The Heroic Hacker: Legends of the Computer Age

Erik Brunvand
Department of Computer Science
University of Utah, SLC, Utah 84112
elb@cs.utah.edu, www.cs.utah.edu/ elb

October 15, 1996

Abstract

The computer hacker has been depicted in the popular press as a socially maladjusted teenager whose goal is to wreak malicious havoc on unsuspecting computer users. In the culture of the computer programmer however, the hacker takes on a far different aspect. The true hacker is raised to heroic status with tales of amazing feats circulated through computer networks in the form of stories and legends. In this paper I discuss the persona of the true hacker, illustrated using hacker legends collected from the Internet.

1 Introduction

You have probably read about computer hackers in the newspaper and seen stories on the evening news about gangs of teenage hackers breaking into computer systems and causing tremendous damage. If so, then your impression of what it means to be a hacker is likely be quite negative. The popular press tends to depict the hacker as a criminal, or at the very least someone who enjoys causing lots of people lots of trouble. He is usually teenage and very smart, but also socially outcast; the overall impression is often one of dangerous intellect of the sort you typically see only in spy movies. In the media's version, the goal of a hacker is often simply to cause computer systems to crash, but sometimes the goals are more directly criminal. Stealing long-distance phone service, credit card numbers, financial information, or other valuable forms of information are all targets of these dangerous individuals.

The term "hacker" has a very different meaning, however, amongst computer programmers. Although the media version of a hacker certainly exists, by far the majority of programmers do not engage in criminal activity. To the working programmer, the term "hacker" is much more likely to be used as a term of respect for another's programming abilities than as an insult. A well-known on-line lexicon of programming terminology, known as either The Jargon File or The Hacker's Dictionary [6], defines a hacker as:

> hacker n. [originally, someone who makes furniture with an axe] 1. A person who enjoys exploring the details of programmable systems and how to stretch their capabilities, as opposed to most users, who prefer to learn only the minimum necessary. 2. One who programs enthusiastically (even obsessively) or who enjoys programming rather than just theorizing about programming. 3. A person capable of appreciating hack value. 4. A person who is good at programming quickly. 5. An expert at a particular program, or one who frequently does work using it or on it; as in 'a UNIX hacker'. (Definitions 1 through 5 are correlated, and people who fit them congregate.)

Because the understood definition of "hacker" relates to a specific attitude towards programming as well as a level of programming talent, not all programmers, not even all good ones, are considered hackers. The Jargon File goes on to say that:

> It is better to be described as a hacker by others than to describe oneself that way. Hackers consider themselves something of an elite (a meritocracy based on ability), though one to which new members are gladly welcome. There is thus a certain ego satisfaction to be had in identifying yourself as a hacker (but if you claim to be one and are not, you'll quickly be labeled bogus)

Implicit in the notion of a meritocracy is the fact that some members of the group represent higher or lower levels of ability within the basic qualifications for group membership. As a programmer, to be known as a "true hacker" is a compliment indeed.

As with any community, programmers have their own heros. If a hero is a figure who is larger than life, one who is endowed with greater skills and powers than ordinary persons, then a hacker hero is one who exhibits tremendous abilities in the aspects that hackers value. The heroic figures in hackerdom are those whose programming feats amaze even skilled programmers. A hacker of this magnitude can, through his programming prowess, make the computer do things that seem magical even to one skilled in the art of programming. Legends of these feats are circulated throughout the community. In this case the communication media are an integral part of the community's identity. Email, newsgroups, the Internet, and the world wide web comprise the virtual home of the hacker community. Physically disparate but connected intimately through these high-speed electronic networks, the hacker community tells legends of their heros' exploits around the high-tech equivalent of a campfire: a glowing computer display.

2 The Culture of the Programmer

To understand the heroic qualities of the legendary hacker, it is important to understand at least a little about the culture from which they come. The culture of the computer hacker is very young. Although I'm sure that there are analogous communities of people skilled in similarly complicated disciplines, the computer hacker culture is connected intimately with the electronic stored-program computer. Computers of this type have existed only since the 1950's. The stored-program qualifier is important. Although it seems obvious today, the idea that a program that controls the actions of the computer can be stored in the same type of memory that is holding the data that the program was working on was in fact a major leap of understanding for the early computer designers.

Computers are built from electronic components that, while complex in their manufacture and their organization into systems, are essentially nothing more than very small, very fast switches. Switching theory, which is the science of studying systems built from switches, is far older than computer science. Telephone systems are an example of a switching network where calls must be routed through a switching network to find the desired receiver. Railroads are another example of a switching network. Consider the problem of making sure that two trains will not try to use the same piece of track at the same time but in opposite directions. The railroad system is a system of switches that control where the trains go and which section of track they will use.

Both of these examples are related to the early examples of hacker culture, but the railroad example is of particular importance. The first computer hackers are generally agreed to have been students at MIT in the 1950's who were part of the Tech Model Railroad Club (TMRC) [8]. This club had (and probably still has) a huge model train layout that was the focus of the club activities. The subcommittee of the club responsible for the complicated wiring of the switches that controlled where the trains went were hackers waiting to happen. The switching network of the TMRC railroad was essentially a special purpose computer, but one where the programming was done either by changing wires, or by real-time control of the switch settings by the operator.

When these railroad-switch-hackers discovered the computers that were being installed at MIT in the 1950's, some of them immediately realized that these large, cumbersome calculating machines were exactly the type of things they had been struggling to create with the model railroad layout. Some would become enthralled with trying to figure out what they could make the computers do. This exploration, for the type of person who would come to be called a hacker, would be simply out of an intense desire to explore the limits of the technology. If the explorations resulted in programs that were useful to others, so much the better, but the motivation of the exploration was simply to gain information about and control over the computer. From the beginning, the hackers found themselves outside the traditional scientific community that was also trying to understand what could and could not be done with these new-fangled computers. It was the hackers, however, that understood how far the technology could really be pushed.

It was the hackers, working in the early hours of the morning because that was the only time they could get access to the machines, who first demonstrated that a computer could do the then unthinkable task of playing a game of chess. Hackers first demonstrated that the computer display could be used in an interactive way. That it, rather than simply be used for static status information on the machine, you could cause the machine to draw shapes that looked like spaceships and play games by causing the ships to move around on the screen and interact with other shapes. Hackers were the first use the computer in an interactive way for entering text. To an engineer of the day, using valuable computer time to simply enter text in an interactive text editor was unthinkable. It was the hackers who understood that interacting with the computer, rather than treating it as a hands-off behind-glass computation resource, was the way of the future.

Because they were operating outside of the official engineering and scientific culture of the day, the hacker developed a bit of an outlaw persona from the first. It's important, though, to make a distinction between a notion of outsider and that of a criminal. Prototype hackers were not criminal; they were simply operating outside the bounds of the established culture of the day. Working mainly at night, without formal scientific backgrounds or training, hackers were simply exuberant, fearless, non-conventional explorers of this new world of computing.

2.1 A Little Bit of Hacker History

The early history of hackers is centered around MIT in the 1950's and 1960's. Naturally curious and intelligent MIT students who had been exploring the phone switching network and the control systems of the Tech Model Railroad Club were drawn to the computers of the MIT Artificial Intelligence Lab (MIT AI Lab). The director of the lab, Marvin Minsky, was sympathetic to the hackers' desire to explore and impressed enough with their accomplishments that he allowed them to have direct access to the machines, even though the true hackers among the group had by then dropped out of school to spend more time hacking. Legendary hacker figures from this time include Peter Deutsch, Bill Gosper, Richard Greenblatt, Tom Knight, and Jerry Sussman.

This was the "golden age" of the computer hacker. The machines were large, slow, cumbersome to use, and it took an extraordinary effort to make them do even the simplest computation. Although this less than 40 years ago, to the programmer of today it is like looking through the mists of time into the genesis of computing. The legendary feats of the early hackers are made all the more amazing by the primitive nature of the machines they were using and the tools they had at their disposal.

As computers spread to other parts of the country, so did the hacker culture. Largely initiated by hackers who had their beginning at MIT, the mid 1960's saw centers of hacker culture develop at other universities such as Carnegie Mellon University, and Stanford University. The Stanford AI Lab (SAIL), under the direction of John McCarthy, for example, became the center for west-coast hacker activity. When the SAIL machine was finally shut down in 1991, hackers sent an email goodbye message to Internet as if the SAIL machine was itself sending a last farewell to its friends [12]. Even commercial research centers were home to hackers. ATT, Xerox, and others all had programmers of legendary skill working for them. Legendary hackers from this second wave of activity include Ed Fredkin, Brian Reid, Jim Gosling, Brian Kernighan, Dennis Ritchie, and Richard Stallman.

The third wave of hacker activity was born in northern California without direct genealogy to the MIT hackers. It started with the Homebrew Computer Club in the San Francisco bay area. This was a group of electronic hobbyists with a common interest in the then radical idea of building their own computers. Because of the size and cost of the early computers, early hackers were restricted to using a small number of machines built by large companies and installed at universities or industrial research centers. This third wave of hackers wanted their own machines that they could not only program at home, but also build and modify the computer hardware at home. It was this group of hackers, which includes legendary figures such as Lee Felsenstein, Steve Dompier, Steve Wozniak, Steve Jobs, and Bill Gates, that formed the foundation for the entire personal computer industry of today.

3 The Story of Mel: A Real Programmer

In addition to the legends of hacker feats based on the exploits of real life hackers, there are also stories that circulate through computer networks that are not connected to any person in particular. There are also stories that feature a semi-anonymous hacker, perhaps named, but not identifiable as a real person. It is a story of this type that I offer as an example of one of the most well known hacker legends. It is called The Story of Mel, A Real Programmer [10].

I first saw this story in 1984 when it was sent to me through email while I was a graduate student at Carnegie Mellon University. At the time it was not attributed to any author in particular, and was sent to me by a friend as a great story about the good old days of hackerdom. The story is told in the first person as the storyteller remembers an encounter with a True Hacker named Mel. This is, to my knowledge, the most widely circulated hacker legend on the Internet and is often pointed to as the best way to begin to understand the culture of the hacker. It is formated in free verse rather than justified prose and feels to my non-folklore-trained ears a bit like an epic poem in style.

I have since discovered that the story was first posted to a Usenet newsgroup on May 21, 1983. The author, Ed Nather, originally wrote and posted the story in prose format. At some point during its travels through the net it was modified into free verse form. It is this free verse form that is by far the most common form of the story as it continues to circulate through the Internet today.

The introduction to the story gives the motivation for the reminiscence as a rebuttal to an article that makes light of current academic thought in computer programming by asserting that "Real Programmers write in FORTRAN." The actual article to which The Story of Mel refers is one published in the computer magazine Datamation in 1983 entitled "Real Programmers Don't Use Pascal" by Ed Post [11]. Pascal is a computer language designed by Nicklaus Wirth to promote so-called structured programming, which forces the programmer into a logical system structure the makes the program easier to understand and debug [16]. FORTRAN is a computer programming language designed for engineering and scientific code, and is one of the, if not the, oldest computer language still in wide use today. As such it is a fairly primitive language when compared to Pascal, and extremely primitive compared to an even more modern language. Constructs that are easily expressed in a modern programming language (like Java, for example[1]) would require a hacker's skill to code in FORTRAN. The main point of The Story of Mel is that perhaps today's programmers think that programming in FORTRAN is hard, but if you look back just a little bit further you can find an example of a programmer who didn't even need FORTRAN to perform amazing feats. The first few stanzas set the stage for the story to follow:

> A recent article devoted to the macho side of programming
> made the bald and unvarnished statement:
>
> Real Programmers write in FORTRAN.
>
> Maybe they do now,
> in this decadent era of
> Lite beer, hand calculators, and "user-friendly" software
> but back in the Good Old Days,
> when the term "software" sounded funny
> and Real Computers were made out of drums and vacuum tubes,
> Real Programmers wrote in machine code.
> Not FORTRAN. Not RATFOR. Not, even, assembly language.
> Machine Code.
> Raw, unadorned, inscrutable hexadecimal numbers.
> Directly.
>
> Lest a whole new generation of programmers
> grow up in ignorance of this glorious past,
> I feel duty-bound to describe,
> as best I can through the generation gap,
> how a Real Programmer wrote code.
> I'll call him Mel,
> because that was his name.

Mel is most likely a fictional programmer, although the story is told as a true story and uses real machines and real languages as examples. This story is clearly written to be read by computer programmers and as such it is full of technical terms and jargon. The very feats that mark Mel as a Real Programmer described later in the story are amazing only to the person who knows something about programming. In a companion to this paper I have included the entire Story of Mel along with extensive annotations that should make it more accessible to non-programmers.

Mel comes from the golden age of hackers when computers were simple enough to be programmed by writing directly in the binary language of the machine. Digital computers process all information in binary notation. The nature of the switches used to build computers is such that the most fundamental operation a computer can do is simply recognize the difference between the representation of a 1 and a 0. If 1's and 0's are all the machine can understand at the most basic level, then programming directly in the language of the machine involves writing down long strings of 1's and 0's. There is a slight simplification of this notation that allows these numbers to be written in base 16 instead of base 2 (the hexadecimal numbers referred to in the first stanza), but the result is largely the same: a sea of mysterious numbers that is all but indecipherable to the normal person.

In order to make these programs easier to write and understand, they can be abstracted into a symbolic form. The first abstraction is assembly language, where machine operations are described

[1] Note that it takes a great deal of programming skill to write the programs that make a language like Java so powerful. Java was designed by second-wave hacker legend Jim Gosling, now at Sun Microsystems

in words and those words are translated into binary numbers by a program called an assembler. A higher abstraction would be a programming language like FORTRAN. Operations are described in a more abstract way in FORTRAN statements, the FORTRAN program is translated into assembly language by a compiler, and then the assembly language is translated into machine code by an assembler. With every abstraction, however, there is a loss of control. For complete control over the inner workings of the machine, direct expression in machine language may be required. Even in the days in which The Story of Mel is set there were assemblers available, but the amazing things that Mel was able to do required the absolute control that comes from rejecting the abstractions and programming directly in the machine's language.

The story describes how Mel used these numbers to control a cumbersome computer from the late 1950's to play blackjack, and how the teller of the story was assigned years later to understand and modify the program to change the odds in favor of the human player and let the customer win. In the end, the teller of the story discovers some of the amazing tricks used by Mel in writing his programs and decides to give up trying to modify the program: both because finding and changing the targeted part of the program would be very difficult, and also because of his awe and respect for the code of a Real Programmer and his feeling that as the work of a Real Programmer the code should not be altered.

4 Hacker Legends in General

4.1 The Heroic Hacker and the Hacker Ethic

The Story of Mel embodies many of the themes and ideas found in other hacker legends. Mel, like other legendary hackers, is a type of prowess hero whose heroic qualities are a result of his extraordinary programming abilities. The hero of prowess is defined by Coffin and Cohen as: "The values they embody reflect the standards, attitudes, and potentials of his society, and most of these heros are historical."[4]. In the society of the programmer the valued skills are related to the ability to make the computer obey commands. Control over the activity of the computer, especially in ways that are unusual, unexpected, obscure, or particularly clever, are the hallmarks of a hacker.

Like heros of prowess in other societies, the hacker who has such strength in one area may have balancing flaws in other aspects of their abilities. The legendary hacker is likely to be described as fairly inept at dealing with other humans. The tendency to work long or odd hours, a disregard for personal appearance, and inability to communicate effectively with other people who are not themselves hackers are all part of the persona of the hacker. In this way, the heroic hacker is weak in exactly the traditional aspects of a hero in other societies. Strength, size, and attractiveness, traditional aspects of a heroic figure, are specifically not found in the hacker hero. This is another indication of the outsider or counter-culture nature of the hacker. Those values that are appreciated in traditional culture are specifically not found in the hacker. Instead, the hacker's prowess is based on more arcane knowledge: power over the logical yet inscrutable computer.

Another aspect of the traditional hero that is, however, present in the hacker hero is that of virtue. In the case of the hacker the virtue, the moral code, is embodied in the Hacker Ethic. The Hacker Ethic is essentially that information should be freely available to individuals who seek to understand the details of any system. The Jargon File [6] defines the Hacker Ethic as:

> Hacker Ethic, the n. 1. The belief that information-sharing is a powerful positive good, and that it is an ethical duty of hackers to share their expertise by writing free software and facilitating access to information and to computing resources wherever possible.

Another example of the Hacker Ethic can be seen in the license agreement of the interactive text editor Emacs. Emacs was written by legendary hacker Richard Stallman, who prefers to be called by his initials which he uses for his login id, RMS. RMS is a second-wave hacker from MIT who, in addition to other feats of hacking, wrote the hacker's favorite text editor. Emacs is completely extensible and includes a fully functional version of the LISP programming language so that customizations to the editor can be written as LISP programs. Emacs is distributed free of charge by the organization founded by RMS called GNU and includes a novel licensing agreement called the "GNU General Public License" or the "copyleft" agreement [14]. The copyleft agreement says that by using the Emacs program you agree to share what you do with other hackers. According to RMS [15]:

> The legal meaning of the GNU copyleft is less important than the spirit, which is that Emacs is a free software project and that work pertaining to Emacs should also be free software. "Free" means that all users have the freedom to study, share, change, and improve Emacs. To make sure everyone has this freedom, pass along source code when you distribute any version of Emacs or a related program, and give the recipients the same freedom that you enjoyed.

4.2 Historical and Magical Themes in Hacker Legends

Another feature of traditional hero legends also found in hacker legends is a historical setting. The Story of Mel is a good example of a legend about hackers in the past. Like the romantic notions of medieval times and the chivalry of the knights, hacker legends tend to be told about the old days of the computer world. Given that "the old days" are only back to the 1950's at the earliest, it is perhaps surprising how much of a historical feel these stories have. The computer industry has moved so rapidly from its birth approximately 40 years ago, and computer programmers are getting started at a younger and younger age, that the early years seem like distant memories to today's programmers. The machines of the 1950's and 1960's could be thought of as the dragons of the past, mean and unforgiving. The hackers who could tame them are the equivalent of the dragon-slayer: the brave knight who by extraordinary skill was able to gain an advantage over the foe. A hacker legend about the old days might talk about programming "the bare metal," meaning writing programs in the most fundamental machine language as Mel did, without benefit of higher level programming tools, or about the single programming insight that inspired the legend. The legends are also likely to focus on the hardships that the early hacker had to deal with in order to make the computer obey his commands,

The feats of the heroic hacker are sometimes referred to in magical terms. Computers seem to be operating largely by magic to many people, so it is not surprising that the person who wields inordinate skill over that machine might be thought of as a magician of sorts. In a hacker legend, the better the hacker, the more magical the feats. Even the terms used by programmers in their profession have an magical aspect to them. Well-regarded programmers in a company are likely to be referred to as "wizards" or "gurus" by not only their peers but by management as well. I have seen computer professionals for major international companies whose business cards give their position as "UNIX Wizard[2]," or "C Guru[3]." Comments (non-program text meant for documentation) in a hacker's program are likely to refer to particularly obscure commands sequences as "magic," or, if they are particularly obscure even to a hacker, as "deep magic."

---

[2] UNIX is a computer operating system developed by hackers at ATT, and favored by hackers even today over other operating systems.

[3] C is a programming language also developed by hackers at ATT, and also one of the programming languages that hackers seem to prefer over others. For a biblical-style account of its development, see [3]

Another legend about the magical aspects of hacking, this one about hacking hardware rather than software, is told about a machine at MIT [9, 7]. In this legend the storyteller comes across a mysterious switch on the side of the MIT AI Lab's PDP-10 computer. The switch has clearly been installed by a hacker and not a computer technician and is labeled by hand with one position marked "magic" and the other "more magic." The switch is currently in the "more magic" position. The switch is flipped to the other position and the machine crashes instantly. They follow the switch to its source, but the wires don't seem to go anywhere that could influence the machine in any way. So they boot the machine and try again. Again as soon as the switch moves from "more magic" to "magic", the machine crashes. The story ends:

> This time we ran for Richard Greenblatt, a long-time MIT hacker, who was close at hand. He had never noticed the switch before, either. He inspected it, concluded it was useless, got some diagonal cutters and diked it out. We then revived the computer and it has run fine ever since.
>
> We still don't know how the switch crashed the machine. There is a theory that some circuit near the ground pin was marginal, and flipping the switch changed the electrical capacitance enough to upset the circuit as millionth-of-a-second pulses went through it. But we'll never know for sure; all we can really say is that the switch was magic.
>
> I still have that switch in my basement. Maybe I'm silly, but I usually keep it set on 'more magic'.

4.3 Doing the Impossible

Part of the appeal of the hacker legend to the programmer is perhaps the very obscurity of the feats described in the legends. The Story of Mel, for example, goes into great detail about the programming style of the hero, but one would have to be at least a little bit of a programmer to understand what was so amazing. A related aspect is that the hacker of legend is often able to perform some feat with the computer that other programmers, or even the computer designer didn't think was possible.

An example of this is a legend that is told about the designer of a micro-controller chip called the 8051. A micro-controller is like a microprocessor but smaller, less powerful, and designed to be included in other devices. The "computer controlled" fuel injection of your car, for example, is controlled by a micro-controller, and probably by the 8051, it being the most commonly used micro-controller for cars. The story goes that the designer of the 8051 is having trouble with his car. He takes it to a garage where the mechanics connect a diagnostic system to the computer control of the engine. Thinking he will impress them he tells them that he designed the chip they are checking. They proceed to bombard him with questions about the chip that he can't answer. The questions indicate that the mechanics have written new code for his chip to improve the performance of the car that makes use of the 8051 in ways that he, the designer of the chip, can no longer understand. The theme of the hackers surpassing the designer by extending things in ways that were never imagined must surely have parallels to other legends in other cultures.

Another legend about doing the impossible is told about third-wave Apple hacker Andy Hertzfeld. According to Byte Magazine [1]:

> Besides everything else he did to help get the first Macintosh out the door, Andy Hertzfeld wrote all the first desk accessories. Most of these were written in assembly. However, to show that desk accessories could also be written in higher-level languages, Hertzfeld wrote a demonstration puzzle games desk accessory in Pascal[4]. Like its plastic counterparts, users moved squares around until the numbers 1 to 9 were in order. As

[4] Recall that the inspiration for The Story of Mel was an article entitled "Real Programmers Don't Use Pascal!"

time began to get short, the decision was made that the puzzle, at 7KB [7KB = 7168 bytes], was too big (and too game-like) to ship with the first Macintosh. In a single weekend, Hertzfeld rewrote the program to take up only 800 bytes. The puzzle shipped with the Mac.

5 The Dark Side: Hackers versus Crackers

There are, of course, programmers who do fit the media's picture of the computer hacker. These programmers use their skills to cause trouble, crash machines, release computer viruses, steal credit card numbers, make free long distance calls (the phone system is so much like a computer system that is is a common target for computer criminals), remove copy-protection, and distribute pirated software. These people may also call themselves "hackers," leading to more confusion. Hackers in the original sense of the term, however, look down on these sorts of activities. Among the programming community, and to a large extent even amongst the illegal programming community, these people are called "crackers" and their activities known as "cracking" to distinguish it from hacking.

The cracker definitely does not follow the Hacker Ethic. Even among legitimate hackers there are those who add to the Hacker Ethic the belief that system-cracking for fun and exploration is ethically OK as long as the cracker commits no theft, vandalism, or breach of confidentiality. It is the hacker who uses his skill for these latter purposes who has crossed over and become a cracker.

Crackers have their own lore, their own heros, and their own set of ethics distinct from the hackers discussed in this paper. They also have much more space in the popular press.

6 Conclusions

Computer programmers as a subculture of the general engineering and scientific community have their own set of heros with aspects based on the values that programmers respect. These heroic figures, called hackers, are not at all like the popular press version of the computer hacker. Legendary hackers are both real and fictional, but tend to share certain common features: extraordinary programming skill, cleverness in the face of difficulty, an ability to suspend all other activities while producing a solution to a problem, an appreciation for a clever solution to a seemingly insignificant problem, weakness in some other aspect to balance their skill as a hacker, and adherence to some form of the Hacker Ethic. Legends of the exploits of the heroic hacker are passed through the virtual community of the programmer, the Internet, using email, newsgroups, and recently web pages. These communication media allow the community of programmers to be in close contact even though they may be physically separated.

As an example of perhaps the best-known hacker legend, I have discussed The Story of Mel, A Real Programmer. This legend embodies many of the aspects of hacker legends in general, and is often used as a way to define what it means to be a hacker. The legendary status of Mel is seen, for example, in another hacker legend from the Internet, this one about the problems of making code fit into the guidance computer of a satellite before lift-off [13]. The story ends with the following summation:

> The solution was both clever and bletcherous. The overflow bit became part of the opcode. To get the extra opcodes, the program would add-to-memory an accumulator value that would cause an overflow when added to the instruction that was in memory. Since the instructions were in ROM they didn't get modified, but the added bit of the overflow gave 8 more opcodes.

I only presume that these opcodes were powerful enough that the extra instructions required to generate them cost less than the instructions they replaced. I have long since lost the article reprint we used. Dave Parnas, who taught the course, had a great time with this particular machine...

All I know is: Mel would have *loved* it!

References

[1] Noted & notorious hacker feats. *Byte*, September 1995.

[2] Jan H. Brunvand. *The Study of American Folklore: An Introduction*. W. W. Norton & Company, 1986.

[3] Ian Chai and Glenn Chappell. The c programming language. Circulated through email, March 1989. Available as `http://www.cs.utah.edu/~elb/folklore/C-bible.html`.

[4] Tristram Potter Coffin and Hennig Cohen. *The Parade of Heros: Legendary Figures in American Lore*. Anchor Press / Doubleday, 1978.

[5] Wayland D. Hand, editor. *American Folk Legend: A Symposium*. University of California Press, 1971.

[6] The jargon file, v3.2.0. Available at `http://dante.creighton.edu:80/jargon`, as well as many other places on the web.

[7] Kerla Jennings. *The Devouring Fungus: Tales of the Computer Age*. W. W. Norton & Company, 1990.

[8] Steven Levy. *Hackers: Heros of the Computer Revolution*. Anchor Press / Doubleday, 1984.

[9] A story about magic. For one location, See appendix A of The Jargon File. Also available at `http://www.cs.utah.edu/~elb/folklore/magic.html`.

[10] Ed Nather. The story of Mel, a real programmer. This story exists in many forms in many places on the web. For one location, See appendix A of The Jargon File. Also available at `http://www.cs.utah.edu/~elb/folklore/mel.html`.

[11] Ed Post. Real programmers don't use pascal. *Datamation*, 1983.

[12] Take me, I'm yours: The autobiography of SAIL. Circulated through email, January 1991. Available as `http://www.cs.utah.edu/~elb/folklore/SAIL.html`.

[13] Posted to newsgroup alt.folklore.computers, April 1991. Available as `http://www.cs.utah.edu/~elb/folklore/satellite.html`.

[14] Richard Stallman. Available wherever GNU Emacs is installed. Also available at `http://www.gnu.ai.mit.edu/copyleft`.

[15] Richard Stallman. Part of the Frequently Asked Questions (FAQ) file about GNU Emacs. Available at `http://www.eecs.nwu.edu/emacs/faq/7.html`.

[16] Nicklaus Wirth. *Algorithms + Data Structures = Programs*. Prentice Hall, 1976.