Living Amongst the Wizards of Cray Research

Edward Barnard

Living Amongst the Wizards of Cray Research

Edward W. Barnard

This book is available at https://leanpub.com/dragons-wizards

This version was published on 2025-12-24

Also by single authorAlso By Edward W. Barnard

From Capone to Cray: Where Computers Really Came From
Unexpected Histories: Demonstrating that Humans are Still Better at Pattern Recognition
Surviving Spring Break on the Mountain: The Power of Experiential Education
How to Create Masters and Mastery in a Classroom Setting
Transcendent Patterns (инвариант): Teaching the Process of High-Tech Mastery in Student-Accessible Fashion
Large Language Model Architecture Patterns in PHP: No Mathematics Required
The Impossible Challenge Manual for Age 14 and Up, Even For Adults: How to accomplish what everyone says you can't
The Wizard's Lens: Learn to Think Like AI
Nobody but Us: A History of Cray Research's Software and the Building of the World's Fastest Supercomputer
Beyond Prompt Engineering
Billy Mitchell's Bombsight: Shaping the B-25 Mitchell Bomber

Contents

Chapter 1. Profile of the “Real Programmer”

The modern technical interview process is badly broken in a way that helps drive women and all kinds of minorities out of our industry.

By recognizing how 1980s “hacker folklore” about the “Real Programmer” mirrors the selection bias in modern coding challenges, we can identify systemic issues invisible to those focused only on current practices.

I personally fit the “Real Programmer” profile and therefore can illuminate these hidden connections from both perspectives. As we bridge modern technical interviews with computing history, we will discover how selection patterns established decades ago continue to shape our industry today.

A Debugging Skill

We are about to dive into the modern technical interview. Our aim is to understand and solve an unexpected problem. As we progress through this chapter, keep an eye on our *methodology* for digging in to, and understanding, the problem at hand. This sort of deep dive, with background research, is an important analysis skill.

Our starting point is the problem itself, the modern technical interview. We will explore variations before stepping back in time to the 1980s and my days at Cray Research. For the solution we will even step further back in time to the days before the U.S.A. implemented Title IX, leveling the playing field between men and women participating in school athletics.

Coding Challenges

Let us begin with an aspect of modern life in high technology: the tech interview that includes a coding challenge. This type of interview seems reasonable on the surface, but I believe it is a key component of the badly broken system that is driving women and people of color out of our industry.

That is a strong statement. To clearly see what I mean, we need to visit several topics; start with the coding challenge.

I have encountered several such coding challenges for job interviews. In each case, the company or recruiter said people had trouble with these. That seemed strange because I did *not* have trouble. In one case, the interviewers were surprised by how quickly I finished the challenge; in another case, no one had yet passed the challenge. This was *not* because I am a better candidate or some such nonsense. What was really going on here?

None of these challenges had anything to do with designing a web application, even though most people in my field do just that. Nor do we generally start from scratch. There is usually an existing

codebase and a problem to be solved. That is why we are having this interview. But these challenges have nothing to do with the reason I am being interviewed. That is weird. Why did the interviewers ask this question? In fact, a pattern emerges.

Online searches confirm the intent: there is a dark side breaking our industry. Sure enough, this type of interview challenge is called *Fizz Buzz*, and it is based on the surprising premise that most computer science graduates, and most professional software developers, “can’t seem to program their way out of a wet paper bag.”

We will first review the reason Fizz Buzz was created. Then, we will take a step back in time, to 1983, and encounter the famous *Real Programmers Don’t Use Pascal*. We will bring those two themes together to see what is broken.

Fizz Buzz

Each of the challenges I encountered was of the form “write the code to make these tests pass.” Each included a clear statement of the problem and instructions for running the tests. None of these challenges had anything to do with building web pages.

That is certainly strange. Why bother with a coding challenge which has nothing to do with the actual job requirements?

Are these Kobayashi Maru scenarios, that is, no-win situations? Yes, and no. One objective of the coding challenge is to observe how you perform during the scenario. It can be a win-win situation rather than a no-win situation.

The Fizz Buzz site explains:[1]

> The “Fizz-Buzz test” is an interview question designed to help filter out the 99.5% of programming job candidates who can’t seem to program their way out of a wet paper bag.

This 99.5 percentage comes from the Coding Horror article “Why Can’t Programmers… Program?” by Jeff Atwood.[2]

Both articles are worth reading in their entirety (and they are short). The Coding Horror comments section debates the merits of Fizz Buzz-style questions, e.g., does not being able to implement recursion mean you are not a good programmer? I will show my snarky recursion solution in the next section.

Coding Horror explains why interviewers are presenting these sorts of challenges. People “can’t seem to program their way out of a wet paper bag.”

This claim is both harsh and unfair. Let’s do the math. If 99.5% of job candidates have trouble with these coding challenges, that tells us either:

- These coding challenges have nothing to do with, and are irrelevant as, predictors of actual job performance; or

- Our training system has nothing to do with, and has no relevance to, our real-world jobs; or
- Both of the above.

The problem, as we will see, is more subtle. We are selecting *against* diversity while *appearing* fair and transparent.

Our next step is to examine Fizz Buzz’s explicitly evil cousin, a specific type of whiteboard interview. But first, let us have some fun with my solution to “can you do recursion?”

Recursion

During September 2010 I was driving through the Ozark Mountains in Missouri and discovered the town of Meta. I definitely was not lost; I was merely enjoying unexpected scenery.* I turned around and returned to the edge of town to take the photo in Figure 1.1, “Meta Data.”

*I was lost.

Figure 1.1. Meta Data

The sign was helpful! It inspired me to stop and take this metadata photo, which in turn, inspired me to get out the map and discover where I was not. I was supposed to take a left turn ten miles back.

The sign is interesting because we use “metadata” constantly in modern software development. Metadata is information about something that is kept separate, not a part of the thing. We see this a lot with websites, databases, and so on.

As another example, digital photos include “exif” data. That includes the date and time the photo was taken, (often) where the photo was taken, and what type of camera took the photo. This information is a form of metadata.

One of the more ridiculous interview challenges is “implement this algorithm using recursion.” The interviewer does not care if you can write good software; if you cannot implement recursion, their company will not be needing you. It does not matter that very few software developers actually use recursion in their day-to-day jobs.

> Recursion occurs when a thing is defined in terms of itself or of its type. Recursion is used in a variety of disciplines ranging from linguistics to logic. The most common application of recursion is in mathematics and computer science, where a function being defined is applied within its own definition. While this apparently defines an infinite number of instances (function values), it is often done in such a way that no loop or infinite chain of references can occur.

This sign carries information about the town of Meta, Missouri, while standing outside, and therefore separate from, Meta. This sign, by definition, is metadata consisting of Meta data, which is totally meta.

An alternative answer to the interviewer's request to solve the problem using recursion might be to hand them your copy of this book with the announcement, "you're in the book." (This answer carries no guarantee of being hired, however.)

Whiteboard Interview

The "Fizz Buzz" test has an explicitly evil cousin, a certain type of whiteboard interview. Let us take a look.

Adrianne Jeffries in "Programmers are confessing their coding sins to protest a broken job interview process" shows how a particular form of whiteboard interview selects *against* a diverse population.[3]

> This interview style, widely used by major tech companies including Google and Amazon, typically pits candidates against a whiteboard without access to reference material—a scenario working programmers say is demoralizing and an unrealistic test of actual ability.

David Heinemeier Hansson, the creator of Ruby on Rails, describes the process as "whiteboard algorithm hazing."[4]

Max Howell adds, "Google: 90% of our engineers use the software you wrote (Homebrew), but you can't invert a binary tree on a whiteboard so f–– off." Someone else in that discussion mentioned they regularly look up material in books *they wrote*.[5]

How does this elitist whiteboard hazing work against people? It selects for recent college graduates who have studied-up on those algorithms. It works against people with real-world experience who have been writing software for a decade or four. It also works against people who did not receive a very-costly education, including working against graduates of coding boot camps. It works against people who are already employed full time, or have other full time responsibilities, with little extra time to study-up on whiteboard answers.

At best, the perpetrators of this system are implicitly selecting for people just like themselves.

Aline Lerner, in *You can't fix diversity in tech without fixing the technical interview* presents charts showing ways this process adversely impacts diverse groups. She explains the process, statistically,

proves to be non-deterministic. Interviewees cannot tell how they did; results are arbitrary. Impostor Syndrome kicks in, and those susceptible stop interviewing altogether.[6]

> At the end of the day, because technical interviewing is indeed a game, like all games, it takes practice to improve. However, unless you’ve been socialized to expect and prepare for the game-like aspect of the experience, it’s not something that you can necessarily intuit.
>
> Also, if you go into your interviews expecting them to be indicative of your aptitude on the job, which is, at the outset, not an unreasonable assumption, you will be crushed the first time you crash and burn. But the process isn’t a great or predictable indicator of your aptitude. On top of that, you likely can’t tell how you’re doing even when you do well.

Lerner also explains the origin of puzzles in software developer interviews. Fortunately, I have not seen such puzzles lately, but ten years ago I was passed a wood-and-string puzzle during an interview. We continued talking. I soon passed it back, taken apart (i.e., solved). The interview team was surprised and said none of their other candidates had solved it. I did not mention I had seen and solved it before.

This broken interview process does not serve the employer well either. We have a concrete well-established process for selecting people who can solve a specific puzzle. We don’t seem to have any such process for the more important skills such as communicating with other people or dealing with unexpected situations, let alone designing for those unexpected situations.

There is a pattern here, and it is nefarious. But to identify that pattern we must first step back in time to 1983.

Insecurities

Home computers appeared in the 1970s. By 1982, Radio Shack had sold over 100,000 of their TRS-80 home computer. Apple worked with MECC, the Minnesota Educational Computing Consortium, to get Apple computers placed in public schools. The plan worked. You may recall some beloved MECC titles like Oregon Trail.

This advent of the TRS-80 (Figure 1.2, “TRS-80 Model 4”), Apple II, Commodore, Atari, and other home computers, transformed our view of our own profession in that this caused fear, fundamental endemic fear, for our jobs, our livelihood. The job fears were real because this was the first generation of children with access to computers. Computer programming had now been a profession for three or four decades. Now, teenagers potentially had equal footing with adults in the job market.[7]

Figure 1.2. TRS-80 Model 4

Figure 1.3, “Apple II+, late 1982,” shows my son Jakob with his first home computer the Apple II+. Purchase price, with additional hardware and software add-ons, was around $2,100 (price in 2022 would be $6,200). He did not yet know how to type, so I took care of it for him for several years. The monitor is to the right of the keyboard. The floppy disk drive is atop the main unit containing motherboard, extension cards, and keyboard. The gray plastic file box on its left was storage for the 5-1/4“ floppy disks. There’s a box of spare floppy disks (for backups) to the right of the monitor. That is a Cray Assembly Language coding form underneath the Cray Research coffee mug. I had been using coding forms to write out the code long-hand for more than 12 years by this point, so using the pre-printed form was second nature. The gold USAF Academy bathrobe came in handy during our first Minnesota winter!

Figure 1.3. Apple II+, late 1982

This may sound ridiculous to the modern ear because many of us have heard questions like “Why should I pay you to build webpages when my 12-year-old could do it?” But in 1983 computer programmers, to this point, were much like rocket scientists. Neither was a profession one could easily learn in high school. Rocket scientists required the resources to design and build towering rockets. Computer programmers depended on entire rooms full of computing equipment.

Meanwhile, the U.S. Constitution’s proposed Equal Rights Amendment was nearly ratified by its original 1979 deadline, receiving 35 of the needed 38 state ratifications. Then Phyllis Schlafly (in my opinion) single-handedly defeated the ratification. The activism and upheaval became part of an ongoing battle, “the battle of the sexes,” which took its name from the highly-publicized exhibition tennis match between Billie Jean King and Bobby Riggs in 1973.

Why do we care? Because these events converged on the lackluster book *Real Men Don’t Eat Quiche: A Guidebook to All That is Truly Masculine*, a bestseller throughout the summer of 1982. This book poked fun at the concept of “Real Men,” contrasting them to men getting more out of life, the “quiche eaters.”[8]

> Quiche eater: “A man who is a dilettante, a trend-chaser, an over-anxious conformist, one who eschews (or merely lacks) the traditional masculine virtues.”

Let me be clear here, lest we miss the context a generation later: The book is a JOKE. A satire. A way of poking fun at real insecurities faced by our society at the time.

Ed Post of Tektronix continued the “quiche eater” joke. He changed the phrase from “Real Men” to “Real Programmers” and spun an over-the-top tale in honor of computer programmers, *Real Programmers Don’t Use Pascal*. This story entered the annals of “hacker folklore” wherein the hero accomplished feats not to be expected from mere mortals.

Post’s underlying message was that “real” computer programmers should not be replaced by the younger generation. He begins, as they all do, by talking of the “good old days” which were really never that good:

> Back in the good old days, the “Golden Era” of computers, it was easy to separate the adults from the children, sometimes called “Real Men” and “Quiche Eaters” in the literature… During this period, the Real Programmers were the ones who understood computer programming, and the Quiche Eaters were the ones who didn’t. The Real Programmer is in danger of becoming extinct, being replaced by high-school students playing Pac-Man with TRASH-80s!

Did you catch the connection? The “Real Programmer” is exactly the person who could handle the “Fizz Buzz” challenge. Fizz Buzz, and the whiteboard interview, explicitly selects for Real Programmers who “understand computer programming.”

Post sounds the alarm:

> There is a clear need to point out these differences: Help employers of Real Programmers to realize why it would be a mistake to replace the Real Programmers on their staff with 12-year-old Pac-Man players (at a considerable salary savings).

Post then enumerates our differences, giving voice to our fears.

> The easiest way to tell a Real Programmer from the crowd is by the programming language he or she uses. Real Programmers use FORTRAN. Quiche Eaters use PASCAL. Real Programmers actually talked in capital letters, you understand.
>
> Niklaus Wirth, the designer of PASCAL, was asked, “How do you pronounce your name?”
>
> He replied, “You can either call me by name, pronouncing it ‘Veert,’ or call me by value, ‘Worth.’” One can tell immediately from this comment that Niklaus Wirth is a Quiche Eater.

In other words, Real Programmers are Serious, and don’t do cute. To fully appreciate the joke, you need to know that one of the differences “under the covers” between FORTRAN and PASCAL was “call by name” versus “call by value.”

Wirth was making a joke about a joke. You will not be surprised, therefore, to learn Wirth's comment has itself made it into the annals of hacker folklore. Nowadays, we could change PASCAL to PHP and change FORTRAN to JAVA with similar effect.

There is an elitist irony here regarding COBOL. COBOL (an acronym for Common Business Oriented Language) was the business language of choice well into the 1980s. Because most large mainframes were purchased for business use, COBOL had by far the largest market share of all “higher level” programming languages. But COBOL programming was not “cool” like FORTRAN. Nor was it “cool” like the object-oriented languages appearing from the 1990s onward. But, funny thing, there likely are still more lines of COBOL running in production in 2025 than all “cooler” languages combined!

Pascal arrived in 1974 as the weirdly-titled book *PASCAL User Manual and Report*.[9] Once again, context is everything. The weird title is a nod to the programming language Algol. All modern C-like programming languages derive from Algol. Algol-58 was described in the 1958 *Preliminary report*,[10] and Algol-60 by the *Revised Report on the Algorithmic Language Algol* 60.[11]

The Pascal book's original title points out another weirdness of the time. FORTRAN (Formula Translation), COBOL (Common Business-Oriented Language), BASIC (Beginner's All-Purpose Symbolic Instruction Code), and ALGOL (Algorithmic Language) were all acronyms and therefore written with upper-case letters. Pascal, for whatever reason, was introduced as PASCAL. Programmers really did talk in capital letters back then!

Wirth's own evolution of Algol-60, which he called Algol-W, was declined by the Algol-68 committee in favor of a more complex language. Wirth moved on to develop Pascal, announcing it with a nod to the “real” Algol (by using the word “Report” in the book title). Such are the politics of computer science.

Pascal thus arrived just in time to be adopted by quiche-eating Computer Science departments and laughed-at by Real Programmers standing fearfully next to their TRASH-80s.

Post's essay disparaged the then-modern theory behind Computer Science (the idea of creating abstractions):

> Real Programmers don't need abstract concepts to get their jobs done. They are perfectly happy with a keypunch, a FORTRAN IV compiler, and some pizza. If you can't do it in FORTRAN, do it in assembly language. If you can't do it in assembly language, it isn't worth doing.

Post's essay is brilliant, describing the many characteristics and situations of our heroic computer programmer. We learn:

> The Real Programmer is capable of working 30, 40, even 50 hours at a stretch, under intense pressure. In fact, he prefers it that way... If there is not enough schedule pressure on the Real Programmer, he tends to make things more challenging by working on one small but interesting part of the problem for the first nine weeks, then finishing the rest in the last week, in two or three 50-hour marathons. This not only impresses the hell out of

> his manager, who was despairing of ever getting the project done on time, but creates a convenient excuse for not doing the documentation.
>
> A real programmer might or might not know his wife’s name. He does, however, know the entire ASCII (or EBCDIC) code table.

This attitude remains true but harmful. Eugene Kim’s article “Yahoo CEO Marissa Mayer explains how she worked 130 hours a week and why it matters”:[12]

> She’s regularly pulled off all-nighters during her time at Google, and even worked from her hospital bed shortly after having her twins last year… Mayer says hard work is an important part of any business and an often overlooked part of Google’s success.
>
> “Could you work 130 hours in a week?” The answer is yes, if you’re strategic about when you sleep, when you shower, and how often you go to the bathroom. The nap rooms at Google were there because it was safer to stay in the office than walk to your car at 3 a.m. For my first five years, I did at least one all-nighter a week, except when I was on vacation—and the vacations were few and far between.
>
> Being there on the weekend is a huge indicator of success, mostly because these companies don’t just happen. They happen because of really hard work.

There is a pattern here. We are *not* selecting for work/life balance, or a diverse workforce. Whether unintentionally or not, we are actually selecting for mental health problems, burnout, and higher rates of project failure. As we saw in “Real Programmers Don’t Use Pascal,” this pattern began at least half a century ago.

Meanwhile, what *are* Fizz Buzz, whiteboard hazing, and heroic folklore all selecting for? I have a theory. I fit the profile.

Profile Describing a Real Programmer

Different people think differently. The Myers-Briggs® Type Indicator (MBTI) attempts to standardize this observation, claiming that people falling into the same category (there are 16 categories) have some characteristics in common.[13]

The MBTI is not considered as authoritative or accepted as it once was. My point here is to note that different people think differently and that some people do have similar characteristics. It is often true that “birds of a feather flock together.”

Suppose that Fizz Buzz, whether intentionally or not, precisely selects for a very specific profile. Suppose it is not selecting the “best” 0.5% of the general population, as intended, but is instead selecting members of a very specific population. It is no coincidence that this selected population displays the same characteristics described as a Real Programmer.

For example, I have asked a few people I know who identify as INTJ on the Myers-Briggs scale. I asked if they are good at identifying patterns. The answer has always been yes. It is a small sample

size, fewer than a dozen people, but I find the result interesting because I too recognize patterns others do not. It is a useful skill in the hard sciences, including hard-core Real Programming.

From what I can remember of the Stanford-Binet evaluations (the “IQ test”), it too selected for these same sorts of characteristics. I remember that one of the sections was to be verbally told a string of digits, and I was to repeat them back in reverse order, with the strings of digits becoming longer and longer.

That is the sort of trick that comes naturally to a Real Programmer whereas other more normal tests are outside a Real Programmer’s comprehension or interest.

As another example, I have been telling people for years that it is really important to understand how computers and software work. I have *always* taken advantage of *my* knowledge in that area. So I have blithely assumed anyone else in our profession with similar knowledge would *also* use that knowledge to advantage.

Post’s essay shares a similar assumption:

> What of the future? It is a matter of some concern to Real Programmers that the latest generation of computer programmers are not being brought up with the same outlook on life as their elders. Many of them have never seen a computer with a front panel. Hardly anyone graduating from school these days can do hex arithmetic without a calculator.

Decades later, it does appear the software industry has survived, even though it remains true that hardly anyone graduating from school these days can do hex arithmetic without a calculator.

I interpret the Fizz Buzz premise to mean that 99.5% of current software developers do not match the “Real Programmer” profile. Post’s essay, in effect, proclaims the Real Programmer is a rare phenomenon. The Jargon File 4.4.7 notes:[14]

> A Real Programmer’s code can awe with its fiendish brilliance, even as its crockishness appalls... and terrify the crap out of other programmers—because someday, somebody else might have to try to understand their code in order to change it. Their successors generally consider it a *Good Thing* that there aren’t many Real Programmers around anymore.

Modern tech interviews are seeking Real Programmers whether they realize it or not. They are simply interviewing “the way it has always been done” without realizing the horrific premise behind Fizz Buzz. The interviews are, therefore, excluding most of humanity, and that particularly means excluding anyone with a life outside of work. Schools, even down to kindergarten and earlier, are pushing for computer-programming literacy, yet we continue to select only for Real Programmers. Our selection process is definitely working against our own best interests.

This, I believe, is why the interviewers are consistently surprised when they *do* encounter someone fitting that profile. That person may or may not be a useful employee once hired, but it is an easy guarantee that he or she can do Fizz Buzz variations all day long, in octal or hexadecimal.

Track and Field

I do not want to give the impression that anyone can become an expert, or master a skill by merely reading a book. Mastery requires work—and determination. It often requires a mentor and an accountability partner.

My senior year in high school was my fourth year on the high school cross-country team. I was so bad that I received the “most improved runner” award *twice* during those four years. Steve, the other senior on the cross-country team that fall, was the team captain.

Figure 1.4, “Author, Center Left With Steve in Front, Center Right,” shows Steve and I during the mile run during Track season in the spring. Our uniforms included white socks with the team colors, green and gold.

Figure 1.4. Author, Center Left With Steve in Front, Center Right

Colleen, now starting her junior year in high school, had performed exceptionally well during the spring track-and-field season. Those boys who ran cross country in the fall usually also competed on the track team in the spring. Girls had no such option; girls had no cross-country team, only track.

To put this into the right historic context, Title IX had only become law two years earlier. This was

the summer/fall of 1974; Title IX became law June 1972. Title IX prohibited sex discrimination in any educational program or activity receiving federal financial aid.[15]

Not until the following summer, July 1975, would Title IX federal regulations be issued in the area of athletics. At that point both high schools and colleges were given three years to comply.

Colleen received permission to join the cross-country team as the only girl, competing amongst the boys. Each of the high schools had a varsity squad and a junior varsity (JV) squad. Varsity competed against varsity, and JV competed against JV, at the various meets.

The Varsity rule was quite simple. If you could keep up, you ran varsity. If you choked, you ran JV and somebody else got to take your spot. Colleen ran exclusively JV, so there was no concern about her taking one of the coveted varsity spots.

Figure 1.5, “Timberline High School Cross Country Team, Late 1974,” shows Colleen at front center. I’m at the back left next to Coach Bykerk. Steve is at back right. The bright yellow jackets indicate the varsity team, and they were numbered 1 through 7. I wore 3 that week and Steve wore 4, but it was usually the other way around. The dark green uniforms were the JV. Normal team size was about double what’s in this photo.

Figure 1.5. Timberline High School Cross Country Team, Late 1974

Colleen, to the best of my recollection, was not there to break barriers. She was there to train in the off-season. Her focus was on the half-mile and mile events during Track season in the spring. It

was pointless for her to train with other girls; they literally could not keep up. Only the boys could push her to her full strengths.

Once cross-country season was over, the three of us (Colleen, Steve, and I) continued to train. We each had our reasons, but we will continue Colleen's story first. Since my mother worked at the high school, it was easy for us to run the 5-6 miles to school every morning. I drove over to each of Steve's and Colleen's houses, picked them up, and left the car at my house with our school clothes in the car. Mom drove the car to school while we were running the route. We then got our clothes from the car at school, showered, and were ready for class.

Winter in western Washington State is known as "the Dark Ages." It is cold, damp, rainy. It is dreary and dark. Many were the times when we teenagers did not feel like running that morning. It was dark and early; we had to run the 5-6 miles and be there in time to shower and be ready for the first class.

The funny thing is, none of us wanted to disappoint the others. We ran to school anyway. We were each accountability partners for the others. We didn't call it that, of course, but that is what we were.

Steve and I, naturally, kept an eye on Colleen's competitions that spring. We knew her body language and running style. We had been there with her, literally every step of the way that year. She, for her part, had spent her winter running with the varsity.

A typical high school track is a quarter-mile oval. It is two laps for the half-mile, four laps for the mile, and for boys, eight laps for the two-mile run. There are strategies for when to be in front, when to be in the back of the pack, whether to pass on the outside or inside, how to handle moving into or coming out of the turn.

For Colleen, however, there was no strategy. She was so far in front of everyone else that there was nothing to discuss. She would not meet anyone else to challenge her until the state championships. Her only challenge was the stopwatch.

That year, as a junior, she placed fourth in the State of Washington for the half-mile and placed fifth in the mile with a time of 5:42.3. The winning time was 5:35.5. The next year, as a senior, she placed sixth with a time of 5:30.6. If she had run that time the previous year, she would have won the event by a clear five seconds!

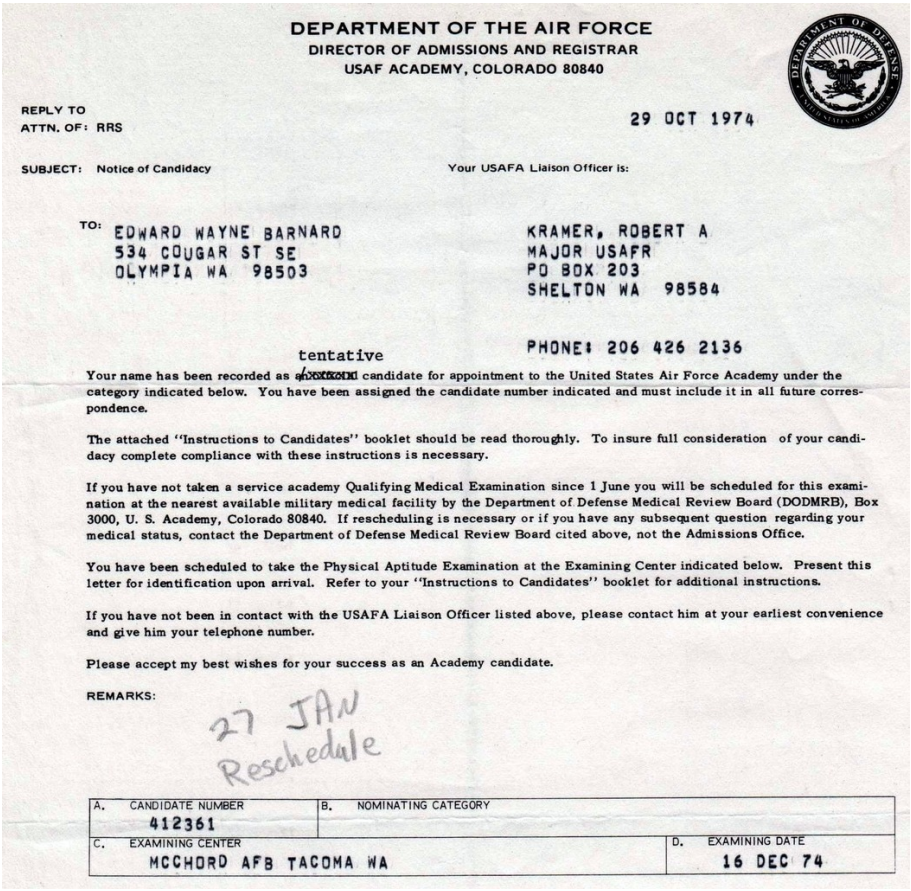
High school graduation, for Steve and I, was June 7, 1975. That would seem trivial, but it was not. Colleen, a junior, was part of the honor guard. She would be standing on the stage for the whole ceremony while Steve and I sat in the audience, basking in our graduating glory. Or so we thought.

Graduation was in the afternoon. That morning the three of us competed in the Third Annual Sound-to-Narrows 7.5-mile run with plenty of hills. (It is now advertised as Washington State's oldest 12K run.) Colleen, to the best of my recollection, came in first for all prep (high school age) women. After the race, we jumped in the car and zoomed to be ready for graduation. Steve and I both had our leg muscles seize up trying to sit still. We could not imagine the agony Colleen was enduring, having to stand still, on stage with everyone watching!

We each had our own reasons for training together that winter. Colleen needed training partners who could keep up with her, even push her. Neither Steve nor I would ever compete in the state finals, but it was fun knowing that she would.

My own reason had nothing to do with track and field. I was preparing for the basketball throw.

I was preparing for the United States Air Force Academy (USAFA)’s arduous admissions process (Figure 1.6, “Orders for Physical Aptitude Examination”). The Physical Aptitude Examination, now called the Candidate Fitness Assessment on the USAFA website, includes a basketball throw.[16] That sounds easy, right? On the contrary, it is weird. It is much like Fizz Buzz.



DEPARTMENT OF THE AIR FORCE
DIRECTOR OF ADMISSIONS AND REGISTRAR
USAF ACADEMY, COLORADO 80840

REPLY TO
ATTN. OF: RRS

29 OCT 1974

SUBJECT: Notice of Candidacy

Your USAFA Liaison Officer is:

TO: EDWARD WAYNE BARNARD
534 COUGAR ST SE
OLYMPIA WA 98503

KRAMER, ROBERT A
MAJOR USAFR
PO BOX 203
SHELTON WA 98584

PHONE: 206 426 2136

tentative

Your name has been recorded as a tentative candidate for appointment to the United States Air Force Academy under the category indicated below. You have been assigned the candidate number indicated and must include it in all future correspondence.

The attached “Instructions to Candidates” booklet should be read thoroughly. To insure full consideration of your candidacy complete compliance with these instructions is necessary.

If you have not taken a service academy Qualifying Medical Examination since 1 June you will be scheduled for this examination at the nearest available military medical facility by the Department of Defense Medical Review Board (DODMRB), Box 3000, U. S. Academy, Colorado 80840. If rescheduling is necessary or if you have any subsequent question regarding your medical status, contact the Department of Defense Medical Review Board cited above, not the Admissions Office.

You have been scheduled to take the Physical Aptitude Examination at the Examining Center indicated below. Present this letter for identification upon arrival. Refer to your “Instructions to Candidates” booklet for additional instructions.

If you have not been in contact with the USAFA Liaison Officer listed above, please contact him at your earliest convenience and give him your telephone number.

Please accept my best wishes for your success as an Academy candidate.

REMARKS:

27 JAN
Reschedule

A. CANDIDATE NUMBER	B. NOMINATING CATEGORY	
412361		
C. EXAMINING CENTER		D. EXAMINING DATE
MCCHORD AFB TACOMA WA		16 DEC 74

Figure 1.6. Orders for Physical Aptitude Examination

The basketball throw tests how far you can throw a basketball while kneeling on your knees. It is an unusual motion. You cannot use your whole body to achieve distance because your knees must remain straight, behind a line. There is a scooping motion, and a rhythm, needed to throw that basketball for distance. It then takes months to develop the strength behind that motion.

That is why the USAFA can immediately tell who practiced the basketball throw month after month, and who did not. They are seeking the young men and women most determined to pass.

My high school’s wrestling coach agreed to set me up with a year-long training program aimed at passing the Physical Aptitude Exam. I was not even on the wrestling team! But he was the right person for the individualized program. Always be open to finding the right mentor in unexpected places.

My coach set up the strength training for the pull-ups, push-ups, shuttle run, and sit-ups. We

cleared my use of the high school gym for solo rope climbs and tossing all the basketballs from one side of the gym to the other, and back again.

The USAFA is completely clear about their intentions with the assessment. Once you are exhausted from all the strength tests, *then* you will undertake a one-mile run. *That* is why I was out every morning running to school with Steve and Colleen. Mastering any skill, whether physical or mental, requires planning and practice. Mastery comes when you practice at all times, in all seasons, under all conditions.

Thanks to Steve and Colleen—and the wrestling coach, and mom—come January 27th of my senior year, I passed. Now I had a new goal: Survive Basic Cadet Training which would begin late June. We continued our daily runs to school.

Notes

1. “Fizz Buzz Test.” Accessed December 27, 2024. http://wiki.c2.com/?FizzBuzzTest.

2. Coding Horror. “Why Can’t Programmers.. Program?,” February 26, 2007. https://blog.codinghorror.com/why-cant-programmers-program/.

3. Jeffries, Adrianne. “Programmers Are Confessing Their Coding Sins to Protest a Broken Job Interview Process.” The Outline. Accessed December 27, 2024. https://theoutline.com/post/1166/programmers-are-confessing-their-coding-sins-to-protest-a-broken-job-interview-process.

4. DHH [@dhh]. “Hello, My Name Is David. I Would Fail to Write Bubble Sort on a Whiteboard. I Look Code up on the Internet All the Time. I Don’t Do Riddles.” Tweet. Twitter, February 21, 2017. https://x.com/dhh/status/834146806594433025.

5. Max Howell [@mxcl]. “Google: 90% of Our Engineers Use the Software You Wrote (Homebrew), but You Can’t Invert a Binary Tree on a Whiteboard so Fuck off.” Tweet. Twitter, June 10, 2015. https://x.com/mxcl/status/608682016205344768.

6. Lerner, Aline. “You Can’t Fix Diversity in Tech without Fixing the Technical Interview.” Diversity Together (blog), August 9, 2018. https://medium.com/diversity-together/you-cant-fix-diversity-in-tech-without-fixing-the-technical-interview-597250e8564d.

7. Licensed creative commons. TRS-80

8. Feirstein, Bruce, and Lee Lorenz. *Real Men Don’t Eat Quiche: A Guidebook to All That is Truly Masculine*. New York: Pocket Books, 1982.

9. Jensen, Kathleen, Niklaus Wirth, P. Brinch Hansen, D. Gries, C. Moler, G. Seegmüller, N. Wirth, G. Goos, and J. Hartmanis. *PASCAL User Manual and Report*. Berlin, Heidelberg: Springer-Verlag, 1974.

10. Perlis, A. J., and K. Samelson. “Preliminary Report: International Algebraic Language.” Commun. ACM 1, no. 12 (December 1, 1958): 8–22. https://doi.org/10.1145/377924.594925. The online article is marked “free access” as of December 2024.

11. Backus, J. W., F. L. Bauer, J. Green, C. Katz, J. McCarthy, A. J. Perlis, H. Rutishauser, et al. “Revised Report on the Algorithmic Language ALGOL 60.” Commun. ACM 6, no. 1 (January 1, 1963): 1–17. https://doi.org/10.1145/366193.366201. The online article is marked “free access” as of December 2024.

12. Kim, Eugene. “Yahoo CEO Marissa Mayer Explains How She Worked 130 Hours a Week and Why It Matters.” Business Insider. Accessed December 27, 2024. https://www.businessinsider.com/yahoo-ceo-marissa-mayer-on-130-hour-work-weeks-2016-8.

13. “Myers-Briggs® Overview.” Accessed December 27, 2024. https://www.myersbriggs.org/my-mbti-personality-type/myers-briggs-overview/.

14. “Real Programmer.” Accessed December 27, 2024. http://www.catb.org/jargon/html/R/Real-Programmer.html.

15. Women’s Sports Foundation. “History of Title IX.” Accessed December 27, 2024. https://www.womenssportsfoundation.org/advocacy/history-of-title-ix/.

16. U.S. Air Force Academy. “U.S. Air Force Academy.” Accessed December 27, 2024. https://www.academyadmissions.com/.

Chapter 2. Think Like a Computer

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Primary

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Popular Electronics

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Turing Tumble

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Crypto

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Flow

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Journey Versus Destination

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 3. What Can the NSA Teach Us About Debugging?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Digital Computers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Intuition

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Superpowers

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 4. Problem in a Box

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Once More

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Loud Whiteboards

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Fitting the Problem Into a Box

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Task Main Loop

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 5. Software Development

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Vectorizing Software

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Seymour's Memo

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

On-Line Tape Software

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 6. My NSA Mug Shot

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

But Where...?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Safe Arrival

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Round Tapes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Channel Extender

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Horseshoes

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Patterns in the Noise

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Channel Commands

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Spotting Weirdness

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Expressing Expertise

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 7. It Was Nothing, They Said

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Dash Pig

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Leap Day

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Y2K Bug

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Perspective

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Wizards

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Magnetic Tape

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Deadline

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Modern Legacy Code

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 8. Ducky Day

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Cray Style

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Octal

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Storytelling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 9. The Transitive Property of Keeping Your Mouth Shut

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Minnesota

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Valley Girls

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Transitive Property

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

9-Track

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

One More Play

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

My Turn to Teach

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Equality

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 10. The Veil

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Initial Visit

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Saudi Hospitality

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Feast

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 11. Oil Across the Water

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Revisionist History

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Grapevine

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Standard Oil

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Relationship

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Video and References

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 12. Desert Shield

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Eastern Province

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Notary Public

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 13. Remembering “Scuzzy”

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Pre-Release IBM Hardware

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Software Division

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 14. Gate Keeping

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Burnout

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Gate Keeping

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Unix Guru

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

On the Shoulders of Giants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Dragon Wrangling

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Chapter 15. Imposter Syndrome

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Imposter Syndrome

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Different Way of Thinking

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Software Training

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Paying it Forward

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Wizard Thinking

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Dark Side

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Shoulders of Giants

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

The Gatekeeper

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Prohibition-Era John Scarne

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

Resources

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.

What Do You Hear?

This content is not available in the sample book. The book can be purchased on Leanpub at https://leanpub.com/dragons-wizards.