a considerable learning effect. By fostering practical skills, awareness of social issues, and personal connections to the subject, the course successfully conveyed both technical knowledge and social implications, contributing to her understanding of programming and its broader societal impact.

Participant PINEAPPLE

Participant Pineapple [A.6.4] had a mixed learning experience throughout the duration of the course. While he appreciated the aesthetic design and the various resources provided, he encountered challenges in transitioning from theory to practice and struggled to find a connection to the act of programming. These difficulties could be attributed to his lack of prior knowledge in coding and difficulties with open tasks with no clear progression, but also provide insight on the fundamental flaws of the course program.

The participant expressed feeling overwhelmed by the abundance of resources and information despite multiple attempts, which made it difficult for him to sort through, and understand the material. This suggests that the course might have presented an information overload for someone with no prior coding experience, and no hint of what a program actually does. The participant's preference for more practical projects indicates that he would have benefitted from hands-on exercises that allowed him to apply the concepts he learned in a more concrete and structured manner: "I had the feeling most of the time that I weren't the person suited for the study. That's why the smallest unfamililiarities, like all the languages, blocked my head.". When we work with an extensive amount of resources, getting lost is not surprising. In this case, the openness of the participant was very helpful, which proves the initial rapport with the participants to benefit the study. Through thoughtful conversa-

tion, the participant still was able to reach the finish line without leaving the study, despite the grim outlook on his role.

Additionally, the participant's struggles with the free nature of the probe study suggest that he may have faced challenges in navigating the open-ended nature of the course. As a music composer, he may have been more accustomed to working within defined parameters and found it challenging to explore coding without clear guidelines or constraints: "If I look at music, when I compose something, or I am writing style copies, I usually take something that already exists, and try to understand it. I try to comprehend the processes someone else had in their head in the first place.". It opens the discussion on the validity of industry-driven coding lessons, since they would very well benefit participant Pineapple in regard to the remarks he voiced.

However, it is important to note that despite his difficulties, the participant still managed to produce impressive submissions. His ability to characterize each chapter in a distinct way was apparent, with pseudocodes looking individual while portraying every chapter very well. This is reminiscent of his approach to creative work, as he had no trouble creating submissions in a rather short time span. Whenever the assignment became the more clear, he did not fail to produce pseudocodes fitting an artificial prompt. Considering this, it is safe to say that even in a novel approach to educate code, the impact and effectiveness of current, already functioning tools should not be dismissed. Guidance and clarity are fluid factors that should be considered when designing learning platforms.

Overall, while there may have been some learning effect in terms of exposure to coding concepts and the exploration of social injustice, the participant's struggles with beginning to create and adapting to the free nature of the study suggest that the course needs to provide more structured guidance and support for individuals with no prior coding experience. Incorporating practical projects and offering clearer introductions to fundamental concepts could help bridge the gap between theory and practice, enabling participants to develop a stronger foundation in creative coding.

4.2.3 Cross-Participant Comparisons

Based on the analysis of the cross-participant comparison, several common themes and differences can be identified among the participants. Participants Grape, Apple, Lemon, and Jackfruit displayed a high level of engagement, and excitement with the course. They actively explored the available resources, attempted programming tasks continuously, and reflected on the concepts taught. They were able to form their own style by traversing the inspiration network, and leaving their unique experiences in their pseudocodes. In contrast, Participant Pineapple struggled with the open-ended nature of the course and had difficulty finding a connection to programming which, after examining their participant's background, revealed that a person

without coding experience remains to have individual needs and should not be addressed as a generalized layperson.

Participants Grape, Peach, and Lemon developed their own personal coding styles and approaches. Grape's personalized syntax with long variable names allowed him to find a balance between readability and personal needs, Peach relied on the aesthetics of interfaces while abstracting and imagining complex background tasks, and Lemon embraced the freedom to explore and express ideas that occurred during the theoretical research. Even participant Pineapple was able to find peace with his problems with the study, but sticking to what he feels most comfortable in through adhering to the concepts presented, creating visually impressive pieces related to certain resources. This demonstrates adaptability and creativity in their programming journeys, and only supports the idea of a universal user who is able to solve unique problems with the help of their personal experiences.

Participants Grape and Apple had minor programming experience, which likely contributed to their ability to engage with the course material more easily. Grape's experience in data analysis and Python programming, as well as Apple's interest in programming for creative purposes, helped them grasp programming concepts and apply them practically. Both were the most consistent in their submissions, and did not require assistance aside from technical difficulties with the program.

Participants Grape, Apple, and Lemon found value in the course's focus on social problems connected to programming. They expressed an interest in discussions on marginalized groups, issues of sexism, and the broader societal impact of coding decisions. The course's aim to introduce these topics and actively engage with it proved successful and require further investigation with more dedicated resources, which would be appreciated by the participants.

Participants Grape, Jackfruit, and Pineapple mentioned the practical applicability of programming skills, and how they may benefit the course. Grape expressed an interest in using programming for data analysis, Jackfruit applied programming in her internship by creating visuals, and Pineapple sought more practical projects to make the course more tangible. These examples highlight the importance of connecting programming to real-world applications for increased engagement and motivation. While the post-digital approach is intriguing, and applicable in contexts that must be established. Certain points of an individual's programming journey require digital focal points to overlap with their beliefs on what programming means to them.

Participants Pineapple and Grape encountered some challenges during the course. Pineapple struggled with transitioning from theory to practice and suggested the need for more structured exercises. Grape initially faced challenges in navigating the course website but found the overall experience enjoyable. Their feedback highlights the importance of balancing openness and guidance in course design, and require

intervention by the course tutor to support individuals.

In conclusion, participants Grape, Apple, Lemon, and Jackfruit had positive learning experiences with the course, demonstrating commitment, practical application, and an understanding of social implications. Participant Pineapple faced difficulties and struggled to find a connection to programming, indicating the need for more structured and guided learning experiences. Incorporating practical exercises, clearer guidelines, and providing a balance between theory and practice could enhance the learning experiences of participants like Pineapple.

4.3 Conclusion

By examining the interconnected themes, and tags of the data set of unique programming course artifacts, it was possible to deduce central design fundamentals that have to be addressed when designing an aesthetic programming course. Accepting the learner to be curious, busy, and of good faith, may not provide individuals with the ability to develop software, but it encourages them to explore the field further reflectively. Addressing social injustice in the technical field sparks processes of reconsideration of one's own presumptions, empathy for minorities affected by the technological complex, and clarity on the grave impact programmers have with their skill set. By seeing the learner as an individual who engages with the novel medium under the impact of their environment, we assume their motivation to be rooted in an inspirational network that can be sparked by the resources provided by the course. Furthermore, designing educating material on digital topics, may benefit from positioning oneself on the spectrum towards post-digitalism. While not all people are comfortable with current implementations of post-digital learning strategies, such as the one applied in this thesis, it is the educator's responsibility to provide assistance for learners to be able to establish their own learning strategy. Giving learners an opportunity to navigate the subject matter by confronting the medium in different ways, paints the picture of a Peri-Digital Learner. A learner who, motivated by complex impulses, is able to approach the medium with or without it.

Chapter 5

Threats to Validity and Future Work

5.1 Threats to Validity

The programming course, and research design, encounter numerous threats to their validity based on a variety of biases influencing a course that is reliant on the participants' creative self-expression.

5.1.1 Credibility Threats

The internal validity is shaped by the credibility of the people involved in the study. To minimize biases, the study was designed with its political form in mind at every step. The course form is shaped by a large amount of personal intervention that all are a threat to the credibility of the work every participant produced.

Researcher Bias As an artist, feminist and programmer, I myself have strong opinions on what programming is and can be. Every instance of choosing fitting resources for participants to explore is biased, and their contents are largely adherent to my own belief system. To mitigate an inspirational framework that is solely dictated by my own interests, the theoretical resources were largely copied from the work of Soon and Cox. Except for more light-hearted media such a video games, I remained close to the framework their course provided.

Participant Bias Participants may have provided responses that were socially desirable, as the resources they consumed inherently shaped their idea of what the study is trying to achieve. While this process is difficult to assess in an uncontrolled setting, participants are encouraged to remain critical of the resources they consume. Through initial rapport that was established as all participants were friends of mine, I urged the participants to stay open-minded, yet attentive.

Sampling Bias With a small amount of participants, the study is not representative to make assumptions on the general applicability of such a course. The goal is to determine whether its ability to educate is apparent, and to determine learning effects, the examination of participants is of the highest priority. To understand the beliefs, and reasoning behind the creative processes of the participants, I as a researcher tried to examine them as closely as possible, which left me with a small sample amount as the most feasible. Only this is how I managed to gather sufficient data to have a grasp on the rich and diverse data set.

5.1.2 Transferability Threats

The generalizability of the course analysis is limited, as the courses stemming from materiality and the post-digital are encouraged to be copied and changed.

Sampling Limitations The course form is targeted towards people with little to no knowledge in programming. Finding these people is not difficult, yet individual beliefs vary tremendously, and no coding experience does not infer a similar readiness to learn it. Finding a set of participants with already established rapport is a challenge that implies that the design must include processes for building it.

Contextual Factors The uniqueness of the research context limits the applicability of the findings in other settings. Participants were in uncontrolled environments as they worked on every task. Technology itself is a shifting phenomenon which changes social barriers with every day. The material provided may become irrelevant every instant. This type of work is largely dependent on the current social context, and formulating it is necessary for every iteration of the course. For this, providing sources that a course is based on is a first attempt at shifting the researcher's bias.

5.1.3 Dependability Threats

The reliability of the course depends on the time span, and setting the participants are partaking in the study. While reproducing the exact artifacts is not possible, the inherent themes emerge from constant reflection, and open discussion with participants.

Researcher Reflexivity During contact with participants, guidelines were present to shape the way I interact with them without intruding on their autonomy. Responses on how to react to starting difficulties were pre-written, and every interview was conducted with the help of an interview guide to not deter the conversation. Topical arguments were up to the participant, and my own reflection was used as a transition to another topic. During data analysis, a diary supported my efforts to

remain as reflexive as possible, since the process of tagging data points and establishing themes is a subjective process that is impossible to conduct without human intervention - especially when examining social and political assumptions by the participants. With every iteration of the analysis, I noted my own thoughts to isolate personal opinion to my best ability.

Data Saturation Further iterations of the course will provide more saturated data, as the many opinions on the topic cannot be grasped by just six participants. While this first iteration made a proof of concept, exploring the vast identities interacting with technology will produce data that will make a more precise point on what *learning to code* can be in social contexts. To enable this, all data is accessible for future iterations.

5.1.4 Confirmability Threats

To overcome instances of objectivity, the study's moral code was outlined by the inspirations that stemmed from cyberfeminist research. Instead of relying on my personal opinions on the interpretation of the data, I relied on concepts such as materiality, and reproduction of materials to make assumptions on learning effects and course quality.

Researcher Positionality To outline my own thoughts during crucial evaluation steps such as the thematic analysis, all tags established for the data points, as well as the reasoning and description is provided in a public repository. While my own perspective on coding in social environments is largely impacted by cyberfeminist research, all relevant concepts under which the study was conducted are cited in the Background section.

Audit Trail Alongside the public repository and its history, the Overleaf document provides historic data of the production of the study. All planning is documented in a central collection of slides that were discussed during consultations with my supervisors. Designs, ideas, and detailed data can be accessed openly through the repository.

5.2 Future Work

Designing new courses inspired by the creative pseudocode approach used in this study can encourage individuals unfamiliar with programming to engage with the subject safely. Incorporating live coding sessions and topical discussion forums can facilitate active engagement, exhaustive feedback, and personal discussions, enhancing the overall course experience. Creating space for sociopolitical discussions in STEM curricula enables voices directly involved in queer computer science research to contribute their insights and implement their ideas in an open context.

Improving the learning platform itself is essential. Conducting investigations on user experience can provide valuable feedback on the effectiveness of the interaction loop between the zine and the website. Simplifying the submission process, adding customization options, and introducing personal profiles can strengthen the connection individuals feel to their work. Enhancing the editor's export capabilities and providing dedicated animation interfaces can enable users to bring their code to life in a more engaging and creative manner. Additionally, incorporating accessibility features such as a color-blind version, minimal animation, and alternative editor navigation can ensure inclusivity and cater to a wider range of users. Technical additions to the platform can expand its accessibility and inspire more people to engage with the educational resources.

Creating a social network around the learning platform, while acknowledging the potential risks associated with such platforms, can enhance motivation and foster a sense of community among learners. Allowing participants to observe and engage with each other's work can create a collective experience that differs from conventional online coding platforms. Exploring how this feature is used by alternative demographics, beyond the typical audience of visual coding websites for kids, remains an unexplored area of study.

Future research on implementing queer feminist findings into technical education should focus on addressing the challenges posed by the rapidly evolving nature of computer technologies and their impact on society. Exploring ways to incorporate social context into the education of artificial intelligence (AI) fundamentals and capabilities is crucial, considering the increasing relevance of AI in social and political discussions. This education should aim to go beyond industry-driven perspectives and emphasize interdisciplinary approaches. Providing resources that discuss AI systems from both social and technical perspectives can offer new insights and perspectives for learners, allowing them to critically reflect on the topic.

Establishing dedicated forums for discussing technical topics related to the course can facilitate further studies and foster a social learning environment. Exploring the insights and inputs generated through forum discussions, which often offer high-quality and specific knowledge, can contribute to the continuous improvement of the learning platform.

While the study focused on the learning experience of laypeople, conducting a dedicated analysis for computer science professionals can help identify weaknesses in the educational material and refine the course content accordingly. Comparing the learning journeys of individuals acquiring programming knowledge from different backgrounds can also lead to valuable insights and improvements in the course design. Additionally, examining the role of creativity in programming through interviews with programmers can provide further understanding and inform future studies.

To establish more robust findings, future studies should strive to include a diverse range of participants from various backgrounds. Efforts should be made to include underrepresented voices in the discussion, and larger participant pools can provide a more representative research sample. Such research endeavors can further emphasize the importance of rethinking programming education within a social and aesthetic context.

By addressing these areas of future work, researchers can contribute to the ongoing development of inclusive and socially contextualized technical education, fostering a more diverse and engaged community of learners in programming and computer science.

Appendix A

Appendix

A.1 Preliminary Questionnaire

Preliminary Questionnaire

Programming Experience

A.1.1 [Programming Knowledge of Users] What have been your experiences with programming platforms and graphical software?

	APPLE	PEACH	JACKFRUIT	PINEAPPLE	GRAPE	LEMON
I know of online learning platforms for programming.	X					
I have used such platforms privately.	X					
I have completed at least one course on such a platform.	X					
I was taught programming in school.	X	X	X			
I attempted to learn programming through a personal computer.					X	
I have used graphical software privately.		X		X	X	X

A.1.2 [View on Programming as a concept] What is your personal opinion on programming? *List adjectives separated by a comma (,) that would describe programming to you.*

APPLE	logic, mathematical, exact, reproducible, fast
PEACH	confusing, difficult
JACKFRUIT	complicated, confusing, admirable, algorithmic, creative
PINEAPPLE	complex, structured, creative, tricky, clear, modern
GRAPE	weird, cursed, janky, logical, nightmarish, beautiful
LEMON	fascinating, mysterious, inaccessible

A.1.3 [Connection to Code]

A.1.3a Where have you experienced programming in daily life?

APPLE	Once in the holidays I did a basic python course on YouTube, but actually I never used that knowledge since then. In everyday life I do just very modest "programming" like adjust an alarm clock or the microwave.
PEACH	Just in university context and excel (if that counts)
JACKFRUIT	Probably I have experienced it using diverse electronic media and social media platforms and also by using word, office etc.
PINEAPPLE	I have not experienced programming yet; I have always used pre-existing CMS to create/design websites or online-courses.
GRAPE	In attempts of game making, creating small programs (with the complexity of a simple calculator) and necessary for analyzing data.
LEMON	- empty -

A.1.3b Which areas of programming would you like to explore and learn about?

APPLE	I think it would be good to learn something about the general principles of code that is used by companies everyday to collect data from us: things like tracking software, algorithms on social media etc.. For university it could sometimes be helpful to code some simple programs for sorting big amounts of data from experiments etc..
PEACH	What areas exist? More complex excel skills and easy programming.
JACKFRUIT	I would like to refresh my knowledge of the basic stuff, but also learn some artistic programming or creating maps of different landscapes and cities. (I know this is probably too much for two weeks)
PINEAPPLE	Since I have no experience at all, I would like to learn the basics in a way that is not too overwhelming.
GRAPE	User friendly interfaces, game and puzzle logic, software creation (a bit more complex than a simple calculator).
LEMON	- empty -

A.1.4 [Personal opinions] Would you agree/disagree with the following statements?

Strongly disagree				Strongly Agree	No opinion / prefer to not say
General programming resources are difficult to find.					
APPLE					PEACH
GRAPE					JACKFRUIT
					PINEAPPLE
					LEMON
I think dedicated online courses could uphold my interest in programming.					
	APPLE	GRAPE	JACKFRUIT	PEACH	
			PINEAPPLE		
			LEMON		
I have an easy time working with new commercial program interfaces (Photoshop, Microsoft Office, Social Media Platforms, Google Suite).					
GRAPE	LEMON	JACKFRUIT	APPLE	PINEAPPLE	
			PEACH		
I try to avoid using code when working with programs of my daily life. (Photoshop Plugins, One Drive Office, Google Sheets Scripts, Blender Addons, Website Builders, Videogame Mods).					
	APPLE		PEACH	PINEAPPLE	
	GRAPE		JACKFRUIT		
			LEMON		
School education has sparked an interest in programming for me.					
PEACH	JACKFRUIT	GRAPE	APPLE		
PINEAPPLE	LEMON				

I would use program code for private projects rather than my professional career.

APPLE		PEACH	JACKFRUIT	LEMON	
			PINEAPPLE		
			GRAPE		

I like to express myself creatively (traditional/digital artwork, crafts, prose, design, dance, etc.).

		APPLE	PEACH	PINEAPPLE	
			JACKFRUIT	LEMON	
			GRAPE		

A creative process is defined by its result.

JACKFRUIT	APPLE	PINEAPPLE			
GRAPE	PEACH				
	LEMON				

I understand what open-source and transparent code is.

JACKFRUIT		APPLE	PEACH		
			PINEAPPLE		
			GRAPE		
			LEMON		

When using apps and programs, I question how data about me is captured and used.

	APPLE	JACKFRUIT	PEACH		
		PINEAPPLE	GRAPE		
		LEMON			

A.2 Zine Companion: LYRE

Feel free to visit their site (HTTPS://AESTHETIC-PROGRAMMING.NET) to see the full course alongside practical programming tasks using the p5.js framework.

preface(){

Thank you for participating in this study on teaching computer science principles. In the following, you will encounter text fragments about programming and its societal and political implications. This zine is yours. Hence, you may highlight or comment the chapters to your liking. At the end of each chapter, you will find a section for your notes and the chapter's KEY CODE which will look like this:

code

Visit HTTPS://ASSEMBLYRE.GLITCH.ME to enter your user code from the front page to enter your personal course hub. With each code, you will unlock digital resources and an online editor to work on your personal PSEUDO-PROGRAMS. There, you will SUBMIT ONE PROGRAM PER CHAPTER.

Pseudo-Programs: In this course, you will not be coding in a conventional way. You will be provided with a graphical editor which will allow you to write and customize texts and images. Your task is to write programs that would represent your ideas regarding the chapter. You may come up with your own syntax, add related images, or write a plain text to express yourself. In the end, you will be asked to reflect on your work. Pseudo-Programs do not execute code, but convey its concepts and ideas. You will find more information on that later.

If you happen to have issues with your task, feel free to contact me. Happy coding!

}

setup(){

It has become commonplace to include programming in educational programes at all levels and across a range of disciplines. Yet this still remains relatively uncommon in the arts and humanities, where learning to program does not align explicitly with the related career aspirations. This raises questions about what does or doesn't get included in curricula, why this may be the case, and which knowledge and skills are considered essential for some subjects and not others. Certain forms of privilege (related to class, gender, race) are clearly affirmed in these choices. For instance, in very general terms, "high culture" has traditionally been described as the domain of university-educated (wealthy, white) people, whilst "low culture" the domain of non-university-educated (working class) ordinary people. Neither high nor low culture, programming cuts across this class divide as both an exclusive and specialized practice that is also one rooted in the acquisition of skills with applied real- world use in both work and play. Yet, despite its broad applicability, access to the means of production at the level of programming remains an issue all the same.

[Learning] to program allows us to think in new ways by introducing different methods and perspectives to raise new questions; programming offers us a better understanding of culture and media systems, subsequently allowing us to learn to develop better, or better analyze, cultural systems; and, finally, programming can help us improve society by creating, designing, and discovering programs. […] [We] see this as a means to

open up different working methods, and, using programming as a basis for our thoughts, to speculate on alternative forms and political imaginaries of programming practice.

[…] [The] argument for programming or coding as a necessary skill for contemporary life seems indisputable, and there are plenty of examples of initiatives related to computational literacy and thinking, from online tutorials to websites such as Codecademy and Code.org. [...] Vee's Coding Literacy also explores these connections, arguing how the concept of literacy underscores the importance, flexibility, and power of writing for and with computers.[1] An important aspect of this is that, not only does this help us to better understand the social, technical and cultural dynamics of programming, but it also expands our very notion of literacy and its connection to a politics of exclusion (as with other non-standard literacies). Furthermore, and given that programming, like other forms of writing, performs actions, it presents itself as a way to reconceive politics too: not simply writing or speaking, arguing, or protesting in public, but also demonstrating the ability to modify the technical processes through which the action is performed, in recognition of the ways in which power and control are now structured at the level of infrastructure.[2]

}

setup(notes){

}

This is your code for additional digital resources.
Visit your personal hub to start coding!

seed

data(){

Variables are used to store data and information in a computer program. You can think of variables as a kitchen container, in which you can put different types of things (like food, kitchen utensils, etc.) in a given container, replace them with other things, and store them for later retrieval. There are two main types of variables: "local variables" that are defined within a structure or a function, can only be used within that block of code; and "global variables" that can be used anywhere in the code. Global variables need to be defined before the setup of the program, usually in the first few lines of code.

Another reason for using variables is that if you have longer lines of code, it is easier to have all the variables that you have declared for the program in an overview. If a variable is used in different parts of a complex program, you can simply change the value of the global variable instead of changing the multiple parts in the entire program, and this is useful for testing/refining the program without locating specific and multiple lines of code for modification. This leads to the reusability of variables. Variables can be used in different functions and more than once.

[In] the era of big data, there appears to be a need to capture data on everything, even from the most mundane actions like button pressing. Moreover a button is "seductive,"[3] with its immediate feedback and instantaneous gratification. It compels you to press it. Similarly in software and online platforms like Facebook, a button calls for

interaction, inviting the user to click, and interact with it in binary states: like or not-like, accept or cancel. The functionality is simple — on or off — and gives the impression of meaningful interaction despite the very limited choices on an offer (like most interactive systems). Indeed this binary option might be considered to be more "interpassive" than interactive, like accepting the terms of conditions of a social media platform like Facebook without bothering to read the details, or "liking" something as a way of registering your engagement however superficial or fleeting. Permission for capture data is provided, and as such our friendships, thoughts, and experiences all become "datafied." Even our emotional states are monitored when it comes to the use of emoticons […].

[Datafication] — a contraction of data and commodification — refers to the ways in which all aspects of our life seem to be turned into data which is subsequently transferred into information which is then monetized.

At the moment, the most widely used web analytics service is provided by Google and contains tremendous amounts of data on website traffic and browsing behavior, including the number of unique visits, average time spent on sites, browser and operating system information, traffic sources and users' geographic locations, and so on. This data can then be further utilized to analyze customers' profiles and user behaviors.

Smart devices like our computers, phones, and other gadgets are commonly equipped with voice recognition — such as Siri, Google Assistant or Alexa — which turns audio input into commands for software, and feedback with more personalized experiences to assist in the execution of everyday tasks. You can find these voice assistants in just about everything now including, everyday objects like microwaves, and they become more and more conversational and "smart," one might say "intelligent," as machine learning develops. These "voice assistants," as they are known, carry out simple tasks very well, and become smarter, and at the same time capture voices for machine learning applications in general. Placing these tangible voice assistants in our homes allows the capturing of your choices and tastes when not facing a screen. In the internet of things, the device serves you, and you serve the device. Indeed we become "devices" that generate value for others.[4]

Fitness and well-being becomes datafied too, and with the setting of personal targets, also "gamified." As the welfare state is dismantled, personal well-being becomes more and more individualized and there is a growing trend for "self-tracking" apps to provide a spurious sense of autonomy. Movement, steps, heart rate, and even sleep patterns can be tracked and analyzed using wearable devices such as the Fitbit, or the Apple Watch. These practices of the "quantified self," sometimes referred to as "body hacking" or "self-surveillance," overlap with other trends that incorporate capture and acquisition into all aspects of daily life.

}

data(notes){

}

This is your code for additional digital resources.
Visit your personal hub to start coding!

sprout

loops(){

The core concept of a loop is that it enables you to execute a block of code many times. For example, if you have to draw one hundred lines that are placed vertically one after the other, you can of course write one hundred lines of code […].

A "for-loop" allows code to be executed repeatedly, and so provides an efficient way to draw the line one hundred times by setting up a conditional structure, counting the number of lines that have been drawn and counting the maximum number of lines. […]

To structure a for-loop, you need to ask yourself:

- What are the things/actions that you want to repeat in a sequence or pattern?
- More specifically, what is the conditional structure and when do you want to exit the loop?
- What do you want to do when this condition is, or is not, met?

Conditional structures are very useful as they allow you to set a different path by specifying conditions. Indeed, a conditional decision is not specific to programming. For example, in everyday life, you might say

"If I am hungry, I should eat some food, if I am thirsty, I should drink some water, otherwise I will just take a nap."

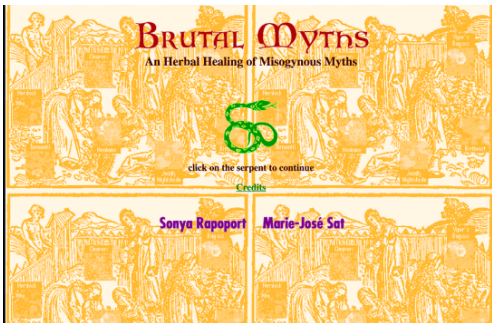
```
if (I am hungry) {  eat some food;} else if (thirsty) {
drink some water;} else{  take a nap;}
```

The above is an example of "pseudocode" to

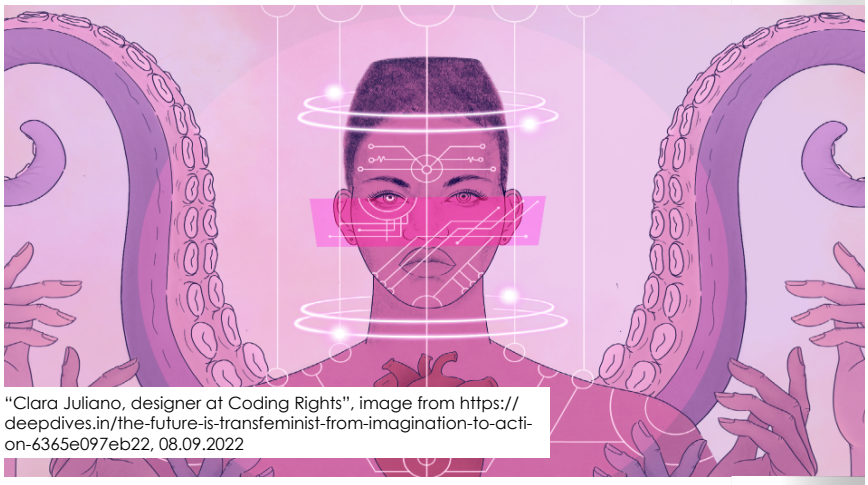
demonstrate what making an everyday decision might look like in programming. The keyword and syntax if is then followed by the condition and checks whether a certain condition holds. As such, the whole if statement is a "Boolean expression" — one of two possible values is possible, true or false, each of which leads to a different path and action. In computer science, the Boolean data type has two possible values intended to represent the two truth values of logic.

Loops offer alternative imaginaries, as is the case of the ancient image of a serpent eating its own tail. Ouroboros, from the Greek, expresses the endless cycle of birth and death, and therefore stands for the ability of processes to infinitely renew themselves. Alongside evocative references to autocannibalism and alchemy, loops are related to control and automation tasks, as well as repetitive procedures in everyday situations such as those heard in repeating sections of sound material in music. [5] In programming, a loop allows the repeated execution of a fragment of source code that continues until a given condition is met, such as true or false. Indeed a loop becomes an infinite (or endless) if a condition never becomes false.
The loop can be thought of as a repeating "if" statement and offers a good way of challenging conventional structures of linear time, and demonstrating how computers utilize time differently. Programming challenges many of our preconceptions about time including how it is organized, how the present is rendered using various time-specific parameters and conditions, as in the case of a

throbber. We hope it is already clear that machine-time operates at a different register from human-time, further complicated by global network infrastructures, and notions of real-time computation.
}



Screenshot, 2020, Firefox v76.0.1 on Mac OS 10.13.3; http://users.lmi.net/sonyarap/brutal/index.html featured in http://www.sonya-rapoport.org/portfolio/brutal-myths-1996/, 08.09.2022



"Clara Juliano, designer at Coding Rights", image from https://deepdives.in/the-future-is-transfeminist-from-imagination-to-action-6365e097eb22, 08.09.2022

loops(notes){

}

This is your code for additional digital resources.
Visit your personal hub to start coding!

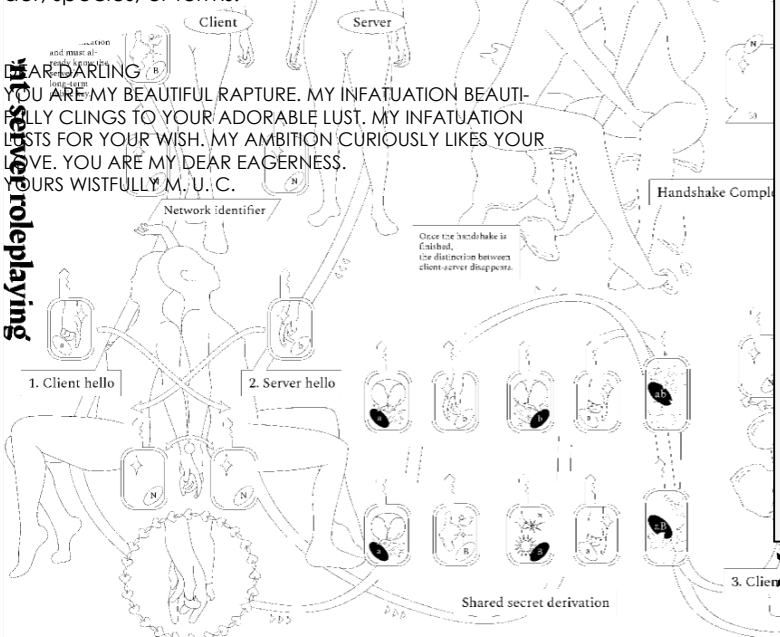
thorn

generation(){

[The] discussion of more love and care in programming brings us to [...] the generative "love-letters" that appeared on the Manchester University Computer Department's noticeboard in 1953. These computer-generated declarations of love were produced by a program written by Christopher Strachey using the built-in random generator function of the M. U. C.(Manchester University Computer, the Ferranti Mark I), the earliest programmable computer. Regarded by some as the first example of digital art,[6] and by Jacob Gaboury as a critique of hetero-normative love, not least because Strachey like Turing was queer.[7] Moreover these letters are arguably more than a longing for same sex love, but human-machine love.

Artist David Link built a functional replica of both the hardware and the original program, following meticulous research into the functional aspects.[8] The main program is relatively simple, and uses loops and a random variable to follow the sentence structure: "You are my — Adjective — Substantive," and "My — [Adjective] — Substantive — [Adverb] — Verb — Your — [Adjective] — Substantive." Some words are fixed and some are optional, as indicated by the square brackets. The program selects from a list of options — adjectives, adverbs, and verbs — and loops are configured to avoid repetition. The software can generate over 318 billion variations. In terms of effect, the dialogue structure is important in setting up an exchange between "Me" (the program writer) and "You" (human reader), so you feel personally addressed. The resulting love letters provide a surpri-
}

sing tenderness of expression that runs contrary to what we consider the standard functional outcomes of computational procedures. This is far from a reductionist view of love, and perhaps the challenge for those making programs is to generate queer recombinant forms in which neither sender or receiver are predetermined by specifying gender, species, or forms.



DEAR DARLING
YOU ARE MY BEAUTIFUL RAPTURE. MY INFATUATION BEAUTIFULLY CLINGS TO YOUR ADORABLE LUST. MY INFATUATION LASTS FOR YOUR WISH. MY AMBITION CURIOUSLY LIKES YOUR LOVE. YOU ARE MY DEAR EAGERNESS.
YOURS WISTFULLY M. U. C.

"Handshake Erotica" 2019, Nahee Kim, https://nahee.app/handshake_erotica.html, 08.09.2022

generation(notes){

}

This is your code for additional digital resources.
Visit your personal hub to start coding!

leaf

abstraction(){

In programming an object is a key concept, but it is also more generally understood as a thing with properties that can be identified in relation to the term subject. Put simply, and following philosophical conventions, a subject is an observer (we might say programmer) and an object is a thing outside of this, something observed (say a program). In this chapter we will learn to further manipulate objects and understand their complexity in line with people who think we need to put more emphasis on non-human things so we can better understand how objects exist and interact, both with other objects, but also with subjects.

Object abstraction in computing is about representation. Certain attributes and relations are abstracted from the real world, whilst simultaneously leaving details and contexts out. Let's imagine a person as an object (rather than a subject) and consider which properties and behaviors that person might have. We use the name "class" to give an overview of the object's properties and behaviors.

Properties: A person with the name Winnie, has black hair, wears glasses and their height is 164 cm. Their favorite color is black and their favorite food is tofu.

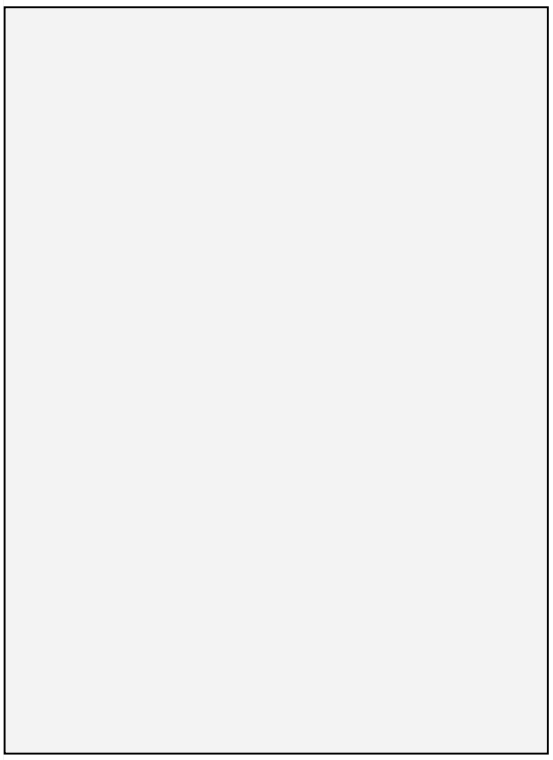
Behavior: A person can run from location A (home) to location B (university). (Pseudoklasse)

[…] object-oriented programming is highly organized and concrete even though objects are abstractions. It's also worth reiterating that OOP is designed to reflect the way the world is organized and imagined, at least from the computer programmers' perspective. It provides an understanding of the ways in which relatively independent objects operate through their relation to other objects.

[...] [Object-oriented] programming is highly organized and concrete even though objects are abstractions. It's also worth reiterating that OOP is designed to reflect the way the world is organized and imagined, at least from the computer programmers' perspective. It provides an understanding of the ways in which relatively independent objects operate through their relation to other objects.
}

Floppy disks of Cyber Rag, image from https://nymag.com/intelligencer/2018/04/claire-evanss-broad-band-excerpt.html, 08.09.2022

abstraction(notes){



}

This is your code for additional digital resources.
Visit your personal hub to start coding!

stem

queery(){

To query something is to ask a question about it, to check its validity, or accuracy. When querying a database, despite the apparent simple request for data that enables selectivity with regard to which and how much data is returned, we should clearly question this operation too. We need to query the query.

Search engines (like Google and Baidu) are a good example of applications that aggregate content and algorithmically return search results according to a keywords search. They promise to answer all our questions, but do not make the underlying processes (and ideology) visible that prioritize certain answers over others. In a query-driven society, search engines have become powerful mechanisms for truth-making and for our making sense of seemingly endless quantities of data, manifested as streams, and feeds — indicative of the oversaturation of information and the rise of the attention economy. [...] The habit of searching, for instance, is transformed into data that is storable, traceable, and analyzable.

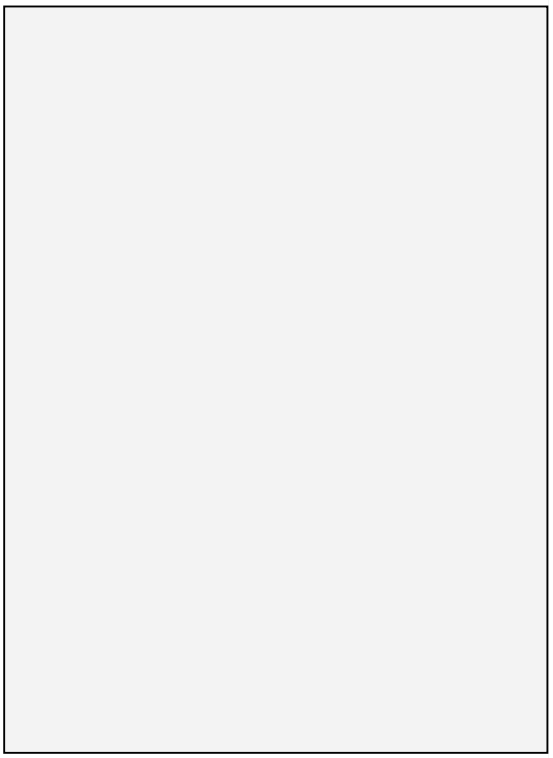
We have already explored some of the processes programs use to capture input data in [Chapter 'Data',] especially data that is connected to physical devices, and in this chapter we expand this exponentially to data hosted on online platforms. We scale up from the capture of data to the storage, and analysis of massive amounts of captured data — so-called "Big data" (or even "Big Dick Data" if we consider this to be a masculinist fantasy)[9] — which is in turn utilized for user-profiling,

targeted marketing, personalized recommendations, and various sorts of predictions and e-commerce, and so on. Subsequently it would seem that: "*We're not in control of our search practices — search engines are in control of us and we readily agree, though mostly unconsciously, to this domination.*" [10]. But arguably it's not quite as deterministic as this, as these operations are part of larger socio-technical assemblages and infrastructures — including data, data structures, and human subjects — that are also constantly evolving and subject to external conditions.

"*Search happens in a highly commercial environment, and a variety of processes shape what can be found; these results are then normalized as believable and often presented as factual [and] become such a normative part of our experience with digital technology and computers that they socialize us into believing that these artifacts must therefore also provide access to credible, accurate information that is depoliticized and neutral.*" [11]

}

queery(notes){



}

This is your code for additional digital resources.
Visit your personal hub to start coding!

root

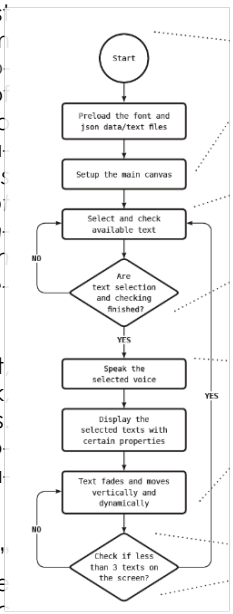
algorithm(){

"1. Begin reading this procedure, unless you have already begun to read it. Continue to follow the steps faithfully; […] 5. Is the subject of the chapter interesting you? If so, go to step 7; if not, go to step 6. 14. Are you tired? If not, go back to step 7; 15. Go to sleep. Then, wake up, and go back to step 7." [12]

The example serves to emphasize that we tend to follow instructions faithfully. However, we might also observe that algorithms are more than simply steps and procedural operations as there are wider cultural and political implications, not least in terms of whether we decide to interpret them on our own terms. In this sense, like cooking, algorithms express cultural differences, and matters of taste, even aesthetics. Extending the analogy to other cultural practices, Knuth quotes Ada Lovelace: *"The process of preparing computer programs for a digital computer is especially attractive, not only because it can be economically and scientifically rewarding, but also because it can be an aesthetic experience much like composing poetry or music."* [13]

Algorithms […] are there to transform, construct and shape data, in order to then classify, rank, cluster, recommend, label, or even predict things. The concern is not how to build an efficient or optimized algorithm, but to understand these operative dimensions better.

In this chapter we will build on "diagramming," particularly the use of flowcharts to elaborate the practical and conceptual aspects of algorithmic procedures. Flowcharts, […] have been considered a fundamental explanatory tool since the early days of computer programming. One of their common uses is to illustrate computational operations and data processing for programming by "converting the numerical method into a series of steps." [14]. But flowcharts can also be considered to be representational diagrams which can also be used to communicate complex logic between programmers and others involved in software production. This is good practice of course, especially for beginners in a learning context, and is essential for communicating ideas in ways that can be easily understood by others. […] Moreover most software applications are not developed by a single programmer but are organized into tasks that are tackled collaboratively by programmers, as for instance when maintaining or debugging a program made by someone else. Collaborative workflows lends themselves to flowcharts.



Conventionally, each step in a flowchart is represented by a symbol and connecting lines that indicate the flow of logic towards a certain output. The symbols all have different meanings [...]:

OVAL: Indicates the start or end point of a program/system. (But this requires further refection on whether all programs have an end.)
RECTANGLE: Represents the steps in the process.
DIAMOND: Indicates the decision points with "yes" and "no" branches.
ARROW: Acts as a connector to show relationships and sequences, but sometimes an arrow may point back to a previous process, especially when repetition and loops are concerned.

}

algorithm(notes){

}

This is your code for additional digital resources.
Visit your personal hub to start coding!

fruit

learning(){

Machine learning is a term coined by Arthur Samuel in 1959 through his research at IBM on game development, with the ultimate goal to reduce or even eliminate the need for "detailed programming effort."[15]. The roots of how computers might begin to write their own programs lie in older discussions of artifical intelligence.

In machine learning, it is commonly understood that the style is learnt from training datasets through techniques to process and analyze large amounts of (natural language) data. As such, machine learning techniques such as "style transfer" rely on a process of generalization in order to identify patterns. However, this "pattern recognition" is clearly not a neutral process as it involves the identification of input data, and the "discrimination" of information.[16] It is clear that there is other kinds of discrimination in such processes […]. Understood this way, pattern recognition is not only about smoothing tasks and making accurate predictions in terms of technical operations but also political operations as it creates "subjects and subjection, knowledge, authority" as well as classification and categorization.

It should be pointed out that although machine learning is part of AI, AI is a broader concept. AI, machine learning and deep learning are terms that are often used interchangeably but there are key distinctions to be made. To explain: "You can think of deep learning, machine learning and artificial intelligence as a set of Russian dolls nested within each other, beginning with the smallest and

working out. Deep learning is a subset of machine learning, and machine learning is a subset of AI, which is an umbrella term for any computer program that does something smart. In other words, all machine learning is AI, but not all AI is machine learning, and so forth."[17]

"If you consider a child's eyes as a pair of biological cameras, they take one picture about every two hundred milliseconds, the average time an eye movement is made. So by age three, a child would have hundreds of millions of pictures of the real world. That's a lot of training examples. So instead of focusing on solely better and better algorithms, my insight was to give the algorithms the kind of training data that a child was given by experiences, in both quantity and quality."[18]

If visual literacy is no longer simply an educational task for humans, but also for machines, then it becomes a question of human-machine literacy in its broadest sense. […] That machines can be said to "see" or "learn" is shorthand for calculative practices that only approximate likely outcomes by using various algorithms and models.

"But remember that I am controlling and using for my own purposes the means of reproduction needed for these programmes […] with this programme as with all programmes, you receive images and meanings which are arranged. I hope you will consider what I arrange but please remain skeptical of it."

What is learnt should not be separated from the means by which it is transmitted, nor the direction of travel from human to machine or from machine to human. More to the point, the production of meaning lies at the core of our discussion, as are concerns about what is being learnt, and to what extent this has been compromised or inflected by reductive ideas of how the world operates.

}

Everest Pipkin: Plotter Drawings 2017, https://everest-pipkin.com/#drawings/plotter.html, 08.09.2022

glossary(){

The following definitions are formulated under my own biases. you are free to re-consider their meaning.

PROGRAMMING - Formulation of concepts through syntax that is meant to communicate functionality to developers and computers. The respective programming languages work as a middle ground for thought and writing.

LITERACY - Here, Digital Literacy. The ability to estimate the potential and influence of (new) digital media and its evolution through lived experience and exploration.

VARIABLES - Control points within programs yielding values. Values can be assigned, read, and detached for the program to function.

FUNCTIONS - A named collections of operations within a program. Given an input in form of variables or values, a function produces an output that can be attached to further variables.

BIG DATA - The practice of collecting, processing, and storing data within the shortest amount of time possible.

loops - Repititions of operations during program runtime until a certain condition is fulfilled.

SYNTAX - Grammar for creating a program. In the nature of programming principles, complex computer operations are put into words that can be understood by humans.

OBJECTS - Complex building blocks in programs described by a respective class. Defined by their own functionality and data structure, they can act as simplifications of the real world.

QUERY - Search and organization of data from a larger data set. Information from large data sets can be deduced by searching data of certain properties.

ALGORITHMS - Conceptual ideas involving the input, process, and output of data. Their realization comes mostly through the use of programming.

MACHINE LEARNING - Iterative calculations to build an automated system for answering questions. By estimating control parameters, the machine is supposed to make precise decisions when confronted with new kinds of data.

}

1 Vee, Coding Literacy.

2 This point is largely derived from Kelty's Two Bits, which uses the phrase "running code" to describe the relationship between "argument-by-technology and argument-by-talk." See Christopher Kelty Two Bits: the Cultural Significance of Free Software (Durham: Duke University Press, 2008), 58. Clearly programmers are able to make arguments as people can in other rhetorical forms, see Kevin Brock, Rhetorical Code Studies: Discovering Arguments in and around Code (Ann Arbor, MN: University of Michigan Press, 2019).

3 Paraphrasing the final lines of Leslie's essay "The Other Atmosphere: Against Human Resources, Emoji, and Devices": "The workers become their own devices. They becomes devices of communicative capitalism […]."

4 Søren Pold, "Button," in Matthew Fuller ed., Software Studies (Cambridge, Mass.: MIT Press, 2008), 34. Users are seduced by the wording of the button not least, and Pold suggests that a button is developed with distinct functionality and signification (Ibid., 31).

5 The logic behind loops can be demonstrated by the following paradoxical word play: "The next sentence is true. The previous is false." Further examples of paradox, recursion, and strange loops can be found in Douglas R. Hofstadter's' Gödel, Escher, Bach: An Eternal Golden Braid (New York: Basic Books, 1999)

6 Jacob Gaboury, "A Queer History of Computing," Rhizome (April 9, 2013). We return to the issue of Turing's sexuality in Chapter 7, "Vocable Code".

7 Noah Wardrip-Fruin, "Christopher Strachey: The First Digital Artist?," Grand Text Auto, School of Engineering, University of California Santa Cruz (August 1, 2005).

8 David Link's LoveLetters_1.0: MUC=Resurrection was first exhibited in 2009, and was part of dOCUMENTA(13), Kassel, in 2012. Detailed description and documentation can be found at http://www.alpha60.de/art/love_letters/. Also see Geoff Cox, "Introduction" to David Link, Das Herz der Maschine, dOCUMENTA (13): 100 Notes - 100 Thoughts, 100 Notizen - 100 Gedanken # 037 (Berlin: Hatje Cantz, 2012).

9 Big data is referred to as "Big Dick Data" by Catherine D'Ignazio and Lauren Klein, to mock big data projects that are characterized by "masculinist, totalizing fantasies of world domination as enacted through data capture and analysis," see "The Numbers Don't Speak for Themselves," in Data Feminism (Cambridge, MA, MIT Press 2020), 151.

10 Big data is referred to as "Big Dick Data" by Catherine D'Ignazio and Lauren Klein, to mock big data projects that are characterized by "masculinist, totalizing fantasies of world domination as enacted through data capture and analysis," see "The Numbers Don't Speak for Themselves," in Data Feminism (Cambridge, MA, MIT Press 2020), 151.

11 Safiya Umoja Noble, Algorithms of Oppression: How Search Engines Reinforce Racism (New York: New York University Press, 2018), 24-25.

12 Knuth, The Art of Computer Programming, xv-xvi.

13 Knuth, The Art of Computer Programming, v.

14 Ferranti Limited, Ferranti Pegasus Computer, programming manual, Issue 1, List CS 50,September 1955

15 Machine learning is a term coined by Arthur Samuel in 1959 during his game development research at IBM which ultimately aimed to reduce or even eliminate the need for "detailed programming effort," using learning through generalization in order to achieve pattern recognition. See Arthur L. Samuel, "Some Studies in Machine learning Using the Game of Checkers," IBM Journal of research and development 3, no.3 (1959): 210-229

16 Clemens Apprich, "Introduction," in Clemens Apprich, Florian Cramer, Wendy Hui Kyon Chun, and Hito Steyerl, eds., Pattern Discrimination (Minnesota: Meson Press, 2018), x

17 See Pathmind's "A.I. Wiki: A Beginner's Guide to Important Topics in AI, Machine learning, and Deep Learning," https://pathmind.com/wiki/ai-vs-machine-learning-vs-deep-learning

18 The Fei Fei Li quote is taken from Nicolas Malevé's article, "'The cat sits on the bed': Pedagogies of vision in human and machine learning," Unthinking Photography (2016), https://unthinking.photography/articles/the-cat-sits-on-the-bed-pedagogies-of-vision-in-human-and-machine-learning.

LYRE

A.3 Course Material Overview

A.3.1 setup() Why to Code

The first step of this course dealt with the societal need and potential of coding. In the following, you will look at existing art, first languages, and art installations.

Open Questions

Look at the creative works others have produced. What might have they used to achieve it? Try to decompose their process and how you would tackle it. What resources would you need? Where would you apply them, and what would be your result?

Who is someone who understands code? What makes people interact with code? Where are technology and code intertwined? Picture possible interactions between you and a machine? Are there possible dangers with people understanding code that you do not?

Is coding a process of writing text, or is it something else? What does your piece mean to you? Would you like to turn your pseudocode into something real? Do you believe society has the capabilities to produce the code you wrote?

The Hello World Collection `http://helloworldcollection.de/`
From a small collection of text files by the German network MausNet, a large corpus of Hello World programs emerged. In both program, and human languages, the source code is compiled in an exhaustive overview.

Markdown Guide `https://www.markdownguide.org/basic-syntax`
Markdown is a language to format plain text. Just like any office application, you can use it to create headings, lists, italics, and much more. Assemblyre uses Markdown for its text field. Check out this reference guide if you would like to make use of it in your own program.

The P5.js Editor `https://editor.p5js.org/`
P5.js is a JavaScript framework that is used by many web artists to create generative artwork. The live-preview gives you immediate feedback on your code to overcome any setup limitations. Try drawing a circle and transform it however you like. Find Help using their Reference sheet.

sasj.nl `https://sasj.nl/portfolio/daily/`
Saaskia Freeke is a designer and coder from Amsterdam specializing in exploring structure, geometry, and playfulness with new media. Daily, she presents generative patterns and animations coded in the programming environment Processing.

Annette Vee: Coding Literacy `http://mitpress.mit.edu/9780262340243/coding-literacy/`
If you got curious about what coding literacy means in a broader context, Annette Vee elaborates on the importance of understanding programming. She explores the boundaries between text and code, painting a picture of a society working under computational standards that are unknown to a majority of the population.

A.3.2 data() Real Data

Before we delve further into programming paradigms, we will look at core issues lying within the concept of data. This chapter is a longer one. The following resources are meant to further communicate its ultimate capabilities, and how they currently have been implemented.

Open Questions

Data forms an opportunity to convey data not only to computers, but to us. What role does data play to a machine compared to a human being? Is data an objective representation of information? What kind of data would you like to gather? How would you try to do that? What would you do with the data and in which ways would you share it, if so?

Imagine a button that would jumpstart a variety of tasks invisible to you. Do these buttons exist and where? How would you make such a button appealing to press?

Have you ever "datafied" yourself? What were your experiences back then compared to this very moment?

VPRO Documentary: Shoshana Zuboff on surveillance capitalism `https://www.youtube.com/watch?v=hIXhnWUmMvw&feature=emb_title`
In this documentary featuring Harvard professor Shoshana Zuboff, the fundamental and grave dangers of capturing unending amounts of data are summarized. A social media network wouldn't need all the data they could get from you - so where exactly does it go?

Vienna Biennale: Uncanny Values `https://process.studio/works/uncanny-values/`
The Austrian design studio Process implemented in their generative art installation on artificial intelligence and uncanny emote. How would one generate such emotes? What makes them intriguing? What does identity mean in the face of artificial intelligence?

Transmediale Artwork Archive `https://archive.transmediale.de/archive/explore?f%5B0%5D=type:artwork`

Since 1988, the transmediale archive features hundreds of artists with their multimedia installations. Browse through their catalog on the many voices surrounding the topic of technology and culture. Transmediale is a festival that implores a critical view on how digital media reflects on every aspect of our life. Do check out their publications as well, if you would like to delve deeper!

Facebook agrees to pay fine over Cambridge Analytica scandal `https://www.theguardian.com/technology/2019/oct/30/facebook-agrees-to-pay-fine-over-cambridge-analytica-scandal`
A comprehensive report on the 2018 discussion on Cambridge Analytica. How do we regulate data protection, and has it really changed since then?

MLTalks: How Data Killed Facts `https://www.media.mit.edu/videos/ml-talks-2018-04-23/`
Jill Lepore, a Professor of American History at Harvard University, argues the ever-changing definition of facts. If you are curious about how data is commonly referenced as a medium of facts, this discussion opens up new ways of thinking about what we ultimately perceive as truth.

A.3.3 loops() Time, but different

Before we delve further into programming paradigms, we will look at core issues lying within the concept of data. This chapter is a longer one. The following resources are meant to further communicate its ultimate capabilities, and how they currently have been implemented.

Open Questions

When do you encounter loops in daily life? Think of them in the digital and analogue context! How do loops represent (or even steal) time? What the conditions of your loop? What are the variables?

Have you encountered infinite loops in daily life? What is needed TO loop? Can we think of lifecycles as loops? Where are the boundaries?

DVD guy `https://www.youtube.com/watch?list=PLCUGKK4FUkbMdnNii8qoRy9_tMvqE8XHB&time_co4&v=4RfakB58kb8&feature=emb_logo`
This video series by constantdullaart reflects on what a loop in real life may look like. How does one feel when watching it go? How would we implement it?

Sort the Court `https://graebor.itch.io/sort-the-court`
Build your own kingdom based on your responses to the townsfolk. See it thrive or burn by your actions of responding binarily. See the narrative unfold in this simple loop of actions, and think about where you might have gone wrong. Are binary

decisions fitting for positions in power? When is an enforced binary helpful, and when is it a burden to others?

A.3.4 generation() Building (Random) Blocks

Use the additional resources as inspiration. Would you consider generative art to be the only way to see code as an art form? What would you like to generate? What are the components that you are generating? Try to go through the iterations manually? How long would the generation take? Where would randomness come into place, and which parts of the real world could be used to simulate said randomness?

When do you encounter loops in daily life? Think of them in the digital and analogue context! How do loops represent (or even steal) time? What the conditions of your loop? What are the variables?

Have you encountered infinite loops in daily life? What is needed TO loop? Can we think of lifecycles as loops? Where are the boundaries?

Generative Artistry `https://generativeartistry.com/tutorials/`
Generative Artistry is a project by Ruth John and Tim Holman and features a variety of fun tutorials on getting into generative artwork. The guides are visualized along your reading progress and detail the thought process of expressing yourself through generation. The guides don't need a framework, but are simple algorithms on the HTML canvas element. Feel free to make your own .html and .js file and code ahead!

Generative Tarot [content note: flashes, moving images, rapid image changes] `https://www.melaniehoff.com/generativetarot/`
A fun and expressive way to get into tarot. This P5.js project features multiple artists that would design their own tarot cards to be visualized randomly at every pull. Which card did you pick? Would you realize your own cards differently?

The Coding Train: 2D Arrays in JavaScript `https://www.youtube.com/watch?v=OTNpiLUSiB4`
The Coding Train is a delightful series of coding problems and uses the P5.js framework for their implementation as well. Feel free to take a peek into this lesson on 2D arrays to understand a fundamental way of structuring your data, and maybe code along!

David Link: Machine Heart `http://www.alpha60.de/art/love_letter/DavidLink_DasHerzDerM`
An essay on how we could define computers as sentient in the context of showing affection. An artificial intelligence producing love letters opens the discussion on love and intimacy in a mathematical context.

A.3.5 abstraction() Matter (over/under) Class

Abstraction constantly takes place in human perception. How do we simplify, why do we care about it, and how can we make these processes consciously? Take a look at some small video games that would make use of that exact process to bring light to new points of view.

Open Questions

Picture an object. What properties does it have, and how can we interact with it? Try to model yourself in an object. Start with a very basic representation of yourself and put your Me-object into simple scenarios! What would the scenario need from you? How would you want to improve your object for new scenarios? Could a computer help you improve your Me-object, or would it make things difficult?

Think of a game that you played sometime! What objects might have been used? Try to imagine a game yourself and how you could implement it! Which parts of the real world would need to be abstracted?

Murder at the Residence Gudul `https://sparklinlabs.itch.io/gudul`
The Residence Gudul is no stranger to the exciting benefits of The Distribute, a nifty television with a one-button device to send you everything you see on the monitor. Help Dowie fix his favorite toy while uncovering an unusual murder in this short, dystopian role play game on consumerism.

Cats are Assholes but we still like them `https://deepnight.itch.io/cats-are-assholes`
Watch your precious stray cats under time pressure, management issues and the ever-growing fear of getting old. A fast-paced game written in mere days. Try finding out more about the Game Jam that this was a part of. What are your thoughts on Game Jams? How do you think people deal with time constraints in software development, and especially in the creative field of indie gaming?

Different Strokes `https://swsteffes.itch.io/different-strokes`
Help this community of artist to improve the submissions of others. Explore the gallery, talk to the personnel, draw something! What would you do if you witnessed your artwork to be drawn over both consensually and consensually? Think about the experiences you had in art communities, or how you imagine them to be. Would you consider collaborative code to be of similar nature?

A.3.6 queery() Computers 2 Action

Data comes in raw form, and it is up to us to govern it accordingly. In this chapter, we try to think about the many steps a computer could take to process a request. What

are we looking for, and what actually do we get? With large databases, developers create contact points for you to see, query and process. These so-called APIs are possibilities for you to explore complex data that others have gathered in the scope of the internet. Which large databases would you like to explore? Whose API would you like to query, and what would you use it for?

Open Questions

Think more about the API that you would like to address! What are its capabilities? What does it tell about the instance providing you with that information? How and why would you process your desired data from that API?

Do you have any concerns regarding the public APIs? How could power dynamics be exploited with such a concept? Who would you share your API with? What if there was an API for you as a human being - or does it already exist?

NAG: Net.Art Generator `https://nag.iap.de/`
As an art installation, the net art generator tackles the possibilities we have when we scour the internet for information. Taking various images that are openly accessible from anywhere, search engines can provide us with data that users barely have a hold-over. Create some net art, and keep in mind the implications of a mere Google search.

The Graph API: Key Points in the Facebook and Cambridge Analytica Debacle `https://medium.com/tow-center/the-graph-api-key-points-in-the-facebook-and-cambridge-analytica-debacle-b69fe692d747`
An exhaustive explanation of the technology prevalent in the scandal regarding the Cambridge Analytica controversy. How much of the data people may track gets actually useful through a detailed query?

Echo Beach `https://tim-sheinman.itch.io/echo-beach`
Tim Sheinman's Echo Beach is a great example of a merge of many art forms. Be the person who would track down notorious musicians in a world where music is against the law. Be a part of a small forum to deduce its seemingly anonymous information to take the artists to jail. Listen to the soundtracks, query the database, and see how you and your character may feel about that. A short and simplistic detective game on vulnerable anonymity, music as a medium of information, and balance between work and art.

A.3.7 algorithm() Building (Complex) Blocks

The core of programming comes down to the algorithm - the combination of all the elements you have learned about. We will take a look at some algorithms and how they manage to convey their message through some short lines of text, their context

of application, and the result they are striving for.

In your course, you have been playing around with the editor to create diagrams of programs from the very beginning. Use this exercise to maybe create a piece that hasn't really fit into any of the other courses. Try to imagine that someone else is reading your code: How can you make it understandable, clear and as transparent as possible? How would you convey the "flow of information" with the graphical tools provided?

You may even just create a flowchart as explained above. Would you stick to the geometric forms presented?

Look at your previous work! What has been the most complex work until now? How could you make it more concise? How could you make it even more complex?

Do you think that simplifying, making your code as clear as possible, is something one should always keep in mind? What is the main problem with communicating your code in the most simple ways while remaining a complex code base? What are the technical challenges?

The Secret Rules of Modern Living: Algorithms `https://www.bbc.co.uk/programmes/p030s6b3/clips`
A short summary of what an algorithm is with a small-scale comparison of two staple algorithms, and a large-scale analysis of the societal context that we perceive.

Christian Sandving: Seeing the Sort: The Aesthetic and Industrial Defense of "The Algorithm" `http://median.newmediacaucus.org/art-infrastructures-information/seeing-the-sort-the-aesthetic-and-industrial-defense-of-the-algorithm/`
An analysis on the cultural and mathematical definition of an algorithm. Christian Sandving tries to overcome the boundaries that would obfuscate the nature of an algorithm, and explains a complex structure of knowledge that is not tainted by a corporate identity.

A.3.8 learning() Cheating the Test

The core of programming comes down to the algorithm - the combination of all the elements you have learned about. We will take a look at some algorithms and how they manage to convey their message through some short lines of text, their context of application, and the result they are striving for.

Open Questions

Where do you see the duality of children learning, and machines learning? Where are its boundaries? From your understanding of designing a learning task, conceptualize a model that could solve some issue. What are the benefits of machine learning to you personally, and where would you draw the line? If machines are fit to learn, how would you implement them in a societal context?

What is the common perception of machine learning in your daily life? How do you think different kinds of people perceive the possibilities of learning models? What could be done to further communicate their potential and its benefits and dangers?

Generate some images or other kinds of media yourself! What do you see? What do you think went into its generation? Would you consider the result to be art?

Ruha Benjamin: Are Robots Racist? `https://www.dropbox.com/s/j80s8kjm63erf70/Ruha%20Benjamin%20Guest%20Lecture.mp4`
With institutions and corporations strongly relying on automation, the dangers are imminent, when programs are written by and tested on largely white men. In the context of race and gender biases, Ruha argues the dangers of current employment practices and how capital employs software that undermines intersectional identities.

Discriminating Systems: The Power in AI `https://ainowinstitute.org/discriminatingsystems.pdf`
In their paper, Sarah Myers et al. put together a comprehensive view on the matter of Gender, Race and Power in AI. Their findings paint a picture of a severe diversity crisis in AI, a research topic that has been largely discussed mathematically, yet has been barely touched in a societal and political context.

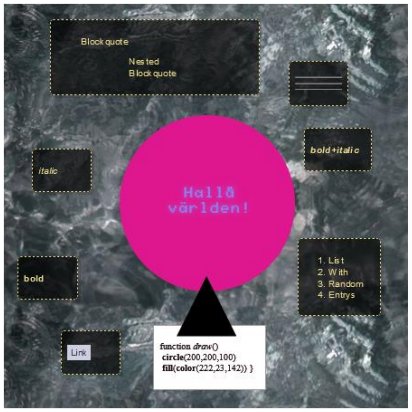
Racial and Gender bias in Amazon Recognition — Commercial AI System for Analyzing Faces `https://medium.com/@Joy.Buolamwini/response-racial-and-gender-bias-in-amazon-rekognition-commercial-ai-system-for-analyzing-faces-a289222eeced`
A showcase of an Amazon controversy surrounding facial recognition in commercial, closed-source software. Despite its dominance in industrial practice, its functionality not only discriminates against minorities, but gives room to weaponizations and exploitation.

To Be or not to Be Hacked? The Future of Democracy, Work, and Identity `https://www.youtube.com/watch?v=tRVEY95cI0o`
From the issues brought up in the resources before, this discussion on technological influence in different cultures sparks arguments on its future, and the historical development of digital norms.

A.4 Pseudoprogram Submissions

A.4.1 Participant APPLE

Submissions **APPLE** - 1
1 – 2 – 3 – 4 - 5 - 8



1: setup() ~ Why to Code

What inspired your piece?

The whole piece was inspired by the given resources. The titel is the Swedish translation of 'hallo world' found in the 'Hello World Collection'. The square shape of the canvas was inspired by the work of Saskia Freeke. The magenta circle in the middle is a dupilcate of the one programmed first by me in the P5.js Editor. The main part of the used code is shown underneath. The text fields around are showing the main formattings of Markdown that i learned from the Markdown guide. One of this field leads to a video from Saskia Freeke that I watched.

How does the topic influence yourself?

The works of Saskia Freeke were quite interesting for me. Maybe if I'm going to be bored some day I could try to learn a bit more about the program 'Processing' that she used.

Would the program or the topic you are tackling influence anyone else?

I guess my piece does not really, but some of the used resources could be helpful in general. Most of all the Markdown guide.

View in Editor

2: data() ~ Real Data

What inspired your piece?

Das Muster des Hintergrunds erinnert an einen Rohrschach-Test. Dieser steht sinnbildlich für das Spannungsfeld zwischen "objektiver" Realität und subjektiver Interpretation. Die Erhebung von Daten kann hierbei als Bindeglied fungieren. (Darstellung rechts oben.) Darunter sind willkürliche, nüchterne, kalt-analytisch wirkende Daten aufgelistet, wie sie gesammelt werden könnten, um eine Person digital zu durchleuchten. Als Symbolbild hierfür ist links eine alte Aufnahme der "Gläsernen Frau" aus dem Dresdener Hygienemuseum dargestellt. Vollig entblößt steht sie vor uns: Nicht nur äußerlich durch das Fehlen von Kleidung, sondern auch ihr Innerest bleibt dem Betrachter nicht verborgen. Dies wird kühl durch das darunterbefindliche Pop-Up quittiert. Die widerstandslose Bestätigung dieser Tatsache erscheint durch den zugehörigen Button alternativlos.

How does the topic influence yourself?

Als Nutzer internetbasierter Dienste weiß ich als Laie oft nicht wie ich mit der wohlmöglich massiven Datenerhebung über mich umgehen soll. Ich habe keinen Überblick über die von mir erfassten Daten und da man fast schon systemisch an gewisse Anbieter gebunden ist, erscheint das Akzeptieren der AGBs (die man wohlgemerkt nie liest) unabdingbar.

Would the program or the topic you are tackling influence anyone else?

Vielleicht freut sich ja jemand über die versteckte Hörbuchempfehlung.

View in Editor

Submissions **APPLE** - 2
1 – 2 – 3 – 4 - 5 - 8

3: loops() ~ Time, but different

What inspired your piece?

Diesmal habe ich eher etwas leichte Kost produziert und ein bisschen mit den Animationen experimentiert. Die Krone verweist auf das Spiel "sort the court" aus den Materialien. In der Mitte steht etwas Pseudocode, der das Thema "loops" aufgreift und zudem eine bewährte Taktik präsentiert, um in eben besagtem Spiel aus der Geldnot zu kommen. Die drei Diamanten auf den Zacken führen a) zu einer Animation des Ouroboros, der im Text besprochen wurde, b) zu einem Spongbob-Video, dass durch einen Loop endlos fortgesetz werden kann und c) zum klassischen DVD-Screensaver, der die Grundlage für die Videos von "DVD-Guy" darstellt.

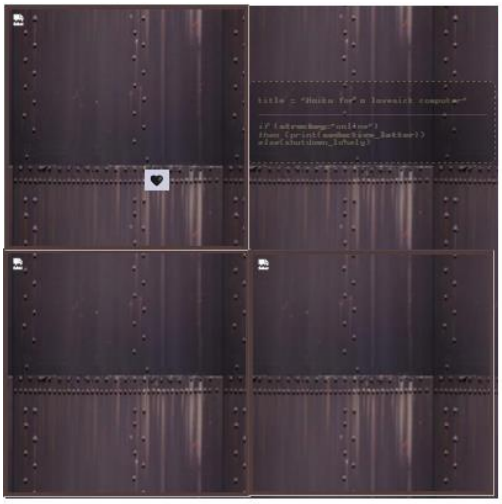
How does the topic influence yourself?

Ich würde sagen, loops bzw. Schleifen sind für Berechnungen im naturwissenschaftlichen Bereich das beste Argument zur Programm-gestützten Auswertung eines Problems. Sie ermöglichen viele kleine, stupide Berechnungen oder Veränderungen in kurzer Zeit durchzuführen. Oft helfen mir gewisse, tägliche Routinen, mich im Alltag zurechtzufinden. Auch diese könnte man gewissermaßen als Schleifen bezeichnen.

Would the program or the topic you are tackling influence anyone else?

Ich denke, es ist ganz schön anzuschauen, wie sich die einzelnen Elemente leicht verzögert bewegen und vielleicht findet noch jemand das Spongebob-Video lustig. Ich denke Schleifen im Allg. haben eine große gesellschaftliche Bedeutung in Form von "Teufelskreisen" oder Spiralen der Angst/Trauer/Gewalt.

View in Editor



4: generation() ~ Building (Random) Blocks

What inspired your piece?

Das Werk ist größtenteils inspiriert durch die Geschichte um Christopher Strachey und den Liebesbriefe-schreibenden M.U.C. Ich dachte mir, der Computer muss tief erschüttert und traurig sein, dass seine liebevollen und zärtlichen Liebesbriefe nie beantwortet werden. Mit der Demo-Version des text-to-image models "Stable Diffusion" habe ich einen Computer selbst ein grafisches Spiegelbild der inneren Situation des M.U.C. generieren lassen. Drei Output-Bilder sind im Werk dargestellt. Auf die Frage aus dem Intro "Would you consider generative art to be the only way to see code as an art form?" habe ich in der verbleibenden Ecke des Quadrats geantwortet. In der Form eines Haikus (zugegeben die einfachere, europäische Form, bei der eine Silbe als eine More interpretiert wird) habe ich aufgezeigt, dass Code auch per se ein (kleines) Kunstwerk sein kann, ohne ausgeführt zu werden oder etwas zu generieren.

How does the topic influence yourself?

Mich faziniert, wie mit Computer-Programmen und selbst simpelster Logik komplexe Grafiken entstehen können. Es kann süchtig machen mit "Stable Diffusion" die absurdesten Szenarien und Bilder zu generieren. Auch habe ich vor einiger Zeit schon mit Fraktalen experimentiert, die entstehen, wenn man in Paint simple Pixel-Strukturen auf regelmäßige Art immer wieder ausschneidet, dreht und anfügt. Ich wäre auch interssiert, irgendwann einmal die Angebote von "Genarative Artistry" auszuprobieren.

Would the program or the topic you are tackling influence anyone else?

Vielleicht gefällt jemanden das kleine, melancholische Haiku oder der versteckte Song.

View in Editor

Submissions **APPLE** - 3
1 – 2 – 3 – 4 - 5 - 8

5: abstraction() ~ Matter (over/under) Class

What inspired your piece?

Ich habe mich damit auseinandergesetzt, wie Objekte aus der realen Welt in digitale Objekte umgewandelt werden können. Hier für steht das Foto einer Rose, das über einen Pfeil mit dem Icon einer Rose aus der Item-Anzeige eines Videospiels verbunden ist. An den Seiten sind Eigenschaften und mögliche Aktionen beider Objekte bzw. des einen Objekts aufgelistet. Die Listenpunkte auf der Höhe des digitalen Objekts sind eher informatisch strukturiert und emotionslos. Die Punkte auf höhe des Fotos sind eher in Prosa und gefühlvoller gehalten. Der Hintergrund erinnert mich an die Aufnahme eines tomographischen Verfahren, bei dem auch reale Objekte (z.B. lebendes Gewebe) digitalisiert wird.

How does the topic influence yourself?

In den Naturwissenschaften werden oft komplexe Objekte aus der realen Welt durch Anwendung eines Modells in vereinfachte, theoretische Objekte überführt. Hierbei muss besonders auf die Modellgrenzen geachtet werden. Außerhalb dieser nimmt das theorische Objekt Eigenschaften an, die sich im realen Objekt nicht mehr widerspiegeln. Kritisch wird es meiner Ansicht auch, wenn Menschen und andere Lebewesen zu Objekten "vereinfacht" werden. Ein theoretisches Objekt mit Laufnummer kann die emotionale, individuelle Situation eines fühlenden Wesens nicht wiedergeben. Dennoch sind für viele Internet-Konzerne individuelle User nur bedeutungslose Listeneinträge.

Would the program or the topic you are tackling influence anyone else?

Vielleicht hinterfragt jemand, wie manche Dinge im Alltag vereinfacht und abstrahiert werden. Vielleicht bekommt jemand Lust einen Tango zu tanzen?

View in Editor

8: learning() ~ Cheating the Test

What inspired your piece?

Die ausgegebenen Materialien von Ruha Benjamin und Joy Buolamwini haben mich inspiriert, über Rassismus und Diskriminierung im Zusammenhang mit modernen Techniken der künstlichen Intelligenz (AI) zu reflektieren. Das vorliegende Werk stellt eine Art (nicht repräsentatives) Experiment dar. Erneut wurde das Tool "Stable Diffusion" eingesetzt. Es wurde jeweils der aufgeführte Input "criminal human" und "good employe" eingegeben und die ersten drei Bilder gespeichert, auf denen Menschen zu erkennen waren. Es zeigt sich, dass in der ersten "negativen" Kategorie keine weiß-europäischen Personen, sondern arabisch-stereotype Personen und POC generiert wurde. Hingegen sind in der zweiten "positiven" Kategorie überwiegend weiß-europäische Personen zu sehen. Somit wurde gezeigt, dass auch "Stable Diffusion" durch das "Training" rassistische Stereotype aufgenommen hat und wiedergibt. Bemerkenswert ist auch die links-mittige Person, die sowohl weibliche (rotes Kopftuch) als auch männliche Züge (grüner Oberlippenbart) übernimmt und als "kriminell" eingestuft wurde. Diese Zuordnung könnte zusätzlich als Queerfeindlichkeit interpretiert werden.

How does the topic influence yourself?

Als weißer Cis-Mann gehöre ich nicht zu einer vulnerablen Gruppe und werde deshalb von den klassischen Algorithmen nicht diskriminiert. Umso erschreckender war für mich zu erfahren, wie selbst alltägliche Algorithmen wie die Gesichtserkennung einer Kamera-App, Menschen, die nicht dem "weißen Norm-Gesicht" entsprechen, diskriminieren.

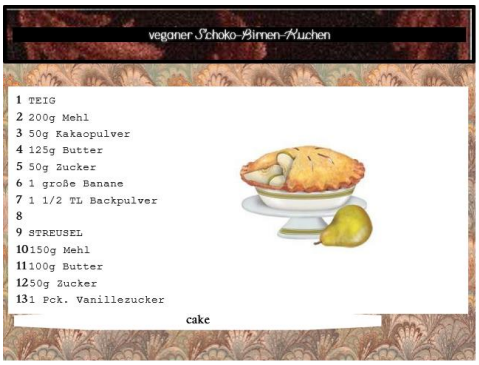
Would the program or the topic you are tackling influence anyone else?

Ich denke, dass gerade in weißen Bubbles noch mehr auf diese Themen aufmerksam gemacht werden muss. Vielleicht könnte mein Werk und die Diskusion darüber einen kleinen Anhaltspunkt liefern.

View in Editor

A.4.2 Participant PEACH

Submissions **PEACH** - 1
1 – 2 – 3 - 4 – 5



1: setup() ~ Why to Code

What inspired your piece?

die cookie crumbles - cake animation & die momentane Birnen Jahreszeit

How does the topic influence yourself?

Ich backe gern & ich mag den Herbst

Would the program or the topic you are tackling influence anyone else?

ja, alle Menschen müssen essen. Viele Menschen mögen Kuchen

View in Editor

2: data() ~ Real Data

What inspired your piece?

I like knitting. This is a scarf design I am thinking about

How does the topic influence yourself?

it is relaxing in a way that I don't have to stop thinking but at the same time it is somehow monotonous

Would the program or the topic you are tackling influence anyone else?

One of my friends probably ends up with it as a christmas gift :D

View in Editor

3: loops() ~ Time, but different

What inspired your piece?

I couldn't think of something better

How does the topic influence yourself?

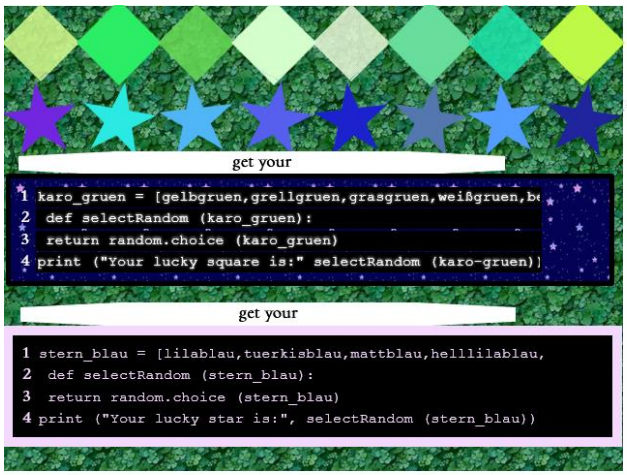
I greet people everyday

Would the program or the topic you are tackling influence anyone else?

yes, people don't like it if you don't greet them

View in Editor

Submissions **PEACH** - 2
1 – 2 – 3 - 4 – 5



4: generation() ~ Building (Random) Blocks

What inspired your piece?

Tarokarten beispiel

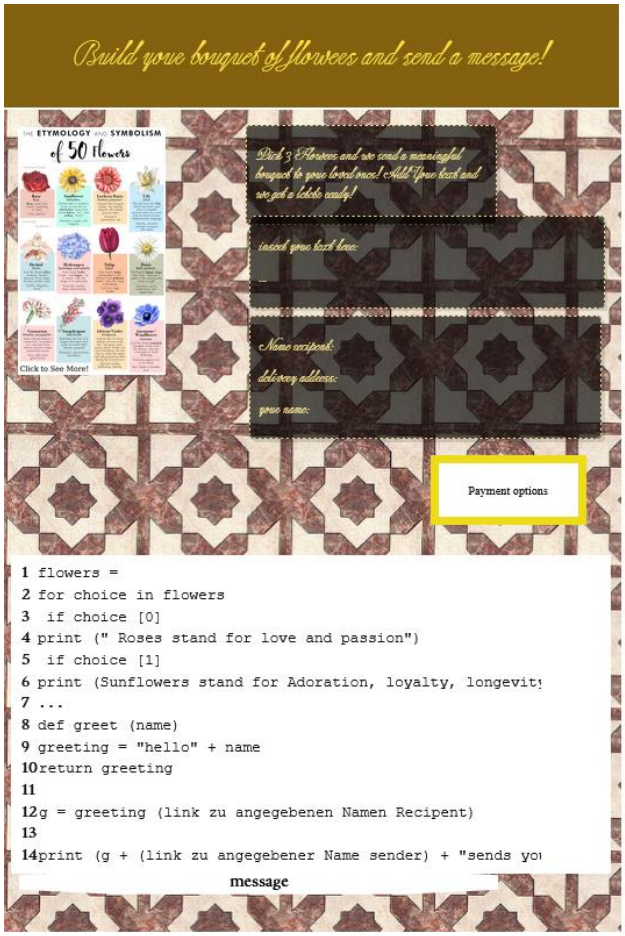
How does the topic influence yourself?

die Tarokarten haben spaß gemacht

Would the program or the topic you are tackling influence anyone else?

abergläubische Menschen vielleicht?

View in Editor



5: abstraction() ~ Matter (over/under) Class

What inspired your piece?

flower bouquet in office and messages vua flowers were big in korea

How does the topic influence yourself?

i like flowers

Would the program or the topic you are tackling influence anyone else?

many people like reseiving flowers but don't noe there meaning

View in Editor

A.4.3 Participant JACKFRUIT

Submissions **JACKFRUIT** - 1
1 – 2 – 3 – 5 – ~~7~~

1: setup() ~ Why to Code

What inspired your piece?

Ich wurde zum einen von Saaskia Freeke inspiriert, was sich mit visuellen Formen alles darstellen lässt und habe mir dann ein eigenes nahe liegendes Thema ausgesucht und überlegt, wie sich diese mit den mir zu verfügenden Formen darstellen lässt.

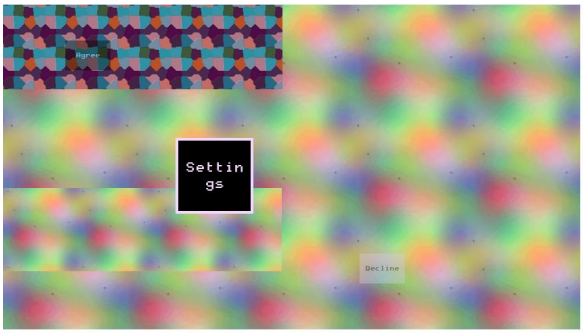
How does the topic influence yourself?

Das Thema soll zum einen zeigen wie geometrische Formen Naturszenarien darstellen können. Ich glaube mir wurde wieder mehr bewusst, wie man mit einfachen Mitteln auch Nuancen darstellen kann, wie beispielsweise Schattierungen.

Would the program or the topic you are tackling influence anyone else?

Es ist ein rein künstlerisches Projekt ohne politische, kulturelle etc. Absichten, aber vielleicht gibt das Thema Formen und Natur Inspiration für andere Darstellungen.

View in Editor



2: data() ~ Real Data

What inspired your piece?

Bei diesem Projekt inspirierte mich die manchmal limitierte Auswahl an Optionen im Internet, sei es beispielweise ob man Cookies zulässt oder nicht und wie diese Entscheidung, die entweder die Möglichkeit gibt etwas zu nutzen, zu sehen etc. oder halt auch nicht.

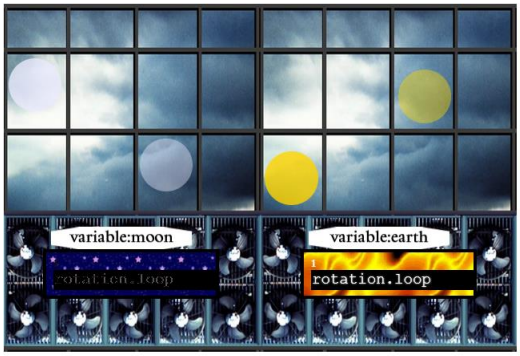
How does the topic influence yourself?

Ich möchte öfter bewusst die Entscheidung treffen, ob ich einen Inhalt im Internet wirklich bedingungslos zustimme oder halt auch in Kauf nehme, diesen dann nicht zu konsumieren.

Would the program or the topic you are tackling influence anyone else?

Ich denke diese Thema ist schon vielen bekannt, aber vielleicht hilft es anderen sich diesem wieder mehr bewusst zu werden, weil es halt auch durch die Limitation bei vielen (mich eingeschlossen) oftmals weniger Beachtung geschenkt wird oder man sich nicht dem Aufwand machen möchte, nach anderen Optionen zu suchen oder sich beispielsweise die ganzen AGBs durzulesen.

View in Editor



3: loops() ~ Time, but different

What inspired your piece?

Zu diesen Programm habe ich mir überlegt, wo fangen loops im täglische Leben an und wie lassen sich diese im visuellen Kontext darstellen. Es war dann ein ziemlich einfaches Thema, welches aber schon mit den größten Einfluss auf das menschliche Leben nimmt. Eine weiter Inspiration war das Buch: "Die Zeit, die Zeit" von Suter

How does the topic influence yourself?

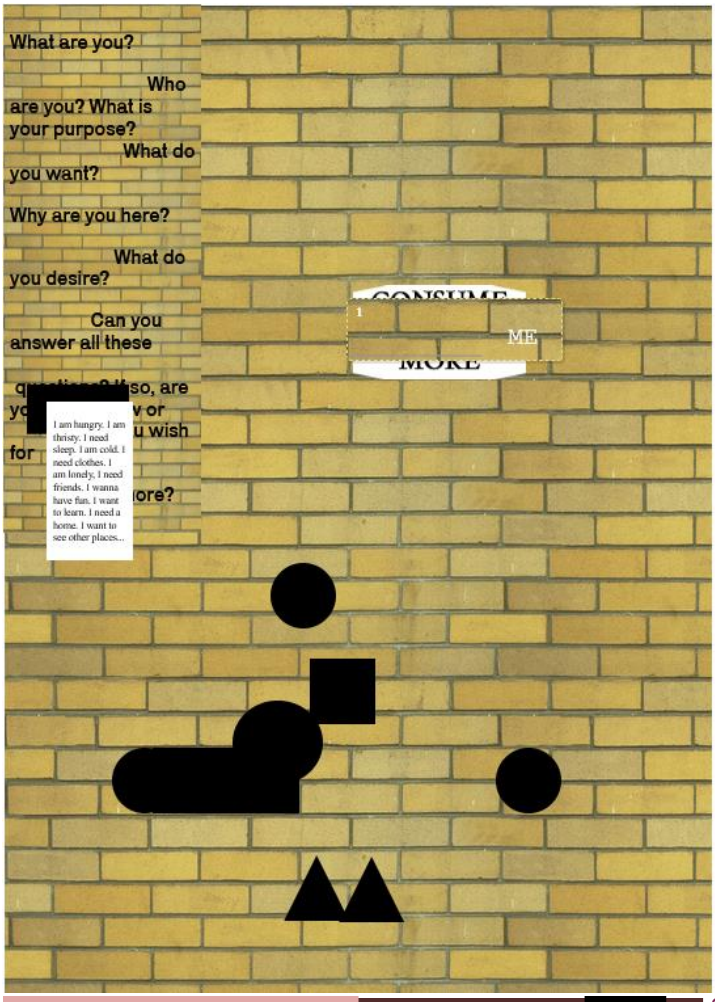
Ich denke wir sind alle an die Rotation der Erde und des Mondes gebunden und dem sich ewigen Kreislauf der Zeit. Die Zeit ist allgemein ein sehr spannendes Thema, wenn man sich mit anderen Darstellungen, Kalendern etc, beschäftigt.

Would the program or the topic you are tackling influence anyone else?

Nur wenn diese Darstellung angezweifelt wird und ob Zeit wirklich durch Rotation entsteht. Also aus philosphischer Sichtweise kann Zeit aufjedenfalls anders gedacht werden.

View in Editor

Submissions **JACKFRUIT** - 2
1 – 2 – 3 – 5 – ~~7~~



5: abstraction() ~ Matter (over/under) Class

What inspired your piece?

Die Spiele, die auf der Hub zur Verfügung gestellt wurden. Beobachtungen von 'modernen' Gesellschaften und Leuten (einschließlich mir selbst) in Hinblick auf Kapitalismus und Consumerism und die Frage, ob Konsum wirklich glücklich macht.

How does the topic influence yourself?

Es ist spannend wie man selbst auch vom Konsum betroffen ist und welche Glückshormone bei einem Kauf ausgeschüttet werden, wie bei einer Droge und wie dieses ganze Bild von der Gesellschaft aufrecht erhalten und befeuert wird.

Would the program or the topic you are tackling influence anyone else?

Es soll die Frage aufwerfen, ob ein Punkt erreicht werden kann an dem ein Mensch keine neuen Sachen, Wünsche etc. mehr braucht/hat oder ob dieser Punkt der vollumfänglichen Zufriedenheit nie erreicht werden kann. Eine sehr individuelle Entscheidung!

View in Editor

7: algorithm() ~ Building (Complex) Blocks

What inspired your piece?

Zu diesem Projekt habe ich mir meine anderen Projekte angeschaut und überlegt, welches am aufwendigsten war und dieses dann versucht in Schritte zu unterteilen und in einen vereinfachtem Code darzustellen.

How does the topic influence yourself?

Ich habe noch einmal überlegt, wie lässt sich etwas vereinfacht darstellen und wie Arbeitsschritte auch verschrifftlicht werden können. Jedoch wurde mir noch einmal sehr bewusst, wie genau und detailreich wirklich komplizierte Codes kreiert werden und welche Arbeit und Passion dahinter steckt.

Would the program or the topic you are tackling influence anyone else?

Es gibt zahlreiche Algorithmen, die sich sicherlich auch mehr bewähren als meiner, aber ich wollte einmal eine andere Form finden, wie ich meine Arbeitsprozesse für ein bestimmtes Projkt von mir darstellen kann. Es ging mir hier nicht um effizienz, sondern auch eher um einen Lernprozess für mich selbst. Die Codes sind natürlich nicht anwendbar, sondern angelehnt oder inspiriert von meinen Erinnerungen an 'richtiges Programmieren'. Also, um die Antwort in einem Wort zusammenzufassen: Nein.

View in Editor

A.4.4 Participant PINEAPPLE

Submissions **PINEAPPLE** - 1
2 – 3 – 4 – 5 – 6

2: data() ~ Real Data

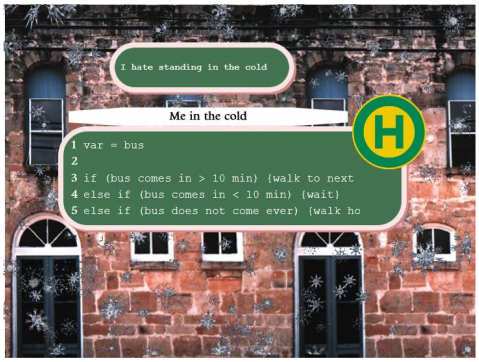
What inspired your piece?

Eine Freundin von mir stellt sich gerne vor, wie es wäre, kurz nach dem Tod einen Rückblick über das eigene Leben mit Daten/Statistiken zu bekommen. Dieses Gedankenspiel habe ich angerissen.

How does the topic influence yourself?

Would the program or the topic you are tackling influence anyone else?

View in Editor



3: loops() ~ Time, but different

What inspired your piece?

Ich habe mir anhand einer Alltagssituation (auf den Bus warten) vorgestellt, wie sich loopähnliche Prozesse in meinem Kopf abspielen.

How does the topic influence yourself?

Would the program or the topic you are tackling influence anyone else?

View in Editor

4: generation() ~ Building (Random) Blocks

What inspired your piece?

Ich habe inspiriert von den generative love-letters einen Satz erstellt, der zufällig mit verschiedenen Variablen gefüllt werden kann, wodurch ernste, lustige oder absurde Inhalte entstehen.

How does the topic influence yourself?

Would the program or the topic you are tackling influence anyone else?

View in Editor

Submissions **PINEAPPLE** - 2
2 – 3 – 4 – 5 – 6

5: abstraction() ~ Matter (over/under) Class

What inspired your piece?

Ich habe exemplarisch überlegt, nach welchen Aspekten ein Objekt für zwei verschiedene Szenarien abstrahiert werden kann.

How does the topic influence yourself?

Would the program or the topic you are tackling influence anyone else?

View in Editor

6: queery() ~ Computers 2 Action

What inspired your piece?

Ich habe versucht, das Gefühl, das das Thema bei mir auslöst, durch Animation darzustellen. Leider geht die Animation des Hintergrunds nach dem Speichern verloren.

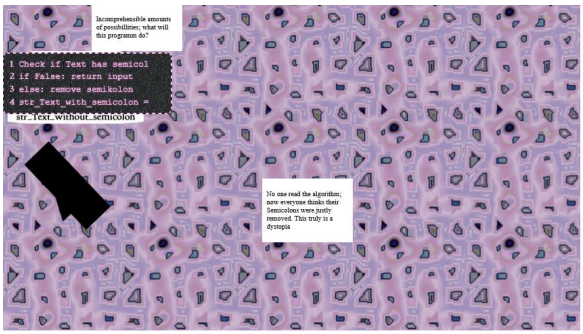
How does the topic influence yourself?

Would the program or the topic you are tackling influence anyone else?

View in Editor

A.4.5 Participant Grape

Submissions **GRAPE** - 1
1 – 2 – 3 – 5 – 7 - 8



1: setup() ~ Why to Code

What inspired your piece?

I felt overwhelmed at the options at my disposal and starter with something purposley dumb which slowely turned into somethin a bit more meaningful.

How does the topic influence yourself?

I became more thoughtful of the digital tools I use for convenience. If I don't know why, or for what thing they optimize I might be getting sold short.

Would the program or the topic you are tackling influence anyone else?

If it were use by everyone the entire concept of an obscure but cool charakter might fade out of existence;

View in Editor

2: data() ~ Real Data

What inspired your piece?

Seeing how powerful data (of people) is and how it is almost exclusively used in shady and secretive ways **against** the people who provided it is awful. Data could be incredebly empowering. It could conway truths about oneself which would be otherwise impossible to find out about.

How does the topic influence yourself?

It's depressing that empowerment isn't incentivised. All data collections projects that don't sell it for profit seem to be really small or obscure, as cool as they are.

Would the program or the topic you are tackling influence anyone else?

Hopefully! If you could analyse yourself in easy, accessible and safe/private ways you could gain incredible control over your own life and behaviour if you so wish. .

View in Editor

3: loops() ~ Time, but different

What inspired your piece?

I know how loops work, but stayed up way to long to finish the flash game...

How does the topic influence yourself?

loops are dope! Powerful tool, but remember to exit them in time

Would the program or the topic you are tackling influence anyone else?

My bedtime only influences me directly I think

View in Editor

Submissions **GRAPE** - 2

1 – 2 – 3 – 5 – ~~7~~ - 8

4: generation() ~ Building (Random) Blocks

What inspired your piece?

The thing on 'generative artistry' was incredebly beautiful and easy to follow. But instead of squares I wanted to paint with sine waves for musical reasons.

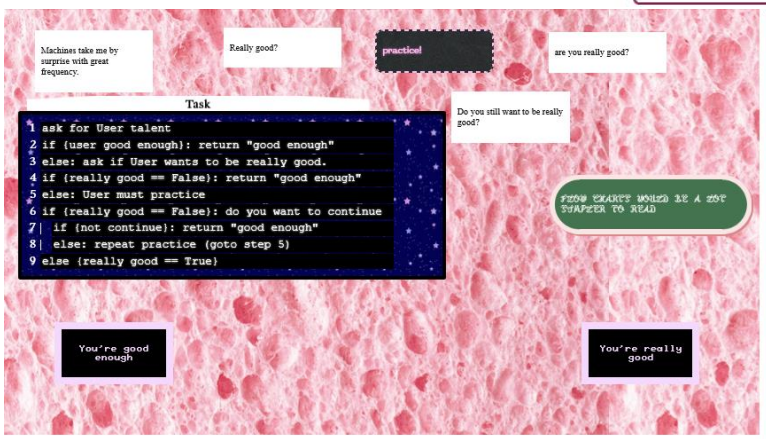
How does the topic influence yourself?

Generative Art is fascinating, I will have to delve deeper into this subject.

Would the program or the topic you are tackling influence anyone else?

Tools to generate random sounds already exist but aren't widely used as far as I am aware. So probably not that much.

View in Editor



7: algorithm() ~ Building (Complex) Blocks

What inspired your piece?

I wanted to create an algorithm out of a simple flowchart structure. I mostly wanted to do something OTHER than a sorting algorithm

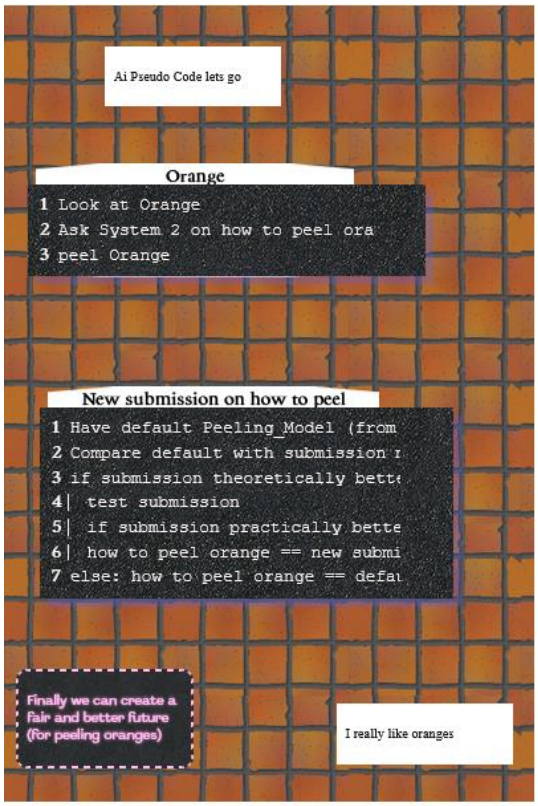
How does the topic influence yourself?

The existence of public relations for algorithms are a bit terryfiying. I wasn't aware of the efforts for algorithm concealment but that is also the point I guess.

Would the program or the topic you are tackling influence anyone else?

Yes! Finally in a handful of steps you can achieve anything (if you care enough). Or not, it really isn't that great a algorithm for most things.

View in Editor



8: learning() ~ Cheating the Test

What inspired your piece?

I took inspiration from 'To Be or not to Be Hacked? The Future of Democracy, Work, and Identity'. The part about the taiwanese mask supply system which could be improved by everyday-people-input seemed so elegant. It saddens me, that this way of handling the public world in general is the exception, not the norm.

How does the topic influence yourself?

Having no chance of offering better solutions to the worlds problems without lot's of work is baffling! I wish this would change

Would the program or the topic you are tackling influence anyone else?

Coding while keeping in mind that other (common) people might have better solutions to offer or just valuble input in general could leed to unimaginable greater outcomes

View in Editor

A.4.6 Participant LEMON

Submissions **LEMON** - 1
1 – 2 – 3 – 6 - ~~7~~

1: setup() ~ Why to Code

What inspired your piece?

the struggle to get a circle done in one of the provided materials for the chapter.

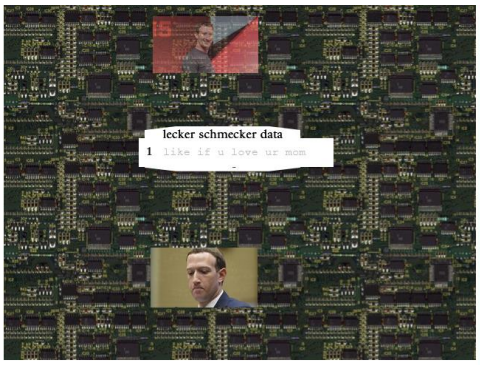
How does the topic influence yourself?

it helped me open up to the assignment more and put my perfectionism aside.

Would the program or the topic you are tackling influence anyone else?

probably not? It would me probably just cause confusion or little chuckles, like an obscure meme.

View in Editor



2: data() ~ Real Data

What inspired your piece?

I was inspired by the documentary about surveillance capitalism which I enjoyed very much.

How does the topic influence yourself?

I'd definetely want to take a closer look on the media I use/ that use me and maybe look into which ones I could step away from.

Would the program or the topic you are tackling influence anyone else?

The program could (capital could, it probably shouldn't) be like a gateway meme into the more serious topic, which then could hopefully lead to people being more aware of the media they are using and to even take action to protect theyr data.

View in Editor

3: loops() ~ Time, but different

What inspired your piece?

i just really like ouroboroi.

How does the topic influence yourself?

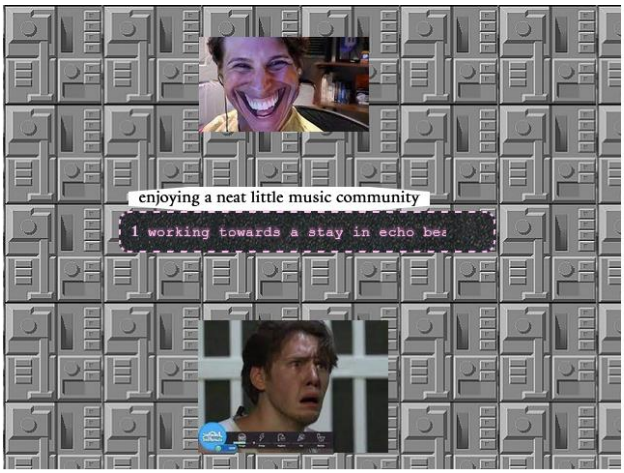
i used to think about the concept of ouroboroi a lot when I was a child, only that my approach was 'if a vaccuume cleaner would start vaccuume itself, how would that play out. how much could it take in of itself, before it breaks?' and this was a fun prompt to reconnect with this topic.

Would the program or the topic you are tackling influence anyone else?

probably not? maybe other weird kids.

View in Editor

Submissions **LEMON** - 2
1 – 2 – 3 – 6 - ~~7~~



6: queery() ~ Computers 2 Action

What inspired your piece?

i enjoyed playing echobeach a lot. The music was amazing, the deducing really hit a nerve for me and just diving in to this community of undercover artists, exploring their work and interactions was just great. But also it was deeply unsettling to be like 'fuck, actually this is pretty messed up' after identifying one of them. It reminded me of enders game and how distancing yourself from others by only seeing them through screens makes cruelty so much easier for a lot of people and how that can be gamified and weaponized for oppressive agencys. But no matter how hard I tried I couldn't find anything to really embody how I felt about this so I just threw something together tbh.

How does the topic influence yourself?

I think that it's not that easy to get the program unless you have some weird nieche knowledge (about the game but also havin watched the jerma dollhouse streem I crossreferenced would maybe help because we have the topic of gamifying interaction with other people again). So.. maybe.

Would the program or the topic you are tackling influence anyone else?

View in Editor

7: algorithm() ~ Building (Complex) Blocks

What inspired your piece?

i never really looked into algorithms that far, but it was very fascinating to see how sorting algorithms work and was pretty impressed with the merge sort algorithm in comparison to the bubble sort.

How does the topic influence yourself?

It helped me to get a (like reaally basic) understanding of algorithms, like.. it demystified them and made just thinking about them more accessible for me i think.

Would the program or the topic you are tackling influence anyone else?

could maybe do the same as for me?

View in Editor

A.5 Interview Guide

Teaching Queer Code: Guide for the Final Interview

Observation: Learning success and quality of the course form

PQ □ How did you feel during the course?

□ (LEARNING EXPERIENCE) WHAT WORDS WOULD YOU USE TO DESCRIBE PROGRAMMING?

- □ What was your impression of the different sources?
- □ Which topics or information were you not aware of before?
- □ Which topics would you like to explore in more depth on your own?
- □ Would you want to discuss the topic with other laypeople?
- □ Would you say that you have been programming in your pseudoprograms?
- □ How do you feel about teaching technical tasks in connection with social analysis?
- □ Would you research other practical resources for learning programming in the future?
- □ Would you consider learning a programming language?
- □ What ideas for implementations woould you like to program?

□ (COURSE QUALITY) WHAT DO YOU THINK ABOUT THE SUMMARY OF YOUR COURSEWORK?

- □ Which of your works would you publish as the face of your course completion?
- □ What was your strategy for creating the works?
- □ What hurdles did you encounter during the creation process?
- □ How did it feel to progress on the site? (Codes, Hub)
- □ How did you work through the zine sections?
- □ How did you find the digital resources?
- □ Which chapters would you want to explore further?
- □ When did you get the most ideas for the submissions?
- □ What did you enjoy most about the course?
- □ What options or content would you have liked to see in the course?

A.6 Interview Transcripts

A.6.1 Participant Apple

<Interviewer>: So, thank you for being here for the last part of the study. I will begin the recording now, is that okay?
<Participant>: Of course.
<Interviewer>: You have participated in programming course. I'd like to ask you how you felt during that and how you enjoyed it.
<Participant>: I found it a bit surprising for a programming course, I believe. What I felt? [pause] Well, obviously it pushed my creativity a little [laughs] I didn't think it would require so much self-expression for me. I thought it would be more structured. It was very open. The material impressed me in very different ways.
<Interviewer>: Would you like to elaborate on that? What were your impressions on the different sources?
<Participant>: Some resources were a bit tough for me. I didn't check out all the material, like the books, or the 3-hour documentations. But there were other things that I found very interesting. Like the more creative things. Like the one artist who creates programmed art. Or the digital workshops where you are able to program something yourself.
<Interviewer>: Did you try to program some of it?
<Participant>: Yes, I didn't go in depth, but in this JavaScript-based environment, I tried to program a circle, give it a color, so it was kind of my first project to make this red circle.
<Interviewer>: I found very cool how you showed a lot of interest in using external sources. You actually had considerable self-study in that regard. What was your strategy for every submission?
<Participant>: Yeah, at first I read the magazine, then I opened the links on your website. I looked at all of it, but skipped the parts that I didn't really have the energy or time for. Some things I read through intensively. I also started taking some notes before I forgot some of the information. Based on my notes, and the open questions that you posed, I thought about what one could at this point. What would inspire me now?
<Interviewer>: Any chapter that stuck to you?

<Participant>: What shocked me most was probably one of the last topics are worked on. That AI is adapting the intrinsic racism of the data it is fed, the images. If you think about it, it's so plausible, but I never really thought about it.
<Interviewer>: You used an image AI tool for your submissions there. How did you come across it?
<Participant>: A friend of mine is a biologist, and does biological computer science on the side. He likes generating such images, and sent some of them to me. And it came to my mind when the course talked about generation. And I thought, I could create something with it.
<Interviewer>: I genuinely loved all your entries. You can see all of them here alongside your reasoning. I assume, you took great inspiration from the world around you to work on them. One of them stuck to me, it was the one based on the exhibition that you visited. Have you attended it?
<Participant>: I have!
<Interviewer>: How did you like it?
<Participant>: It was some time ago, but it was definitely very surprising as well. I didn't read on it much. Okay hygiene, it must be about how can I prevent infections, and stuff like that. It was a lot more broad as it was a medical museum with many experiments to try out. And the screening of the private person, that glass person, maybe I can do something with that. It was about the whole world play with it. When I was there, there was another exhibition on race theory because the museum itself had an obvious problem with it back then. They talked about how race in general is not a real thing, but a social concept, and that putting humans into these kinds of groups is nonsensical.
<Interviewer>: I think it's very important to look at these topics through this interdisciplinary lens as you do. Do you remember any exhibitions on further digital topics?
<Participant>: [pause] I don't believe it was the main topic. I really like visiting nature museums, those always have some more technical sections. But there wasn't really anything based on computer programming.
<Interviewer>: Do you believe that pseudoprograms like those here, would work as parts of an exhibit?

<Participant>: Yeah definitely. I think it had a lot of potential for something like that. I really loved one of the resources in this digital gallery where one could walk through there, edit the work of others and rate it. I wasn't confident enough to do something by myself there, but walking through it was fascinating. There is always so much digital art in these exhibitions with projections, installations. Maybe not just two-dimensionally, but maybe even in a room where elements change and move around.
<Interviewer>: Would you like to find more practical resources on the topic? Working on a programming language?
<Participant>: I had some very simple Delphi programming in school back then. In university, I thought to myself, okay, you're a researcher, you better get into programming, so I learned some Python basics. I didn't really need them yet, though.
<Interviewer>: Any ideas of what it may be? Something professional? Something private?
<Participant>: I believe it would be more for a practical application in terms of data analysis. Do you know this memory game where you have squares leading up to a big square, and you have to memorize the pattern sequence? Each round, a new part of the pattern lights up or makes a sound, and you have to reproduce it every time. I tried to program something like this, but discontinued it. I might attempt the project again, though. Just to not forget it. I remember it from my childhood from some random games.
<Interviewer>: I love this. Would you feel confident in exchanging this newly found knowledge from the course with others who didn't get the insight?
<Participant>: I think that with topics like search engines, AI, or data collection, those are all current topics. And I believe I had considerable input on it by now that I could definitely bring up in discussions. In more technical discourse, I couldn't really keep up, though.
<Interviewer>: You have all your course submissions here. How does it feel looking at them?
<Participant>: I think it's so interesting what I did there.
<Interviewer>: I think it's apparent that you went through some progress where you at first were pretty tied to the resources that you were given, and then you kept hiding

hyperlinks in there for me to find, which I really loved. And in the end, you confronted yourself with the more difficult sources where you took a stance on the racial injustice through AI. And I think you did a great job putting it all together with the help of the external tool. Is there any piece that you would consider your main one, one that you would choose as the face of your course progression?

<Participant>: [pause] Not directly. I like many of them. Maybe the love letters? But yeah, definitely, as you said, the last one.

<Interviewer>: When were the moments you got the most ideas during your work on a chapter?

<Participant>: Definitely while I was reading the resources. There I had most situations where my mind caught thought. But after further reflection, and looking at my notes, doing the dishes, I eventually found something. It really depended on the topic.

<Interviewer>: Yeah, it was apparent that you continuously worked through the course. You didn't do everything all at once. Was there anything you found the most fun?

<Participant>: [laughs] I liked the little minigames that you had scattered throughout the material. It was pretty relaxing in-between all the reading. Also, the conceptualizing, and assembling everything was even more fun. While I worked on the task, created some elements, I got even more ideas on what I could add. It created this flow where the submissions eventually formed.

<Interviewer>: I can relate to that. The course itself was rather short for the amount of topics addressed. Let's say we had all the possibilities, what would be your wishes for the course form?

<Participant>: Maybe the little technical difficulties where things wouldn't work as they should be, more opportunities for bug fixing in general. Aside from that, I felt like the texts in the zine were rather academic. It would be great to have more of an introduction to ease you into these difficult topics, so people who are unfamiliar with any of it can get a new view on computer science. Maybe some super basic concepts if they have absolutely no connection to it.

<Interviewer>: Awesome! Those were all of my questions to you. Thank you again!

A.6.2 Participant Peach

<Interviewer>: Thank you so much for taking part in my study. This will take about half an hour I think, then is it okay that I record your voice?
<Participant>: Yes.
<Interviewer>: So roughly speaking, do you have any thoughts in general about the course, how did you find it? Did you have fun with it? How did it feel?
<Participant>: I had fun, at first I didn't really know what I was doing. Started out like this and didn't really know what to do. Just tried everything and looked at all the material, and then I asked you again.
<Interviewer>: Did you have any particular problems with this open task? That you didn't have exactly one task, but that you had to try out something yourself?
<Participant>: Yes, I think, because all the courses, which I had already participated in, we had just a task to program this and that and this and that should come out at the end. And the first time I just tried it out and thought yes okay try it out. The second time I thought somehow that the tool would look different. So somehow what I had looked at before would come up and that I would have to work it in somehow. And then there was nothing, and then I thought, did I do that wrong now? But once I understood that I can just program anything, and not to use a language, then it was then it was quite cool because you [pause] Then you could think how you would implement it, but it didn't matter now that the program probably wouldn't work. [laughter]
<Interviewer>: Yeah, maybe you remember the question: how would you, what words would you use to describe programming then like this.
[pause]
<Participant>: Complex, for sure. And partly frustrating. So I can imagine you sitting there for ages in front of the code and then, ah, it probably doesn't work like that because I didn't do this and that. And to think like a computer, that you have to feed in every little bit [pause] I found that frustrating. I still find that frustrating. But also not very observing somehow. So it drags you in, and I can totally understand that people kind of sit for six hours at a time to get their program to work.

<Interviewer>: You mean the sources, as well as your task, have helped you to understand how to approach this course. What was your impression of the sources. Did anything stick in your mind?
<Participant>: Yes, actually, not only the course, but generally dealing with the topic. For example, I mean, of course you've heard that a lot of data is collected from you. So now I actually dealt with it, and then I really had enough time to take a look at it and generally think about how much everything is actually programmed.
<Interviewer>: Would you then also feel confident to discuss the topic with others?
<Participant>: Yeah, well, I think so. I think there are people who know a lot more about it than I do. But yes.
<Interviewer>: Back to my question. Did you have any topics that particularly stuck in your head? Any chapters that resonated with you. You mean data, obviously? Would you say that you would expand on that? If you saw articles on the subject, which ones would you click on?
<Participant>: I would watch the videos with the discussions. These videos I found nice, the discussion videos. With articles, it depends on what news and what newspaper it is.
<Interviewer>: So you would prefer to stay on that more sociological, philosophical level rather the technical side. But still, do you find that even coming up with a syntax has helped you a little bit in dealing with the subject? So would you say that when you were programming, that you internalized the topics more?
<Participant>: The data now, or in general?
<Interviewer>: Generally in the course, yes.
<Participant>: [Pause] As soon as it comes to loops, I think you get kind of stuck. So of course I understand how a loop works, but I find that thinking it through [pause] At some point, you always lose the thread and think to yourself, oh, did I already include that? If I turn there now, what will happen? I find that totally difficult. But in itself, I find it fascinating to think about how a computer thinks, because then you realize what you can actually do with a computer. If that's the way it is, then it will take forever until artificial intelligence can really understand feelings. That's because it can really only turn one path and not take any shortcuts.
<Interviewer>: Yes, that's always the difficulty, because we

have this level of abstraction that we create in programming, that we take away from reality in order to be able to capture it in data. That's what makes it so complicated.
<Participant>: Yes, but also somehow totally simplified, because the real world is more complicated than that and there are ten thousand other things involved that you can't represent. Maybe you could, but you'd spend ages on one little thing to represent all the influences alone.
<Interviewer>: If the loops cause such problems. The thinking of these loops is often a big hurdle – you are not alone with this one, because you often have to oversee the broader context. It represents what happens in a time space that you can't really keep track of during programming. If there were programming languages that didn't have that. If it was more graphical, to maybe create a web page without any generation. Would you be up for doing something like that?
<Participant>: I can't imagine right now. So I thought maybe I should have used arrows to draw what goes where, but i didn't do that.
<Interviewer>: You did write a very good loop, I think.
<Participant>: Yes, but that's still short. Some of the information that's processed is really crass.
<Interviewer>: Yes, if there's a second loop nested, it becomes critical again.
[laughter]
<Participant>: This one also turned out cool [points to submission 4]. Then I gave up on that thing and just [pause] added dot dot dot.
<Interviewer>: That's okay, too.
[laughter]
<Participant>: So I thought, surely you can still loop this, but it occurred to me that I can feed each one in. I didn't feel like doing that, so I said, you just imagine that I fed everything in one by one.
[laughter.]
<Interviewer>: While we're on your submissions, do you have any particular feelings when you look at it like that? [shows submissions].
<Participant>: So the first two I think are boring now because there's not that much happening except that one looks like a Pinterest page. [First submission] I think that's when I was just hungry on the first one and had

pears. And then from here [second submission] I had more fun. Foremost, having an idea of what to program, I think that's one of the hardest things. It reminds me of university, when you have to write a paper, and the most difficult thing is to find a topic. And it's the same with programming. When I think about what I want to do with it, okay, but I would have to invest so much time in writing it, because I'm just not used to it. So if I now think of a text analysis that picks out words for me, I would be quicker to pick out the texts myself than to program that now. Because I sit there for ages. That's why I always sat there, okay, you're going to do this now, but what do you actually want?

<Interviewer>: But you had some pretty special ideas nonetheless! What were your strategies?

<Participant>: I took a look at your special material. Okay, something like that, but simpler. [pause] [third delivery] Yes, that's the easiest. There is this Hello World. And then I thought, "Hello, Artur. [laughter] [delivery four] And I couldn't think of a game, so I just designed a sales page for flowers.

<Interviewer>: But I found the ideas quite specific. Here were flowers, here were friends, the tarot cards were inspired by the one game. The knitting instructions, cake baking, these are all hobbies that you internalize yourself, that are part of your identity. Would you say that it was easier for you when you said, I'm going to relate this to myself a little bit and take a theme that I've already been in touch with.

<Participant>: Yes, because you don't have to deal with a subject again. I always thought, what would I need? I'm sure you can program a lot, but I don't care what others do. I had no idea for ages, but I just planned what I could give you for Christmas. First I had the idea to knit a scarf by myself, which took way too much time and now only one person gets a scarf. [laughter] And then I just did that and played with the fact that you can insert pictures.

<Interviewer>: How did you find it in general to try out the editor and what you can do there?

<Participant>: Well, the first two programs I didn't understand that it didn't run at all. I thought to myself, now I don't know if it works! And I was pretty sure that it didn't. [laughs] And then I kind of saw it more

as a design element. With the one, I just thought, that's perfect for a recipe. That's how I came up with the recipe too, because I thought "that's a perfect list!". And then it also said cake [code OUT element in editor] and then I thought you could embellish that. And then I thought, well, this doesn't have much to do with programming, it's more like a little ideas page.

<Interviewer>: You don't think you were programming in any way, but graphically creating something?

<Participant>: With the first two, yes. With the others, I was trying to think more like that. And then I imagined if I put it in like this, now how would that play out, and then what's going to happen to it? But the first two are just Pinterest pages.

<Interviewer>: So you knew that you didn't have any knowledge of software architecture that would generate that.

<Participant>: Yeah, I thought, you can't really see that, and I just imagined that it would just run in the background. And then I wanted the adjectives to be assigned to it. But then I lost the thread at some point.

<Interviewer>: You got a zine from me. What was your strategy? How did you go about it?

<Participant>: I worked through it chapter by chapter. I did that once, and then I went to the next chapter. I read the zine first and then went to the page, read through the material, then had like the first idea of what it was about. Okay, this is about loops, this is about data, and then I looked at that, and then I always tried to find what I had from the programming course. That was not so successful.

<Interviewer>: You looked at your own notes on this?

<Participant>: Yes, I had taken notes at that time and I thought that I could use those afterward. But partly I thought [pause] I don't remember exactly.

<Interviewer>: On the internet, you had also searched for infos?

<Participant>: Yeah. For the random stuff, I just googled, "How can I find out random things in Python?" or something, and then that came up. Oh, that's so nice and short!

<Interviewer>: That's really short. Most of the time other people have figured it out for you already.

<Participant>: Yeah, I thought, nowhere does it say I can't google the internet. Then I partially pieced together

from what I had from notes. I figured, okay, this was under the note loops, then it must fit together. I know something was there with Java, but I thought, I only know Python. No, I don't feel like Java right now. One programming language I know at least at 1% and I thought, I'll take that and I'll stick with it. That's what I was thinking about a little bit. And then at work I have to do a lot with Excel and I always try to use If–Then, so that at least you don't completely forget it. I will never program complex structures, but at least if–then.
[laughter]
<Interviewer>: To understand in general how it works is always very helpful. You can also go really deep into Excel.
<Participant>: I know! Somewhere I had found that you can draw a timeline and I had tried to find something about it, but there was not so much explanation. Then the idea was thrown away in the team. But then I thought, oh, there is so much more to Excel than meets the eye.
<Interviewer>: There are also very complex loops there. You can go totally nuts there.
<Participant>: Once I had a little bit of time, and then I programmed a whole evaluation somehow. I was a bit proud of that. There was a column with numbers, and everything else ran back to it. Then it turned green, if it was positive it turned red for what we didn't have to pay attention to anymore. I think my bosses thought I spent too much time on it, that I could have done other tasks. But I looked at it over the week and there were a lot of IF loops.
<Interviewer>: That's great. I'm always glad to hear that you do get back into a similar mindset then, even if it's just for a second.
<Participant>: Yeah, I think it's super handy because it saves you so much time. Also, something like VLOOPKUP in Excel.
<Interviewer>: That's also a loop.
<Participant>: Yeah, but one that I don't have to program anymore. [But then it doesn't find the data because the name has to be entered differently. Then I have to go through it again and find the name from the table and put it into the next one.
<Interviewer>: Yes, these manual processes are not lost. Oftentimes one must intervene then nevertheless.

<Participant>: Yes, but you still save quite a lot of time. I always write everything down in my workbook and everything.
<Interviewer>: Roughly speaking, is there anything you particularly enjoyed about the course?
<Participant>: Yes, once I understood that I could enter the programming, I had a lot of fun. I had a lot of fun with the backgrounds.
<Interviewer>: Did you like them?
<Participant>: Yes.
<Interviewer>: Nice.
[laughter]
<Participant>: So I had fun imagining how everything would work, even though I know most of it wouldn't have worked, and I would have gotten frustrated. But then I thought, that's cool when it all goes like that and the stars move. To imagine it working like that someday.
<Interviewer>: That makes me very happy. And to wrap this up, just speculatively, do you have any ideas that you would like to see in a course like this? Functions, other media, what else were you missing?
[pause]
<Participant>: Yeah, just my confusion at the beginning. I think that it was mentioned somewhere, but I didn't understand that that's what a pseudoprogram is. I understood that it is executed somewhere, but that something happens when you click on Go. Something like an error. I couldn't quite figure out what a pseudoprogram was.
<Interviewer>: I totally understand that. So, a very basic, simple explanation of what exactly is going on here.
<Participant>: But otherwise, you had Google. [laughter]
<Interviewer>: Do you have any idea what would be the face of all your programs?
<Participant>: I think this is the coolest [points to submission 3]. The background is the coolest, but I was still fiddling with it, so that it rotates and blinks in different directions. And then I was really happy that I found this program.
<Interviewer>: Nice. I'm really excited about it. That concludes the interview. Thank you so much.

A.6.3 Participant Grape

<Interviewer>: Thanks again for taking part in this last interview. I will start the recording now, is that fine by you?
<Participant>: Yes
<Interviewer>: So you were a part of my study as a substitute for someone who couldn't make it, unfortunately. Foremost, I want to ask you how you liked the course.
<Participant>: It may be a little in the past, but I recall being involved with most of the resources on there, which were plentiful. So I believe it couldn't have been very bad, or else I wouldn't even have stuck around. [laughs] Some of the things I found so cool, I bookmarked them!
<Interviewer>: Oh, cool, what exactly?
<Participant>: For example, the digital artistry thing. I was really impressed with that.
<Interviewer>: What did you like the most about it?
<Participant>: Making cool stuff with the simplest codes is nice as it is. What I found most impressive were the thoughts behind them and how genius the final artwork becomes. It's just fundamentally beautiful.
<Interviewer>: Awesome. You might recognize this question: With which adjectives would you describe programming?
<Participant>: [pause] Scary, time-intensive, frustrating, pretty neat!
<Interviewer>: I believe you had some programming experience. You already programmed something for university. Would you like to specify what you programmed in particular?
<Participant>: Absolutely! Things I made that actually worked were in university, where I analyzed data. For most of them, I didn't really know what I was doing, but it kinda worked out, like drawing polar plots in Python. This took me two days and was quite gruesome [laughs]. And yeah, small analysis algorithms like a k-means thing. Before the whole university stuff, I re-coded a calculator in C# to have a little text box that would translate glyphs into English letters.
<Interviewer>: That's actually more than I assumed! They sound like really cool projects. Were you able to find some of the things you did back then in the study?
<Participant>: So in general, pseudocode, when you are dealing with stuff like analysis algorithms, where you don't know where to start, is genius. Most of the time, I have to annotate every line I write, so I know what it

actually does. If I don't, then there won't be any problem-solving. I believe there was one thing I made [pause] with semicolons. Just some recursive thing.

<Interviewer>: Yeah, finding out when there's a semicolon in a text. I find it remarkable how you developed your own style in these submissions. The course itself barely teaches you explicit syntax, and you made it up for yourself in your own language. Where did this come from?

<Participant>: When I create variables, and I don't see their purpose, like the solutions on Stack Overflow, I don't see myself coding with them successfully. That's why when I try to do something difficult, I do something like ArrayWithThisValue1. If there's an error, it's easier for me to find it again.

<Interviewer>: Definitely. Many people have the most interesting strategies for naming conventions. Some write an N and others write a whole sentence.

<Participant>: Yeah, and the funniest names are the best ones. I'd rather have those to stay engaged.

<Interviewer>: When we think of self-made syntax, societal problems play a big role. The course tried to convey the technical part of programming while discussing the social problems accordingly. Would you say it helped you in any way? Were you able to find personal conclusions on some of the topics that may have been new to you?

<Participant>: Yes. One thing that blew my mind was a project that was an app about distributing masks where all users could contribute, and it worked out super efficiently. I thought it was fantastic. People aren't as stupid as one thinks, and when implemented correctly, it can be unbelievably efficient compared to software by the government. When you have attempts that are utter failures. As an example, imagine you would be able to self-proclaim that you have Covid in the Covid alert app. You wouldn't have to get a PCR test for that. This wouldn't have been a hassle to implement. And the other thing I see in science pretty often where marginalized groups are ignored in coding processes, like issues of sexism. Or like the alert app with the high white collar crime rate. That definitely stuck with me.

<Interviewer>: Definitely. The people who write these programs oftentimes have no control over or knowledge about the impact they have on society.

<Participant>: It's generally so important to see how you

filter data and how can you get your results. And if you are not educated, terrible things will happen.
<Interviewer>: Correct. Would you be confident in discussing these topics with other laypeople?
<Participant>: I think so. I am not much of a debater.
<Interviewer>: Not really debating. Just an exchange of information on the topic.
<Participant>: Yeah! I was already successful in taking two of my friends out of a terrible argument about a white representative of a black rights organization, and the so-called cancel culture. Luckily, we agreed that despite his degree in African Studies, people don't want to be represented by him.
<Interviewer>: I'm glad you found an agreement with them.
<Participant>: Me too!
[laughter]
<Interviewer>: Do you have any plans on pursuing more programming? Any projects you ever want to realize?
<Participant>: Nothing in particular right now. I used to start and stop RPG making projects. The coding in the RPG Maker is horrendous, though.
<Interviewer>: Oh yeah, I remember. Those were some of my first steps into programming.
<Participant>: This lever is supposed to open the door, why is it not doing just that?
[laughter]
<Participant>: But yeah, I am sure something like that will happen eventually. Aside from that, professional things. When you have a lot of data that you need to analyze, and you want to do cool stuff in it, you can't avoid programming.
<Interviewer>: Are you having fun visualizing data?
<Participant>: The programming itself is real funny, the visualizing; however, it's great to have a result, but you come across some terrible sub-steps. And with data visualization, it's hard to see whether the code you wrote will generate anything useful. It's fun as long as I have the opportunity to quickly debug.
<Interviewer>: Well, then, I sent you some of your course submissions. Did you take a look at them?
<Participant>: I have, yeah.
<Interviewer>: What are you feeling when you look at them?
<Participant>: The more I look at them, the more I love them. This is fantastic. I think it's crazy, I struggle a lot

of feeling competent when processes take a long time to finish. I think it's so cool to see what I did here. I feel funny.
<Interviewer>: Of all the submissions, is there one that sticks out to you and could represent your whole course completion?
<Participant>: I feel like, judging by vibe, I'd take the third one with the ugly pink sponge as a background.
<Interviewer>: Excuse me, it's one of my favorite backgrounds [laughs]. All these backgrounds are publically sourced that encourages using them all over the internet, so I felt like it fit the project very well !
<Participant>: That's awesome. I think the sponge one is just perfect. And it shows very well how I approached the study.
<Interviewer>: How did you approach it, then? What was your strategy?
<Participant>: I believe, I needed a bit of time to find an idea. That's why I just looked at everything, for some, especially the ones that impressed me the most, I immediately had something. For the one with white background, I was like, I guess I'll just do something.
<Interviewer>: Oh yeah, it's the one where you spend all your night with the game with the kingdom.
<Participant>: Oh, this one! Yeah, and we played a second time through it to get the true ending.
[laughter]
<Participant>: I believe, the actual coding part always came in between. I thought about what's going in and out, what's done to the input, is there anything saving and analyzing going on, and what is supposed to come out of there? Those were the questions I thought about at first. Are there more than one output possibility? And then I gave it the last touch in the hopes that people who are not me will understand it.
<Interviewer>: Yeah, I find it very impressive that you wrote for every submission at least a small code snippet. I can't stress enough that you did not need to do that! In general, the whole course took place on a website. Do you recall?
<Participant>: Yeah, it was definitely special.
<Interviewer>: How did it feel like working with it?
<Participant>: The beginning where you had to understand

where to input what code was a bit janky. But right after the second submission, I found it pretty cool. The first problem was that you when you wanted to look at past resources, you had to enter your secret code again. And for some submissions, I never really knew if it was uploaded or not. [laughs]
<Interviewer>: Yeah, I wish I could realize all of it again on a dedicated server to giver learners a proper experience, with cool and direct feedback.
<Participant>: But I mean, having a website that works and doesn't explode is quite the accomplishment. And I definitely saw websites way worse than that.
[laughter]
<Interviewer>: I am very happy to hear that. [laughs] You already mentioned that you liked many of the digital resources, especially the games. What can you remember from it? What did you find the coolest?
<Participant>: This Echo Beach, such a wonderful game. I will say that it felt a little like a "Good job kids, you've worked hard, now play some games".
[laughter]
<Participant>: The expert interviews, they were pretty long, especially because I listened to them right after another. Those were some cool people who had cool stuff to say. No regrets there. And especially the person talking about the white-collar crimes.
<Interviewer>: What was the most fun to you?
<Participant>: The actual code-writing I found to be pretty lit. Specifically, every time I put real code snippets in there, I thought about how implementable it all actually would be. Pseudocode is one thing, but what if it would take a million year to compute? Solving those micro-problems was intriguing. Especially when you have these super smooth solutions, that you most-likely just copied from someone else. But you still can be happy about that. I have an input, and output, and thinking about everything taking place in between is very entertaining.
<Interviewer>: I am so happy to hear that. Now, as a final question, let's say this course will evolve. What would you like to see for the future of it? How would you try to make it appeal to others?
<Participant>: Making things appealing is always such a challenge. While I was very interested in the topics, I found it pretty difficult to stay motivated with the

longer videos. If things got even more complex thematically, I would welcome that. Maybe even a small levelling system, or hidden leaderboards. Fun incentives in general! With discussion, it also really depends on who you are addressing in your course.
<Interviewer>: Thank you so much for all that insight. We're through! It was such a delight.

A.6.4 Participant Jackfruit

<Interviewer>: Thank you so much for participating in this final interview. Is it okay if I record our conversation from now on?
<Participant>: Yeah!
[laughter]
<Interviewer>: Awesome. So you participated, albeit a little while back, in a course on programming. Can you tell me how it generally felt attempting programming that is not as conventional as you may know?
<Participant>: Yeah, I can remember the beginning relatively well. When you came to us, and I just responded with a yes because I had some free time on my hands. But I immediately was afraid of what I got myself into here! [laughs] I can't recall anything from my computer science course back in school. I thought I would get guided back into it step by step, and then I had to [laughs] actually program something! [laughs] This was mostly my concern. But no, it wasn't like that at all, and I found that super nice to just dabble in it and get creative. Because of the little information boxes that you added, some things came back to me on what computer science were actually like. It's like 7 years back, after all. Yeah. That was fascinating. It was fun to collect yourself some projects and just work on them. Unfortunately, the tool kinda messed them up. [laughs]
<Interviewer>: Great. Yeah, as I told you earlier, it's not really about what comes out in the end, but the fact that you actually created something is already an accomplishment. Since you did have computer science lectures, were you able to identify some of the principles you learned there in the course?
[pause]
<Participant>: First, I had to work myself back into binary code. I didn't have to, but I did. I did try to calculate

some values and failed miserably. [laughs] But I recall that there was a chapter at the very beginning where it said something about height, open bracket, weight, open bracket, it all came back to me because we used this in the lessons when we worked with HTML. It was a lot, but most of it was lost in the limbo.

<Interviewer>: Now with your current knowledge, with which words would you describe programming now?

<Participant>: So how I feel right now?

<Interviewer>: Yeah! What's programming to you? How does it feel to program or to hear about it?

<Participant>: It's still very abstract to me because it barely has anything to do with what I do in my free time. It's not necessarily an interest of mine, either. That's why it's also still difficult for me. But I find it interesting that when it's portrayed in films and shows, or when you tell me about how it all works, it's still a bit of a mystery. Generally, I do understand what's kind of behind it, but it's still portrayed in such an extreme light, like with things as the dark net, or cryptocurrency [laughs], it's feels like such a stereotype. But in the end it's more about using a website or dealing with programs.

<Interviewer>: Yeah, you're describing digital literacy here. The intuition to use such programs. The moment we don't have this ability, we get vulnerable to attacks by people that are really skilled in it. Can you recall any other sources from the study? Anything you found interesting? You really liked generative artistry, apparently. Maybe you played some of the games, as well, or watched some videos!

<Participant>: Yeah, I can actually recall many of the resources because I checked them all out. One in particular was this game where you play the role of a king, and then people approached you with questions, and you had to answer while keeping them happy [laughs] and I failed miserably!

[laughter]

<Participant>: That, I still remember. Then there were some video sequences, but I honestly forgot about most of them. I did watch them, and oh yeah, there were like interviews that went for over an hour. I zapped in but oh no ... I remember this one video on black people and how the system discriminates based on skin color because it

wouldn't fit the white norm.
<Interviewer>: Yeah, it was a report on Amazon and how it trains their system to automatically deal with job applications. Based on the faces of their workers, the system was trained to accept people that fit that standard, yet the trained data was devoid of people of color, which strongly preferred accepting white people. It's one of the reasons why, even though we have no idea what happens behind the curtains of such systems, we are affected by them. I also would like to commend you on the fact that you used so much pseudocode in your submissions. Can you tell me a little about that? What were your inspirations?
<Participant>: Yeah, I mostly was inspired by the resources and asked you specifically for an example. Of course, I didn't want to reproduce it, but it definitely influenced what I wanted to create.
<Interviewer>: You mention that you had problems to get really into it. Where were those initial hurdles?
<Participant>: At first, I didn't really know what my task was. At first, you had this brochure, then the code part, it was all clear. [pause] Then I read the questions and was like: Those questions are pretty big! What exactly do you want from me? [laughs]
<Interviewer>: Yeah, they were very open-ended, but I think you managed to overcome that pretty well. It all turned out great in the end. Did you have any particular strategy for your work? How was every program formed?
<Participant>: At first, I read the text, oh I wish I had it here, oh well, then I made some notes on what stuck out to me. It was generally inspired by what I was currently doing in life as well. I looked at the site, and its references. Especially with those shapes, I was pretty determined to create a whole landscape with them. I believe I sat there for a whole hour planting those trees. [laughs] It was just so much fun at this moment. I am having so much fun! And then I thought: What smart thing could I do with time?
[laughter]
<Interviewer>: Yes! And I think it was so cool! If you'd like, we can talk about it for a second. I didn't read the book, but I looked at some excerpts, and the main point of the novel was the time nihilism. That man who doesn't see time like we do, but defines it by the

changes that occur. What are your thoughts on this regard ?
<Participant>: Yeah, I don't know! I found the book very inspiring. It's crazy how, in general, I feel like every year goes by faster. Especially, the last weeks just moved past me. My mom and my grandma keep saying that it will move even faster now. But I can't even imagine that because it's already so fast. It's very interesting to me how the experiences you have such an impact on your feeling for time.
<Interviewer>: Another topic that I found very fitting, especially because I know you, is the consumerism that is implied by technology. Do you have any thoughts on how this model of capitalism and programming fit together?
<Participant>: I immediately think of advertisements, marketing. But also the emotional technology that is designed in a way that we are tempted to buy their products.
<Interviewer>: You have been navigating a small website during the study. Can you recall it?
<Participant>: Yeah.
<Interviewer>: When you entered the site, what did you do, how did it feel, were there any frustrating moments?
<Participant>: [laughs] Yeah, some. Foremost, I looked through every single site. Then, I chose what I wanted to work on, started creating, saved it, put it back in, and then it didn't work anymore. That's when I was a little frustrated. [laughs] A big inspiration for me were all the questions. What can I even use on the platform, what backgrounds were there? That was my main source of inspirations.
<Interviewer>: Did you like the backgrounds?
<Participant>: Yeah, they were cool! I had an idea for a house, but I kinda dropped it.
<Interviewer>: Was there something that you found especially fun?
[pause]
<Participant>: Yeah, the trees and the lake. [laughs]
<Interviewer>: I am so happy you liked doing that, even though the editor isn't designed for stuff like that. It's so cool to me that you were able to overcome the limitations for your program. Do you feel like with all the things we've discussed, you would be able to have that discussion with other people who are not involved?

<Participant>: Yeah, with my mom [laughs] No, but generally with my family, where I know there isn't much expertise to be arguing against. That's where I feel most confident voicing my own opinions.
<Interviewer>: Do you think you will be confronted with programming in the future?
<Participant>: I actually was in my internship. I was part of the merch team, and since I already was working on something similar in your course, I accepted. Of course, the actual computer scientist did most of the heavy lifting, but I worked with a tool called Canva where you were able to design something. That's where I used some of that knowledge. Then I had some more practice with Gimp for a local festival, where I designed some festival ribbons.
<Interviewer>: When I designed the editor, I was actually really inspired by similar software, so I'm happy you noticed that. I think it's a great entry point to work with digital interfaces. Now, if you ever went through a similar course again, is there something you wish it did better? Any additions you would like to see?
<Participant>: No, I actually found it perfect. It was very informative, yet diverse. The whole material was a lot of fun. It was really time-intensive, but you did warn me. And I also did take my time with it because I wanted and was able to. Back then [laughs] So I don't think that I would do anything differently.
<Interviewer>: That's great. Thank you so much again for your valuable input.

A.6.5 Participant Pineapple

<Interviewer>: I started the recording. May I record your voice for this study?
<Participant>: Yes.
<Interviewer>: You participated in a programming course. I would like to ask you to elaborate on how you liked the course in general.
<Participant>: I found the overall aesthetic very pleasing because it was designed in many different ways. On the one side you had a little magazine with info texts. And those were very nice to look at with images, and then it also was connected to the Online. It was the fact that you had a room for yourself to process your ideas or

notes, both on paper and even online. And then you had a variety of resources.
<Interviewer>: Did it help you to jump between the two media?
<Participant>: I remained flexible. I read a little and then started looking at the resources.
<Interviewer>: Did you work on everything at once or was spread over a longer time frame where you read something?
<Participant>: Actually, they were multiple attempts. I started, looked at the resources and was a little overwhelmed.
<Interviewer>: Why do you think were you overwhelmed? What were the causes?
<Participant>: [pause] I had the feeling most of the time that I weren't the person suited for the study. That's why the smallest unfamiliarities, like all the languages, blocked my head. Later, there were so many resources.
<Interviewer>: It was a lot at once.
<Participant>: And I couldn't really sort it all out. The small descriptions were nice, but for me personally it wasn't enough.
<Interviewer>: If you ever attended such a course in the future, would you wish for a better introduction, or a better chapter on fundamentals? That's how it sounds to me.
<Participant>: Yes, exactly. Because from the very beginning, and I think that's good, you have this social discussion, and where you find it in daily life, I understood all that. But still, real coding kept being referenced, and that's where I maybe lost the connection.
<Interviewer>: Yeah, to catch that thought, how would you describe programming?
<Participant>: Oh god, I think in the beginning questionnaire we had to answer that as well. Now it's like, after?
<Interviewer>: Yeah!
[laughs]
<Interviewer>: Yeah, how would you describe it now?
<Participant>: I mostly thought about how programming could be, but I still don't know how it actually is. I don't really have the feeling that I actually programmed.
<Interviewer>: That's good! So you were mostly interested in reading in the course, rather than doing the actual programming?

<Participant>: Yeah, now, I didn't really know how I could apply that editor. It was the same as PowerPoint or when I worked with Moodle [laughs]. Well, okay, I can do that! Adding a picture, some animations, it's fun, but it felt superfluous.
<Interviewer>: You really lacked this bridge to the feeling of actually programming.
<Participant>: Yeah.
<Interviewer>: Do you believe that more practical projects would have helped you with programming?
<Participant>: I believe so. It was in the resources, but it was a lot where I didn't know where to start. I looked for a video where someone made a basic explanation of a robot where you could understand everything step by step. It was a guide, so you could do it at least once.
<Interviewer>: Obviously, you had a lot of self-studying going on. Independent of the resources, you looked at things by yourself that could answer your questions. What was that in particular?
<Participant>: Well, entry level programming.
<Interviewer>: So, just the whole basis?
<Participant>: Yes.
<Interviewer>: Were they YouTube videos? Google?
<Participant>: Yeah, one video with that robot, I remember, just something to click through. I didn't really stick with it for the entirety. [laughs]
<Interviewer>: Yeah, it can be a little difficult because you get so much information on problems that you aren't necessarily trying to solve.
<Participant>: Yeah, and sometimes I just sat down, and felt like I didn't have the mental capacity to get creative at this very moment.
<Interviewer>: Yeah, I understand.
<Participant>: I just wanted to send out the results. [laughter]
<Interviewer>: Do you feel like you could ever try out a programming language?
<Participant>: Yeah, I think so.
<Interviewer>: Any ideas what you could program? Some you might need, something fun?
<Participant>: Yeah, in the beginning, something to copy. If I look at music, when I compose something, or am writing style copies, I usually take something that already exists and try to understand it. I try to comprehend the

processes someone else had in their head in the first place.
<Interviewer>: That's cool! I have all of your work here alongside your reasoning. As you look at it, how does it feel?
<Participant>: I think there are some fun ideas among them [laughs]. And still, I get all the feelings of doubt that I had when I made them.
<Interviewer>: I find it so interesting to hear about your doubts because those are some really cool submissions. Especially this one is my favorite [points to submission 3]. I feel like you embodied the essence of the chapter very well, and as you said, you copied, but in this very individual and creative way. It's more than just a love letter!
<Participant>: Yeah, I would have really liked to program this with a little machine that spits out these sentences, with the words saved in a little database.
<Interviewer>: Yeah, I can definitely imagine something like that. It sounds like a nice project to learn how to program. [pause] What submission would you choose as the face of your course? Anything you feel especially proud of? Something summarizing your experiences?
[pause] I don't think so.
<Interviewer>: [laughs] No worries, that's fine. How did your strategy look like when you designed your submissions? You already said that you had initial problems every time, but I believe you sent them all in a very brief time frame.
<Participant>: Yeah, those submissions weren't the things I tried out over the weeks, but I made them all one train ride.
[laughter]
<Interviewer>: For that, I find them very different and unique if you don't know that they were created on a train ride. They all tackle a different issue. I believe you also have many notes on all the topics? Did you use them for these submissions?
<Participant>: No, I didn't. At some point I said, screw it, I didn't even use any of the resources any mroe.
<Interviewer>: You did. [points to submission 3] [pause] Well, as an inspiration.
<Participant>: Well, yeah, the texts. And the examples that you sent me. When I saw them I knew, okay, those are more

general situations.
<Interviewer>: I understand. So of all the resources that you got, which were the most appealing to you?
<Participant>: From the examples?
<Interviewer>: In general. You had texts from the zine, some videos, more texts, games, was there anything you found most cool or helpful?
<Participant>: [pause] The games were fun, even aside from the course. Yeah, but how I could turn them into something new was difficult for me.
<Interviewer>: Do you remember, if you can recollect some of the chapters, any topics that you found interesting or would want to go in greater detail with?
<Participant>: I would like to try out the more practical things. Any practical projects.
<Interviewer>: Gotcha. Of all the sources provided, were there any revelations? Something new that I didn't know before?
<Participant>: Yeah, seeing daily things through the lens of computer science, and how it is a different way to think.
[pause]
<Interviewer>: So generally speaking, is there anything in the course that you had the most fun with?
[pause]
<Participant>: Well, the reading and playing, it was all pretty fun.
<Interviewer>: So, exploring the resources – what there is?
<Participant>: For me, it was strongly separated. For me, it was on the one side, soaking in the information. But producing it was a totally different thing for me.
<Interviewer>: Yeah, am I right in the assumption that you had the most issues with the collaging? Why would I even do that? It just didn't click?
<Participant>: Yeah.
<Interviewer>: Yet again, to me your submissions are really amazing, though. They go deeply into coding.
<Participant>: Well, yeah, there, I knew what abstraction is. It's similar to linguistics. There, you also have things like feature semantics.
<Interviewer>: Oh yeah, right!
<Participant>: Where you create feature sets that are iterated in your head to recognize something. When programming, or designing, you have to do it the other

way around.
<Interviewer>: Thank you so much, you have been a great help. That concludes the interview.

A.6.6 Participant Lemon

<Interviewer>: Thank you for participating in this last interview for the study. May I record your voice?
<Participant>: Yes.
<Interviewer>: So you have been part of a course. Can you tell me how you generally enjoyed it?
<Participant>: I think it was really fun. I think it had a lot of cool resources and felt very different to what I expected a programming course to be. I was also very happy that you could make notes in the zine because I could just write there when I had an idea on the topic. But the most fun I had probably with one of the games.
<Interviewer>: I remember you were especially liking that one game. [pause]
<Participant>: Echo Beach!
<Interviewer>: Yeah, I would love to discuss that a little later. First, you might recall that question, I would like to ask you how you would describe programming. What adjectives would you use?
<Participant>: Ah, I don't remember what I wrote in that questionnaire.
<Interviewer>: Don't worry about that. How would you describe it now?
<Participant>: I think I used to find it very abstract and didn't really have a clear idea about it. But now I just find it very funny. Looking back on what I submitted, I didn't really think too much about it and just tried to make something that matched my thoughts while I looked at the resources after you helped me.
<Interviewer>: Oh yeah, I remember we had some initial problems. Why was that? Was it the open questions that I posed?
<Participant>: I don't think so. I found it very refreshing to have these big questions, but it was overwhelming at first when you had all these problems and thoughts posed. I didn't really know where to even begin or whether I should even be able to answer them all or not. After we talked, I actually understood that I can just do whatever I want, so that's what I did. Looking at the examples

you sent me, I felt more free to submit whatever, and suddenly I had a lot more fun with the course.
<Interviewer>: What topics would you say were the most interesting ones to you?
<Participant>: I really enjoyed the discussions and the gallery sites. But at some point it was just okay, I get it, and I didn't really know where to go from there, so I mostly created something that jumped to my mind while I looked at everything. I enjoyed the video on data because I never really understood what was happening there, but now I feel like I got a good idea on the problems of what is done to your data online.
<Interviewer>: Would you say you feel more equipped to talk about it to laypeople?
<Participant>: [pause] I think? Maybe not on a technical level, but I am sure I could discuss it with people who have never heard of it before. I wouldn't be able to tell them how to code, though. [laughs]
<Interviewer>: That's totally fine [laughs]. Do you see yourself looking for other practical technologies in the future? A programming language of some sorts?
<Participant>: I don't know, I think I really enjoy the graphical aspect of it, so I would rather stick to something more fun and weird rather than actually program. I definitely would like to fund out more about the different topics, though.
<Interviewer>: So since you really enjoyed the more fun parts of the course, I would love to talk about the different games you seemed to have enjoyed.
<Participant>: Yeah, I really liked Echo Beach and Kingdom game. I played it for way too long because I screwed up the first time around, and I really wanted to make those townspeople happy [laughs].
<Interviewer>: [laughs] Yeah, I remember us talking about these games a lot. What was it that captivated you with Echo Beach?
<Participant>: It was such a cool concept and I loved finding out more about those characters and listen to their music. It was so shocking when you realized that you would be able to endanger these people with just one click, when I actually really liked them. That way of putting you in the role of the person doing these terrible things was frightening.
<Interviewer>: Definitely. It was mentioned briefly in the

course. This is a great example of Desktopization where people tasked with morally questionable jobs will be put in front of a display that will not inform of the social problems at hand.
<Participant>: Yeah, I think it's crazy to think about how one click can impact the lives of many, and I am sure this reflects in many interfaces that we use. That's why this game really stuck in my head.
<Interviewer>: So, I would like to show you your submissions of the course. How does it make you feel looking at it.
<Participant>: [laughs] Some of those are just so crusty and silly. Looking at them now really makes me love what I did here. Like this one [points to submission 2]. It's just a silly meme, but I had so many thoughts on the topic, and merging these two ideas just made sense in my head. [laughs]
<Interviewer>: Yeah, I think these submissions really shows how you overcame your fear of circles [laughs].
<Participant>: Wait [pause] Oh yeah, this one was so frustrating because at first I thought I had to program this circle to move around, but the interface just did not work for me, so I decided to just stick to this image.
<Interviewer>: Don't worry, you weren't the only person who had trouble with the editor. I feel like after this, you developed your own strategies to work on the submissions. What exactly were your strategies with that?
<Participant>: Well, I usually read a chapter and made some notes in the zine where the notes section was. I then went on the site and looked through all the resources and just started creating.
<Interviewer>: If you look at the entirety of your submissions, is there any piece that you would like to define as the face of your study – something you might feel most proud of.
<Participant>: Let me take a look [pause]. It's so difficult to choose because I love all of them. They all stand for one particular thought I had during the study, and I love their general look. They just look so raw and silly, and I am kind of proud of them [laughs]. I guess I would choose this one [points to submission 2], it's just this weird mix of memes that really stands for my whole approach to the course.
<Interviewer>: Great. We talked about your initial hurdles,

but do you remember having trouble with the editor itself ?
<Participant>: I don't remember having any trouble with that . I just really liked all the graphics and how it looked, and it all worked fine. Especially when I found the button to animate, such as the code for the ouroboros [laughs].
<Interviewer>: Was there anything else that you particularly liked and remember just now?
<Participant>: Sorting algorithms.
<Interviewer>: Sorting algorithms? Can you elaborate on what you found most interesting about them?
<Participant>: Yes, I don't remember which one I found the coolest, but I think it was Merge Sort. I think it's awesome to know how you can make everything sort, and it made me feel really good thinking about it. It just works . [laughter]
<Interviewer>: Did you research some of them?
<Participant>: Yeah, it was part of the resources. That one video on algorithms!
<Interviewer>: Oh yeah, right! I am happy you enjoyed that.
<Participant>: Yes, it was very concise and to the point and I quickly got some of it if not everything.
<Interviewer>: So to sum up this interview, I would like to ask you something speculative. Is there anything you would like to see in the future of such a course? Would you change something? What else do you desire for a course in this field?
<Participant>: Honestly, I think it's really cool as it is. I just want more, really. Of course other courses might use more linear tasks, but I feel like it's something that doesn't really exist. Maybe more resources, more specific topics, and stuff like that. I really enjoyed it .
<Interviewer>: That is a genuine relief. Thank you so much for your kind words and your participation in the study.

Bibliography

1, 2

URL https://de-de.facebook.com/business/learn. 1, 2

URL https://techdevguide.withgoogle.com/paths/new_to_cs/. 1, 2

Stacy Alaimo and Susan Hekman. *Material Feminisms*. Indiana University Press, 2008. ISBN 9780253349781. URL http://www.jstor.org/stable/j.ctt16gzgqh. 2

Antje. Making your activism accessible, Apr 2023. URL https://commonslibrary.org/making-your-activism-accessible-ggsn/. 2

Ruha Benjamin. Are robots racist? reimagining technology, equity, and the new jim code, 2019. URL https://www.dropbox.com/s/j80s8kjm63erf7/Ruha%20Benjamin%20Guest%20Lecture.mp4. 4.1.3, 4.2.2

Besart. Most popular degrees in the uk - 2023 rankings, Feb 2023. URL https://www.studying-in-uk.org/most-popular-degrees-uk/. 1

Simon Campbell. Codecademy, an early (and now profitable) pioneer of coding education, raises $40m in new funding. https://www.edsurge.com/news/2021-02-23-codecademy-an-early-and-now-profitable-pioneer-of-coding-education-raises-40m-in-new-funding, 2021. Accessed: 2022-12-29. 2

Victoria Clarke and Virginia Braun. Thematic analysis: a practical guide. *Thematic Analysis*, pages 1–100, 2021. 3, 3.5, 4.2

Leaf Corcoran. itch.io. https://itch.io/, 2013. Accessed: 2022-10-25. 3.4.2

Francesca da Rimini, Josephine Starrs, Julianne Pierce, and Virginia Barratt. Cyberfeminist manifesto for the 21st century. https://vnsmatrix.net/projects/the-cyberfeminist-manifesto-for-the-21st-century, 1991. Accessed: 2022-11-29. 2

Elizabeth de Freitas and Nathalie Sinclair. New materialist ontologies in mathematics education: The body in/of mathematics - educational studies in mathematics, Jan 2013. URL https://link.springer.com/article/10.1007/s10649-012-9465-z. 2, 3.1.2

Tomi Slotte Dufva. *Creative Coding as Compost(ing)*, pages 269–283. Springer International Publishing, Cham, 2021. ISBN 978-3-030-73770-2. doi: 10.1007/978-3-030-73770-2_16. URL https://doi.org/10.1007/978-3-030-73770-2_16. 2, 4.1.3

Michelle Goldberg. What is a woman?, Jul 2014. URL https://www.newyorker.com/magazine/2014/08/04/woman-2. 3.1

University Harvard. URL https://pll.harvard.edu/subject/programming. 2

Bruno Imbrizi. Online course - creative coding: Making visuals with javascript (bruno imbrizi), 2021. URL https://www.domestika.org/en/courses/2729-creative-coding-making-visuals-with-javascript. 2

Aaron D. Knochel and Ryan M. Patton. If art education then critical digital making: Computational thinking and creative code. *Studies in Art Education*, 57(1):21–38, 2015. doi: 10.1080/00393541.2015.11666280. URL https://doi.org/10.1080/00393541.2015.11666280. 2

Jill Lepore and Andrew Lippman. Mltalks: How data killed facts, 2020. URL https://www.media.mit.edu/videos/ml-talks-2018-04-23/. 4.1.3

Olia Lialina. Turing complete user. *Contemporary Home Computing*, 14, 2012. 2, 3.3.2, 4.1.3, 4.1.3, 4.2.2

Rebecca Ann Lind and Colleen Salo. The framing of feminists and feminism in news and public affairs programs in u.s. electronic media. *Journal of Communication*, 52:211–228, 2002. 3.1

Ingrid Lunden. Sololearn raises $24m for its bite-sized, duolingo-like mobile-first coding education app. https://techcrunch.com/2021/07/21/sololearn-raises-24m-for-its-bite-sized-duolingo-like-mobile-first-coding-education-app, 2021. Accessed: 2022-12-29. 2

C Dianne Martin and Elaine Yale Weltz. From awareness to action: Integrating ethics and social responsibility into the computer science curriculum. *Acm Sigcas Computers and Society*, 29(2):6–14, 1999. 2

Victoria Masterson. These are the degrees that will earn you the most money when you graduate - and the ones that won't, Oct 2021. URL https://www.weforum.org/agenda/2021/10/stem-degrees-most-valuable/. 1

Tuuli Mattelmäki. Applying probes–from inspirational notes to collaborative insights. *CoDesign*, 1(2):83–102, 2005. doi: 10.1080/15719880500135821. URL `https://doi.org/10.1080/15719880500135821`. 3.1.2

Eric Matthes. *Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction to Programming*. No Starch Press, kindle edition edition, 5 2019. 2

Kylie Peppler and Yasmin Kafai. Creative coding: Programming for personal expression. *Retrieved August*, 30(2008):314, 2005. 2, 4.1.3

Jose Portilla. Python bootcamps: Learn python programming and code training - udemy, Mar 2021. URL `https://www.udemy.com/course/complete-python-bootcamp/`. 2

Tim Rodenbröker. trcc: Tim rodenbröker courses, 2019. URL `https://timrodenbroeker.de/courses/`. 2

Mindy Seu. Cyberfeminism index. `https://cyberfeminismindex.com/`. Accessed: 2022-10-25. 3.4.2

Winnie Soon and Geoff Cox. *Aesthetic programming: A handbook of software studies*. Open Humanities Press, 2020. 1, 2, 3, 3.3.1, 3.3.3, 4.2.2, 5.1.1

Marlene Stoll, Martin Kerwer, Klaus Lieb, and Anita Chasiotis. Plain language summaries: A systematic review of theory, guidelines and empirical research. *Plos one*, 17(6):e0268789, 2022. 2

Bjarne Stroustrup. *Programming: Principles and Practice Using C++*. Addison-Wesley Professional, paperback edition, 5 2014. 2

Rana Tassabehji, Nancy Harding, Hugh Lee, and Carine Dominguez-Péry. From female computers to male computrs: Or why there are so few women writing algorithms and developing software. *Human Relations*, 74:001872672091472, 03 2020. doi: 10.1177/0018726720914723. 2

Annette Vee. *Coding literacy: How computer programming is changing writing*. Mit Press, 2017. 3.3.2

Kalindi Vora, Sarah McCullough, and Sara Giordano. Asking different questions in stem research: Feminist sts approaches to stem pedagogy. *ADVANCE Journal*, 3 (1), 5 2022. doi: 10.5399/osu/ADVJRNL.3.1.10. 2, 4.1.3

Ben Williamson. Political computational thinking: policy networks, digital governance and 'learning to code'. *Critical Policy Studies*, 10(1):39–58, 2016. doi: 10.1080/19460171.2015.1052003. URL `https://doi.org/10.1080/19460171.2015.1052003`. 2

Lillian Xiao. Creative coding with p5.js: Examples and project inspiration, 2021. URL `https://www.codecademy.com/resources/blog/creative-coding-p5js-examples/`. 2

Audrey Tang Yuval Noah Harari and Puja Ohlhaver. To be or not to be hacked? the future of democracy, work, and identity, 2020. URL `https://www.youtube.com/watch?v=tRVEY95cI0o`. 4.1.3, 4.1.3

Saadia Zahidi. The future of jobs report 2023, Apr 2023. URL `https://www.weforum.org/reports/the-future-of-jobs-report-2023/in-full`. 1