< Chapter 7 >

The ENIAC Girls Vanish

What makes programming so often inhospitable for women?

This is something Cate Huston has thought about a lot during her 15 years as a coder. She started hacking as a kid in boarding school in Scotland, then went to study computer science at college in Edinburgh, where she quickly discovered, no surprise, it was a boys' club; there were few other women in her computer science class. But it didn't deter her. She craved the weird, head-spinning challenges of coding, like "the time I had to implement my own floating-point math for an app on a Motorola mobile phone during an internship." She dropped out and joined Google in 2011, helping code the mobile and tablet apps for products like Google Docs and Google Plus.

On the surface, Google touted itself as a "nice" workplace, where executives talked bouncily about inclusion and diversity. But when Huston arrived in 2011, she found that many of her mostly male colleagues could be frosty and pedantic, and some were clearly convinced that women weren't suited to coding. One day she asked a coworker about how to mount a hard drive on an Android device to take a photo; he replied by sending her some code "that I wrote two years ago when I was an intern," as he added snippily. (He was also too late; she'd already figured it out.) During "code reviews"—when colleagues appraise each other's code, offering suggestions for improvement—male colleagues would spend hours nitpicking over tiny details in her code or frowning and saying *I would have done it differently*. ("And I'd have to go look into and discover that the way they did it wouldn't have worked," she says.) One colleague was so endlessly antagonistic about her code—and aggressive during code reviews—that her manager, she says, barred the guy from meeting with her alone or emailing her without also copying the manager. "By that point, it happened so often that it seemed normal to me," she adds. "But my manager was, like, no," this was unacceptable. Indeed, she got so used to being harangued by male coders that she worried she was mimicking their behavior. "I wondered if *my* code reviews were usually harsh because I'd learned to do code review that way, by having it done to me by these guys."

Still, the actual programming work remained fun. She got to help program the mobile interface for Google Plus, which back then was a high-profile product for the company. Many of her coworkers were collegial and respectful, and she tried to focus on them. But then comments would come out of the blue that stunned her. “Women should be in the kitchen, not writing code,” one coder told another female Googler, a friend of hers. “Women just don’t *like* coding, women just like pretty things,” others would tell her. And then there was the frenemy head-gaming, quiet undercutting moves. She’d show up at work to discover a colleague had written a piece of code for her (“just wanted to save you time!”); she watched as female colleagues were pulled off a project, claiming, “Look, I know this project isn’t really your style.”

Was she actually inadequate? By the most objective standards possible, it didn’t seem so. She was getting perfectly solid reviews from superiors. But the message from these young-guy colleagues was like a constant hum of background radiation: We don’t really think you’re suited to be here. When someone pointed out only 15 percent of Google engineers were female, they’d argue that, well, it was probably genetic. Google was a meritocracy; they only hired the best, right? If they weren’t hiring women, it must be because women just didn’t have enough in-born logic or grit.

And on top of that were actual bursts of flat-out harassment. One coworker called her a “cunt” to one of her friends; and although Google took some action in response to her complaint, in the end, she didn’t feel they took it seriously enough. Then there was a tech worker for another firm who started feeling her up on a flight to a conference—“He was treating me like a part of the in-flight entertainment,” Huston says. And there was the coder from another Google office who would stalk young female interns so persistently that a female Googler sent a memo to the women in her office warning others about him when he was traveling to visit: “Please stay away from him.”

“It was very normal for me to find other women crying in the bathroom—and for *me* to be crying in the bathroom once a week was normal.” She even, she notes drily, began looking for new jobs that would allow her to work from home once a day so she could at least do her weekly breakdown in private. “For me, crying at work was so normal that I decided to *optimize for it in my job search*,” she says, with mordant wit. “I don’t want to cry in the bathroom anymore! I want to cry on my sofa and have access to my Clinique products after to repair things.”

After three years, she was heading into her middle career, and she was sick of being undervalued. Huston fled Google and was later hired as the head of mobile development for Automattic, the firm that builds apps for blogging using WordPress, the popular open source blog software. When I spoke to her, she'd been there for a year, managing a staff of 25, and she'd found it a breath of cool air. Her boss, WordPress cofounder Matt Mullenweg, had actively recruited her; women hold many high-up positions in the firm. Plus, WordPress's code is all open source, so the culture is less know-it-all than at Google. And, hey, nobody'd called her a bitch!

"I just love to write code," she says, half brightly, half sadly.

Computer programming is a strange aberration in the world of high-stakes, high-pressure professional work. In the last few decades, the number of women has grown rapidly in many such fields. In 1960, they were only 3 percent of lawyers; by 2013, they were 33 percent. In the same time frame, women went from 7 percent of physicians and surgeons to 36 percent. In many parts of science and technology, it's the same story. Women were 28 percent of biologists, and were fully 53 percent by 2013; they've risen from 8 percent of chemists to 39 percent today.

One big exception? Computer programming. Back in 1960, 27 percent of workers in computing and mathematical professions (they're grouped together in US government stats) were women, and that number rose until 1990, when it reached about 35 percent. But then it reversed trend—and started falling. By 2013, the participation rate for women was back to 26 percent. Things had regressed to worse than the point they were at in 1960. Nearly every other technical field has had increasing numbers of women arrive. Programming is the one field where things went backward, and women were actually chased away. Why?

It's often noted that the first computer programmer ever was a woman: Ada Lovelace. As a young mathematician in Victorian England, she met Charles Babbage, the inventor who was trying to create an Analytical Engine. The Engine was a steampunk precursor to the modern computer: Though designed to be made of metal gears, it could execute loops and store data in memory. More even than Babbage, Lovelace grasped the enormous potential of computers. She understood that because they could modify their own instructions and memory, they could be far more than rote calculators. To prove it, Lovelace wrote what is often regarded as the first computer program in history, an algorithm that the Analytical Engine could use to calculate the Bernoulli sequence of numbers.

True to coder form, it contained a bug! Perhaps even truer to form, though, Lovelace had a clear and bombastic view of her own brilliance. “That *brain* of mine is something more than merely *mortal*; as time will show,” she wrote in a letter. When she listed her personal qualities, item 2 was “my immense reasoning faculties.” (“No one knows what almost *awful* energy & power lie yet undeveloped in that *wiry* little system of mine,” she added in another letter.) Lovelace intuited the enormous power that computer programmers would one day wield. It would be like, she said, being an “*Autocrat*” of information, commanding “the most *harmoniously* disciplined troops—consisting of vast *numbers*, & marching in irresistible power to the sound of *Music*.” Alas, Babbage never managed to actually build an Analytical Engine, so Lovelace died of cancer at 36 without ever having seen her code executed.

When the age of truly electronic computers began in the 1940s, women were again at the center of the action, as Janet Abbate recounts in *Recoding Gender*. Men, certainly, were central to this new machine age—but at this point, they felt all the rugged glory and heroism lay in making the hardware. The first programmable digital computer in the US, the ENIAC, was a more than 30-ton behemoth made of 20,000 vacuum tubes and 70,000 resistors. Getting that contraption merely to function? That was the manly engineering task. In contrast, programming it—figuring out how to issue instructions to the machine—seemed menial, even secretarial. Women had long been involved in the scut work of doing calculations. In the years leading up to the ENIAC, many companies used huge electronic tabulating machines they bought from companies like IBM; they were really just glorified adding-and-subtracting machines, but very useful for, say, tallying up payroll. Women frequently worked as the punch-card operators for these machines; they’d punch holes in punch cards that represented, for example, how many hours an employee worked, then feed them into the tabulating machine to add them up. It was noisy, painstaking, and inglorious work.

So when ENIAC came around, programming seemed—to the men in charge—to be similar enough to menial punch-card work that they happily hired women to be the ENIAC's first programmers. Indeed, the first ENIAC programmer team was all-female: Kathleen McNulty, Betty Jennings, Elizabeth Snyder, Marlyn Wescoff, Frances Bilas, and Ruth Lichterman, known later as the "ENIAC Girls." The men who ran ENIAC would figure out what they wanted the program to do; they'd spec out the code, as it were. It was up to the ENIAC women to physically crawl around—and even inside—the machine, hooking up wires that "programmed" the machine to execute the instructions. It was head-scratching, pioneering work; they wound up understanding how ENIAC worked even better than many of the men who'd built it.

"We could diagnose troubles almost down to the individual vacuum tube," as Jennings, one of the women, noted. "Since we knew both the application and the machine, we learned to diagnose troubles as well as, if not better than, the engineer."

They invented groundbreaking ideas in coding. Snyder realized that if you wanted to debug a program that wasn't running correctly, it'd help to have a "break point," a moment when you could stop a program midway through its run. When she pitched the idea to the male engineers running ENIAC, they agreed to implement it; and to this day, coders use break points as a key part of debugging. Indeed, the ENIAC women were particularly adept debuggers, the first coders to discover that software never works right the first time. In 1946, the ENIAC heads wanted to show off the computer to reporters for the first time. They asked Jennings and Snyder to write a program to calculate missile trajectories. After weeks of intense hacking, they had it working, except for one embarrassing bug: The program kept running after the missile had landed. The night before the demo, Synder suddenly intuited the problem. She showed up early the next day and flicked a single switch inside ENIAC, fixing the bug. "Betty could do more logical reasoning while she was asleep than most people can do awake," Jennings later said.

Nonetheless, the women got little credit for their pioneering work. At that famous press conference, the managers of the ENIAC project didn't mention or introduce the women; that's how unimportant coding, and the work of the women, was considered.

After the war, coding jobs shifted from the military to the workplace, and industry desperately needed more programmers—and thus some way to make coding easier than having to onerously write cryptic, number-based "machine code.” Here, women again wound up being pioneers. They designed some of the first “compilers.” These were programs that would let you create a programming language that more closely resembled actual English writing. A coder could thus write the English-like code, and the compiler would do the hard work of turning it into 1s and 0s for the computer. Grace Hopper was wildly productive in this field, often credited as creating the first compiler, as well as the “FLOW-MATIC” language aimed at nontechnical businesspeople. Later, she worked with a team to create COBOL, the language that became massively used by corporations, and the programmer Jean Sammet from that group remained influential in the language’s use for decades. (Her desire, she said, was “to put every person in communication with the computer.”) Fran Allen became so expert in optimizing Fortran to run swiftly that, decades later, she became the first female IBM fellow. Women were at the forefront of bringing coding to the masses, by dramatically exploding the number and style of programming languages.

Coding jobs exploded in the ’50s and ’60s, and for women, this weird new field was quite receptive to them: Since almost nobody knew how to code, in the very early days, men had no special advantage. Indeed, firms were struggling to figure out what type of person would be good at coding. You needed to be logic-minded, good at math, and meticulous, they figured. In this respect, gender stereotypes could work in women’s favor. Some executives argued that women’s traditional expertise at fastidious pastimes like knitting and weaving imparted precisely this mind-set. (The 1968 book *Your Career in Computers* argued that people who liked “cooking from a cookbook” would make good programmers.) Mostly, firms gave potential coders a simple pattern-recognition test, which many women readily passed. Most hires were then trained on the job, which made the field particularly receptive to neophytes of any gender. (“Know Nothing about Computers? Then We’ll Teach You (and pay you while doing so),” as one ad enthused.) Eager to recruit women, IBM even crafted a brochure entitled *My Fair Ladies*. Another ad by the firm English Electric showed a bob-haired woman chewing a pen, noting that “Some of English Electric Leo’s best computer programmers are as female as anything.”

Even some black women could find a toehold, so hungry was the field for talent. In Toronto, a young black woman named Gwen Braithwaite had married a white man, and they discovered that given the racism of the time, nobody would rent to them. That meant they had to buy a house, which meant she needed a job. After seeing an ad for “data processing” jobs, Braithwaite showed up and convinced the all-white employers to let her take the coding-aptitude test. When she placed in the 99th percentile, the supervisors thought she'd pulled a prank and grilled her with verbal questions. When she passed those with flying colors, they realized she was the real thing and hired her. She became one of the first female coders in Canada, and she led several big projects to computerize insurance companies. "I had it easy," she later told her son. "The computer didn't care that I was a woman or that I was black. Most women had it much harder."

By 1967, there were so many women programming that *Cosmopolitan* magazine commissioned a feature on "The Computer Girls." Illustrated with pictures of beehived women piloting computers that looked like the control deck of the Starship *Enterprise*, the story described this crazy, new Wild West that was paying women $20,000 a year—or over $140,000 in today's money. Coding had quickly become a rare white-collar professional field in which women could thrive. For an ambitious woman in the '60s, the traditional elite professional fields—surgery, law, mechanical engineering—accepted almost no women. Programming was an outlier; the proportion of female coders was fully one in four, a stunningly high figure, given the period. The options for women with math degrees were limited—teaching high school math or doing rote calculations at insurance firms, for example—so "women back then would basically go, 'Well, if I don't do programming, what *else* will I do?' " notes Janet Abbate, a professor in the department of Science, Technology, and Society at Virginia Tech, who has closely studied the era. "At the time back then, the situation was very grim for women's opportunities."

Women even pioneered their own entrepreneurial opportunities, as Abbate's research has documented. One female coder, Elsie Shutt, learned programming while a college student doing summers with the military at the Aberdeen Proving Ground. In 1953, she was hired to code for Raytheon, where the programmer workforce "was about fifty percent men and fifty percent women," she said. "And it really amazed me that these men were programmers, because I thought it was women's work!" She left the job to have children, and she quickly discovered that motherhood killed her chances of getting another programming job; the '50s and '60s may have been welcoming to female coders, but it wanted them childless. No firms were willing to offer part-time work, even to superb coders. So Shutt founded Computations, Inc., a for-hire consultancy that would write code for corporations. Remarkably, Shutt hired stay-at-home mothers as part-time programmers; if they didn't know coding already, she trained them.

They'd look after their kids during the day, then code at night, renting time on local computers. "What it turned into," as Shutt told Abatte, "was a feeling of mission—in providing work for women who were talented and did good work and couldn't get part-time jobs." *Businessweek* dubbed Shutt's workforce the "pregnant programmers," running a story that was illustrated by a picture of a baby in a bassinet in a home hallway, with the mother in the background, hard at work on a piece of code. (The article's title: "Mixing Math and Motherhood.")

By the late '60s and '70s, though, the field of coding began to tilt male. As the historian Nathan Ensmenger has documented, as programming became more crucial to firms and coding projects got bigger, they needed to promote coders to management; it didn't feel right to be putting women in such important positions. Culturally, the industry was beginning to professionalize. It was moving away from the "anyone can do this" ethos and demanding advanced degrees and accreditation at a time when women were less likely to have either. And the very image of what a coder looked like was becoming more male in the central-casting eye of employers. The industry was increasingly believing coders *ought* to be introverts who had terrible people skills and unkempt grooming. On top of it all, the economics of coding were becoming more lucrative, too, making it obviously less secretarial and menial than it seemed back in its ENIAC days. As sociologists have long noted, when a field suddenly becomes increasingly well paid and prominent, men who previously spurned it rush in: *We'll take this over now, ladies, thank you very much.*

"One of the big takeaways is that technical skill does not equate to success," says Marie Hicks, a former UNIX administrator who became a historian and studied the same shift taking place in the UK. "They wanted people who were more aligned with management."

The prominent women in coding could see things changing as the '70s wore on. Fran Allen of IBM watched as she became the increasingly rare female coder in the room. "As it became a profession . . . it became an avenue that women were pretty much shut out of—in general," Allen told Abatte. "There were fewer women." There were still pockets of prominent women, particularly in professional societies, and areas like compilers. But "in lots of places, there was a huge glass ceiling." She stuck through to the end, though, and by the early '00s she was still hacking at IBM, helping to create Blue Gene, the computer that would become the basis of Watson, IBM's elite artificial intelligence.

If we wanted to pinpoint a moment when things flipped, we could look at one year in particular: 1984. That's when women started moving out of bachelor's computer science degrees.

In the decade leading up to 1984, both men and women were increasingly interested in programming. Ten years previously, for example, a study found that the number of men and women who expressed an interest in coding as a career was equal; men enrolled more than women did in bachelor's computer science programs (women were only 16.4 percent), but women were certainly intrigued by the prospect of doing it. Women soon began to act on that interest. Their participation in computer science programs rose steadily and rapidly throughout the late '70s and early '80s, such that by the 1983–84 academic year, women were fully 37.1 percent of all student coders. In only one decade, they'd more than doubled their participation rate.

But that was the peak. From 1984 onward, the percentage slid downhill, slumping even more in the '90s, such that by the time 2010 rolled around, it had been cut in half. Only 17.6 percent of the students in computer science programs were women.

What happened? Why was there such a sudden and dramatic inflection point?

One thing that transpired was the shift in when kids could learn to program. As personal computers emerged in the late '70s and early '80s, it altered the population of students who were streaming in to computer science degrees.

In the years leading up to 1984, pretty much every student who showed up on a college campus to learn programming had never touched a computer before. Frankly, they'd probably never even been in the room with a computer before. Computers, in those decades, were rare, expensive machines mostly available only at companies or research labs. So all the students were on equal footing. They were nearly all new to programming. They learned their first "Hello, World!" program at the same time.

But beginning in the late '70s and early '80s, the first generation of personal computers arrived—like the Commodore 64 or the TRS-80. Now, a teenager could mess around with a computer at home, spend weeks and months slowly learning the major concepts of programming in their spare time: for loops, if statements, data structures. So beginning in the mid-80s, some of these showed up for their first class having *already done* a lot of programming. These kids were remarkably well prepared for the introductory work—perhaps even a little jaded about what the first Comp Sci 101 class would offer. As it turns out, the kids who'd had this previous experience were mostly boys, as two academics discovered, when they researched the reasons why women's enrollment was so low.

One of these two researchers was Allan Fisher, then the associate dean of the computer science program at Carnegie Mellon University. CMU had founded a computer science undergraduate program in 1988, and for the first few years, up into the early '90s, Fisher noticed that the proportion of women in the major was consistently below 10 percent. So in 1994, Fisher decided to figure out what was going on and how to attract more women into CS. He partnered with Jane Margolis, a social scientist (and now a senior researcher in the UCLA Graduate School of Education and Information Studies), to embark on an ambitious study. For four years, from 1995 to 1999, she and her team interviewed about one hundred of CMU's computer science undergraduates, male and female, and she and Fisher wrote up their findings in the book *Unlocking the Clubhouse*.

Margolis discovered that these teenagers arriving at CMU with substantial experience were far more likely to be boys. Boys had been given much more exposure to the machines than girls; for example, boys were more than twice as likely to have been given a computer as a gift by their parents. And if parents bought a computer for the whole family to use, the parents most often put it in their son's room, not their daughter's. On top of that, the fathers tended to have almost an "internship" relationship with their sons, working through BASIC manuals with them, figuring out coding and encouraging them, while very rarely doing that with their daughters.

"That was a very important part of our findings," Margolis told me. Nearly every girl at CMU said "it was the father who worked with her brother, and they had to fight their way through to get some attention with their father." The mothers, in contrast, were more absent from the home-computing scene. Girls, even nerdy girls, picked up these messages and adjusted their enthusiasm accordingly. And these were, of course, pretty familiar roles for boys and girls. By the early '80s, parents had for decades or centuries tacitly—and openly—nudged their kids into these categories: Boys do the technical stuff, while girls play with dolls and socialize. It wasn't terribly surprising to Margolis that when a new tech came on the scene, it fit into that groove.

At school, too, the girls got the message that computers were a boy thing. Male nerds would form computer clubs, in part as a relief to finally have their own arena away from the torments of jock culture. The cliques of boys that formed around computers created "a kind of peer support network," as Fisher puts it. But they often wound up, intentionally or not, becoming exclusionary puts it. But they often wound up, intentionally or not, becoming exclusionary themselves—snubbing, as studies in the ’80s found, not just girls who came by, but often the black and Latino kids, too. And the girls also policed their own behavior around computers. Being a girl nerd, they realized, was going to be harder than being a boy nerd. Male nerds were certainly picked on, but being single-mindedly obsessive about something, anything (football, cars, computers) was encouraged in boys. Girls, by contrast, have long been harshly judged—including by their own parents—for seeming too narrowly focused on a single subject. But coding rewarded the obsessive.

This helped explain why CMU’s first-year classes were starkly divided—between the sizable amount of men who were already confident in basic programming concepts and the women and minorities who were frequently complete neophytes. A cultural schism had emerged. The girls—and some of the boys who didn’t have previous experience—would start doubting their ability. How would they ever catch up?

As Margolis heard from the students (and from faculty, too), it started to feel that if you *hadn’t* already been hacking obsessively for years, why in God’s name were you there? The stereotype emerged: The only “real programmer” was the one who “had a computer-screen tan from being in front of the monitor all the time,” Margolis notes. “The idea was, you just have to love being with a computer all the time, and if you don’t do it 24/7, you’re not a ‘real’ programmer.” The truth is, many of the hacker men themselves didn’t fit this monomaniacal stereotype. They had hobbies; they did social things; they wanted a more rounded life. But there was a double standard: Whereas it was okay for the men to want a rounded existence, women who expressed the same wish were judged for not being “hard-core” enough. The experienced male students would talk openly about how the inexperienced women didn’t seem up to scratch and were unsuited for computing. If the women asked a question in class, they’d get snarked at: *How obvious*. By the second year, many of these women, riddled by doubt, began dropping out. The same was true for the few black and Latino students who also arrived on campus without teenage hacking experience. They, Margolis found, were dropping out far more often, too, at a rate of 50 percent (though she cautions that it’s hard to generalize from that statistic, because there were so few enrolled to begin with—only four.)

Here’s the thing, though: A student’s decision to drop out—or, conversely, to stay—was not correlated with raw coding talent, it seems. Many of the women who dropped out were getting perfectly good grades, Margolis discovered. Indeed, some who decided to leave had been top students. And the women who did persist, and made it to the third year of the degree, discovered that by then they'd generally caught up to the teenage hackers. The degree was, in other words, a leveling force. Doing BASIC as a teen might teach you a ton of cool skills, but the pace of learning at college was so much more intense that by the end of the degree, everyone—the total neophytes and the hacker teens alike—eventually wound up graduating with the same amount of knowledge about coding and making software.

This is a counterintuitive finding, of course. We'd assume the teenage hackers would remain eternally ahead. CMU itself, operating under the assumption that self-taught teenage coders would forever outclass neophytes, had actively selected for teenage coders; it was more likely to admit applicants with previous experience into the CS degree. But as Allan Fisher told me, this intuition was wrong. "It turned out that having prior experience is not a great predictor, even of academic success," he says.

One can't solely blame the hacker boys, who'd created a high school culture of programming that prepared them for college, for being sexist or exclusionary. They were kids, pursuing their love and interest in coding, and the adults around them—the ones whom you'd hope might have been better at being less sexist—were validating the boys-only culture. What's more, the early pioneering work of women in programming had by now been generally erased, making it a lot harder for women of the '80s to understand not only that they could belong here but also that women had been, in fact, among the biggest pioneers in the field. Meanwhile, mass culture was hammering home the boys'-club message. Hollywood was energetically producing hit movies (from *Revenge of the Nerds* to *Weird Science* to *Tron* to *WarGames*) in which the computer nerds were nearly always white or, more occasionally, Asian young men. Video games, a major conduit into becoming intrigued with computers, were, as studies at the time found, pitched far more often at boys. The mass public sense of "What does a programmer look like?" was becoming heavily male and white.

It didn't help, amid all this cultural love bombing, that women themselves tend to rate their technical competence lower than men. When Lilly Irani—then a graduate student of Science, Technology, and Society at Stanford University—did a study of the computer science students similar to Margolis's, she took men and women and asked about their level of "confidence in solving problems with computers." Irani knew that the women and men had essentially identical grades. But *they* didn't know that. And the confidence rates, it turns out, were gendered: Women rated their confidence an average of 7.7 out of 10, and men rated theirs at 8.4 out of 10. More striking was when they were asked to compare themselves to their fellow students: Did they think they were more or less confident than their peers? Women pegged themselves, on average, as being half a point less confident than the other students, while men rated themselves as being “six tenths of a point *more* confident than their peers.” Despite being identical in performance, the men were simply more confident, more convinced they *felt* like real coders. The women were precisely the reverse.

Some of the women’s sense of inadequacy appeared to be based in their preferences. Some research has found that men more often like software tinkering for the sheer pleasure of tinkering; women, in contrast, more often thought coding would be cool because they wanted to have an impact on the world. So it was at CMU: As Margolis found, most of the women didn’t like the idea that they’d just be sitting in a room with a computer for the rest of their lives. This was another factor that drove women away during the first two years of computer science, because that’s precisely what students traditionally do during that period: They’re heads-down, learning how to write algorithms and work with data structures, without the work being connected to any larger mission. It’s the definition of tinkering. To be sure, some of the CMU men also didn’t like the idea of ignoring their hobbies and social lives while coding endlessly at abstract tasks. But this “geek myth”—that the only real coders were endlessly glued to their keyboards—was more damaging to the women’s sense of belonging, Margolis found.

Nonetheless, the women who managed to stick it out for the later years discovered that things flipped. In the later years of computer science at Stanford and CMU, students moved on to working in teams to build full-featured apps; at Stanford, they also began learning a new language, Java. And here, the comfort level of men and women shifted. Since everyone at Stanford was now struggling to master a new language, it reduced some of the perceived differences between students. Plus, working in teams required new skills in managing projects and collaboration, again putting the students on more equal footing—plus bonding them together, because the success or failure of the project was a common fate. For the women who actually made it to that stage, their confidence suddenly soared. They loved it.

But there were fewer of these survivors reaching this epiphany. At CMU, by this point, about half of the women who’d started out had already left.

There was another force that drove women away from computer science beginning around 1984: the "capacity crisis."

Computer science schools had become victims of their own popularity. Programming was hot. So many students were rushing to enroll that universities ran into a supply problem: They didn't have enough professors to teach that many students. To make matters worse, good professors were getting hired away by the already fat and rapidly ballooning salaries in the private sector. "Industry was paying *so* much more," recalls Eric Roberts, who taught computer science at Wellesley College and Harvard in the '80s, before moving to Stanford. (He's currently at Reed College in Portland.) He remembers making offers to five different people before finding one willing to teach.

How did the schools cope with their limited teaching resources? By chasing students away. Some began imposing "weeder" courses, which students had to pass before they'd be accepted into the computer science major. These were classes with intentionally punishing workloads, and which moved along so quickly they left behind anyone who didn't immediately "get it." Some professors would offer little to no assistance to anyone confused by an assignment; *Figure it out on your own or you don't belong here* was the message. (Many professors overloaded, in any case.) Indeed, some schools demanded perfect scores on first-year pre-CS classes—at Berkeley, you needed a 4.0 average on these—before you'd be allowed into the major. The schools, in other words, created an environment where the people who'd be most likely to get through were the ones who came in already exposed to coding—which was, again, mostly boys, and generally white ones.

"Every time the field has instituted these filters on the front end," Roberts says, "that's had the effect of reducing the participation of women in particular." It's also possible that by weeding out so many kids, the programs wound up actively sorting for kids who were extra competitive and who never suffered from any deflated sense of their worth. Roberts adds, "Women are socialized to like much less the kinds of competitive situations in which there are real losers. And as a result they were quicker to leave the stage."

The capacity crisis hit both men and women, in its first few years. As the schools chased students away, the number of CS degrees overall—for both men and women—declined by over 40 percent. But by the mid '90s, the crisis abated, and computer science programs began to grow again, taking in more students. By now, though, the culture of "who goes into coding" was becoming set. When the crisis eased and more students surged back in, most of them were men. The interest among women had been blunted. It never recovered to where it had been in the late ’70s and ’80s. And for the women who did show up, they often felt like sore thumbs. In a room of 20 students, perhaps 5 or fewer would be female.

With so few women around, computer science departments became hotbeds of testosterone. In 1991, the computer scientist Ellen Spertus wrote a report on women’s experiences in programming classes. She catalogued a landscape of guys who’d snigger about women’s presumed inferiority; professors who’d tell female students “you’re *far* too pretty” to be studying electrical engineering; when a few CS students at CMU asked men to voluntarily stop using pictures of naked women as their computer desktop wallpaper, the men angrily replied that this was censorship straight out of “the Nazis or the Ayatollah Khomeini.” A similar study at MIT produced equally bleak tales. Male students would muse about women’s mediocrity: “I really don’t think the woman students around here are as good as the men,” one said. Behavior in research groups “sometimes approximates that of a locker room,” the report found, with men rating how “cute” female students were, in front of them. (“Gee, I don’t think it’s fair that the only two girls in the group are in the same office. We should share,” said one.) “Why do you need a degree for marriage?” others would ask women coders. When women raised their hands in class, they’d typically get ignored by professors and talked over by other students. They’d be told they weren’t aggressive enough; though when they actually did behave aggressively, challenging other students or contradicting them, they’d hear comments like, “You sure are bitchy today; must be your period.” Some would even grope them, massaging their shoulders or fondling their breasts.

As women fled computer science, the workplace, too, increasingly became a monocrop of mostly white men. According to data from the Bureau of Labor Statistics, in the US in 2017, about 20 percent of workers categorized as “computer programmers,” “software developers,” or “web developers” were women. If you added in other job categories that involve technical chops—such as database administration or statisticians—it goes up a bit higher, to 25.5 percent. The ratios for black employees were 6 percent and 8.7 percent, respectively, and Latinos were in that range, too (6 percent and 7.3 percent), which is in each case about half, or slightly more than half, their rate of participation in the rest of the private-sector workforce. (Other stats I've found suggest things might be lower; Data USA, for example, pegged the ratio of black coders at 4.7 percent in 2016.) In the more rarefied world of the top Silicon Valley tech firms, the numbers were sometimes skimpier: An analysis by *Recode* suggested that 20 percent of Google's technical employees were women, while only 1 percent were black and 3 percent were Hispanic. Facebook was nearly identical, and Twitter was 15 percent, 2 percent, and 4 percent, respectively.

These were sufficiently small minorities that, many would tell me, arriving for work felt like they were intruding on some weird, cloistered, all-white priesthood. Certainly, every underrepresented coder I spoke to told me stories of individual colleagues or managers who treated them as equals. But nearly every one also described the daily fight of having to prove themselves over and over again to the wide swathe of industry peers who, tacitly or openly, assumed they didn't have serious technical chops, that they *couldn't.*

One coder, Stephanie Hurlburt, was a classically nerdy math-head who'd cut her teeth doing deep work on graphics. "I love C++, the low-level stuff," she tells me. She'd worked for a series of firms, including Unity (which makes a popular game-design tool), and then for Facebook on its Oculus Rift VR headset, cranking mad hours to release their first demo. Hurlburt was accustomed to shrugging off neg hits. There were many: She'd been told, including by many authority figures she admired, that girls weren't wired for math. While working as a coder, if she expressed ignorance of nearly any niggling concept in graphics, some male colleagues would pounce. "I thought you were at a higher math level," one sniffed. At one firm, she was sexually harassed; when she later tried to raise concerns about the company's overall toxic environment with HR, they were—in a pattern that I heard repeatedly from female coders—unwilling to address it.

So Hurlburt eventually founded a start-up called Binomial, programming, along with Rich Geldreich—a friend who shared her love of graphics math—a tool that helps compress the size of "textures" in graphics-heavy software.

Working for herself, she figured, would mean no dealing with undermining bosses. But still, she couldn’t entirely escape the default assumption that women aren’t techies. When she and Geldreich would go into meetings to sell the product, some customers would assume she was just the marketing chick. “I don’t know how you got this product off the ground when you only have one programmer!” one customer told Geldreich. Still, owning the company gave Hurlburt some moments of sweet karma. At one of her first coding jobs years earlier, the chief technical officer had told her “I shouldn’t waste time working in a field my brain wasn’t made for.” Now, years later, the staff working under that CTO has been sending her emails asking if they can use her texture-compression code, because they desperately need to speed up their own software. She told me she hadn’t figured out if she would sell it to them yet. “People who *aren’t* sexist get the best compression!” she laughs.

Women coders get mistaken as PR lightweights. Black and Latino programmers also get mistaken for security or housekeeping. That’s one of the things that happened to Erica Baker, a build-release engineer for Slack when I met her in 2017. Baker had grown up as a nerdy African American kid teaching herself QBasic and Hypercard, and eventually wound up working for Google in Atlanta. She faced some racist comments: She had a coworker who said “Does your boyfriend beat you?” she recalls. She eventually moved to Google’s Mountain View office, where at one point a temp mistook her for security, and another employee mistook her for an administrative assistant. When she asked to be put in training programs to move up into new technical levels, she never got in, though she “saw white dude after white dude get the chance to do it.” Her time at Google wasn’t uniformly bad, she notes; several of her fellow Google employees were more supportive and mentored her. (One executive noticed the incident with the temp when it occurred and checked in with her to make sure she was okay.) But eventually, she says, “I had realized that, even though it’s the best place for the majority of people at Google, it was not the best place for those of us who were the minority.” In 2015, Baker moved to Slack, which had a somewhat higher proportion of minority-group coders than many other valley firms. (Across Slack’s worldwide offices, women were 34.3 percent of those in technical jobs; in their US offices, 12.8 percent were from “underrepresented racial and/or ethnic backgrounds”; and 8.3 percent identified as LGBTQ. Later, she moved on to be a senior engineering manager for Patreon.)

Levels of harassment are high, too, for LBGTQ employees in tech. When the Kapor Center for Social Impact surveyed tech workers to find out why they’d voluntarily left a tech job, 24 percent of LBGTQ workers said they’d voluntarily left a tech job, 24 percent of LBGTQ workers said they'd experienced “public humiliation or embarrassment” at work. They were also the group in the survey that reported the highest rates of being bullied, at 20 percent. (Fully 64 percent said bullying “contributed to their decision to leave.”)

Not every minority-group coder I interviewed had trouble being taken seriously. Of the scores of women I spoke to for this book, a small number said they’d experienced essentially no problems in the tech workplace. “I don’t find it,” noted Yan Zhu, a well-known expert in encryption, who was working as an engineer helping to create the new web browser Brave. “I think that I’m not very sensitive to this, honestly.” She certainly knew many tales of discrimination: “I’ve talked to a lot of women whose experiences don’t match mine at all. . . . I don’t want to discount theirs.”

And while blatant harassment and assault get the most headlines, many coders I spoke to said the bigger problem was the side-eyed doubt, the undermining comment. It was less often a guillotine than death by a thousand tiny cuts. Studies bear this out: One analysis of 248 performance reviews for tech engineers found that women were considerably more likely than men to receive reviews with negative feedback; men were far more likely to get reviews that had “only constructive feedback,” with no negative material.

Another part of what froze women out of many coding jobs is the concern, among start-ups, about “culture fit.” A start-up involves a small number of people working often in tight quarters for long hours. This leads, as you might imagine, to founders primarily smiling upon hires who are socially and culturally similar to them. One female coder arrived in Silicon Valley after being scouted by a firm, and the founders took her out to bars three times for long, breezy hangout sessions. They never discussed work. She was baffled. “When are they going to interview me for the job?” she asked a friend, a veteran of the valley. “Oh, those drinking sessions *were* the interview,” her friend assured her. “That’s what they do. They want to know if you can hang.” Many start-ups I interviewed told me they simply hired their close friends from Harvard, Stanford, or MIT.

“It’s all this loosey-goosey culture thing,” says Sue Gardner. After her stint running the Wikimedia Foundation, she decided to study the reasons so few women were employed as coders, and surveyed over 1,400 women in the field, doing sit-down interviews with scores more. She realized the takeover by guys in the ’90s had now turned into a self-perpetuating cycle. Since everyone in charge was mostly a white guy, they preferred hiring people like them; they only recognized talent when it walked and talked as they did. For example, when hiring a coder, most firms relied on whiteboard challenges, in which a prospective employee is asked to write code—often a sorting algorithm—live, on a whiteboard. This type of performance bears almost zero resemblance to what coders actually do in their jobs. But whiteboard questions *do* resemble classroom work at Ivy League schools. So it feels familiar to the guys doing the hiring, many of whom are only a few years out of college, as Gardner notes.

The same cycle governs the all-crucial arena of getting venture-capital funding; they fund people who seem to map onto successful folks they've seen before. Indeed, they even have a term for it: "pattern matching." The venture capitalists, nearly all white men who made their money from the '80s to the early '00s, take a shining to young entrepreneurs who look and behave like younger versions of themselves. In 2008, the leading VC John Doerr said this explicitly, when he described the tech entrepreneurs he most admired. "They all seem to be white, male nerds who've dropped out of Harvard or Stanford and they absolutely have no social life. So when I see that pattern coming in—which was true of Google—it was very easy to decide to invest."

"What I came to realize," Gardner tells me, "is that it's not that women are excluded. It's that practically *everyone* is excluded if you're not a young white man who's single."

This sort of subtle preference—and quiet, understated jabs at women—are more often the barrier to women in coding. But it certainly isn't hard to find stories of flat-out sexism and boiler-room boys'-club behavior, either.

There was the female coder who told me of being assaulted in the offices of a well-known, bold-face tech firm. There's the penchant coders have for including porn in their presentations—like the Ruby on Rails coder who put porn images in his slide deck. There was the Ubuntu lead coder who said in a speech that he was excited about their new software release because "we'll have less trouble explaining to girls what we actually do." There was the guy who, in a seminar on database queries, illustrated how to optimize queries with the example of ranking women by "hotness"—"`WHERE sex='F'AND hotness>0 ORDER BY age LIMIT 10`." There was the leader of a Bitcoin meetup who responded to a female Facebook client-solutions manager who showed up, saying "You don't look like someone who would even know about Bitcoin!," followed by "Women don't usually think in terms of efficiency and effectiveness." (What's more, another attendee groped her while she was there.) Women who talk about these sorts of experiences online face clear threats and harassment; when former Google engineer Kelly Ellis retweeted examples of harassment she got, it just invited even more. ("How much do you charge to spread them?" one replied, in a later-deleted tweet.) And there were tales of tech start-ups that operated as what seemed like thinly veiled frat houses, such as Upload VR—where, as a lawsuit alleged, male employees designated one room with a bed as a "kink room," and a manager once announced he was going into the bathroom to "rub one out" so he could continue to focus while coding.

This frat-like nature of tech start-ups was, in part, the emergence of another trend in the mid-'00s: the arrival of the "brogrammer."

As the portmanteau suggests, the brogrammer—a frattier, bro-ier coder—is a bit of an evolution away from traditional machine-minded men who'd colonized the field. There had been plenty of frat-style environments in the late '90s and early '00s; Facebook's early keg-equipped, graffiti-bedecked office was a good example of that. But the trend accelerated, many techies say, after the 2008 financial crisis. Because many banks contracted, investment banking became a less reliable route for young, cocky, hard-drinking men of the Ivy League to make instant riches. So they moved to the next gold-rush industry—software in Silicon Valley, where investment money flowed at endless silly start-up concepts. (The same trend had appeared, in a smaller format, in the '90s during the dot-com boom, when many Madison Avenue types, from the world of advertising and marketing, glommed on to the start-up scene.)

Eric Roberts remembers seeing a few more of these guys appear in his classroom after the financial collapse. They just wanted to secure a pathway to wealth, and they certainly didn't suffer from lack of confidence. Some were deeply interested in software; others, to his dismay, weren't. "One of the most disturbing things," Roberts tells me, "was that we had a number of students who were majoring in computer science who *hated* the field. But they knew that was where their millions were coming from." As he adds: "It's sort of boiler-room ethic of going and working like mad to become the next billionaire, that we saw so much in the run-up to the crash. And when that fell apart, where did those people go? The only game in town was computing."

Conversely, even as the brogrammers were emerging, many midcareer women were realizing Silicon Valley was never going to give them a break—and were leaving the field. When Sue Gardner surveyed those 1,400 female techies, they told her the same story: In the early years, when they were junior coders, they'd shrugged off the ambient sexism they encountered. They loved programming, were ambitious, and were excited by their jobs. But over time, “they get ground down,” Gardner says. As they rose in the ranks, they found few-to-zero mentors willing to take them on. As many as two-thirds either experienced harassment or saw it happen to someone, as she found when examining *The Athena Factor* (another study of women in tech); and one-third reported that their managers were more supportive and friendly to the men they worked alongside. It’s often assumed that having kids is the moment when women get sidelined in tech careers, but Gardner found that wasn’t often the breaking point for these midcareer women. It was more that they got sick of having equally or less-qualified men around them receive better opportunities and treatment.

Ironically, the higher these women rose, the worse and more blatant the harassment could get. This was particularly true when female programmers decided to found their own firms and had to interact with the world of venture capital, where about 96 percent of the investors are men. Female founders who’d make a pitch for venture capital would get propositioned for sex by investors. Women who actually became venture capitalists themselves discovered the world could be cartoonishly piggish. That’s what happened to Ellen Pao: While working as an investor for Kleiner Perkins Caufield & Byers, a tech CEO on a private-jet flight bragged about meeting porn star Jenna Jameson, then asked the members what sort of sex workers—“girls”—they liked. (“Ted said that he preferred white girls—Eastern European, to be specific,” Pao later wrote.)

So midcareer women in tech were leaving. Sick of years of that crap and not seeing it getting any better—indeed, seeing it getting worse—they voted with their feet. It wasn’t that they’d leave coding entirely. They’d go off and find technical jobs where they’d do programming and oversee coders, but not at software firms. They’d do it in the fields of medicine, in law, in government—anywhere, really, but Silicon Valley.

“What surprised me was that they felt ‘I did all that work!’ They were *angry*,” Gardner tells me. “It wasn’t like they needed a helping hand or needed a little extra coaching. They were mad. They were not leaving because they couldn’t hack it. They were leaving because they were skilled professionals that had skills that were broadly in demand in the marketplace, and they had other options. So they’re, like, ‘Fuck it—I’ll go somewhere where I’m seen as valuable.’ ”

The irony is profound. In the early days of coding, women flocked to software because *it* was the place that offered more opportunity, compared to the sexist fields like law. But that situation has now completely inverted itself. Software is the laggard, the boys' club.

Does it *matter* that the world of tech is mostly white guys? One could, in a purely bloodless way, entertain that question. To which there are two obvious answers: Sure, as a matter of economic opportunity, it certainly does. This is a question about "whether half the human race" is going to get to participate in "a certain kind of work, work that is exciting, lucrative, prestigious, and actually very desirable to many of us," as one female engineer said at a roundtable on gender for Y Combinator.

But it's also a problem for the rest of the world outside coding, too. That's because the monoculture of tech affects the type of products that get made. When you have a homogenous cohort of people making software and hardware, they tend to produce work that works great for *them*—but can be useless, or even a disaster for people in other walks of life.

Consider the engineers at the VR firm Magic Leap, who excitedly developed their augmented-reality headset only to be told by women on staff that it was wildly uncomfortable for many women, since ponytails interfered with the headband, and the device required you to holster a small computer on your belt, which many women don't wear. (The engineers appear to have ignored this input: "None of the proposed changes were made to the design," as a lawsuit alleged.) Or as my friend the comedian and writer Heather Gold has written, consider the UI design Google and Apple introduced in their video chat for groups, where the software is designed by default to take whoever is talking and increase the size of their face on everyone's screen. As she points out, this winds up exacerbating the well-known dynamic of face-to-face meetings: White dudes who bloviate nonstop usually wind up commanding all the attention.

Or consider the case of Twitter, where an originally all-male design team failed for years to appreciate their service's growing problem of abuse on their platform. It was likely quite invisible to them; men experience far fewer threats and less harassment on social networking. If there had been even a few women on the original team who'd been active online in, say, the preexisting early '00s world of blogging, they might have given a heads-up warning: Hey, women online get targeted and harassed, even for pretty innocuous posts! Let's figure out how to get ahead of this problem! ("An opinion, it seems, is the short skirt of the internet," as the social critic and writer Laurie Penny once cracked.) Forewarned thusly, Twitter might have implemented more features to minimize abuse and worked harder to instill a more civil culture among its user-base, as companies like Flickr had previously attempted. But instead it allowed malevolent users to flourish—well unto the point that flat-out neo-Nazis were happily thriving on the social network.

By 2016, the Twitter designers' lax approach to harassment had become an economic liability. Growth had flatlined, but corporate suitors were leery of investing. Disney had decided against buying Twitter, worried that, as the *Boston Globe* put it, "bullying and other uncivil forms of communication on the social media site might soil the company's wholesome family image."

Former Twitter CEO Dick Costolo, after years of denying abuse was a problem, openly recanted. "I wish I could turn back the clock and go back to 2010," he says, "and stop abuse on the platform by creating a very specific bar for how to behave on the platform." That would've been much easier to do if there had been someone around to warn them.

In the summer of 2017, a 28-year-old senior software engineer named James Damore wrote a memo arguing that there was a different reason for the dearth of female coders: It was biological. As antifeminist theories often do, this argument quickly went viral. Damore worked for Google, a company that—at least publicly—had embraced the notion that gender discrimination was a problem. They'd instigated training sessions to alert staff to "implicit bias," unconscious expressions of bigotry. In meetings, managers discouraged sexist language. None of this stuff had significantly increased Google's ratio of women in technical jobs yet (to do that, you'd have to hire many more women, not just talk about it), but the topic was in the air.

Damore, though, saw these policies as political correctness that avoided the real facts. In a note on the company's internal bulletin board, titled "Google's

Ideological Echo Chamber," he argued that the gender gap wasn't necessarily a cultural problem. The real issue was that men were more suited—cognitively, biologically, based on evolutionary adaptation—to be coders. Damore argued that research shows women were more prone to feeling anxious and were less competitive, thus making it harder to thrive in the hard-driving world of Google. The company could, he wrote, adapt more to women by changing its culture to accommodate these traits, such as "allowing those exhibiting cooperative behavior to thrive," or making tech less stressful. But he also insisted that Google's current attempts to increase the hiring of women and minorities had led to reverse discrimination. After reading the memo over a few times, I found his message pretty clear: The overall distribution of men and women in coding was pretty biological.

Damore himself clearly identifies with the modern model of the ultrafocused, highly competitive, socially awkward male coder. When we met near his apartment in Mountain View, he told me that he'd played competitive chess as a child, before becoming fascinated by game theory, evolution, and physics. As an adult, he was diagnosed as a high-functioning autistic, which, he added, explains why he's so intrigued by systems. Damore was fluent in the work of Simon Baron-Cohen, a British clinical psychologist and professor of developmental psychopathology at the University of Cambridge, who argued that because male brains get such a larger bathing in testosterone in vitro, they are hardwired to be systemizers, interested in things, while women are hardwired to be interested in people. Damore himself never formally studied computer science; while working toward his degree in systems biology, he read a book on algorithms, started doing coding challenges, and won a Google code contest. Once the company recruited him, he worked on search infrastructure.

Damore told me he didn't think women at Google were treated more poorly than men. As he saw it, aggro male coders say snippy critical things about their male colleagues, too—and also talk over them, and take credit for their ideas. "That happens to everyone, I feel," he told me. Men just didn't mind, because they were less sensitive, he said. Essentially, from Damore's perspective, programming workplaces were culturally "male," alienating to women, in a way that was at least partly innate. "I'm not denying that there is sexism or anything," he adds. "But that's definitely not the full picture."

Indeed, Damore argued that the act of talking about sexism in the workplace could make things worse for women: "I really think that there's nothing more disempowering than saying the system is rigged against you." I told him that sexist behavior in tech seemed straightforwardly real—there were copious sexist behavior in tech seemed straightforwardly real—there were copious stories and documented examples, including scores I'd heard myself. "I think a lot of this also gets overblown by the media," he responded, and went on to argue that men faced sexism, too, because people presumed they were physically dangerous, and they were overrepresented in highly risky jobs like coal mining or garbage collectors. Listening to Damore was, I realized, like reading the standard arguments on men's rights activists websites: that offenses against men were equal to, or even outweighed, those against women.

If it's true that women are seldom coders because of biology, I asked him, does that also explain why so few US programmers are African American or Latino? Here, he claimed, the issue *wasn't* biology. "I think culture plays a much larger role than genetics in a lot of these things," he mused. An African American child raised in a high-achieving Asian family would be high-achieving, too, he argued. In essence, Damore was claiming the gender gap had roots in biology but the racial gap did not.

Damore's memo was leaked to the public, most likely by his colleagues at Google, and the company swiftly fired him. In the aftermath, Google CEO Sundar Pichai wrote an email to staff explaining that he couldn't employ someone who judged coworkers based on their sex. "To suggest a group of our colleagues have traits that make them less biologically suited to that work," Pichai noted, "is offensive and not OK." More to the point, as former Google engineer Yonatan Zunger noted, any manager would now find it difficult to put Damore on a team. How would a female coder feel having their code reviewed by Damore, knowing he believes their biological makeup was a liability?

Damore is not alone, though, in this argument that biology is a key to all mythologies. Other coders at Google posted in support of him on their internal forums. Nearly every female coder I spoke to had stories of coworkers citing the same research Damore did, dilating on how nature has decreed rigid male and female roles. When Kelly Ellis, that former Google coder, tweeted about sexual harassment she'd experienced, a tech worker emailed her to explain it was all evolution at work—"We have hundreds of thousands of years of evolutionary history to deal with. Men are wired to approach you as an egg donor first."

There are pretty clearly self-serving reasons why some programmers prefer to believe that biology explains why men are employed in coding far more than women. It's a very pat, neat theory. If you accept it, then you can believe nothing needs to change, that merit obtains. It also suggests, despite reams of evidence to the contrary, that white male coders have no social and cultural advantages over others—no extra encouragement, nudges, mentorship—and that their success is purely a matter of talent and work.

"Programmers tend to be extremely self-assured of their own rationality and objectivity," says Cynthia Lee, who should know: She's a programmer who was an early stage hire at several tech start-ups, became an expert in high-performance computing, and now teaches computer science at Stanford. "They have a very big blind spot for biases on their side, because they don't think they *have* blind spots." Lee wrote an essay for *Vox* explaining why so many women in tech were enraged by the memo: because they spend their days responding wearily, over and over, to precisely this sort of biological skepticism. "When men in tech listen to the experiences of women in tech," she wrote, "they can come to understand how this manifesto was throwing a match into dry brush in fire season."

Is it possible that Damore is right—and biology does explain why so few American women are coders?

No. Many scientists who spend their days seriously studying sex differences say biology alone simply isn't that strong a determinant of preference and ability. Certainly, psychologists have long documented lower levels of self-confidence among many professional women and students. But the detectable differences in boy and girl cognition and behavior are too small to explain such divergence in life and career paths; instead, they're like a biological signal that gets heavily amplified by cultural feedback into life-changing decisions and preference. The portion attributable to pure genetics becomes a force that is small and scientifically murky—particularly when compared to the manifold, well-documented examples of phosphorescent sexism radiating in the halls of tech workplaces.

Indeed, if women's biology made them temperamentally unsuited to coding—and uninterested in it—it's difficult to explain why they were so prominent in the early years of American programming. After all, the coding back then was, if anything, harder than today's programming. It was a head-bustingly new field, in which you had to do math in binary and hexidecimal formats, and there were no helpful internet forums offering assistance with your fiendish bug. It was just your brain in a jar, solving hellish problems.

What's more, if women were so biologically neurotic that they couldn't endure the competitiveness of coding, then the ratio of women-to-men in programming ought to be similar around the world.

But it isn't. In India, over 40 percent of the students studying computer science are women. And this is despite its being even harder to be a female coder there; India has such rigid gender roles that female college students often have an 8:00 p.m. curfew, meaning they can't even stay to work late in the computer lab, as the social scientist Roli Varma found when she studied them. The women had one big cultural advantage over their US peers, though, Varma found: They were far more likely to be encouraged by their parents to go into the field. The Indian coders reported much more equal treatment when girls; only 12 percent said their male peers had been given more exposure to computing, compared to the majority of girls in the US who watched their fathers shower their brothers with nerd attention. (Indeed, the women regarded coding as a safer job because it was an indoors profession, exposing them to less of India's rampant street-level sexual harassment.) It was, in other words, considered normal in India that girls would code. The picture is similarin Malaysia, where in 2001—precisely when US women's participating in computer science had slid into a trough—women were 52 percent of CS majors and 39 percent of the PhDs at University of Malaya, a large public university in Kuala Lumpur. These international comparisons are the sort of A/B test of which Silicon Valley is so fond. The Occam's razor explanation for why US women aren't much in tech seems pretty clear: It's not biology. It's culture.

"Computing being very masculine—that is not a universal image," Varma tells me. And of course, if you wanted to show the fragility of biological and evolutionary explanations, it's worth noting that they could be flipped on their head: You could argue that women's evolutionary history ought to make them *better* at coding. All those centuries of being "gatherers," attentive homemakers, and all that knitting and weaving: Hey, wouldn't that imbue them with the fastidiousness and precision crucial for coding? Indeed, in the early days of the '50s and '60s, as we've seen, this is precisely how executives argued for women's suitability for programming. (One computer-firm head in 1968 claimed that "girls sometimes make better programmers than boys" because "an intelligent girl who has the patience to do embroidery has just the right mentality to do the job.") Once you have your conclusion in mind—*Women are biologically natural coders! No, they're not!*—you can easily craft an evolutionary story (and a definition of what "programming" entails) to get you there, which illustrates precisely why it's difficult to use biology and evolution to explain complex society-level modern phenomena.

If anything, the nature-nurture debate shows the power of the stories society tells itself about talent. People used to say that women were natural coders—they were the winners of that meritocracy. Then in the ’80s a different tale emerged.

The very word *meritocracy* has an ironic provenance. When the British politician Michael Young wrote his novel *The Rise of the Meritocracy* in 1958, he intended it as a biting satire—of governmental attempts to determine an individual’s value based on an IQ test. The term was later adopted as an unironic, literal concept, something that deeply disappointed Young.

It turns out that believing your field is purely meritocratic has lousy side effects. As the cognitive scientist Sarah-Jane Leslie found in a fascinating study, women and African Americans fare less well in domains where people believe that “raw, innate talent is the main requirement for success.” Stats show they’re well represented in fields such as molecular biology and neuroscience, where it’s understood that hard effort, teamwork, and rigor carry the day; but in fields that pride themselves on requiring raw, isolated genius—like math or philosophy—they’re not. Other experiments verify this. Two scientists from MIT and Indiana University took MBA students who had management experience and gave them two hypothetical companies to manage: one where the “core company values” included being meritocratic and one that didn’t emphasize that value. Then the students were given written job evaluations for a hypothetical male employee and a female employee, who had identical and positive evaluations. When the MBA students were told the company was “meritocratic,” they gave the men 12 percent higher bonuses. Why? Because, the scholars suspect, when you prime someone with the idea of meritocracy, it seems to give them permission to act on their internal biases—evoking, it would seem, the cultural idea of the man as more likely to be the lone genius.

Is it possible to reverse the historic shift of women out of coding?

Back in the ’90s, CMU’s Allan Fisher decided to try. Impressed by Jane Margolis’s findings, he and the faculty instituted several changes designed to break the confidence-destroying loop of neophytes who’d glance around their first classroom, worry they’d never catch up with their confident teen-hacker peers, and begin the slow drift toward leaving. One strategy was to create classes that clustered students by experience: The kids who'd been hacking since youth would start on one track; the newcomers to coding would have a slightly different curriculum, allowing them more time to catch up. CMU also offered extra tutoring to all students, which was particularly useful for the newcomers to programming. If Fisher could get them to stay through the first and second years, he knew, these students would catch up to their peers. They also changed the early courses so they showed how code could affect the real world—so a new student's view of programming wouldn't be just an endless vista of algorithms disconnected from any practical use. He wanted students to get a glimpse, earlier on, of what it's like to make software that works its way into people's lives. Back in the '90s, before social media and the mainstream web, the impact code could have on daily life wasn't so easy to see.

The faculty, too, shifted their culture. For years they had tacitly endorsed the idea that the kids who came in already knowing code were, in an essential way, born to it. "CMU rewarded the obsessive hacker," Fisher notes. But the faculty now knew this wasn't true; they'd been confusing *previous experience* with *raw aptitude*, as Fisher puts it. They still wanted to encourage those obsessive teenage hackers, but they now understood that the neophytes were just as likely to bloom rapidly into remarkable talents, and were just as important to encourage; and students pick up on faculty's expectations and respond to them. "We had to broaden how faculty see what a successful student looks like," he notes. Meanwhile, led by professor Lenore Blum, CMU began building more community participation and support groups for women studying CS, events that made them more visible to their fellow students. Admissions, too, changed; it no longer gave so much preference to students who'd been teen coders.

There was no silver bullet, no single policy that improved things, Fisher notes. "There's really a virtuous cycle," Fisher adds. "If you make the program accommodate people with less experience, then people with less experience come in"; then faculty become more used to seeing how neophyte coders evolve into accomplished ones, and how to teach that type.

CMU's efforts succeeded, dramatically. Only a few years after these changes, the percentage of women entering CMU's computer science degree boomed, rising from 7 percent to 42 percent; and graduation rates for women rose to almost match those of the men. The school vaulted over the national average, and indeed improved on the historical record. When word got out of CMU's remarkable success, other schools began using approaches similar to Fisher's. At

Harvey Mudd College in 2006, the school created an intro-to-programming course aimed specifically at complete newbies, and they rebranded their Intro to Java as Creative Problem Solving in Science and Engineering Using Computational Approaches—which, as the president Maria Klawe tells me, "is actually a better description of what you're actually doing when you're coding." These changes weren't always smooth. Mudd has a famously intense curriculum, and the neophyte students drawn into the program have found it hard going. But it, too, eventually worked: By 2018, 54 percent of Harvey Mudd's grads in the computer science major were women, Klawe notes.

There's also been a broader cultural shift, too. In the last few years, women's interest in coding has begun rapidly rising across the entire US. In 2012, the percentage of female undergraduates "who plan to major in computer science" began to rise—for the first time in 25 years, since the collapse in the mid-'80s.

What's behind this? Possibly, women are being influenced by the growing national conversation about how they stopped coding. There's also been a boomlet of institutions training and encouraging underrepresented techies to enter the field, like Black Girls Code and CodeNewbie. And on purely economic terms, coding's appeal has become something akin to law in the '80s and '90s. It's seen as a bastion of good-paying and engaging work.

Perhaps most subtly, the reputation of coding as a profession is changing. Software truly has eaten the world. So anyone who wants to help other people is realizing that programming, as social critic Douglas Rushkoff puts it, is a "high leverage point." In an age when Instagram and Snapchat and iPhones are part of the warp and woof of daily life, potential coders worry less that the job will be isolated, antisocial, and distant from reality. People who used to want to do something "creative" or artsy stayed away from code, including many women. Research shows this is no longer true.

"Computer science today is attracting a more creative woman," notes Linda Sax, an education professor at UCLA, who has pored over decades of demographical data detailing which students, and what genders, opt in or out of STEM fields. This shift is abetted by the fact that it's much easier to learn programming without doing a full degree, using free online code schools, relatively cheaper "boot camps," or even meetup groups for neophytes—all of which emerged only in the last ten years. "This whole thing that let you imagine building an app in a weekend, that was not there back in the '80s and '90s," says Irani, the academic who studied at the Stanford computer science program. "Coding is, on its face, a more social thing now."

There’s even been some pushback to the frat-house-style culture of many Silicon Valley firms. Consider Uber: In February 2017, the programmer Susan Fowler single-handedly shone a light into its culture, with a blog post describing “one very, very strange year” of working there. The company had a famously hard-driving, macho culture. Fowler enjoyed the insane pace of work and her talented colleagues; in her few spare hours, she authored a best-selling book on programming “microservices.” But she quickly discovered that harassment ran wild at Uber. Not long after Fowler had arrived, her manager propositioned her, via chat, for sex. When she told HR and top management, they said her manager was a “high performer” and this was his first offense, so “they wouldn’t feel comfortable giving him anything other than a warning and a stern talking-to.” That didn’t seem true, though. Fowler later encountered other Uber women who’d previously reported the same man for the same behavior. When Fowler continued to complain about how female employees were treated, HR officials struck back in a creepy fashion—demanding the personal email addresses the women used, implying the women were part of a conspiracy, and telling Fowler that “people of certain genders and ethnic backgrounds” may not be suited for coding jobs. About a week after *that,* Fowler fled Uber for a new job. She wrote her story up on the blog post, but, hey: Another blog post complaining about sexism in the valley? What impact could that have?

A huge one, it turns out. Uber’s board had already been hit with years of bad press about its employees’ behavior. Employees had used a “God View” internal mapping tool to help ex-boyfriends stalk their ex-girlfriends; a senior executive had threatened to send private investigators after a female journalist. Then CEO Travis Kalanick had crowed about how he’d gotten so much sexual attention from Uber that he calls it “Boob-er.” Fowler’s story—so carefully documented, with so much official malfeasance by HR—seemed like a particularly humiliating piece of news, and it was the straw that broke the camel’s back. In a few short months, Kalanick had been forced out. And almost as if a cracking of the ice had begun, over the next few months, female coders and entrepreneurs told stories of prominent investors harassing and propositioning them, including Dave McClure, founder of 500 Startups, and venture capitalists Chris Sacca and Justin Caldbeck.

These shifts are only a small beginning. Most coders and experts I spoke to could easily, if wearily, generate a long to-do list of ways the industry could begin to break up its monoculture. They could hire less on “culture fit.” They could recruit from more schools outside the Ivy League. They could kill the “algorithm challenge” whiteboard interview. “They have created their interviews to select for one specific type of candidate, which is, in the case of Google, for example, a Stanford grad. I mean, they design their whole company around it,” to the point where it looks like a Stanford grad school, notes Erica Baker. And perhaps most of all, tech firms—and venture-capitalist titans—would need to actually punish and fire employees who harass others.

Sue Gardner keenly wants a wider range of people going into tech. But she also worries it might be *unethical* of her to encourage the young women she meets to go into the field. She knows they’ll arrive excited, thrive early on, but —unless things change—gradually get beaten down. “The truth is, we can attract more and different people into the field, but they’re just going to hit that wall in midcareer, unless we change how things happen higher up,” she says.

Right now, though, the truth is that there are oodles of young people from all walks of life who want in. They admire tech’s self-image as a meritocracy; they crave a field that’s truly like that. Indeed, few people are hungrier for a true meritocracy than those who’ve historically been sealed out of so many industries and so many other opportunities. An industry run *truly* on hacker ethics—of being judged purely on the quality of your running code, with no one remarking on your identity? That would be a thrilling utopia for them. If the world of programming actually lived up to its implicit ideals, it would be a beacon for any historically screwed-over group. Indeed, that’s precisely why previously picked-on nerd boys found it so glorious when they stumbled upon it in the ’80s. It was the first zone they’d ever found that rewarded their interests and intellect.

In the summer of 2017 I showed up to a TechCrunch hackathon in New York, where 750 coders and designers were given 24 hours to dream up and create a new product. On Sunday at lunch, the teams presented their creations to a panel of industry judges, in a blizzard of frantic elevator pitches. There was “Instagrammie,” a robot that would automatically recognize the mood of an elderly relative; there was “Waste Not,” an app to reduce food wastage. Most of the contestants were coders who worked at local high-tech firms or computer science students at nearby universities.

But the winners, it turns out, were . . . a group of high school girls from New Jersey. In only 24 hours, the trio created reVIVE, a virtualreality app that tests kids for signs of ADHD by having them play a series of games, while also checking their emotional state using IBM’s Watson AI. After the students were handed their winnings onstage—a huge trophy-sized check for $5,000—they flopped into chairs in a nearby room to recuperate. They’d been coding almost nonstop since noon the day before and were bleary with exhaustion.

“Lots of caffeine,” said then 17-year-old Amulya Balakrishnan, clad in a blue T-shirt that read “WHO HACK THE WORLD? GIRLS.” They told me that they’d even impressed themselves by how much they pulled off in 24 hours. “Our app really does streamline the process of detecting ADHD,” said Akshaya Dinesh, then 17. “It usually takes six to nine months to diagnose, and thousands of dollars! We could do it digitally in a much faster way!”

They were aware it was weird, and neat, for three young women to win at TechCrunch. The third member of the group, Sowmya Patapati, pointed out that collectively they had plenty of previous hackathon experience. Dinesh alone had competed in over 25 computer science events. It was how she’d learned to build software, really: She’d taken computer science classes at her school, but that education had been far eclipsed by what she’d learned from free online courses and hackathons. “When I walked into my first hackathon it was the most intimidating thing ever,” she said. “I looked at a room of eighty kids: Five were girls, and I was probably the youngest person there!” But once she’d done a few, her confidence grew. Patapati and Balakrishnan, for their part, took programming courses at school, and had skipped a junior prom and a friend’s birthday party to come to TechCrunch. “Who needs a party when you can go to a hackathon?” Patapati joked.

Being young, brown, and female in this environment brought extra attention, not all of it good. “I’ve gotten a lot of comments like, ‘Oh, you won the hackathon because you’re a girl! You’re a diversity pick,’” said Balakrishnan. When the prize was announced online, some postings were bitter: “There were quite a few engineers who commented, ‘Oh, it was a girl pick, obviously that’s why they won.’ ” On balance, though, they found the world of hackathons deeply welcoming to neophytes. Everyone was there to learn and experiment. People swapped knowledge; they felt like equals in many ways, even to the pros.

“I really do love the hackathon culture and community,” Dinesh said. “It’s the best possible way for people who don’t have resources, like me, to get into this. Without hackathons, I wouldn’t know *anything*.” They all planned to study computer science at college. They can feel the promise of reward here, of the strange powers they can wield; they love the idea of getting ahead purely on their raw, cerebral skill.

“That’s just the thing about tech,” Balakrishnan says. “It doesn’t matter what age or what gender, no matter who you are, you just never know what you can build.”