Introduction

In September 2017, Zeus enrolled in Clearwater Academy in hopes to be among the 20,000 people who would graduate from a computer code bootcamp that year. He bought into an uncertain Cold War-era promise (Guzdial, 2021) that computer scientists and private industry had resurrected in 2011 amidst a so-called software crisis: anyone who learns to code in less than a year at one of these intensive training programs can quickly get an entry-level job in software development. You don't have to spend four years in college and accrue enormous amounts of student loan debt. The son of a Cuban father and a mother of Spanish descent (I'll explain participants' racial identities and justification for inclusion later in this introduction), Zeus described himself as "mulatto—too light for the Blacks and too dark for the whites." He was five-foot-nine and weighed two hundred pounds. During the day, when he wasn't learning web development and employability skills at Clearwater Academy, Zeus taught children mixed martial arts at his mentor's dojo. At night, Zeus put his fighting skills to use as a bouncer at a Wild West-themed bar popular among white undergraduates. It was the only job where he could legally beat up white supremacy. But that bar wasn't a good match for his temperament: you fit in by being in college and keeping up with college students' fads. Recalling that fraught time going to Clearwater Academy, Zeus explained, "A big thing they did was like, someone comes up to you, and they have like an ice, and you're supposed to get down on one knee, and you're supposed to just kill it. And like, I didn't fit in with that, you know?" Icing is a drinking prank that began as an internet meme in 2010. Someone hides a bottle of Smirnoff Ice vodka in a place where one doesn't expect to find it and would be embarrassed if they got caught with it, like during work. The person who finds the ice must point the bottle in the air triumphantly, get down on one knee, and drink the entire bottle. If you don't, you lose your reputation as a "bro."

Zeus graduated from Clearwater Academy in December 2017. With web design added to his repertoire of literacy practices, Zeus could at least get away from the ice pranks, seek better job opportunities, and get a little social mobility. Instead, he threw together a hodgepodge of technological and physical labor to make ends meet: Zeus worked for a landscaping company while practicing graphic design for a local agency and volunteering for the Technology Education and Learning Support (TEALS) Program, Microsoft's high school computer science outreach project. Landscaping was a return to a life he once knew before Clearwater Academy, but the graphic design and computer science work took Zeus' literacy practices to another level. Under the mentorship of an experienced graphic designer, Zeus learned how to create marketing materials like brochures, policy booklets, banners, tablecloths, and car wraps using Adobe's design suite software. His design work was terrible, Zeus admitted, but over time his mentor required fewer and fewer design drafts, a testament to his improving skills. The job wasn't front end web design, but it still built on his experience with art and drawing, something Zeus had been practicing since he was a kid. Meanwhile, mentoring high school kids in computer science exposed Zeus to the limits of his own computer programming knowledge. He taught in "nice fancy top buildings" which housed mostly white students. They worked with Scratch, a platform for teaching computer programming using visual blocks on a screen, and then later they learned the basics of Python. "But yeah, these kids were geniuses," Zeus thought. "They were smarter than me!" He felt like he was back in Clearwater Academy, struggling to keep up and get the coding into his head. Each day Zeus spent his time adapting to their needs, "showing them, trying to test these things out, see if they liked it, see what best fit them."

Zeus' true dream job was serving in the United States Army. Growing up, he loved the 1998 film *Glory*. Starring Morgan Freeman and Denzel Washington, the movie told the story of the 54th Massachusetts Infantry Regiment, one of the first all-Black Union military units that fought in the United States Civil War. The patriotism and heroics of Black Civil War soldiers inspired Zeus for years. But he thought his previous life selling drugs and getting in trouble with the law made him ineligible for military service until one day he ran into an old friend who was an Army recruiter. Zeus learned that if his crime wasn't a felony and seven years had passed, he could join the Army. He took the opportunity, and by September 2018, nine months after graduating from Clearwater Academy, Zeus was on his way to basic training in Georgia. After that, the Army assigned him to a base in El Paso, Texas, along the U.S.-Mexico border. He labored as an infantryman first and then moved to tanker. The armed tank battalion did regular drills where Zeus shot and loaded a lightweight machine gun called M240 Lima. Drills involved more than just shooting bullets; he had to pay attention to terrain, enemy movements, left-flank and right-flank tanks, and the infantry themselves, if they joined the drill.

Despite serving in the Army, the influence of coding continued. He felt the pressure of needing to learn as much as he could about computers, because Zeus couldn't anticipant when his Army service would end. He had to know about computers, to satisfy an insatiable hunger for learning about technology. It was imperative that he still spend a few minutes or hours reading about back-end programming, to practice coding exercises online using his phone and a little portable keyboard. He wasn't dreaming of a long illustrious career in the Army, after all; Zeus traded his body for "benefits, education, computers." He enrolled in online classes with American Military University. Between orders from his superiors, Zeus would sit in his tank in 105 -135 degrees Fahrenheit reading about Network Plus, an exam for certification in information technology infrastructure.

His relationships with other soldiers would eventually take him out of the tank and into Army base offices. Zeus learned a lot about how human beings are just as connected along networks as computers. But more complicated. In the civilian world, biases, prejudices, and oppression against marginalized social identities take front row; however, in the military "the value, you know, of a person is really understood, … like, you know, the nation's defense comes first of everything … we're the first line of defense." In that system, racial identities flatten or become less a basis for judgement. During basic training in Georgia, Zeus' unit had to stand in freezing rain; Zeus started to get closer to his comrade to keep warm, but Zeus only irritated him. "And he was like, 'Man, get the fuck away from me. I'm not gay!'" Zeus recalled. "I was like, I looked at him like, 'You crazy? You crazy? You crazy? You crazy? You on the hard coals or something?'" Eventually he relented and stood closer to Zeus. Looking back, Zeus figured, "We could have been lovers. Like, we could have confused him. Like, for real, like, honestly, like, that's how cold it is. But like, yeah, you have to get rid of all mannerisms." What rises to the top is what you can trade with your comrades, the "valuable information and things that they find dear to them, you know?" So, Zeus could make friends with people from Lagos or witness a Chinese American make lo mien for Mexican friends who make churros for Black friends who teach white people how to play basketball.

Trading your value can also reveal the selfishness of comrades. As squad leader in the infantry, Zeus drove an eighteen-year-old junior soldier 45 minutes away from the El Paso base out past some mountains to meet a Latina. Zeus had a bad feeling about the idea, but the junior soldier offered money and his own vehicle. Not long after dropping him off, Zeus got a call from the junior soldier. "He called me and said, you know, like, 'Hey, bro, pick me up.' … Latinas live with their parents, you know, like in traditional family. So her parents came home and like, he was like, 'Pick me up, bro. They're gonna kill me,' you know, type stuff." Zeus was driving at night on unfamiliar roads, so he drove the speed limit as he turned back. At a green light, Zeus turned right but the driver behind him was over the speed limit and rear-ended Zeus. The car tailspinned and crashed. Zeus left the accident with a concussion and a banged-up body; in return, the junior soldier accused him of stealing the car. Zeus understood the point of the betrayal, because "People are out for their best interest when they're tired out there and that goes to show like a lot of people's lifelines is just their vehicle when they're on a base because you're getting the heat. You can get called up for any reason. You want to have a reason to save yourself some time." The COVID-19 pandemic saved him from being discharged. The deadly respiratory virus disrupted all military objectives; everyone had to defend themselves from disease and support each other emotionally and mentally. Zeus proved his innocence while he recovered with regular speech pathologist appointments and pain killers.

The accident helped Zeus take another turn in his digital literacy practices since graduating from Clearwater Academy. The Army thought he didn't get along others, so they sent Zeus to work in base headquarters. All that time reading in hot stuffy tanks was finally going to pay off. Zeus worked on hardware like antennas and radios so tanks, administrators, and soldiers on the ground had consistent communication. Zeus also had close ties with high-ranking military officers. He drove the captain's jeep and communicated with him or communicated with others on his behalf through email, text messages, and phone calls. Less time in the heat and more time in the air conditioning of an office, Zeus hadn't landed the entry-level front end developer position Clearwater Academy had intended, but he did find information technology.

What does Zeus' story have to do with coding literacy? Using literacy for workforce development and a pathway toward social mobility has persisted for decades. President Lyndon B. Johnson, for example, made adult literacy education a national priority in 1966, the same year composition and rhetoric scholars set a new research agenda for learning what students do with writing. However, then, as now, the priority relies on skills gap rhetoric as a significant threat to the nation. This rhetoric suggests that a significant number of adults have a skills deficit, and they need resources to learn the basic mechanics of reading and writing (Bannon, 2016; Jacobson, 2016). Literacy opens doors to job training (there is no work without reading and writing). Computer programming is just the latest type of writing pulled into the rhetorical campaign politicians (Zinshteyn, 2016) and tech CEOs (CNBC Television, 2019) have used to get more people into computer science. Getting more people to labor in designing our digital environments, they argue, strengthens the economy, produces innovative technologies, and begins to diversify an industry dominated by white men. But Zeus barely did any computer programming on the job after finishing Clearwater Academy. Looking back on those short months at Clearwater Academy, Zeus admitted he wasn't good at coding anyway. In a 2023 follow-up interview, Zeus would describe the practice in this way: "[C]licking, you know, just trying things out that sounded like elementary, you know, and getting a bad, like no result for like hours, days at hand …"

Zeus had expressed what I call Black coding Discourse, his own language to describe coding literacy's relevance to his life and he to it. He developed this Discourse about coding literacy from his time creating, navigating, and then reflecting on a vast ecosystem of experiences, materials, and people he had encountered. From doing coding literacy at Clearwater Academy and even through the TEALS program, Zeus learned that he could not learn coding well, and he could not get an entry-level job. Coding was like wading through a swamp, not getting anywhere productive, just sinking, and sinking. Instead, Zeus discovered new ways of thinking and being in the wider technological economy as a racially marginalized person. Learning coding at Clearwater Academy fueled his interest in working in information technology and that provided the gratification he always needed from literacy and work. Getting his hands on the hardware of digital technology suited him; Zeus could better imagine how people put concepts, ideas, and theories into the objects of computers for distribution along internet cables under the ocean and via satellites from space. The physical engineering of computers made more sense to him. Zeus meandered from circumstance to circumstance, no longer feeling the pressure of the economy to become more literate through the tech pipeline. By, in the words of Miles Morales, doing his own thing (Dos Santos et al., 2023), Zeus' journey to join the ranks of information technology specialists in the Army led to a more affirming sense of being and purpose. Zeus' story solidifies coding literacy as a complex, nuanced social and material activity that represents multiple "figured worlds" (Gee, 2011, pp. 69–72) and ways of being in a community. More important, he represents assets-based framing of so-called failure for Black and other racially marginalized people. In this book I take a closer look at what happens to Black people inside an already leaky tech pipeline and make visible the knowledge and processes Black adult learners bring to computer code bootcamps and develop further throughout the learning process.

The following question animates this study: To what extent do computer code bootcamps achieve their goal to raise low-income Black people's social mobility and diversify the tech profession? I address this question by reporting on a year-long ethnographic study of low-income Black adult learners learning web design at Clearwater Academy to address racial inequalities in their lives. This study took place throughout 2017 and then the year following into the summer with follow-up interviews. One follow-up interview with Zeus took place five years later in 2023. In this study, I find that despite claims to color-evasive education, coding literacy education for work is credentialed and distributed in a racial hierarchy that valorizes whiteness. Participants' accounts make visible how learning computer programming depends on and are shaped by racial and economic ideologies about literacy in ways not discussed in public discourse on the wonders of computer code bootcamps. While the promise that Black adult learners can use computer programming for upward social mobility ultimately does not deliver, I show that Black adult learners create Black tech ecosystems and Black coding Discourse from their encounters with the practices, traditions, and beliefs of software development that center whiteness. In these ecosystems, often adaptative to each participants' individual circumstances, Black adult learners invest in each other's well-being and coding literacy learning. This mutual investment helps them develop a variety of knowledges, practices, and frameworks for navigating a computer code bootcamp and the software developer profession. From these interactions with coding literacy practices and its contexts, participants speak Black coding Discourse, which articulates their relationships with computer programming and its place in their literacy repertories. Black coding Discourse and Black tech ecosystems account for the material, social, cultural, and linguistic resources that should rise "above the fold" of conventional computer code bootcamp narratives and challenge definitions of success.

Black tech ecosystems and Black coding Discourse tie down overhype or mythical pronouncements that new emerging technologies will bring utopias by examining how they work in the lives of Black people and what they believe about those technologies. Journalist Laurie Penny observed that "the wants and needs of young, healthy, middle-class people with connections and a reasonable amount of spare cash" are overrepresented in Silicon Valley's user base; their problems can be solved easily, but "Structural social injustice and systemic racism are harder to tackle" (Penny, 2014). When technologies work for the people they were designed for, they seem amazing, useful, unproblematic. Pain points – short-term granular issues that impact users' experience with a product – takes priority for designers. However, they do not consider the long-term societal harms of their designs (Schmidt, 2022). In this case, the users who the designers were built for possess the political, cultural, and financial resources to protect themselves. We should look closely at the disadvantaged people who were elided during the technological design process, the ones without the means to protect themselves when technology goes awry.

This book makes contributions to three major areas: Black technoculture studies, adult education, and labor. Black tech ecosystems and Black coding Discourse describes the beliefs and practices of coding within Black technoculture. André Brock explains that "Technoculture is the set of relations between, and politics of, culture and technology" and Black technoculture specifically "incorporate[s] the materiality, temporality, and meaning-making capacities of the Black digital and its practitioners as a technological mediation of the Black 'post-present'" (Brock, 2020a, pp. 7–8). Black technoculture elides deficit thinking about Black digital practice and theorizes the joy of these practices. For Brock, Black cyberculture is an essential part of Black technoculture, focusing on the everyday, banal interactions of Black users such as on Black Twitter. Brock explains that research on technology often stop at artifacts and practice but "cultural beliefs about technology are assumed to be universally held or limited to the use-case, often leaving the technologies bereft of critical scrutiny" (Brock, 2020a, p. 10). Cultural beliefs about technology reveals "how people visualize themselves (or others!) and their computer-mediated actions" (Brock, 2020a, p. 10). While I don't take up all the nuances of his matrix for Black technoculture (Brock, 2020b, pp. 227–228), I find that cultural belief and Black digital practice overlaps with understanding literacy as an ideological social practice. Investigating the linguistic and multimodal practices in mundane local contexts and even following them across physical and digital borders as materials, reveal beliefs, traditions, and values. Learning coding literacy for work may miss these beliefs and prioritize interests of institutional whiteness and patriarchy. This book explores how Black tech ecosystems and Black coding Discourse describe the digital cultural beliefs hidden underneath those priorities by closely attending to the circulation of language and action inside and outside a computer code bootcamp. As Black cyberculture investigates mundane interactions on digital platforms, this book approaches Black people's relationship with and use of computer programming as another aspect of Black technoculture.

My study also contributes to scholarship that investigates the lived realities and outcomes of community-based adult education. Jeffrey Grabill argues that schools, workplaces, and community literacy organizations overlap; they often work together in defining what literacy activities matter for people. He writes that "any attempt to understand literate practices without understanding the institutions that make certain practices possible and valuable fails to account for how and why literacy practices look the way they do" (Grabill, 2001, p. 7). "Communities and institutions," Grabill (2001) later writes, "are interrelated and constructed" (p. 87). In alignment with this thinking, other scholars—some of them outside literacy studies—theorize the link between a local community's digital literacy practice and the institutions that sponsor them (Brandt, 2001). From her study on low-income women developing their digital literacies in a nonprofit, Virginia Eubanks (2011) advocates for critical technological citizenship. She writes that learning technology for work benefits people but "it simply cannot produce the kind of critical consciousness we all need in order to imagine alternatives to the status quo" (Eubanks, 2011, p. 154). Critical technological citizenship critiques how technologies shape civic participation, especially from the perspective of those marginalized people most impacted by these technologies. There are few empirical studies on computer code bootcamps, but three notable projects come to mind: Ashley Rae's (2022) study on three computer code bootcamps and workshops for marginalized communities; Kate Miltner's (2019) qualitative work on a coding school that reinforces inequality through several gatekeeping methods;

and a small-scale interview with computer code bootcamp graduates by Kyle Thayer and Amy Ko (2017). Each reveal empirically the lives of coders during and after their time in computer code bootcamps. France Winddance Twine (2022) devotes a chapter to all-women computer code bootcamps. Her analysis on college-degree holding white women attempting to shift from non-tech careers into coding reveal that their pre-existing capital and digital literacies helped them easily shift into tech. Clive Thompson (2019) also shares a chapter on blue collar workers taking up computer code bootcamps in response to the local coal industry's decline.

The scholarship above address computer code bootcamps from technical and professional communication, communication studies, computer science, and journalism. A social view of coding literacy and race adds to this impressive body of work by highlighting Black people's experiences in a regional labor market. Labor and race in Silicon Valley have been extensively explored (Zlolniski, 2006; Pellow & Sun-Hee Park, 2002; Gray, 2019; Meehan, 2021; Park, 1999), but, as I write in Chapter 4, the hiring and workplace literacy practices of Silicon Valley companies resonate with smaller tech companies across the country and globe (Takhteyev, 2012). This book highlights the need to trace these influences in less known markets for a more realistic view of coding literacy, labor, and race for Black people with computer code bootcamps as their only adult education. In addition, I apply Grabill's argument that to change meaning and value of literacy, we must change how different institutions relate to one another. Black participant's discourse about the value of computer programming in their lives and the subsequent ecosystems that it creates suggest possibilities for how computer code bootcamps and the workplace interact. One possibility is that they mutually prioritize Black lived experiences and Black technical knowledge in curricula and workplace practices.

Black Tech Ecosystems reveals how social mobility as a disparate outcome of learning coding literacy results from a complicated relationship among literacy, labor, and race. Clearwater Academy distinguishes itself from other educational institutions for computer programming like K-12 schools, universities and colleges, and extracurricular programs: they are a worksite, so their adult learners work fulltime (35 – 40 hours a week) except without benefits and without pay. Black adult learners in Clearwater Academy do not learn computer programming as conventionally understood in educational settings; they learn *how to labor with computer programming* according to multiple social and cultural dimensions in the tech industry. They labor for Clearwater Academy and the tech sponsors hoping to extract their talent. From a critical race perspective, I suggest that narrowing coding literacy to adult education for work maps the raciolinguistic belief that certain practices with reading and writing produces good workers onto computer programming. Coding literacy gets subscribed into a white supremacy agenda, making it prestigious or unique in digital literacies for some racial identities but not others. Under those conditions, Black participants rethink their digital literacy and coding literacy as tools and practices for redefining what labor looks like for themselves. I contend in this book that Black participants find liberation from barriers to laboring with computer programming and the typical narratives of what counts as Black labor in the tech industry. This calculated look at the unique circumstances of low-income Black adult learners attending a computer code bootcamp shows how computer programming, as an emerging literacy, might facilitate a more dispersed set of opportunities that is deeply connected to the specific needs of Black people rather than the economic and diversity needs of tech in the United States. While policymakers wring their hands over how to get Black people into working for tech, this book considers how Black adult learners labor in a computer code bootcamp and reconceptualize labor after they graduate. Across four chapters I show how Black coding Discourse and Black tech ecosystems lead to these significant revelations for Black adult learners.

New Literacies; Same Old Story

To appreciate the theories advanced in this book, I describe the significance of computer code bootcamps to histories of Black labor and literacy in the United States. First, I establish computer code bootcamps as a logical response to changing economic needs; theories on literacy as an economic resource and coding literacy as a type of writing solidifies why computer programming draws my interest as a literacy studies scholar. Without their knowledge, advocates for coding literacy—such as Code.org—rely on familiar workforce development rhetoric to make their literacy campaign a national effort that is everyone's responsibility—teachers, parents, politicians, venture capitalists, and tech companies. The coding movement also suggests that *everyone* really can learn to code, so it targets both children and adults as potential workers. I later describe theoretical concepts from literacy studies, history, education, and sociology to help me highlight the continuous struggle Black communities have with the narrative that more literacy, or more education, will close wealth gaps while bringing diversity to technological designs. This book builds on, takes inspiration from, and ultimately diverges away from this knowledge to understand the role coding literacy plays in racial capitalism (the idea that capitalism thrives off the labor of racially marginalized people) and the knowledge economy and what downstream impact they have on Black lives.

Kids, Coding Makes You Powerful ... And Marketable

The modern coding movement began in 2009 when the United States federal government declared the second week of December Computer Science Education Week. Microsoft also called for a national plan for computer science education; the company's software engineer Kevin Wang founded the Technology Education and Learning Support (TEALS) program in the same year. These are kernels of the beginning, however. The movement accelerated after Hadi Partovi and Ali Partovi, twin brother venture capitalists who had been early investors in Facebook and Airbnb (Hadi) and Zappos and Dropbox (Ali), founded Code.org in 2012. They offered a new literacy campaign that built on the National Science Foundation, Congress, and Microsoft's goals and outcomes. They agreed that everyone can and should learn to code, and getting computer science curricula in more public schools can make that happen. The website offers a library of free tools and resources for kids to learn computer programming but in 2012 Code.org was also a platform for this iteration of the movement. The nonprofit leveraged the power of social media and celebrity. In 2013, Code.org released "What Most Schools Don't Teach" which features stories of known and less-known influencers in the tech industry on how they got into coding: Mark Zuckerberg, Bill Gates, Jack Dorsey, and other founders of popular digital platforms. The video next switches to commentary by music celebrity will.i.am and professional basketball player Chris Bosh on how computer programming is a necessary skill in the 21st century. Although the impetus for Code.org is to get kids coding (Singer, 2017), as declared in the opening minutes of the video, the narrative shifts as the video reaches the conclusion: working for Big Tech companies, replete with images of open concept offices, pool tables, and food from the best executive chefs around, is the goal (Code.org, 2013). Coding is fun but, when you work for software companies, you can make money while having fun.

That viral video, and Code.org, would have significant impact on the state of computer science education in public schools. In 2022, Code.org reported that fifty-three percent of U.S. public schools offer foundational courses in computer science, and thirty-seven states had adopted at least five policy recommendations that were jointly created by The Code.org Advocacy Coalition, Computer Science Teachers Association, and the Expanding Computing Education Pathways Alliance. But what powers those guidelines may be the tech industry itself: As of 2023, Code.org lists Microsoft and Amazon as Platinum Supporters (each donating $3,000,000 or more) and Coinbase and Google as Gold Supporters (donating between $1,000,000 and $2,999,999). Google would follow Microsoft and create their own computer science curricula for public schools, and Apple delivers a curriculum for public schools in Swift, the computer programming language that powers the company's iOS operating system. Apple does not donate to Code.org. Other non-profit organizations provide extracurricular programs for youth, especially Black youth, such as #YesWeCode (now called Dream Corps TECH). These programs extend their reach and longevity with tech company partnerships such as the collaboration between Kimberly Bryant's Black Girls Code and Google. Some speculate, and rightly so, that these companies aren't just investing in computer science education for children out of kindness, but rather they seek to be literacy sponsors (Brandt, 2001) of future workers.

The viral video's shift to sleek offices, sumptuous food, and recreational rooms makes a stronger appeal for coding labor and reflects the reality of the economy's so-called dire need for software developers. Digital technologies are so essential in our everyday lives that the U.S. Bureau of Labor Statistics predicts that software developer employment will grow 22 percent between 2020 and 2029, more than any other occupation's average. At the same time the workforce remains majority white and male. The top five tech companies in the world—Meta, Amazon, Apple, Netflix, and Google—have been criticized for this lack of diverse workforce. Despite their efforts to recruit and retain more Black coders since 2014, little progress seems to have been made across all these companies. Partnering with the NAACP, the Kapor Center reports that between 2014 and 2021, Black representation in the tech workforce increased just one percent with 3.7 percent of technical roles held by Black people (The Kapor Center & NAACP, 2022). Of course, coding for empowerment alone doesn't put food on the table; the one percent owns these companies. They may not *intentionally* seek workers, but these programs nevertheless create pipelines to meet demand. Getting workers out of public schools would take years, and even then, not enough racially marginalized people will continue down the so-called tech pipeline.

However, extracurricular coding programs are factories for extracting ideas from youth without compensating them for their labor. For example, across the United States K-12 students participate in weekend-long competitions coding mobile apps and websites. Winners in these hackathons may get prize money or free software. Corporate sponsors appear at these events giving the ethos that the sudents' skills and ideas matters in the knowledge economy and supplying them with social and emotional capital. If a team does well enough, they could pitch their idea to investors at the hackathon. But there's an air of advertising at these hackathons: young coders use corporate sponsors' technologies to create their apps, and these competitors could market these companies to their peers. Under a "fictional expectation" that these coders will one day make it big, tech sponsors get to "outsource work and crowdsource innovation" (Zukin & Papadantonakis, 2017, p. 177) for cheap, even if most ideas don't become reality. Popular media also perpetuates coding literacy as child labor. Black Girls Code and Girls Who Code position girl coders as learning computer programming for fun while raising up a generation that will one day correct the ongoing whiteness and cisgender patriarchy of computer science and the tech industry. However, Nickelodeon's television show *Game Shakers* (2015 – 2019) presents white girl coders as talented workers prepared to make money as children in the present, *not as future working adults*. A caveat here, however, is that *Game Shakers*' main characters, Babe and Kenzie, find success partly through the "childish" behavior of their business partner Double G, a Black adult creative director. His behavior contrasts from the more serious and creative attitudes of the show's white teens, suggesting that "the creative white coding girl as the ideal creative subject and erases the possibility of the Black coding girl" (Knotts, 2022, p. 127). In both real and imagined media, coding for youth means coding for work and labor exploitation.

Computer Code Bootcamps: We Will Get You to Work

The coding movement also reaches out to adults who never had the opportunity to learn computer science in childhood but may have interest in joining the ranks of software developers and other tech-related jobs. Computer code bootcamps provide an alternative pipeline to diversifying tech at a faster rate than traditional learning and labor recruitment through universities; these programs teach adults computer programming within a year at a relatively low cost. While in 2012 the industry often offered training in front-end design and backend programming, computer code bootcamps have adapted to the labor market by providing other in-demand skills such as in cybersecurity, machine learning, algorithms, cloud computing, and game development. Computer programming remains the most in-demand with 46 percent of bootcamps covering the industry (Juberg et al., 2023). Required social distancing in response to the COVID-19 pandemic increased online course offerings and they have not slowed down since 2023. Whatever programming language or software learned for whatever purpose, what awaits graduates are lucrative entry-level jobs.

There are several resources that track the computer code bootcamp industry, all of which use different methodologies. I cite two well-known sources—Course Report and Credit Karma—to show that regardless of the method, they seem to consistently point in the same direction: computer code bootcamps, and bootcamps that train adults in other tech skills, continue to be popular. In 2013, just thirty computer code bootcamps existed, and despite a spate of significant closings such as Dev Bootcamp and some questioning the worth of other computer code bootcamps, Course Report lists 500 bootcamps in the United States and Canada in its 2020 report. In 2019, computer code bootcamps graduated over 23,000 adult learners with eighty-three percent of surveyed alumni finding work in tech. The average worker, according to Course Report, is a thirty-one-year-old bachelor's degree graduate with seven years of work experience in their back pocket and no computer programming experience. Six percent of graduates are Black or African American, just one percent less than the percentage of Black coders in the U.S. tech industry (Eggleston, 2021). In 2023, Credit Karma's data gathered from LinkedIn showed that 115,673 people graduated from computer code bootcamps between 2021 and 2022. To understand the significance of graduate numbers, Credit Karma reports on years from 2012 – 2022, showing that nearly 300,000 people report on LinkedIn alone being a graduate of a computer code bootcamp. The revenues continue to grow for computer code bootcamps: In 2018, the industry generated $240 million dollars in profit. By 2022, computer code bootcamps took in nearly $730 million, suggesting that in the next few years computer code bootcamps could be on track to become a billion-dollar industry (Juberg et al., 2023).The model can pull in people looking to switch careers quickly, from non-tech work to work in tech; they extend the so-called meritocracy of software development, and argue for financial uplift for marginalized people. If marginalized people fail to add representation to tech, that's on them and not the industry's "obstacles or gaps on the path between gaining skills and getting a job" (Abbate, 2018, p. S150).

Think of computer code bootcamps as what sociologist Chris Benner (2002) calls labor market intermediaries (LMIs): organizations that match employers with workers who have the skills and competencies they need. There are three types of LMIs: private sector LMIs operate as temporary work agencies or online labor platforms (Corbel et al., 2022), connecting literate workers and the employers looking for contractors; membership-based LMIs advocate for unions; and public sector LMIs offer trainings and certifications, like stand-alone computer code bootcamps or colleges and universities that adopt the computer code bootcamp model into their adult education programs (Dudley & Rindlisbacher, 2021). LMIs had been helping workers find long-term jobs for decades until after the 1970s when the economy changed dramatically and strengthened the need for public sector LMIs. Supply chains, trade, investment, and capital flows became integrated around the world. Information technologies allowed simultaneous work, which flattened hierarchies and removed the need for middle management. Industries created new products and faster production processes to remain competitive in the global economy. These three changes in the economy since the 1990s, Benner argues, has changed the nature of labor. Silicon Valley relies on different "forms of flexible labor—networking, mobility, and nonstandard employment" to remain competitive among peers (Benner, 2002, p. 38). For example, firms need workers who can complete several tasks and change their skillset quickly according to the firm's needs (flexible work). These work tasks occur under short-term or nonstandard contracts (flexible employment) (Benner, 2002). What is known as the gig economy now, mediated by platform literacies and gig literacies (Corbel et al., 2022), has its roots in the rise of information technology systems in a global economy. This model for work in tech recalls critiques of human capital: it treats workers not as human beings but as mere task-completers for the engines that run digital ecosystems. I describe later that racism exacerbates this perspective of Black workers.

Because Silicon Valley demands speed and efficiency, gathering information on what skills a firm needs from workers can be costly and time consuming for both employers and workers. LMIs like computer code bootcamps bridge the two and help them reduce the cost of transaction and time for finding work and workers. Computer code bootcamps are canaries in the coal mine: they plug into the shifting needs of the tech industry (Bennett & Steinberg, 2022) and then communicate those needs with training or certification for unemployed people or people looking for new work. At the same time, employers and employees can use computer code bootcamps to access power to find jobs and negotiate the terms of their contracts. Public sector LMIs play a crucial role in producing "good literacy practices" so adults can become "good workers" in a changing and uncertain economy. Although the results can be mixed, LMIs in the public sector have immense value to regional labor markets (Pennell, 2007) and can help racially marginalized people find some pathways into the Fourth Industrial Revolution (Farrell et al., 2021). However, computer code bootcamps can still worsen inequality in the tech industry. Workforce development for diversity and equity often suggests to the public that the goal is to hire "permanent" employees. When data on diversity in the tech workforce considers temporary or contract work, the reverse happens: tech companies contract out more work to Black Indigenous People of Color (BIPOC) than white workers. BIPOC contract workers are the hidden labor often not accounted for in discourse on the lack of diversity in the tech industry. What continues, and what I'll delineate further in the next section, is an "occupational segregation" that undermines the argument that coding literacy offers opportunities of social mobility for marginalized people (Tech Equity Collaborative & Project Include, 2021).

The computer code bootcamp industry also traffics in what sociologist Tressie McMillan Cottom calls Lower Ed—for-profit educational institutions offering short-term credentials that thrive off socioeconomic inequality. "All of higher education benefits from inequality in some way," writes Cottom, "but only for-profit colleges exclusively, by definition, rely on persistent inequalities as a business model" (Cottom, 2017, p. 21). Similar to public sector LMIs, the rise in short-term credentials from for-profit institutions comes as a response to "structural changes in how we work, unequal group access to favorable higher education schemes, and the risk shift of job training, from states and companies to individuals and families, exclusively for profit" (Cottom, 2017, p. 11). Lower Ed institutions exist because marginalized communities cannot access higher education; Lower Ed absorbs the "breadcrumbs" that fall from the table. For-profit institutions offer themselves as a safe alternative by exploiting the narrative that a good education leads to social mobility, and they target the most marginalized of people. Cottom explains how for-profits make a profit: Once adult learners register for classes and stay in those classes for the first few weeks, the institution claims their federal or private student loans. Black adult learners enrolling in a for-profit college may already be saddled with debt or other forms of economic inequality; other Black adult learners may max out their student loans before graduating. Even those who do finish with a for-profit degree, may find jobs that do not pay enough, keeping them in debt for years. Cottom writes that Lower Ed does not "[challenge] [our faith in education'] market imperatives" and they maintain "the status quo of race, class, and gender inequalities in education and work" (Cottom, 2017, p. 12).

Cottom has already clocked computer code bootcamps as coming from the same era as for-profit colleges, targeting well-off adults who have the means to take on debt in exchange for coding literacy (Logic Magazine, 2017). Poor racially marginalized people making a similar investment in computer code bootcamps will already run through the familiar cycle of economic inequality as experienced in for-profit colleges and universities. Financial models to reduce the costly consequences of private student loans do exist for low-income racially marginalized people, such as the GI Bill, employer sponsorship, scholarships, living stipends, the Workforce Innovation and Opportunity Act, installment plans, deferred payment tuitions, and income shared agreements (ISA). However, fewer bootcamps offer ISAs these days. ISAs are contract agreements that adult learners will pay a percentage of their future income to computer code bootcamps. The deferred payment plans ask computer code bootcamp graduates to pay back tuition in monthly installments over time; and graduates know what that monthly bill will be upfront. ISAs, however, hide what the monthly bill will be. In 2021, the Consumer Financial Protection Bureau (CFPB) found computer code bootcamps mispresent ISAs as a debt-free options when in fact they are student loans (*CFPB Takes Action Against Adult learner Lender for Misleading Borrowers about Income Share Agreements*, 2021). In 2022 a bipartisan group of senators—Mark Warner, Todd Young, Marco Rubio, and Chris Coons—introduced the ISA Adult Learner Protection Act. The bill was referred to the Committee of Finance but had not been taken up for consideration again until as recently as 2023 when these senators reintroduced the bill.

Given their position in a job market that puts the responsibility of retraining on individuals, computer code bootcamps are another option for literate people to accumulate new literacies according to felt pressures of the macro-economy (Brandt, 2001). If computer programming really is the next necessary skill, coding literacy continues a long history of evolving literacy practices. Reading remained the dominant necessity for economic activity throughout the 20th century until the late 1990s when mass production gave way to a knowledge economy; driving data and information require workers to spend more time writing than reading (Brandt, 2014). Although the current coding movement is a decade old, the question of whether all workers should learn coding is still an "it-depends" answer: It depends on what you want to do with computer programming. What's the purpose and goal? To what ends? Some computer programming languages were designed for specific functions over others (languages made for front-end design may not map well into backend programming where data structure management happens). It remains unclear to what extent computer programming really matters for multiple occupations in the general workforce; coding still doesn't have the same reach as writing in print or even multimodal composing and revision on the job (Lauer & Brumberger, 2019). Occupations other than software development do not and will not adopt coding literacy despite discourse on its power and usefulness. They need logical integration into their everyday workplace practice. For example, journalists had begun using coding in their work during the 1960s when journalist Philip Meyer famously collaborated with computer scientists to analyze conviction rates in Philadelphia. Meyer (2002) would later advocate that journalists use computer assisted technology to help them analyze publicly available data for compelling storytelling. Computer programming advances these technologies from the 1970s and 1980s by integrating sophisticated data visualizations into reporting. Data journalism has become an important context for understanding coding as an intermediatory tool for structuring data. Coding isn't just a technology but also a situational and relational way of writing with other symbol systems (Lindgren, 2021).

Diane E. Bailey and Paul M. Leonardi's (2015) research on occupational technology choices demonstrate that there are no easy answers for why an occupation adopts a technology. The notion that as technologies advance, more jobs integrate them without question could not be more complicated. Bailey and Leonardi explain that it's not just about finding the right tool for the right job or adopting new technologies because of the social setting. From their extensive observations of three product engineers, Baily and Leonardi find that people with similar responsibilities use different technologies because of occupational factors: the market they work in and their knowledge. Baily and Leonardi write, "In other words, these occupational factors, whose roots are largely historical, social, and economic, not technological, nonetheless shape the role of new technologies in each occupations' work" (2015, p. 11). Occupational factors lead data journalism to adopt some computer programming languages over others (such as the usefulness of JavaScript for Ray in Lindgren's study), but other occupations may not have the factors that require coding literacy to get work done. Coding literacy's clearest pathway toward social mobility is software development, not necessarily other occupations that integrate computer programming. In this occupation, success isn't just knowing the practices of computer programming, but also navigating and performing the discourses and behaviors—the sociocultural expectations—of tech.

This study looks over the shoulders of for-profit computer code bootcamps and focuses on the nonprofit computer code bootcamps standing behind them. Clearwater Academy operates in the poverty-busting business for racially marginalized people, which is a heavy burden as the United States federal government has created financial policies like tax credits that subsidizes wealth at the expense of the poor (Desmond, 2023). Nevertheless, computer code bootcamps, and other educational institutions that invest in this model, draw on the access doctrine (Greene, 2021), a philosophy originating with the Internet boom of the 1990s that suggests access to digital technology will solve poverty. Daniel Greene (2021) writes that schools and libraries appeal to the access doctrine to receive grant funding that help them reinvent themselves into makerspaces and bootcamps. These spaces offer technologies and training that assist the poor. The access doctrine, however, is less useful for making social mobility happen for poor Black families and more attractive as a political tool. The institutions, not necessarily the patrons, benefit the most as they receive a lifeline to remain relevant in their local communities. Because software developer positions open faster than tech companies can fill, some advocates for coding notice an opportunity to finally let some of that wealth from the tech industry come to the racially marginalized people. The coding bootcamp has a simple formula for Black adult learners who answer the call to learn computer programming for work: they get jobs, tech companies get a more diverse workforce, and the United States maintains global leadership in innovative and, supposedly, more inclusive digital technologies. But the structures around computer programming deploys a process of exploiting Black knowledge and Black bodies. Coding literacy learning can continue legacies of using technology to exploit and oppress marginalized people.

I further describe this connection between Black labor and technology from a historical perspective in the following section.

Technology for Black Discipline and Exploitation

Safiya Noble (2018) argues that Google's Search Engine results reflect ideologies of race and gender. Taking inspiration from her groundbreaking work, I searched "Black labor" on Google Images on June 24, 2023, to learn how Google represents Black people working. The first results sprung onto my screen were photos of Black men in black and white prison uniforms working in fields; the other images showed marches in the Civil Rights Movement, such as the well-known I Am a Man photo taken by photographer-turned-FBI informant Ernest Withers. Google's algorithms show commonplace images of Black labor—either working in the fields or working for their freedom. This popular framing of Black people at work simplifies their contributions in the United States. Black labor practices include techné of marginality, which is

> the critical and marginal standpoint from which historically marginalized cultural groups experience the world and then engage rhetorically. When marginalized people navigate systems not designed for their inclusion, they not only apply this critical marginality to the labor that is required to circumvent, subvert, renegotiate the systems for their own survival and success, but they also leave the specialized communication and navigation infrastructures (i.e. technical communication) in place to sustain the labor moving forward. Put another way, a critical understanding of one's own marginality is a way of seeing and knowing, and therefore is a techné—a flexible, dynamic, powerful, strategic, transferrable, transformative tool that can be used to do technical communication work. (Shelton, 2019, pp. 98–99)

Although Cecilia Jackson's framework focuses on the unpaid labor of social movements for Black liberation, I'm most interested in how her theory describes Black labor in the United States as a meeting point among culture, technology, and Black technical knowledge. The distribution of Black technical knowledge, free and enslaved, brought significant wealth throughout the colonial and antebellum periods. In South Carolina, white colonizers knew little about growing rice, but their African slaves used techniques learned in the Rice Coast of West Africa to work the marshy soil into plantations (James, 2006). Black slaves' expert knowledge of land manipulation and water control helped their white masters accumulate wealth for generations from the seventeenth century until after the Civil War, when emancipation effectively ended rice plantation economies in the South (Carney, 2006). Northern colonial cities became famous because Black people cleared the land first, and then, years later, wealthy whites employed the skills and knowledge of Black slaves as "brick masons, carpenters, cabinetmakers, sailmakers, bakers, coopers, and tailors" (Trotter, 2019, p. 4). Their expertise in ironwork, furnaces, and forges made them valuable workers for building cities that sit on the land their ancestors worked (Sinclair, 2006), land that originally belonged to Indigenous people. Meanwhile, Black women covered domestic service in white master's homes; the occupation came with the twin hazards of sexual and physical abuse (Trotter, 2019).

Although white slave owners did train Black slaves in these skills in the decades that followed, they enslaved Black people because they were technologically sophisticated. But that knowledge also made them a threat to white supremacy. Before ratification of the thirteenth and fourteenth amendments, the United States Constitution categorized Black people as noncitizens, property that was ineligible for filing their own patents. Even free Black people who contributed significant inventions in maritime trade, carpentry, and mechanical engineering struggled to claim ownership of their work. The era of Reconstruction saw an explosion in filed patents from Black inventors, however. So much so that the U.S. Patent Office created an exhibit of inventions from Black people. The list of patents suggests a wide range of occupations Black people had, not just working in the fields but in barbershops, restaurants, and tailoring (James, 2006). I'm painting broad strokes of Black techné of marginality—some coming from West Africa and others created amidst slavery and Jim Crow—but the brief examples here suggest "Black technophilia" (Everett, 2009, p. 20) for and with technological advances going back centuries. White supremacy exploited that Black technical knowledge for wealth, stole Black technical knowledge for their own selfish reputation, and then restricted Black technical knowledge to perpetuate the racist idea (Kendi, 2016) that Black people are intellectually inferior (Jefferson & Forbes, 2022), and that slavery is a civilizing tool that saves Black people from themselves.

Exploitation of and restrictions on Black technical knowledge include *using* technology to discipline Black bodies. These structures kept Black people from doing mental occupations—no education or use of literacy keeps Black people in manual labor or domestic work; no occupation and no literacy means no wealth and no property ownership. In *Dark Matters: On the Surveillance of Blackness*, Simone Browne (2015) examines how race undergirds the surveillance of Black bodies. Brown examines the *Brooks* slave ship design as a type of Panopticon, an architectural structure designed to manage (Black) people using light, shadows, mirrors, walls, and limited space. Slave plantations used other technologies to monitor and discipline Black slaves: a slave pass system to restrict and monitor Black people's movements, advertisements in newspapers about escaped slaves, and, recalling Patricia Hill Collins (2009), controlling images of Black women to justify economic exploitation in domestic work. White slave masters used literacy as a technology for plantation surveillance and delivered violence when Black slaves failed to follow these literacy practices. When the defeated British Army left New York in 1783 at the end of the American Revolution, they took with them three thousand Black people designated "Loyalists" between April and November. The British Empire published a list of these escapees in the *Book of Negroes*, "the first government-issued document for state-regulated migration between the United States and Canada that explicitly linked corporal marker to the right to travel" (Browne, 2015, p. 67). The book listed, among other things, a physical description of the escaped slave—scars and other unique markers on their bodies. This accurate detail ensured that the British Empire could fairly compensate American slave owners for any slaves the Empire took. A final example: in the wake of an armed Black insurrection in New York, the city council passed lantern laws to regulate the movement of Black and American Indian bodies around the city. Any "unattended slave was mandated" (Browne, 2015) to carry a lantern; the lantern "made it possible for the black body to be constantly illuminated from dusk to dawn, made knowable, locatable, and contained within the city" (Browne, 2015, p. 79).

These methods for discipline and regulating Black bodies would cross over into computing in the early nineteenth century across the Atlantic Ocean. Charles Babbage, one of the key architects of modern computing, used plantation management systems to design technologies that would bring order to labor in capitalist societies (Whittaker, 2023). With the British Empire abolishing West Indian slavery in 1807, the government wrestled with new questions: how does capitalism, productivity, and profit carry forward without slavery? How does capitalism continue despite the worker strikes that were prevalent throughout the Empire? Babbage suggested borrowing the mechanisms and logics of plantation management to control working class whites in factories. The old methods from this management philosophy could easily port over to the rest of economy. For example, employers could keep "records on the number of workers needed to complete a task and [track]their speed, individual outputs per day and per task, the tools and implements required to complete work, and the capacities required to accomplish a given effort" (Whittaker, 2023). Management could divide and standardize labor. This method would create a set of expectations each worker had to meet while on the job; knowing their exact task helped strengthen worker surveillance already present in the record keeping strategy.

Babbage designed his famous Difference Engine to automate hand calculations and create "error-free logarithmic tables, which were crucial in astronomy and for maintaining British hegemony in maritime trade" (Pasquinelli, 2023, p. 52). His concept for this early version of the computer (Babbage would later conceive of the Analytical Engine, its successor, with Ada Lovelace) drew on labor division techniques in factories, which they had borrowed from the plantation management system. Babbage assigned each part in his invention a single task, and when brought together those parts and tasks constituted a machine. The punch cards carrying instructions came down from upper management, and the workers would insert those cards into the machine itself. Those same workers would also take responsibility for making sure the machine worked. The creators of the punch cards maintained their power: they had the knowledge necessary to operate the machine, while "the computers" were unskilled monitors of the machine. From their (literal) high vantage point, management could surveil the "computers" below to ensure productivity. Work remained regimented, consistent, and predictable and thus could be automated (Whittaker, 2023). Computing continues a legacy of plantation surveillance through its process of individual tasks (functions) working together to run software. Ultimately, Babbage took inspiration from plantation management and theorized how machines may *automate* tasks in production.

Babbage's ideas came to fruition throughout the 20th century in Britain and the United States. Gender, class, and race solidified Babbage's hierarchy for automated production. Machine work in the British Civil Service "disciplined workers in accordance with certain gendered and classed labor ideals predicated on the heteronormative concept of a male breadwinner wage and unpaid domestic work for women within the nuclear family" (Hicks, 2017, p. 22). These ideologies of gender and class shaped the kinds of work women did in Britain before and during World War II and then in peacetime when the government wanted to leverage computing to regain its lost power and influence. The British government believed data processing on electromechnical computers was low-skilled manual labor suitable for women. Realities of their labor portrayed both Babbage's and patriarchy's expectations: machines like the Colossus were incredibly complex, and data processing required as much intellectual work as the management and supervisor office work men did (machine work was thought to be subclerical, the bottom of the bottom of Britain's war effort and subsequent peace time expansion of computing). The mental and physical work of women codebreakers led to the D-Day landing on Normandy, a major turning point for the Allies. Yet women's significant contribution would be hidden in future representations. Women's machine work would be rendered "Taylorized and skill-less" in future discourse, depicting machines as working on their own while women simply "tended" to their maintenance (Hicks, 2017, p. 43).

In the United States, a competitor of Britain in the global computing market in the 20th century, software, gender, and labor had a less top-down expansion and exclusion. In the 1940s and 1950s, many firms invested in computer hardware, anticipating its revolutionary change to everyday life and work; the role of the computer programmer, however, was simple low-skilled clerical work. Software was not as valued in computing during the mid-20th century. Like in Britain, men computer engineers took over the most important steps of the programming process. Step six—"static coding", according to the authors of *Planning and Coding of Problems for an Electronic Computing Instrument*, was women's work (Ensmenger, 2010, p. 36). Follow the plan the man has made. However, the computer programmer role required more complicated thinking than anticipated, and they soon joined all steps of the software development process. New programming languages like FORTRAN, COBOL, Algol, and Unix helped solidified their needed contributions (Abbate, 2018).

However, the programmer also challenged existing hierarchies (Ensmenger, 2010). Although coding transformed into a central role by the 1960s and beyond, the initial model still echoes Babbage's vision of monitoring labor. Modern digital technology both automates tasks and surveils workers. For example, Amazon notoriously surveils workers in its highly automated wish fulfillment centers (Delfanti, 2021). Recalling Silicon Valley's labor market, digital technology design processes encourage short-term contract work for specific tasks. The place of the software programmer looks different from what Babbage conceived, and the reality of their importance is well-accepted in modern technology design. However, this historical look, though brief, shows how race, technology, and labor centers whiteness. Black adult learners learning coding literacy must buy into this ongoing legacy if they hope to succeed. Chapter 3 and 4 demonstrate how participants wrestle with using computer programming for their self-defined form of liberation and the white-centered interests of the industry. The Conclusion concedes that the tech industry, not just software development specifically, has revised expectations and practices for many kinds of people in the past, but the process for challenging these legacies and adapting to Black coders remains a steep hurdle.

The struggle between whiteness and labor as empowerment for racially marginalized people shows in 20th-century recruitment efforts. Like in the 21st century, industry professionals believed there was a shortage of computer programmers. Many corporations had come to rely on these highly skilled technology operators by the 1960s, and that necessitated more workers. Historians now question that wisdom, a skepticism we may have now, as, like the literacy crisis, there is always a software crisis. (There is another kind of software crisis, in which engineers had a problem with developing software, from getting work done on time to getting advanced technologies to complete desired tasks. In this book, I'm referring to the crisis of needing more workers.) Back in the 1960s, the desire to solve the so-called software crisis overlapped with intense scrutiny on the diversity of tech companies. In 1968, the newly formed US Equal Employment Opportunity Commission (EEOC) called in New York's top corporations to explain the lackluster effort to diversify their workforce. Most notable on that list was IBM, which was then a major distributor of computer technology to businesses across the nation (McIlwain, 2020). Four years before, IBM began the Fort Rodman Experiment, a government-supported project that turned a military base outside a Massachusetts majority white community into a technical education program. The program would train 750 Black and Brown poor high school dropouts in computer programming each year. For fourteen months, these young people from across the country would live, eat, and learn coding together in small cohorts. White college graduates, some of them having served in Peace Corps, taught the courses.

IBM was ahead of the curve but not the only one trying to train Black people. In that same year, The *New York Times* published an article on how coding could be attractive to Black people, and in 1967 the Commerce Department started a program to recruit and train Black men in coding, arguing that, race, class, or gender mattered little in tech; they could work and earn money on their merit alone (Abbate, 2018). Computer code bootcamps across the country sprung into being, as many as 700 by 1969. Computer programming was advertised as a source of empowerment in addition to social mobility. However, trade schools and electronic data programming (EDP) schools failed to deliver on these promises. In her article on the historical link between EDP schools and modern computer code bootcamps, Kate Miltner found that a "combination of fraudulent practices, inferior quality training, poor reputations, and larger structural biases" coalesced as barriers into software development for marginalized people (Miltner, 2022, p. 266).

IBM's Fort Rodman Experiment failed for several similar reasons, including deficit views of the young Black men they trained and complaints from the local white community that they didn't want Black people around their neighborhood. President Lyndon B. Johnson would fall in line with racist demands and shut the program down. However, that didn't stop IBM from trying other training programs. In the 1968 commission hearing, IBM wrote a letter to the EEOC boasting about its efforts to recruit and hire racially marginalized people. Their efforts to diversify their work force created barriers to maintain whiteness. IBM's Programmer Aptitude Test, for example, "reinforced the idea that computer programmers were born not made. Blacks who failed to pass the test were seen as unteachable" (McIlwain, 2020, p. 34). And during a 1974 speech to the company board and stockholders, CEO Frank Cary spoke highly of their diversity programs but admitted that whiteness was strong among the managers. They frequently asked Cary how they could hire more racially marginalized people without undermining racial hierarchies, with Black people being subordinate to white people. Computer code bootcamps as sources for adult education may perpetuate institutional racism; appealing to market logics without interrogating race, and other forms of inequality that the industry reflects, advances nothing for Black people and other marginalized people. Identity-based projects more readily address structural inequality and racial stereotypes, such as Dream Corps TECH, Code as a Second Language, and Black Girls Code. They provide a counternarrative to the assumptions and beliefs about race in dominate discourse about coding. This study examines how Black people develop new discourses and practices from attending a computer code bootcamp that tries to get both racial justice and inclusive workplaces. It describes what happens on the ground for low-income people looking to develop their literacies for social mobility and well-being. In doing so, a more complicated and progressive story comes to the surface.

Limitations of Coding Literacy and Paths to Liberation

I suggest that interests in computer programming for liberation overlaps with literacy studies' interests in reading and writing. Some concepts from the field can show that the coding movement already rests on shaky theoretical ground. These concepts also show why the racial view on everyday Black people in computer code bootcamps helps create an accurate picture of coding literacy's ties to racism and labor. First, the coding movement updates the literacy myth, which originally theorized the belief that reading and writing is necessary for a range of positive life and societal outcomes, such as economic development, democratic practice, and upward social mobility (Graff, 1979). The myth of reading and writing, and later digital literacy broadly throughout the 1990s, has evolved into a coding literacy myth in 2012 (looping back to the 1960s' coding movement, too). The literacy myth turns the word "literacy" into a god-term that, once attached to an adjective, makes anything essential (visual literacy, financial literacy, film literacy etc.) (Wysocki & Johnson-Eilola, 1999) to galvanize powerful politicians and funders to advance the movement's mission. Code.org calling computer programming a new literacy, as essential as reading, writing, and arithmetic raises eyebrows and opens pocketbooks.

Unlike the ways "literacy" gets thrown around in public discourse, coding *is* a literacy; it *is* a type of writing. As a form of writing, computer programming works not only as a cognitive skill but also a social and material practice; coding operates alone as a method of communication while also being the foundation for how we write digitally (video, audio, images, etc.) (Vee, 2017). Computer programming is symbol system that develops meaning for human beings and computers, which can act on the "flow of information through social systems"; what information matters becomes reality for digital platforms themselves, shaping how people interact with each other across space, time, and geography (Ko, 2016, p. 33). Computation awards power to programmers for controlling "others' computers, and by extension, designing and deciding how people experience the extent of their lives" (Ko, 2016, p. 33). A social view of literacy, then, will steer away from treating computer programming as a "magical entity" that speaks for itself, a neutral moving target flowing through histories of literacy; instead, literacy studies recognizes computer programming as subject to human beings' "social and machinic rituals" (Chun, 2008, p. 311). Critical race studies unveils meritocracy and objectivity from tech to find the moves everyday low-income Black people make when they encounter computer programming.

The expansion of coding literacy creates a mentality or pressure that others need to learn computer programming (Vee, 2017). Literacy campaigns leverage the above concepts or rather the above concepts steer literacy campaigns like the coding movement: the myth that coding is a panacea for social ills, the necessity that everyone has it, and coding is a type of writing with power in our lives. Coding literacy campaigns connect local context with efforts to build or strengthen national defense and the national knowledge economy. They also target populations who have been historically left out of literacy education. The responsibility to strengthen a nation through economic productivity passes to Black and Brown people, women, LGBTQ+ people, the poor, and their intersecting social identities, who need to keep up. The irony is that they fell behind because an oppressive state for decades told them to *stay behind*. Coding literacy campaigns do not pay an education debt for years of institutional exclusion (Ladson-Billings, 2006). Instead, computer programming, like reading and writing, "serves as a badge, a sign of initiation into a select group" (Arnove & Graff, 2020, p. 439). Hence, Code.org, and tech companies like Apple, swoop down into states to spread the gospel of computer science education; they offer resources and funding to hire and train computer science teachers. Individuals create bootcamps and partner with local and major tech companies to design curricula that meet industry standards and form a pipeline for bootcamp graduates to join a prestigious class of people. The coding literacy campaign recognizes disparities in the tech workforce across gender, race, ethnicity, and sexuality and so targets these populations to get them onboard. The extent to which they achieve their goals—whether for empowerment or further control over marginalized people into pre-made roles—gets messy when marginalized people assert agency with coding literacy. They can either use computer programming according to the plan of their sponsors or resist and find new pathways. In some cases, as I discuss in Chapter 4, they have no choice but to pivot because the opportunities dry up or are too chained up.

My point in this section is to show just how much the coding movement attracts literacy studies. Code.org, and other organizations leading the campaign, dig deep into literacy history, perhaps unknowingly, and look extremely familiar to scholars like me. Computer code bootcamps are just a piece of many social and material resources sent to communities of color. Computer code bootcamps stand on the front lines of tech labor and the economy. They align with a familiar history of reading and writing's social meanings from age to age and share in a longer history of technological exploitation. For this reason, I journey to understand how the most vulnerable members of Black communities interact with these institutions and how they position themselves in this new turn in our social history of digital literacy.

The conceptual background here shows how technology as utopia play out *against* Black people and labor. The promises and hope of computer programming have been withheld selfishly in the past, and despite efforts to engage Black people in coding now continues that exclusion. In other words, the digital literacy myth was never meant to help Black people; Black people were meant to be subjugated and exploited to perpetuate that myth for others to believe in. What computer code bootcamps and other literacy campaigns do is offer fantasies, not hope. I'm reminded of Alex, one of my participants attending Clearwater Academy in spring 2017, who critiqued half-done racial justice in the United States. He argued that the Emancipation Proclamation, the Thirteenth Amendment, and the Fourteenth Amendment had not delivered total justice for Black people. I think this exchange between us represents his ideas better than any summary or paraphrase:

> **Alex:** I'm an African male captive in America … Well having been. Ok. Well, shit. Yeah. To give a little further explanation of that: If you take a zebra in the year 1920 and you took the zebra from Africa and then put it in a zoo in Canada. And then eventually you let it and all of its babies out of the zoo. Would it be a Canadian zebra?
>
> **Antonio:** No. [laughter]
>
> **Alex:** No? Oh okay. [laughter]
>
> **Antonio:** Well, it's not from Canada, right? [laughter]

> **Alex:** Hell nah, it ain't from Canada. And I'm not from America. We ain't come from here. We ain't come from here. We was captured and brought here. And then let go. And still here. So I am a descendent of African captives. An African captive in America. 'Cause they ain't never talking about sending me back. When have they paid for me a plane ticket? When you get released you, get sent home! Unless, of course, your home has been decimated and destroyed and all the records of your existence and history has been washed from the history of the Earth. I mean are you fucking kidding me? Yeah, I'm an African captive in America. To me, if to nobody else.

I know I'm suggesting that I align myself with Afropessimism, the idea that Black life is ontologically always a social death. From my multiple engagements with Clearwater Academy's thoughtful instructors and from having at times intimate conversations with Black participants and other adult learners attending the computer code bootcamp, my stance is that structures as they are do not work. We need something new. I align with and take inspiration from Adam J. Banks' work on how Black people seek transformative access with digital literacy practices. His explanation I quote at length:

> African American rhetoric, then, is intended to document the ways Black people have hacked or jacked access to and transformed the technologies of American life to serve the needs of Black people and all its citizens. … By transformative access, I mean that African Americans have always argued for a genuine inclusion in technologies and the networks of power that help determine what they become, but never merely for the sake of inclusion. African American rhetorical practices call attention to the ways that the interfaces of American life, be they public facilities, education, employment, transportation, the legal system, or computer technologies, have always been bound up in contests over language, and have always been rhetorical—about the use of persuasion, in these cases, toward demonstrably tangible ends. (Banks, 2006, p. 45)

Transformative access shows that literacy, despite the mythic quality literacy sponsors give to it, has the potential to intervene in social inequalities. That isn't necessarily within hegemonic power structures, social institutions, and the systems created to control how people perceive and interact with their social reality. Instead, the intervention comes through revelations and clarifications about what's in front of us but it's so mundane we don't give it total thought. Black coding Discourse about coding literacy and the Black tech ecosystems that Black adult learners in this book create twists the relationship between labor and race slightly. The study participants say, "I am not just *becoming* a coder; my *relationship with* labor and tech is changing." This nuanced look at literacy and Black labor in computer code bootcamps details how Black adult learners develop knowledge and practices out of their interacting with computer programming. I investigate the downstream impact this knowledge has on their lives that's not completely considered in popular assumptions on what coding literacy should do for Black people. *Black Tech Ecosystems* delivers critical imaginings of Black people discovering the truth of their knowledge and definitions of labor using coding literacy as an intervention tool. In the following section I discuss my study's context, data collection, and methods for analysis. I also briefly introduce you to the participants. I give some theoretical attention to racial matrices, which makes my using "Black" to describe participants more complex for this study design and its analysis. Finally, a special note: the names of all people, cities, and schools in this study have been changed. I have renamed the city where I conducted my research Sakowin. This acknowledges that Clearwater Academy resides on the ceded lands of the Očhéthi Šakówiŋ. The size of this territory provides confidentiality on the exact location of Clearwater Academy.

Clearwater Academy in Context

In the early 20th century a few women brought to the upper midwestern city Sakowin the Social Justice Cooperative (SJC), a nationwide movement to fight racism, sexism, and poverty. The movement had begun in Europe in the century before. These women established a Sakowin chapter, and over a hundred years later in 2014, it established Clearwater Academy as its latest justice-informed program. SJC adopted the growing computer code bootcamp model and wrapped it around equity: recruit low-income racially marginalized people and women to train them in web design over an intensive three months. They would be an important resource in a small, thriving, and growing majority white tech hub. Between 2014 and 2019, the workforce in the tech sector grew by nearly fifty percent; by the end of the first year of the COVID-19 pandemic, technology publications reported that Sakowin had the largest migration of tech workers in the United States. A major research institute listed Sakowin as one of the best cities for future innovations in tech. Entrepreneurship in tech was a major attraction.

An internationally known local research university called Sakowin University fueled this innovation with its sophisticated computer science and business programs, vast connections with national and international major companies, and stellar research faculty; throughout the city startups popped up left and right, and an incubator space in downtown Sakowin encouraged young ambitious entrepreneurs to play around and experiment with business ideas of their own. One of the largest healthcare software companies in United States was founded just outside the city. With a large well-funded research university in Sakowin, many computer science graduates could gravitate to that company. But the drawback with hiring young computer science majors was that they tended to leave town. Get a fantastic education. Get some work experience. Make a lot of money. Pay off their student loans (if they had any) and then jet to the biggest company they could find. A "brain drain" was a problem in Sakowin. Despite the possibilities of tech innovation and social mobility, Sakowin was still burdened with significant disparities. The story was typical of many progressive cities in the United States: Black people were poorer than white people; the city, and its state, incarcerated more Black men than white men. Sakowin professed to white liberal progressivism, but residents often had a Not In My Backyard (NIMBY) mentality.

Clearwater Academy positioned itself in this thriving tech hub as an opportunity creator to address some of these disparities in the city. They partnered with local tech companies to know what skills and competencies were needed in the regional labor market. Clearwater Academy's instructors developed a curriculum around those needs. The computer code bootcamp would provide something the university could not: local diverse talent that intending to stay and live in the city. Clearwater Academy had another advantage: it was one of the few accredited computer code bootcamps in the nation. In the industry's early days, the promise of computer code bootcamps was in doubt because many were not vetted by institutional accreditors like the Higher Learning Commission; these bootcamps had no standards to follow other than what tech professionals told them would be needed in their content. But Clearwater Academy had partnered with a respected local community college; adults graduating from Clearwater Academy could use their credentials for credit hours at the college should they want to try for an associate degree. They didn't have to struggle as much to find a clear pathway into college. The associate degree could be a ticket into attending the Sakowin University, or they could get a good job. Working in tech was the ideal but getting any higher paying job was the more realistic goal.

Clearwater Academy's curriculum split between two skills covered by two instructors. Richard was the technical skills instructor. He had two firsts under his belt: one of the few Black men to design websites during the dot-com era of the 1990s, and one of the first ones to get a Gmail account. These days many users must create email addresses by stylizing their name (using periods between first name, middle name, and last name, adding numbers, or making up some other name, for example) so the address is unique. But if someone's email is just their name—no styling, no numbers—they were probably the first to get a Gmail account (or they have a unique name). That was Richard. His email address was just a sign of how long he had designed websites for businesses before deciding to give back through community teaching. Richard trained adults in web design using HTML (Hypertext Markup Language), JavaScript, and CSS (Cascading Style Sheets) over the intense three and a half months. (Note that HTML is a markup language. A markup language defines and creates a document and can fill that document with text, and CSS helps style the text and bring in other multimodal elements like images; programming languages give instructions to computers to automate tasks. Computer programmers would not consider HTML or CSS programming languages.) These three languages, however, were just the minimum for adult learners to learn.

Richard encouraged adults to learn other languages while attending Clearwater Academy. He could provide resources to get them started, but he would not teach them any scripting language or programming language other than web design. Richard followed Google's learn how to learn model for pedagogy; that was an essential skill for any coder.

Adult learners in the program used a variety of tools to awaken and sharpen their coding practices, but FreeCodeCamp was the primary tool. The popular nonprofit offers hundreds of online activities and projects across multiple areas of computer programming and tech. After completing several hours, users can receive a free certificate. As of July 2023, the website boasts that 40,000 graduates have completed their courses. Code Academy was another tool for learning web design, although it was not as often used and was soon abandoned during my visits in 2017. Like FreeCodeCamp, Code Academy offers easier exercises in web design but provides no certificate. Wordpress was the central tool for projects in class, although adult learners could branch off and create their own website or app from scratch. To bring them fully into the software process, Richard taught adults how to design logos, résumés, websites, and mobile apps in the design program Sketch before coding them into existence. Once a week, volunteer professional coders visited Clearwater Academy for one-on-one tutoring sessions.

Jessica was the case manager and employability skills instructor. After completing a degree in spiritual formation, Jessica spent a year working for a church. For her "it was awful," so Jessica switched to business. While she enjoyed working in that world, Jessica felt inspired to give back, and she switched to child protection services. The job wasn't the right fit for her either. Jessica found her way to Clearwater Academy. She had no idea what she was doing, but her starts and stops in social work, learning how human development works, and working in the business sector all came together at the computer code bootcamp. Richard thought of her as the project manager between the two of them “because without her, I would be lost.” Employability skills included résumé and cover letter writing, mock interviews, elevator pitches, collaboration, problem-solving, wage negotiation, and networking (virtual via LinkedIn and in-person).

Some of the project management frameworks in software development like Scrum solidified collaboration and problem-solving skills. Clearwater Academy most resembled a worksite in its attendance policies. Bootcamp trainees needed to arrive on time for “work” each day and behave like professionals, although they did not have to dress professional all the time. To graduate, they needed to log 400 hours of work time. If they missed some hours, the trainees could make up for the time by writing blog posts or attending Meetups—activities, gatherings, and events for people with similar interests that occurred in a variety of locations throughout Sakowin. Meetsups were opportunities to network with other coders or professionals in the industry, so attending these was required in general, too.

Throughout the three months, Richard and Jessica took adult learners on tours of local tech companies, where they learned and witnessed the inner workings of the industry. Speakers would visit to discuss a variety of life topics from finance and budgeting to time management. Richard and Jessica understood that their program was intense and was asking a lot of adult learners: instead of being at work making money, spend your work week learning something with no pay. Many adults were poor, had many life responsibilities outside of the computer code bootcamp, and had limited resources. Clearwater Academy offered limited social support services: gas cards, bus passes, rent assistance, and access to career services were all in place to ensure most adult learners would complete the entire semester. Teaching coding and employability skills was a collective effort between Richard and Jessica. Both were always in the room passing off teaching responsibilities according to their lesson plan. They were sincere and gave lots of tough love to adult learners slacking off and falling behind. The classroom atmosphere flowed from serious and professional to fun and light-hearted. Other than the desks facing the front of the room (which could be rearranged for group work), I most clearly understood Clearwater Academy as a worksite when an adult learner wasn’t behaving professionally (falling asleep in class, talking out of turn, being late, etc.).

For the first two or three years, Clearwater Academy had a rough start. The program had to hunt for trainees. Instructors and staff put up fliers around town and used existing career services and other social support programs to bring in adult learners. Retaining those adult learners was just as hard. A cohort of 25 adult learners could drop to fifteen or ten in just a few weeks. Childcare, housing, healthcare, mental health, and run ins with the law could cut off low-come adults from access to coding literacy And those who made it to the end had a bigger hurdle: the instructors before Richard and Jessica focused on teaching web design, neglecting employability skills. So, when those graduates presumably had no idea how to behave in the workplace: they might try to cook fried chicken in the break room, sell marijuana to a superior, or go on multiple smoke breaks.

Richard and Jessica had a lot of work to do when they appeared on the scene. Having been one of the first Black men to leverage coding into a lucrative career during the 1990s, Richard brought with him a reputation that tech companies could respect and trust. And Jessica had experience in social work to provide emotional and social support. By the time I arrived at Clearwater Academy in 2017, the computer code bootcamp had earned its reputation in Sakowin. They had gone from hunting for recruits to being flooded in referrals, many of them from alumni telling their friends and family to attend. The retention rate for the first class of adults was thirty percent; by the fifth class in 2016, retention increased to ninety-one percent. Many tech companies were delighted to partner with Clearwater Academy to further the diversity mission. Four years after I completed my study in 2018, Clearwater Academy remains strong, welcoming new cohorts of adult learners even through the COVID-19 pandemic by offering online options. They have since spread their training model to other Social Justice Cooperative chapters nationwide. And that wage problem? Bootcamp trainees now receive a wage grant that pays them fifteen dollars an hour while they attend Clearwater Academy. Richard and Jessica have moved on to other jobs and no longer work for Clearwater Academy, but they left behind a legacy of change.

Finding and Joining Clearwater Academy

A friend in one of my graduate seminars connected me with Clearwater Academy. She had heard about my interest in studying Black coders, and I was having a hard time finding them (not surprising given I was in a majority-white city attending a majority-white university). I first attempted to access Black coders at the major healthcare software company outside of town; I was lucky enough to email someone in their communications department who told me they would investigate how they could help. They never responded to my follow up emails. Clearwater Academy, however, was different. My friend had volunteered at the computer code bootcamp and knew Richard well. She could introduce me to the executive director of the local SJC chapter. With a memo of understanding in my hand, I spoke with the director honestly about how my interest in learning about coding and social justice aligned with Clearwater Academy's goals. I could not promise any insight on the effectiveness of their curriculum design; however, I could, in a small way, bring to light a model for computer code bootcamps that could change conversations on coding literacy education for Black adult learners. With approval from my university's institutional review board and formal permission from Clearwater Academy, I began my year-long ethnographic study.

Recruiting participants was straightforward. On the first day of class for both the spring and fall semesters of 2017, I pitched a simple idea: to gather stories about Black people learning computer programming so that others—tech companies, other computer code bootcamps, and computer science teachers—can learn how to welcome Black adult learners into software development. In the spring semester, seven Black-identified adults agreed to participate. In the fall semester, five agreed to participate. I had no money to offer them, which was a questionable ethical conundrum for me. These were low-income Black people giving some of their time and energy to speak with me, to allow me to do observations, and to collect their literacy artifacts (résumés, cover letters, blog posts, copies of class notes, documents that supported their brainstorming of ideas, etc.). I offered my skills as a writing instructor in return. For three months in the spring and again in the fall, I brought university-level training in teaching to help participants with writing their résumés, cover letters, and blog posts, and with practicing their elevator pitches. When time came for them to formally practice their elevator pitches, Richard and Jessica asked me to weigh in on their performances. I refused to withhold my assistance from other non-participating adult learners; anyone who wanted help got it.

Although I shared racial identities with all but two participants, I could sense that I experienced race and class differently from them. For example, late in the 2017 spring semester, I met with Kevin and a few other Black adult learners during breaktime, and we struck up a conversation about our childhoods. Kevin recalled eating government cheese. In 1981, President Ronald Reagan signed the Agriculture and Food Act of 1981, which included a provision to distribute five hundred and sixty million pounds of cheese the government had been stockpiling. States could request the cheese to give to welfare recipients, Food Stamp recipients, people on social security, the elderly, and community organizations like churches. The government cheese plan so happened to coincide with Reagan cutting the federal food stamps program's budget, so handing out processed cheese was a weird substitute for real food. Nevertheless, Kevin remembering government cheese drew a line between us. My parents grew up poor in rural Alabama, but they navigated racism to reach a middle-class life. I didn't eat government cheese growing up; I ate real cheese.

As I interviewed these participants and talked to them informally, I noticed how my own upbringing could influence my analysis of their experiences.

My experience could lead me to wonder why these Black adult learners had not done better or to judge them for making a series of life choices that led them to this moment—Clearwater Academy. But reading scholarship, attending conferences, and listening to classmates throughout coursework taught me that the choices Black people make respond to structural racism. What I witnessed was a form of Black technical knowledge brought on by years of navigating a racist system that led them to put hope into computer programming. I could not resist that system either. I was a Black graduate student in a majority white city in a state with the highest rate of incarceration for Black men in the country. The gap between us was tenuous and small at best. Having such a small yearly income, I could sense that one choice—or a cascade of expensive events—would put me on the same path as them. By doing in-depth analysis of hundreds of hours of interviews and field notes that cover 200+ hours of participant observation, and then writing them into this book, I have strengthened my joy for Black life in the United States.

Although I use the word “Black” in the title of this book and throughout its chapters, the word has more complexity in this context, and in the context of the diaspora in general. It’s well-documented that race is a social construction, and the United States distributes privileges, rights, and resources according to a perceived hierarchy of worth, placing white racial identities at the top. But even the definition of white is a moving target. Race can be a catch-all term that too neatly categorizes people as having a consistent identity. This perspective can leave conflicting dimensions of race unaccounted for. One dimension frequently used by sociologists, and literacy scholars like me who tend to borrow theories and methodologies from across the humanities and social sciences, is racial identity and racial self-classification. Racial identity is a person’s subjective view of themselves while racial self-classification is the official answer someone gives on a survey or close-ended question (Roth, 2016). While I did rely on racial identity to recruit participants, I ran up against other dimensions of race that shaped how some participants identified themselves: observed race and reflected race, how others view your race and *your understanding* of how others view your race respectively. Thus, a complex web of identification came to light for three participants.

During my call for participants, I used the word “African American,” a term I thought unproblematic. My conception of African American was quickly challenged by three adults. First, Nadine, a single mother attending Clearwater Academy in spring 2017, was from Sudan but came to the United States as a refugee. Other Black people saw her as African and white people saw her as African American. I write more about Nadine in later chapters but suffice to write here that Nadine identified with the African American experience in the United States based not on who she was—Sudanese—but how others treated her in a weird space between two worlds. In the fall 2017 cohort I met Pierre whose mother was Black and his father Irish. During a literacy history interview, Pierre felt that white people treated him as if here were completely Black, based on frequent microaggressions from his co-workers at a restaurant. Although Zeus was the son of Cuban and Spanish parents, his phenotype—the color of his skin—marked him as a combination of white and Black. In this book he jumps between self-descriptors of his race: mulatto, (as read at the start of the Introduction) and Mexican (quoted in Chapter 3). But found himself in life often framed as a Black person, an identity and experience he could not separate from his ethnicity. Once a Nigerian friend in the military even called him white, which was a shock to Zeus. If that was a consistent view from everyone, he joked, he could've then leveraged that identity for some privileges and awards.

Each racial dimension can reveal different experiences with race and new theoretical and methodological insights. They trouble my notions of African American, Black, and Black American. "Black" comes with a small asterisk; its definition encompasses the self-identification of adults and those adults who have coded themselves as Black because their social experiences, and not their racial identity itself, defined so much of their existence. I move from African American to "Black" to include the diaspora whose many racial identities, and racial dimensions, results in part from the subjugation of their ancestors at the hands of white colonizers.

This book reports on a year-long ethnographic study at Clearwater Academy. To understand how Black adult learners labored in a computer code bootcamp I used a diverse set of methods at my disposal: literacy life history interviews, participant observations, document analysis, and ecological theories of writing (this method described in detail in Chapter 2) helped me cover the complicated twists and turns of learning how to labor in Clearwater Academy. At the end of each semester, I asked participants if I could do a follow up on how Clearwater Academy may have changed their lives: In what ways did they end up using coding literacy and to what ends? Here the concepts learned from Clearwater Academy had a chance to be seen in action three months and then six months after. Throughout the following chapters, I rely primarily on interviews and informal conversations with participants, but I also draw from descriptions and reflections from my field notes and official documents from Clearwater Academy to ground my analysis in their context or augment my analysis to drive home my argument. As readers can imagine from my brief positionality statement above, I also bring in my personal experiences learning Python or my own relationships with participants to give greater detail and granularity to the picture of what it means to be Black in a computer code bootcamp.

Critical discourse analysis (Gee, 2011) and grounded theory (Charmaz, 2014) help me arrive at the concepts I argue about throughout the book: Black coding Discourse and Black tech ecosystems. Critical discourse analysis postulates that language doesn't just convey information; it reflects our sense of doing and being in the world among others. However, language also accrues power within communities. The social structures of our world, the institutions that protect those social structures, and the inequalities that can result from self-interest get tied up to how we name and describe people and our interactions with them. If discourse helps shape worlds, it can also intervene in social and political issues. Critical discourse analysis investigates how people use language for transformative change. When Black participants describe the coding practices and sociocultural contexts of software, they name possible worlds for themselves: what the current logic of software development might do to them as Black people and what they can do differently from that logic. As they work on coding and learning employability skills from the tech industry, Black participants in this study construct new ecosystems that suit them. New practices for surviving and thriving using technology spring to life and they develop a different discourse that reflects what's happening in that ecosystem. Grounded theory as a method of analysis helps me deeply investigate the complexity of language and meaning to understand Black participants' motivations, values, and beliefs. Line-by-line, and sometimes in chunks of sentences related to one another, get coded and recoded into categories and themes that describe what's happening for these Black adult learners. Events I witnessed and wrote about in field notes provide a richer tapestry of what coding literacy means to Black adult learners in Clearwater Academy from the interviews. I detail an assets-based study that calls for pathways toward a different future for computer science education.

The methodological diversity I draw together—literacy history interviews, individual and focus group interviews, observation, ecological theory of writing, and document analysis—are related yet distinct methods that allow me to view in-depth the lives of Black adult learners in a computer code bootcamp. In, *Qualitative Literacy: A Guide to Evaluating Ethnographic and Interview Research*, Mario Luis Small and Jessica McCrory Calacro (2022) argue that exposure is the bedrock of a strong ethnographic study; therefore the number of participants may not matter as much as the amount of time spent with them. Although participants in this study are small, the multiple methods I use allow me to find multiple contours of Black coding Discourse and Black tech ecosystems that flows in and around their lives in-depth. Interviews capture narratives, personal backgrounds, values, beliefs, and motivations, but participant observation and literacy artifacts deepen my understanding of the knowledge, processes, and practices Black adult learners gathered from their time in Clearwater Academy and how they applied these ideas in their lives after graduating. Their storied lives reveal motivations and beliefs for the practices they use before, during, and after attending a computer code bootcamp. These uttered discourses reflect the world they desire and seek to create through a complex ecosystem of people, objects, and emotions. These two in-depth discoveries—Black coding Discourse and Black tech ecosystems—form the chapters that follow.

What You Can Expect: An Outline of the Chapters

The book describes Black adult learners' relationships with each other, with themselves, with their families and friends, and the worlds that structures these relationships when they voluntarily enter Clearwater Academy. It treats computer code bootcamps as a key touchpoint in Black adult learners' participation in the coding movement. All these forces and the bootcamp structures their relationships with coding literacy and how Black adult learners leverage computer programming in an economy that supposedly is poised to reward them. *Black Tech Ecosystems* tells a complicated empirical narrative of Black knowledge and lived experience applied to coding literacy education for work. This approach allows me to pay attention to the material and social circumstances that govern their flow through digital ecosystems as historically excluded designers and co-designers of digital environments.

Chapter 1 describes the literacy work histories of Black women adult learners attending Clearwater Academy. While society in the United States often malign low-waged work, I find that these women's patterns of low-waged work contained significant digital literacy practices. Oppression from one job position to the next only solidified their expectations for labor and tech, and these desires led them in part to Clearwater Academy as a step toward new imagined worlds. This chapter shows that conceptually digital literacy links to motivations for coding literacy, and coding literacy becomes a tool toward healthier familial relationships and the kinds of labor that rewards them independence and flexibility. Chapter 2 describes how Black adult learners bring together the infrastructures of Clearwater Academy and the networks of their own lives to foster carework practices to stay plugged into an intensive curriculum. To understand how low-income adults revise their lives for three months of training in computer programming, I use network maps which help emphasize how learning coding literacy amidst the material consequences of racism is highly contextual and may vary from adult to adult. This chapter demonstrates how a computer code bootcamp that centers Black desire can influence the literate lives of Black adult learners.

Based on separate in-depth group interviews with Clearwater Academy's instructors and with Black participants, Chapter 3 suggests that Clearwater

Academy "codes" Black adult learners as "functions" that can fit into existing software programs created by majority-white companies. Rather than disrupt their software programs, Black functions seem to assist in perpetuating whiteness in tech. Interviews with instructors and their adult learners reveal that they struggle with reconciling efforts to create racial justice through coding with the ideological and economic benefits to majority-white tech companies. Chapter 4 considers the realities of transitioning from a computer code bootcamp to the workplace as a Black coder. Drawing on the post-graduation lives of four participants, this chapter demonstrates that, unlike the rhetorical claims explored in the Introduction, coding literacy itself does not help Black adult learners overcome the sociomaterial and cultural barriers to social mobility. However, the work-based coding literacy they did learn continues to echo in their lives in other ways and promotes sustainable lives. Their learning coding literacy also gave them clarity that other literacies already exist in their repertoires and could afford more opportunities than computer programming.

In the conclusion, I reflect on the drama that unfolds among coding literacy myths, computer code bootcamps, and the digital cultural competencies Black adult learners learn about coding. First, I summarize the core of the book: the study of Black coding Discourse in computer programming reveals knowledge, values, and practices that facilitate a desirable Black tech ecosystem. Escaping from the tech pipeline, Black adult learners frame computer programming as a type of writing distinct from other kinds of digital literacy practices yet no better rewarding than existing relationships and Black literacy and rhetorical practice. Then I switch from theory to praxis: I suggest computer code bootcamps may be better equipped to center and draw inspiration from Black adult learners' socio-cultural resources and values to guide curricular design. Vocational training programs may use a critical race technology theory framework (Tanksley, 2022, 2023), which challenges stories of technological progress with teaching in-depth how technologies have often developed under a white racial frame to serve stratification and oppression. Learning coding, then, is not merely an economic opportunity but one that purposely undoes this ongoing legacy in their own lives. An anti-racist computer code bootcamp suggests Black adult learners develop cultural competencies about computer coding and bring those frameworks into the design of technologies. Recognizing and supporting the ways Black adult learners fulfill their techno-lust for coding to find survival and sustainability, based on these participants' lives, is a starting point for such interventions.