```
#======= THIS IS THE JARGON FILE, VERSION 4.2.2, 20 AUG 2000 =======#
 010
 001
 111
```

This is the Jargon File, a comprehensive compendium of hacker slang illuminating many aspects of hackish tradition, folklore, and humor.

The Jaron File is a common heritage of the hacker culture. Over the years a number of individuals have volunteered considerable time to maintaining the File and been recognized by the net at large as editors of it. Editorial responsibilities include: to collate contributions and suggestions from others; to seek out corroborating information; to cross-reference related entries; to keep the file in a consistent format; and to announce and distribute updated versions periodically. Current volunteer editors include:

Yash Tulsyan `yashtulsyan@gmail.com`

Although there is no requirement that you do so, it is considered good form to check with an editor before quoting the File in a published work or commercial product. We may have additional information that would be helpful to you and can assist you in framing your quote to reflect not only the letter of the File but its spirit as well.

All contributions and suggestions about this file sent to a volunteer editor are gratefully received and will be regarded, unless otherwise labelled, as freely given donations for possible use as part of this public-domain file.

From time to time a snapshot of this file has been polished, edited, and formatted for commercial publication with the cooperation of the volunteer editors and the hacker community at large. If you wish to have a bound paper copy of this file, you may find it convenient to purchase one of these. They often contain additional material not foudn in on-line versions. The two 'authoriezd' editions so far are described in the Revision History section; there may be more in the future.

Contents

Introduction

This document is a collection of slang terms used by various subcultures of computer hackers. Though some technical material is included for background and flavor, it is not a technical dictionary; what we describe here is the language hackers use among themselves for fun, social communication, and technical debate.

The 'hacker culture' is actually a loosely networked collection of subcultures that is nevertheless conscious of some important shared experiences, shared roots, and shared values. It has its own myths, heroes, villains, folk epics, in-jokes, taboos, and dreams. Because hackers as a group are particularly creative people who define themselves partly by rejection of 'normal' values and working habits, it has unusually rich and conscious traditions for an intentional culture less than 60 years old.

As usual with slang, the special vocabulary of hackers helps hold their culture together – it helps hackers recognize each other's places in the community and expresses shared values and experiences. Also as usual, *not* knowing the slang (or using it inappropriately) defines one as an outsider, a mundane, or (worst of all in hackish vocabulary) possibly even a **suit**[64]. All human cultures use slang in this threefold way – as a tool of communication, and of onclusion, and of exclusion.

Among hackers, though, slang has a subtler aspect, paralleled perhaps in the slang of jazz musicians and some kinds of fine artists but hard to detect in most technical or scientific cultures; parts of it are code for shared states of *consciousness*. There is a whole range of altered states and problem-solving mental stances basic to high-level hacking which don't fit into conventional linguistic reality any better than a Coltrane solo or one of Maurits Escher's 'trompe l'oeil' compositions (Escher is a favorite of hackers), and hacker slang encodes these subtleties in many unobvious ways. As a simple example, take the distinction between a **kludge**[48] and an **elegant**[42] solution, and the differing connotations attached to each. The distinction is not only of engineering significance; it reaches right back into the nature of the generative process in program design and asserts something important about two different kinds of relationship between the hacker and the hack. Hacker slang is unusually rich in implications of this kind, of overtones and undertones that illuminate the hackish psyche.

But there is more. Hackers, as a rule, love wordplay and are very conscious and inventive in their use of language. These traits seem to be common in young children, but the conformity-enforcing machine we are pleased to call an educational system bludgeons them out of most of us before adolescence. Thus, linguistic invention in most subcultures of the modern West is a halting and largely unconscious process. Hackers, by contrast, regard slang formation and use as a game to be played for conscious pleasure. Their inventions thus display an almost unique combination of the neotenous enjoyment of language-play with the discrimination of educated and powerful intelligence. Further, the electronic media which knit them together are fluid, 'hot' connections, well adapted to both the dissemination of new slang and the ruthless culling of weak and superannuated specimens. The results of this process give us perhaps a uniquely intense and accelerated view of linguistic evolution in action.

Hacker slang also challenges some common linguistic and anthropological assumptions. For example, it has recently become fashionable to speak of 'low-context' versus 'high-context' communication, and to classify cultures by the preferred context level of their languages and art forms. It is usually claimed that low-context communication (characterized by precision, clarity, and completeness of self-contained utterances) is typical in cultures which value logic, objectivity, individualism, and competition; by contrast, high-context

communication (elliptical, emotive, nuance-filled, multi-modal, heavily coded) is associated with cultures which value subjectivity, consensus, cooperation, and tradition. What then are we to make of hackerdom, which is themed around extremely low-context interaction with computers and exhibits primarily "low-context" values, but cultivates an almost absurdly high-context slang style?

The intensity and consciousness of hackish invention make a compilation of hacker slang a particularly effective window into the surrounding culture – and, in fact, this one is the latest version of an evolving compilation called the 'Jargon File', maintained by hackers themselves for over 36 years. This one (like its ancestors) is primarily a lexicon, but also includes topic entries which collect background or sidelight information on hacker culture that would be awkward to try to subsume under individual slang definitions.

Though the format is that of a reference volume, it is intended that the material be enjoyable to browse. Even a complete outsider should find at least a chuckle on nearly every page, and much that is amusingly though-provoking. But it is also true that hackers use humorous wordplay to make strong, sometimes combative statements about what they feel. Some of these entries reflect the views of opposing sides in disputes that have been genuinely passionate; this is deliberate. We have not tried to moderate or pretty up these disputes; rather we have attempted to ensure that *everyone's* sacred cows get gored, impartially. Compromise is not particularly a hackish virtue, but the honest presentation of divergent viewpoints is.

The reader with minimal computer background who finds some references incomprehensibly technical can safely ignore them. We have not felt it either necessary or desirable to eliminate all such; they, too, contribute flavor, and one of this document's major intended audiences – fledgling hackers already partway inside the culture – will benefit from them.

A selection of longer items of hacker folklore and humor is included in **Appendix A**[65]. The 'outside' reader's attention is particularly directed to the Portrait of J. Random Hacker in **Appendix B**[67]. Appendix C, the **Bibliography**[69], lists some non-technical works which have either influenced or described the hacker culture.

Because hackerdom is an intentional culture (one each individual must chose by action to join), one should not be surprised that the line between description and influence can become more than a little blurred. Earlier versions of the Jargon File have played a central role in spreading hacker language and the culture that goes with it to successively larger populations, and we hope and expect that this one will do likewise.

A Few Terms

Linguists usually refer to informal language as 'slang' and reserve the term 'jargon' for the technical vocabularies of various occupations. However, the ancestor of this collection was called the 'Jargon File', and hacker slang is traditionally 'the jargon'. When talking about the jargon there is therefore no convenient way to distinguish it frmo what a *linguist* would call hackers' jargon – the formal vocabulary they learn from textbooks, technical papers, and manuals.

To make a confused situation worse, the line between hacker slang and the vocabulary of technical programming and computer science is fuzzy, and shifts over time. Further, this vocabulary is shared with a wider technical culture of programmers, many of whom are not hackers and do not speak or recognize hackish slang.

Accordingly, this lexicon will try to be as precise as the facts of usage permit about the distinctions among three categories:

- 'slang': informal language from mainstream English or non-technical subcultures (bikers, rock fans, surfers, etc).
- 'jargon': without qualifier, denotes informal 'slangy' language peculiar to or predominantly found among hackers – the subject of this lexicon.
- 'techspeak': the formal technical vocabulary of programming, computer science, electronics, and other fields connected to hacking.

This terminology will be consistently used throughout the remainder of this lexicon.

The jargon/techspeak distinction is the delicate one. A lot of techspeak originated as jargon, and there is a steady continuing uptake of jargon into techspeak. On the other hand, a lot of jargon arises from overgeneraliztion of techspeak terms (there is more about this in the **Jargon Construction**[??] section below).

In general, we have considered techspeak any term that communicates primarily by a denotation well established in technical dictionaries, or standards documents.

A few obviously techspeak terms (names of operating systems, languages, or documents) are listed when they are tied to hacker folklore that isn't covered in formal sources, or sometimes to convey critical historical background necessary to understand other entries to which they are cross-referenced. Some other techspeak senses of jargon words are listed in order to make the jargon senses clear; where the text does not specify that a straight technical sense is under discussion, these are marked with '[techspeak]' as an etymology. Some entries have a primary sense marked this way, with subsequent jargon meanings explained in terms of it.

We have also tried to indicate (where known) the apparent origins of terms. The results are probably the least reliable information in the lexicon, for several reasons. For one thing, it is well known that many hackish usages have been independently reinvented multiple times, even among the more obscure and intricate neologisms. It often seems that the generative processes underlying hackish jargon formation have an internal logic so powerful as to create a substantial parallelism across separate cultures and even in different languages! For another, the networks tend to propagate innovations so quickly that 'first use' is

often impossible to pin down. And, finally, compendia like this one alter what they observe by implicitly stamping cultural approval on terms and widening their use.

Despite these problems, the organized collection of jargon-related oral history for the new compilations has enabled us to put to rest quite a number of folk etymologies, place credit where credit is due, and illuminate the early history of many important hackerisms such as **kluge**[48], **cruft**[40], and **foo**[43]. We believe specialist lexicographers will find many of the historical notes more than casually instructive.

Revision History

The original Jargon File was a collection of hacker jargon from technical cultures including the MIT AI Lab, the Stanford AI lab (SAIL), and others of the old ARPANET AI/LISP/PDP-10 communities including Bolt, Beranek, and Newman (BBN), Carnegie-Mellon University (CMU), and Worcester Polytechnic Institute (WPI).

The Jargon File (hereafter referred to as 'jargon-1' or 'the File') was begun by Raphael Finkel at Stanford in 1975. From this time until the plug was finally poulled on the SAIL computer in 1991, the File was named AIWORD.RF[UP,DOC] there. Some terms in it date back considerably earlier (**frob**[43] and some senses of **moby**[51], for instance, go back to the Tech Model Railroad Club at MIT and are believed to date at least back to the early 1960s). The revisions of jargon-1 were all unnumbered and may be collectively considered 'Version 1'.

In 1976, Mark Crispin, having seen an announcement about the File on the SAIL computer, **FTP**[43]ed a copy of the File to MIT. He noticed that it was hardly restricted to 'AI words' and so stored the file on his directory as AI:MRC;SAIL JARGON.

The file was quickly renamed the JARGON > (the '>' caused versioning under ITS) as a flurry of enhancements were made by Mark Crispin and Guy L. Steele Jr. Unfortunately, amidst all this activity, nobody thought of correcting the term 'jargon' to 'slang' until the compendium had already become widely known as the Jargon File.

Raphael Finkel dropped out of active participation shortly thereafter and Don Woods became the SAIL contact for the File (which was subsequently kept in duplicate at SAIL and MIT, with periodic resynchronizations).

The File expanded by fits and starts until about 1983; Richard Stallman was prominent among the contributors, adding many MIT and ITS-related coinages.

In Spring 1981, a hacker named Charles Spurgeon got a large chunk of the File published in Stewart Brand's *CoEvolution Quarterly* (issue 29, pages 26–35) with illustrations by Phil Wadler and Guy Steele (including a couple of the Crunchly cartoons). This appears to have been the File's first paper publication.

A late version of jargon-1, expanded with commentary for the mass market, was edited by Guy Steele into a book published in 1983 as *The Hacker's Dictionary* (Harper & Row CN 1082, ISBN 0-06-091082-8). The other jargon-1 editors (Raphael Finkel, Don Woods, and Mark Crispin) contributed to this revision, as did Richard M. Stallman and Geoff Goodfellow. This book (now out of print) is hereafter referred to as 'Steele-1983' and those six as the Steele-1983 coauthors.

Shortly after the publication of Steele-1983, the File effectively stopped growing and changing. Originally, this was due to a desire to freeze the file temporarily to facilitate the production of Steele-1983, but external conditions caused the 'temporary' freeze to become permanent.

The AI Lab culture had been hit hard in the late 1970s by funding cuts and the resulting administrative decision to use vendor-supported hardware and software instead of homebrew whenever possible. At MIT, most AI work had turned to dedicated LISP Machines. At the same time, the commercialization of AI technology lured some of the AI Lab's best and brightest away to startups along the Route 128 strip in

Massachusetts and out West in Silicon Valley. The startups built LISP machines for MIT; the central MIT-AI computer became a **TWENEX**[58] system rather than a host for the AI hackers' beloved **ITS**[46].

The Stanford AI Lab had effectively ceased to exist by 1980, although the SAIL computer continued as a Computer Science Department resource until 1991. Stanford became a major **TWENEX**[58] site, at one point operating more than a dozen TOPS-20 systems; but by the mid-1980s most of the interesting software work was being done on the emerging BSD Unix standard.

In April 1983, the PDP-10-centered cultures that had nourished the File were dealt a death-blow by the cancellation of the Jupiter project at Digital Equipment Corpiration. The File's compilers, already dispersed, moved on to other things. Steele-1983 was partly a monument to what its authors thought was a dying tradition; no one involved realized at the time just how wide its influence was to be.

By the mid-1980s the File's content was dated, but the legend that had grown up around it never quite died out. The book, and softcopies obtained off the ARPANET, circulated even in cultures far removed from MIT and Stanford; the content exerted a strong and continuing influence on hacker language and humor. Even as the advent of the microcoputer and other trends fueled a tremendous expansion of hackerdom, the File (and related materials such as the **Some AI Koans**[65] in **Appendix A**[65]) came to be seen as a sort of sacred epic, a hacker-culture Matter of Britain chronicling the heroic exploits of the Knights of the Lab. The pace of change in hackrdom at large accelerated tremendously – but the Jargon File, having passed from living document to icon, remained essentially untouched for seven years.

Eric S. Raymond `<esr@snark.thyrsus.com>` maintained the File with assistance from Guy L. Steele Jr. `<gls@think.com>` between 1991 and 2011. However, as of 2003, this file was contaminated by Eric Raymond's politics (as noted by frustrated hackers, including Richard Stallman). Complains were many (some can be found here `<http://www.ntk.net/2003/06/06/>`) and, until 2011 there were no revisions, major or minor. The various revisions are referred to here by their file numbers, but the Raymond period in general will hereafter be referred to as *ESR-2003*.

ESR-2003 contained nearly the entire text of a late version of jargon-1 (a few obsolete PDP-10-related entries were dropped after careful consultation with the editors of Steele-1983). It merged in about 80% of the Steele-1983 text, omitting some framing material and very few entries introduced in Steele-1983 that are now obsolete.

ESR-2003 casted a wider net than the old Jargon File; its aim is to cover not just AI or PDP-10 hacker culture but all the technical computing cultures wherein the true hacker-nature is manifested. More than half of the entries now derive from **Usenet**[59] and represent jargon now current in the C and Unix communities, but special efforts have been made to collect jargon from other cultures including IBM PC programmers, Amiga fans, Mac enthusiasts, and even the IBM mainframe world.

The 2.9.6 version became the main text of *The New Hacker's Dictionary*, by Eric Raymond (ed.), MIT Press 1991, ISBN 0-262-68069-6.

The 3.0.0 version was published in September 1993 as the second edition of *The New Hacker's Dictionary*, again from MIT Press (ISBN 0-262-18154-1).

If you want the book, you should be able to find it at any of the major bookstore chains. Failing that, you can order by mail from

Teh MIT Press 55 Hayward Street Cambridge, MA 02142

or order by phone at (800)-356-0343 or (617)-625-8481.

The Jargon File is currently maintained by Yash Tulsyan `<yashtulsyan@gmail.com>` in response to the static nature of ESR-2003, as well as the many criticisms layed out against it. It was forked from version 4.2.0.

The maintainer(s) are committed to updating the on-line version of the Jargon File through and beyond paper publication, and will continue to make it available to archives and public-access sites as a trust of the hacker community.

Here is a chronology of the high points in the on-line revisions since Steele-1983:

Version 2.1.1, Jun 12 1990: the Jargon File comes alive again after a seven-year hiatus. Reorganization and massive additions were by Eric S. Raymond, approved by Guy Steele. Many items of UNIX, C, USENET, and microcomputer-based jargon were added at that time.

Version 2.9.6, Aug 16 1991: corresponds to reproduction copy for book. This version had 18952 lines, 148629 words, 975551 characters, and 1702 entries.

Version 2.9.7, Oct 28 1991: first markup for hypertext browser. This version had 19432 lines, 152132 words, 999595 characters, and 1750 entries.

Version 2.9.8, Jan 01 1992: first public release since the book, including over fifty new entries and numerous corrections/additions to old ones. Packaged with version 1.1 of vh(1) hypertext reader. This version had 19509 lines, 153108 words, 1006023 characters, and 1760 entries.

Version 2.9.9, Apr 01 1992: folded in XEROX PARC lexicon. This version had 20298 lines, 159651 words, 1048909 characters, and 1821 entries.

Version 2.9.10, Jul 01 1992: lots of new historical material. This version had 21349 lines, 168330 words, 1106991 characters, and 1891 entries.

Version 2.9.11, Jan 01 1993: lots of new historical material. This version had 21725 lines, 171169 words, 1125880 characters, and 1922 entries.

Version 2.9.12, May 10 1993: a few new entries & changes, marginal MUD/IRC slang and some borderline techspeak removed, all in preparation for 2nd Edition of TNHD. This version had 22238 lines, 175114 words, 1152467 characters, and 1946 entries.

Version 3.0.0, Jul 27 1993: manuscript freeze for 2nd edition of TNHD. This version had 22548 lines, 177520 words, 1169372 characters, and 1961 entries.

Version 3.1.0, Oct 15 1994: interim release to test WWW conversion. This version had 23197 lines, 181001 words, 1193818 characters, and 1990 entries.

Version 3.2.0, Mar 15 1995: Spring 1995 update. This version had 23822 lines, 185961 words, 1226358 characters, and 2031 entries.

Version 3.3.0, Jan 20 1996: Winter 1996 update. This version had 24055 lines, 187957 words, 1239604 characters, and 2045 entries.

Version 3.3.1, Jan 25 1996: Copy-corrected improvement on 3.3.0 shipped to MIT Press as a step towards TNHD III. This version had 24147 lines, 188728 words, 1244554 characters, and 2050 entries.

Version 3.3.2, Mar 20 1996: A number of new entries pursuant on 3.3.2. This version had 24442 lines, 190867 words, 1262468 characters, and 2061 entries.

Version 3.3.3, Mar 25 1996: Cleanup before TNHD III manuscript freeze. This version had 24584 lines, 191932 words, 1269996 characters, and 2064 entries.

Version 4.0.0, Jul 25 1996: The actual TNHD III version after copy-edit. This version had 24801 lines, 193697 words, 1281402 characters, and 2067 entries.

Version 4.1.0, 8 Apr 1999: The Jargon File rides again after three years. This version had 25777 lines, 206825 words, 1359992 characters, and 2217 entries.

Version 4.1.1, 18 Apr 1999: Corrections for minor errors in 4.1.0, and some new entries. This version had 25921 lines, 208483 words, 1371279 characters, and 2225 entries.

Version 4.1.2, 28 Apr 1999: Moving texi2html out of the production path. This version had 26006 lines, 209479 words, 1377687 characters, and 2225 entries.

Version 4.1.3, 14 Jun 1999: Minor updates and markup fixes. This version had 26108 lines, 210480 words, 1384546 characters, and 2234 entries.

Version 4.1.4, 17 Jun 1999: Markup fixes for framed HTML. This version had 26117 lines, 210527 words, 1384902 characters, and 2234 entries.

Version 4.2.0, 31 Jan 2000: Markup fixes for framed HTML. This version had 26598 lines, 214639 words, 1412243 characters, and 2267 entries.

Version 5.0.0, 25 Jun 2011: The Jargon File comes alive yet again after an 8 year hiatus, maintained by Yash Tulsyan. Was forked from 4.2.0, revised in light of criticisms against ESR-2003. This version has no word count as of yet.

Version 5.0.1, 5 Jan 2012: The Jargon File is being fixed and upsated, OSI politics are replaced in favor of OSI-FSF agnostic politics. This version has no word count as of yer.

Version numbering: Version numbers should be read as major.minor.revision. Major version 1 is reserved for the 'old' (ITS) Jargon File, jargon-1. Major version 2 encompasses revisions by ESR (Eric S. Raymond) with assistance from GLS (Guy L. Steele, Jr.) leading up to and including the second paper edition. Usually later versions will either completely supersede or incorporate earleir versions, so there is generally no point in keeping old versions around. One should also remember that due to the long hiatusen between major versions, there have been many forks. We hope that if you find issue with our version, you will send us suggestions to improve it. If, however, you challenge the legitimacy of this edition, then by all means fork it.

Many thanks to ESR, who maintained ESR-2003, and I renew his statement of thanks that follows:

Our thanks to the coauthors of Steele-1983 for oversight and assistance, and to the hundreds of Usenetters (too many to name here) who contributed entries and encouragement. More thanks go to several of the old-timers on the Usenet group alt.folklore.computers, who contributed much useful commentary and many corrections and valuable historical perspective: Joseph M. Newcomer <jn11+@andrew.cmu.edu>, Bernie Cosell <cosell@bbn.com>, Earl Boebert <boebert@SCTC.com>, and Joe Morris <jcmorris@mwunix.mitre.org>.

We were fortunate enough to have the aid of some accomplished linguists, David Stampe <stampe@hawaii.edu> and Charles Hoequist <hoequist@bnr.ca> contributed valuable criticisms; Joe Keane <jgk@osc.osc.com> helped us improve the pronunciation guides.

A few bits of this text quote previous works. We are indebted to Brian A. LaMacchia <bal@zurich.ai.mit.edu> for obtaining permission for us to use material from the *TMRC Dictionary*; also, Don Libes <libes@cme.nist.gov> contributed some appropriate material for his excellent book *Life With UNIX*. We thank Per Lindberg <per@front.se>, author of the remarkable Swedish-language 'zine' *Hackerbladet*, for bringing *FOO!* comics to our attention and smuggling one of the IBM hacker underground's own baby jargon files out to us. Thanks also to Maarten Litmaath for generously allowing the inclusion of the ASCII pronunciation guide he formerly maintained. And our gratitude to Mark Weiser of XEROX PARC <Marc_Weiser.PARC@xerox.com> for securing us permission to quote from PARC's own jargon lexicon and shipping us a copy.

It is a particular pleasure to acknowledge the major contributions of Mark Brader <msb@sq.com> and Steve Summit <scs@eskimo.com> to the File and Dictionary; they have read and reread many drafts, checked facts, caught typos, submitted an amazing number of thoughtful comments, and done yeoman service in catchign typos and minor usage bobbles. Their rare combination of enthusiasm, persistence, wide-ranging technical knowledge, and precissionism in matters of language has been of invaluable help. Indeed, the sustained volume and quality of Mr. Brader's input over several years and several different editions has only allowed him to escape co-editor credit by the slimmest of margins.

Finally, George V. Reilly <georger@microsoft.com> helped with the TeX arcana and painstakingly proofread some 2.7 and 2.8 versions, and Eric Tiedemann <est@thyrsus.com> contributed sage advice throughout on rhetoric, amphigory, and philosophunculism.

Part I

How Jargon Works

Jargon Construction

There are some standard methods of jargonification that became established quite early (i.e., before 1970), spreading from such sources as the Tech Model Railroad Club, the PDP-1 SPACEWAR hackers, and John McCarthy's original crew of LISPers. These include verb doubling, soundalike slang, the '–P' convention, overgeneralization, spoken inarticulations, and anthropomorphization. Each is discussed below. We also cover the standard comparitives for design quality.

Of these six, verb doubling, overgeneralization, anthropomorphization, and (especially) spoken inarticulatinos have become quite general; but soundalike slang is still alrgely confined to MIT and other large universities, and the '–P' convention is found only where LISPers flourish.

- **Verb Doubling**[14]: Doubling a verb may change its semantics
- **Soundalike Slang**[15]: Punning jargon
- **The '–P' convention**[15]: A LISPy way to form questions
- **Overgeneralization**[16]: Standard abuses of grammar
- **Spoken Inarticulations**[17]: Sighing and <*sigh>ing
- **Anthropomorphization**[17]: Homunculi, daemons, and confused programs
- **Comparatives**[18]: Standard comparatives for design quality

Verb Doubling

A standard construction in English is to double a verb and use it as an exclamation, such as "Bang, bang!" or "Quack, quack!". Most of these are names for noises. Hackers also double verbs as a concise, sometimes sarcastic comment on what the implied subject does. Also, a doubled verb is often used to terminate a conversation, in the process remarking on the current state of affairs or what the speaker intends to do next. Typical examples involve **win**[61], **lose**[50], **hack**[45], **flame**[43], **barf**[39], **chomp**[40]:

> "The disk heads just crashed." "Lose, lose."
> "Mostly he talked about his latest crock. Flame, flame."
> "Boy, what a bagbiter! Chomp, chomp!"

Some verb-doubled constructions have special meanings not immediately obvious from the verb. These have their own listings in the lexicon.

The **Usenet**[59] culture has one tripling convention unrelated to this: the names of 'joke' topic groups often have a tripled last element. The first and paradigmatic example was alt.swedish.chef.bork.bork.bork (a *Muppet Show* reference); other infamous examples have included:

alt.french.captain.borg.borg.borg
alt.wesley.crusher.die.die.die
comp.unix.internals.system.calls.brk.brk.brk
sci.physics.edward.teller.boom.boom.boom
alt.sadistic.dentists.drill.drill.drill

Soundalike Slang

Hackers will often make rhymes or puns in order to convert an ordinary word or phrase into something more interesting. It is considered particularly **flavorful**[43] if the phrase is bent so as to include some other jargon word; thus the computer hobbyist magazine *Dr. Dobb's Journal* is almost always referred to among hackers as 'Dr. Frob's Journal' or simply 'Dr. Frob's'. Terms of this kind that have been in fairly wide use include names for newspapers:

Boston Herald ⇒ Horrid (or Harried)
Boston Globe ⇒ Boston Glob
Houston (or San Francisco) Chronicle ⇒ the Crocknicle (or the Comical)
New York Times ⇒ New York Slime
Wall Street Journal ⇒ Wall Street Urinal

However, terms like these are often made up on the spur of the moment, Standard examples include:

Data General ⇒ Dirty Genitals
IBM 360 ⇒ IBM Three-Sickly
Government Property – Do Not Duplicate (on keys) ⇒ Government Duplicity – Do Not Propagate
for historical reasons ⇒ for hysterical raisins
Margaret Jacks Hall (the CS building at Stanford) ⇒ Marginal Hacks Hall
Microsoft ⇒ Microsloth
Internet Explorer ⇒ Internet Exploiter

This is not really similar to the Cockney rhyming slang it has been compared to in the past, because Cockney substitutions are opaque whereas hacker punning jargon is intentionally transparent.

The '–P' convention

Turning a word into a question by appending the syllable 'P'; from the LISP convention of appending the letter 'P' to denote a predicate (a boolean-valued function). The question should expect a yes/no answer, though it needn't. (See **T**[58] and **NIL**[52].)

At dinnertime:

Q: "Foodp?"

A: "Yeah, I'm pretty hungry." or "T!"

At any time:

Q: "State-of-the-world-P?"

A: (Straight) "I'm about to go home."

A: (Humorous) "Yes, the world has a state."

On the phone to Florida:

> Q: "State-p Florida?"
>
> A: "Been reading JARGON.TXT again, eh?"

[One of the best of these is a **Gosperism**[44]. Once, when we were at a chinese restaurant, Bill Gosper wanted to know whether someone would like to share with him a two-person-sized bowl of soul. His inquiry was: "Split-p soup?" – GLS]

Overgeneralization

A very conspicuous feature of jargon is the frequency with which techspeak items such as names of program tools, command language primitives, and even assembler opcodes are applied to contexts outside of computing wherever hackers find amusing analogies to them. Thus (to cite one of the best-known examples) Unix hackers often **grep**[44] for things rather than searching for them. Many of the lexicon entries are generalizations of exactly this kind.

Hackers often enjoy overgeneralization on the grammatical level as well. Many hackers love to take various words and add the wrong endings to them to make nouns and verbs, often by extending a standard rule to nonuniform cases (or vice verse). For example, because

> porous ⇒ porosity
> generous ⇒ generosity

hackers happily generalize:

> mysterious ⇒ mysteriosity
> ferrous ⇒ ferrosity
> obvious ⇒ obviosity
> dubious ⇒ dubiosity

Another class of common construction uses the suffix '–itude' to abstract a quality from just about any adjective or noun. This usage arises especially in cases where mainstream English would perform the same abstraction through '–iness' or '–ingness'. Thus:

> win ⇒ winnitude (a common exclamation)
> loss ⇒ lossitude
> cruft ⇒ cruftitude
> lame ⇒ lamitude

Some hackers cheerfully reverse this transformation; they argue, for example, that the horizontal degree lines on a globe ought to be called 'lats' – after all, they're measuring latitude!

Also, note that all nouns can be verbed. E.g.: "All nouns can be verbed", "I'll mouse it up", "Hang on while I clipboard it over", "I'm grepping the files". English as a whole is already heading in this direction (towards pure-positional grammar like Chinese); hackers are simply a bit ahead of the curve.

The suffix "–full" can also be applied in generalized and fanciful ways, as in "As soon as you have more than one cachefull of data, the system starts thrashing," or "As soon as I have more than one headfull of ideas, I start writign it all down." A common use is a "screenfull" meaning the amount of text that will fit in one screen, usually in text mode when you have no choice as to character size. Another common form is "bufferfull".

However, hackers avoid the unimaginative verb-making techniques characteristic of marketroids, beancounters, and the Pentagon: a hacker would never, for example, 'productize', 'prioritize', or 'securitize' things. Hackers have a strong aversino to bureaucratic bafflegab and regard those who use it with contempt. The terms for 'free software' are an example – 'free software' is the original, most hackish term, and those who use it are a significant minority. It calls to mind freedom, and most who use it also use copyleft licenses, such as the GNU General Public License. ESR coined the term 'open-source' in order to appeal to businesses, and those who use it are a majority of hackers and hacker-friendly (or at least hacker-saturated) businesses. The GNU GPL is popular here as well, but copycenter licenses are gaining in popularity. Then businesses coined the bureaucratic buzzword 'crowdsourcing', in reference to outsourcing, and this is rare amongst hackers, at least when talking of software.

Similarly, all verbs can be nouned. This is only a slight overgeneralization in modern English; in hackish, however, it is good form to mark them in some standard nonstandard way. Thus:

win ⇒ winnitude, winnage
disgust ⇒ disgustitude
hack ⇒ hackification

Further, note the prevalence of certain kinds of nonstandard plural forms. Some of these go back quite a ways: the TMRC Dictionary includes an entry which implies that the plural of 'mouse' is **meeces**[51], and notes that the defined plural of 'caboose' is 'cabeese'. This latter has apparently been standard (or at least a standard joke) among railfans (railroad enthusiasts) for many years.

On a similarly Anglo-Saxon note, almost anything ending in 'x' may form plurals in '–xen' (see **VAXen**[60] and **boxen**[39] in the main text). Even words ending in phonetic /k/ alone are sometimes treated this way; e.g., 'soxen' for a bunch of socks. Other funny plurals are 'frobbotzim' for the plural of 'frobbozz' (see **frobnitz**[43]) and 'Unices' and 'Twenices' (rather than 'Unixes' and 'Twenexes'; see **Unix**[59], **TWENEX**[58] in main text). But note that 'Unixen' and 'Twenexen' are never used; it has been suggested that this is because '–ix' and '–ex' are Latin singular endings that attract a Latinate plural. Finally, it has been suggested to general approval that the plural of 'mongoose' ought to be 'polygoose'.

The pattern here, as with other hackish grammatical quirks, is generalization of an inflectional rule that in English is either an import or a fossil (such as the Hebrew plural ending '–im', or the Anglo-Saxon plural suffix '–en') to cases where it isn't normally considered to apply.

This is not 'poor grammar' (which is itself a somewhat barmy idea), as hackers are generally quite well aware of what they are doing when they distort the language. It is grammatical creativity, a form of playfulness. It is done not to impress but to amuse, and never at the expense of clarity.

Spoken Inarticulations

Words such as 'mumble', 'sigh', and 'groan' are spoken in places where their referent might more naturally be used. It has been suggested that this usage derives from the impossibility of representing such noises on a comm link or in electronic mail, MUDs, and IRC channels (interestingly, the same sorts of constructions have been showing up with increasing frequency in comic strips). Another expression sometimes heard is "Complain!", meaning "I have a complaint!"

Anthropomorphization

Semantically, one rich source of jargon construction is the hackish tendency to anthropomorphize hardware and software. This isn't done in a naive way; hackers don't personalize their stuff in the sense of feeling empathy with it, nor do they mystically believe that the things they work on every day are 'alive'. What is

common is to hear hardware or software talked about as though it has homunculi talking to each other inside it, with intentions and desires. Thus, one hears "The protocol handler got confused", or that programs "are trying" to do things, or one may say of a routine that "its goal in life is to X". One even hears explanations like"...and its poor little brain couldn't understand X, and it died." Sometimes modelling things this way actually seems to make them easier to understand, perhaps because it's instinctively natural to think of anything with a really complex behavioral repertoire as 'like a person' rather than 'like a thing'.

Comparatives

Finally, note that many words in hacker jargon have to be understood as members of sets of comparatives. This is especially true of the adjectives and nouns used to describe the beauty and functional quality of code. Here is an approximately correct spectrum:

> monstrosity brain-damage screw bug lose misfeature
> crock kluge hack win feature elegance perfection

The last is spoken of as a mythical absolute, approximated but never actually attained. Another similar scale is used for describing the reliability of software:

> broken flaky dodgy fragile brittle
> solid robust bulletproof armor-plated

Note, however, that 'dodgy' is primarily Commonwealth Hackish (it is rare in the U.S.) and may change places with 'flaky' for some speakers.

Coinages for describing **lossage**[50] seem to call forth the very finest in hackish linguistic inventiveness; it has been truly said that hackers have even more words for equipment failure than Yiddish has for obnoxious people.

Hacker Writing Style

We've already seen that hackers often coin jargon by overgeneralizing grammatical rules. THis is one aspect of a more general fondness for form-versus-content language jokes that shows up particularly in hackish writing. One correspondent reports that he consistently misspells 'wrong' as 'worng'. Others have been known to criticize glitches in Jargon File drafts by observing (in the mode of Douglas Hofstadter) "This sentence no verb", or "Too repetetetive", or "Bad speling", or "Incorrectspa cing." Similarly, intentional spoonerisms are often made of phrases relating to confusion or things that are confusing: 'drain bramage' for 'brain damage' is perhaps the most common (similarly, a hacker would be likely to write "Excuse me, I'm cixelsyd today", rather than "I'm dyslexic today"). This sort of thing is quite common and is enjoyed by all concerned.

Hackers tend to use quotes as balanced delimiters like parentheses, much to the dismay of American editors. Thus, if "Jim is going" is a phrase, and so are "Bill runs" and "Spock groks", then hackers generally prefer to write: "Jim is going", "Bill runs" and "Spock groks". This is incorrect according to standard American usage (which would put the commas and the final period inside the string quotes); however, it is counter-intuitive to hackers to mutilate literal strings with characters that don't belong in them. Given the sorts of examples that can come up in discussion of programming, American-style quoting can even be grossly misleading. When communicating command lines or small pieces of code, extra characters can be a real pain in the neck.

Consider, for example, a sentence in a **vi**[60] tutorial that looks like this:

> Then delete a line from the file by typing "dd".

Standard usage would make this

> Then delete a line from the file by typing "dd."

but that would be very bad – because the reader would be prone to type the string d-d-dot, and it happens that in vi(1) dot repeats the last command accepted. The net result would be to delete two lines!

The Jargon File follows hackish usage throughout.

Interestingly, a similar style is now preferred practice in Great Britain, though the older style (which became stablished for typographical reasons having to do with the aesthetics of comma and quotes in typeset text) is still accepted there. *Hart's Rules* and the *Oxford Dictionary for Writers and Editors* call the hacker-like style 'new' or 'logical' quoting. This turn British English to the style Latin languages (including Spanish, French, Italian, Catalan) have been using this style all along.

Another hacker habit is a tendency to distinguish between 'scare' quotes and 'speech' quotes; that is, to use British-style single quotes for marking and reserve American-style double quotes for actual reports of speech or text included from elsewhere. Interestingly, some authorities describe this as correct general usage, but mainstream American English has gone to using double-quotes indiscriminately enough that hacker usage appears marked [and, in fact, *I* thought this was a personal quirk of mine until I checked with Usenet –ESR]. One further permutation that is definitely not standard is a hackish tendency to do marking quotes by using apostrophes (single quotes) in pairs; that is, 'like this'. This is modelled on string and character literal

syntax in some programming languages (reinforced by the fact that many character-only terminals display the apostrophe in typewriter style, as a vertical single quote).

One quirk that shows up frequently in the **email**[42] style of Unix hackers in particular is a tendency for some things that are normally all-lowercase (including usernames and the names of commands and C routines) to remain uncapitalized even when they occur at the beginning of sentences. It is clear that, for many hackers, the case of such identifiers becomes a part of their internal representation (the 'spelling') and cannot be overridden without mental effort (an appropriate reflex because Unix and C both distinguish cases and confusing them can lead to **lossage**[50]). A way of escaping this dilemma is simply to avoid using these constructions at the beginning of sentences.

There seems to be a meta-rule behind these nonstandard hackerisms to the effect that precision of expression is more important than conformance to traditional rules; where the latter create ambiguity or lose information they can be discarded without a second though. It is notable in this respect that other hackish inventions (for example, in vocabulary) also tend to carry very precise shades of meaning even when constructed to appear slangy and loose. In fact, to a hacker, the contrast between 'loose' form and 'tight' content in jargon is a substantial part of its humor!

Hackers have also developed a number of punctuation and emphasis conventions adapted to single-font all-ASCII communications links, and these are occasionally carried over into written documents even when normal means of font changes, underlining, and the like are available.

One of these is that TEXT IN ALL CAPS IS INTERPRETED AS 'LOUD', and this becomes such an ingrained synesthetic reflex that a person who goes to caps-lock while in **talk mode**[58] may be aksed to "stop shouting, please, you're hurting my ears!".

Also, it is common to use bracketing with unusual characters to signify emphasis. The asterisk is most common, as in "What the *hell*?" even though this interferes with the common use of the asterisk suffix as a footnote mark. The underscore is also common, suggesting underlining (this is particularly common with book titles; for example, "It is often alleged that Joe Haldeman wrote _The_Forever_War_ as a rebuttal to Robert Heinlein's earlier novel of the future military, _Starship_Troopers_."). Other forms exemplified by "=hell=", "\hell/", or "/hell/" are occasionally seen (it's claimed that in the last example the first slash pushes the letters over to the right to make them italic, and the second keeps them from falling over). On FidoNet, you might see #bright# and ^dark^ text, which was actually interpreted by some reader software. Finally, words may also be emphasized L I K E T H I S, or by a series of carets (^) under them on the next line of the text.

There is a semantic difference between *emphasis like this* (which emphasizes the phrase as a whole), and *emphasis* *like* *this* (which suggests the writer speaking very slowly and distinctly, as if to a very young child or a mentally impaired person). Bracketing a word with the '*' character may also indicate that the writer wishes readers to consider that an action is taking place or that a sound is being made. Examples: *bang*, *hic*, *rings*, *grin*, *kick*, *stomp*, *mumble*.

One might also see the above sound effects as <bang>, <hic>, <ring>, <grin>, <kick>, <stomp>, <mumble>. This use of angle brackets to mark their contents originally derives from conventions used in **BNF**[39], but since about 1993 it has been reinforced by the HTML markup used on the World Wide Web.

Angle-bracket enclosure is also used to indicate that a term stands for some **random**[56] member of a larger class (this is straight from **BNF**[39]). Examples like the following are common:

> So this <ethnic> walks into a bar one day ...

There is also an accepted convention for 'writing under erasure'; the text

> Be nice to this fool^H^H^H^Hgentleman,
> he's visiting from corporate HQ.

reads roughly as "Be nice to this fool, er, gentleman ...". This comes from the fact that the digraph ^H

is often used as a print representation for a backspace. It parallels (and may have been influenced by) the ironic use of 'slashouts' in science-fiction fanzines.

A related habit uses editor commands to signify corrections to previous text. This custom is fading as more mailers get good editing capabilities, but one occasionally still sees things like this:

> I've seen that term used on alt.foobar often.
> Send it to Erik for the File.
> Oops...s/Erik/Eric/.

The s/Erik/Eric/ says "change Erik to Eric in the preceding". This syntax is borrowed from the Unix editing tools ed and sed, but is widely recognized by non-Unix hackers as well.

In a formula, * signifies multiplication but two asterisks in a row are a shorthand for exponentiation (this derives from FORTRAN). Thus, one might write 2 ** 8 = 256.

Another notation for exponentiation one sees more frequently uses the caret (^, ASCII 1011110); one might write instead 2^8 = 256. This goes all the way back to Algol-60, which used the archaic ASCII 'up-arrow' that later became the caret; this was picked up by Kemeny and Kurtz's original BASIC, which in turn influenced the design of the bc(1) and dc(1) Unix tools, which have probably done most to reinforce the convention on Usenet. (TeX math mode also uses ^ for exponention.) The notation is mildly confusing to C programmers, because ^ means bitwise exclusive-or in C. Despite this, it was favored 3:1 over ** in a late-1990 snapshot of Usenet. It is used consistently in this lexicon.

In on-line exchanges, hackers tend to use decimal forms or improper fractions ('3.5' or '7/2') rather than 'typewriter style' mixed fractions ('3-1/2'). The major motive here is probably that the former are more readable in a monospaced font, together with a desire to avoid the risk that the latter might be read as 'three minus one-half'. The decimal form is definitely preferred for fractions with a terminating decimal representation; there may be some cultural influence here from the high status of scientific notation.

Another one-line convention, used especially for very large or very small numbers, is taken from C (which derived it from FORTRAN). This is a form of 'scientific notation' using 'e' to replace '*10^'; for example, one year is about 3e7 seconds long.

The tilde (~) is commonly used in a quantifying sense of 'approximately'; that is, ~50 means 'about fifty'.

On Usenet and in the **MUD**[51] world, common C boolean, logical, and relational operators such as |, &, ||, &&, , ==, +, >, <, >= and =< are often combined with English. The Pascal not-equals, <>, is also recognized, and occasionally one sees /= for not-equals (from Ada, Common Lisp, and Fortran 90). The use of prefix '!' as a loose synonym for 'not–' or 'no–' is particularly common; thus, '!clue' is read 'no-clue' or 'clueless'.

A related practice borrows syntax from preferred programming languages to express ideas in a natural-language text. For example, one might see the following;

```
In <jrh578689@thudpucker.com> J. R. Hacker wrote:
>I recently had occasion to field-test the Snafu
>Systems 2300E adaptive gonkulator. \ The price was
>right, and the racing stripe on the case looked
>kind of neat, but its performance left something
>to be desired.

Yeah, I tried one out too.

#ifdef FLAME

Hasn't anyone told those idiots that you can't get
```

```
decent bogon suppression with AFJ filters at today's
net volumes?
#endif /* FLAME */

I guess they figured the price premium for true
frame-based semantic analysis was too high.
Unfortunately, it's also the only workable approach.
I wouldn't recommend purchase of this product unless
you're on a *very* tight budget.

#include <disclaimer.h>
--
                == Frank Foonly (Fubarco Systems)
```

In the above, the #ifdef/#endif pair is a conditional compilation syntax from C; here, it implies that the text between (which is a **flame**[43]) should be evaluated only if you have turned on (or defiend on) the switch FLAME. The #include at the end is C for "include standard disclaimer here"; the 'standard disclaimer' is understood to read, roughly, "These are my personal opinions and not to be construed as the official position of my employer."

The top section in the example, with > at the left margin, is an example of an inclusion convention we'll discuss below.

More recently, following on the huge popularity of the World Wide Web, psueudo-HTML markup has become popular for similar purposes:

```
<flame>
Your father was a hamster and your mother smelt of elderberries!
</flame>
```

You'll even see this with an HTML-style modifier:

```
<flame intensity="100%">
You seem well-suited for a career in government.
</flame>
```

Another recent (late 1990s) construction now common on USENET seems to be borrowed from Perl. It consists of using a dollar sign before an uppercase form of a word or acronym to suggest any **random**[56] member of the class indicated by the word. Thus: '$PHB' means "any random member of the class 'Pointy-Haired Boss' ".

Hackers also mix letters and numbers more freely than in mainstream usage. In particular, it is good hackish style to write a digit sequence when you intend the reader to understand the text string that names that number in English. So, hackers prefer to write '1970s' rather than 'nineteen-seventies' or '1970's' (the latter looks like a possessive).

It should also be noted that hackers exhibit much less reluctance to use multiply nested parentheses than is normal in English. Part of this is almost certainly due to influence from LISP (which uses deeply nested parentheses (like this (see?)) in its syntax a lot), but it has also been suggested that a more basic hacker trait of enjoying playing with complexity and pushing systems to their limits is in operation.

Finally, it is worth mentioning that many studies of on-line communication have shown that electronic links have a de-inhibiting effect on people. Deprived of the body-language cues through which emotioanl state is expressed, people tend to forget everything about other parties except what is presented over that ASCII link. This has both good and bad effects. A good one is that it encourages honesty and tends to break down

hierarchical authority relationships; a bad one is that it may encourage depersonalization and gratuitous rudeness. Perhaps in response to this, experienced netters often display a sort of conscious formal politesse in their writing that has passed out of fashion in other spoken and written media (for example, the phrase "Well said, sir!" is not uncommon).

Many introverted hackers who are next to inarticulate in person communicate with considerable fluency over the net, perhaps precisely because they can forget on an unconscious level that they are dealing with people and thus don't feel stressed and anxious as they would face to face.

Though it is considered gauche to publicly criticize posters for poor spelling or grammar, the network places a premium on literacy and clarity of expression. It may well be that future historians of literature will see in it a revival of the great tradition of personal letters as art.

Email Quotes

Email Quotes and Inclusion Conventions

One area where conventions for on-line writing are still in some flux is the marking of included material from earlier messages – what would be called 'block quotations' in ordinary English. From the usual typographic convention employed for these (smaller font at an extra indent), there derived a practice of included text being indented by one ASCII TAB (0001001) character, which under Unix and many other environments gives the appearance of an 8-space indent.

Early mail and netnews readers had no facility for including messages this way, so people had to paste in copy manually. BSD Mail(1) was the first message agent to support inclusion, and early Usenetters emulated its style. But the TAB character tended to push included text too far to the right (especially in multiply nested inclusions), leading to ugly wraparounds. After a brief period of confusion (during which an inclusion leading consisting of three or four spaces became established in EMACS and a few mailers), the use of leading > or > became standard, perhaps owing to uts use in ed(1) to display tabs (alternatively, it may derive from the > that some early Unix mailers used to quote lines starting with "From" in text, so they wouldn't look like the beginnings of new message headers). Inclusions within inclusions keep their > leaders, so the 'nesting level' of a quotation is visually apparent.

The practice of including text from the parent article when opsting a followup helped solve what had been a major nuisance on Usenet: the fact that articles do not arrive at different sites in the same order. Careless posters used to post articles that would give with, or even consist entirely of, "No, that's wrong" or "I agree" or the like. It was hard to see who was responding to what. Consequently, around 1984, new news-posting software evolved a facility to automatically include the text of a previous article, marked with ">" – but this too has led to undesirable workarounds, such as the deliberate inclusion of zero-content filler lines which aren't quoted and thus pull the message below the rejection threshold.

Because the default mailers supplied with Unix and other operating systems haven't evolved as quickly as human usage, the older conventions using a leading TAB or three or four spaces are still alive; however, >-inclusion is now clearly the prevalent form in both netnews and mail.

Inclusion practice is still evolving, and disputes over the 'correct' inclusion style occasionally lead to **holy wars**[45].

Most netters view an inclusion as a promise that comment on it will immediately follow. The preferred, conversational style looks like this,

```
> relevant exceprt 1
response to excerpt
> relevant excerpt 2
response to excerpt
> relevant excerpt 3
response to excerpt
```

or for short messages like this:

```
> entire message
response to message
```

Thanks to poor design of some PC-based mail agents, one will occasionally see the entire quoted message after the response, like this

```
response to message
> entire message
```

but this practice is strongly deprecated.

Though > remains the standard inclusion leader, | is occasionally used for extended quotations where original variations in indentation are being retained (one mailer even combines these and uses |>). One also sees different styles of quoting a number of authors in the same message: one (deprecated because it loses information) uses a leader of > for everyone, another (the most commmon) is > > > > , > > > , etc. (or >>>> , >>>, etc., depending on the line length and nesting depth) reflecting the original order of messages, and yet another is to use a different citation leader for each author, say > , : , | , } (preserving nesting so that the inclusion order of messages is still apparent, or tagging the inclusions with authors' names). Yet another style is to use each poster's initials (or login name) as a citatio leader for that poster.

Occasionally one sees a # leader used for quotations from authoritative sources such as standards documents; the intended allusion is to the root prompt (the special Unix command prompt issued when one is running as the privileged super-user).

Hacker Speech Style

Hackish speech generally features extremely precise diction, careful word choice, a relatively large working vocabulary, and relatively little use of contractions or street slang. Dry humor, irony, puns, and a mildly flippant attitude are highly valued – but an underlying seriousness and intelligence are essential. One should use just enough jargon to communicate precisely and identify oneself as a member of the culture; overuse of jargon or a breathless, excessively gung-ho attitude is considered tacky and the mark of a loser.

This speech style is a variety of the precisionist English normally spoken by scientists, design engineers, and academics in technical fields. In contrast with the methods of jargon construction, it is fairly constant throughout hackerdom.

It has been observed that many hackers are confused by negative questions – or, at least, that the people to whom they are talking are often confused by the sense of their answers. The problem is that they have done so much programming that distinguishes between

```
if (going) ...
```

and

```
if (!(going)) ...
```

that when they parse the question "Aren't you going?" it may seem to be asking the opposite question from "Are you going?", and so to merit an answer in the opposite sense. This confuses English-speaking non-hackers because they were taught to answer as though the negative part weren't there. In some other languages (including Russion, Chinese, and Japanese) the hackish interpretation is standard and the problem wouldn't arise. Hackers often find themselves wishing for a word like French 'si', German 'doch', or Dutch 'jawel' – a word with which one could unambiguously answer 'yes' to a negative question. (See also **mu**[51])

For similar reasons, English-speaking hackers almost never use double negatives, even if they live in a region where colloquial usage allows them. The thought of uttering something that logically ought to be an affirmative knowing it will be misparsed as a negative tends to disturb them.

In a related vein, hackers sometimes make a game of answering questions containing logical connectives with a strictly literal rather than colloquial interpretation. A non-hacker who is indelicate enough to ask a question like "So, are you working on finding that bug now or leaving it until later?" is likely to get the perfectly correct answer "Yes!" (that is, "Yes, I'm doing it either now or later, and you didn't ask which!").

International Style

Although the Jargon File remains primarily a lexicon of hacker usage in American English, we have made some effort to get input from abroad. Though the hacker-speak of other languages often uses translations of jargom from English (often as transmitted to them by earlier Jargon File versions!), the local variations are interesting, and knowledge of them may be of some use to travelling hackers.

There are some references herein to 'Commonwealth hackish'. These are intended to describe some variations in hacker usage reported in the English spoken in Great Britain and the Commonwealth (Canada, Australia, India, etc. – though Canada is heavily influenced by American usage). There is also an entry on **Commonwealth Hackish**[40] reporting some general phonetic and vocabulary differences from U.S. hackish.

Hackers in Western Europe and (especially) Scandinavia report that they often use a mixture of English and their native languages for technical conversation. Occasionally they develop idioms in their English usage that are influenced by their native-language styles. Some of these are reported here.

On the other hand, English often gives rise to grammatical and vocabulary mutations in the native language. For example, Italian hackers often use the nonexistent verbs 'scrollare' (to scroll) and 'deletare' (to delete) rather than native Italian 'scorrere' and 'cancellare'. Similarly, the English verb 'to hack' has been conjugated in Swedish. And Spanish-speaking hackers use 'linkar' (to link), 'debuggar' (to debug), and 'lockear' (to lock).

European hackers report that this happens parly because the English terms make finger distinctions than are available in their native vocabularies, and partly because deliberate language-crossing makes for amusing wordplay.

A few notes on hackish usage in Russian have been added where they are parallel with English idioms and thus comprehensible to English-speakers.

Lamer-speak

From the early 1980s onward, a flourishing culture of local, MS-DOS-based bulletin boards has been developing separately from Internet hackerdom. The BBS culture has, as its seamy underside, a stratum of 'pirate boards' inhabited by **cracker**[40]s, phone phreaks, and **warez d00dz**[61]. These people (mostly teenagers running IBM-PC clones from their bedrooms) have developed their own characteristic jargon, heavily influenced by skateboard lingo and underground-rock slang.

Though crackers often call themselves 'hackers', they aren't (they typically have neither significant programming ability, nor Internet expertise, nor experience with UNIX or other true multi-user systems). Their vocabulary has little overlap with hackerdom's. Nevertheless, this lexicon covers much of it so the reader will be able to understand what goes by on bulletin-board systems.

Here is a brief guide to cracker and **warez d00dz**[61] usage:

- Misspell frequently. The substitutions

  phone ⇒ fone
  freak ⇒ phreak

  are obligatory.
- Always substitute 'z's for 's's. (i.e. "codes" → "codez").
- Type random emphasis charactrs after a post line (i.e. "Hey Dudes!#!#!#!$").
- Use the emphatic 'k' prefix ("k-kool", "k-rad", "k-awesome") frequently.
- Abbreviate compulsively ("I got lotsa warez w/ docs").
- Substitute '0' for 'o' ("r0dent", "l0zer").
- TYPE ALL IN CAPS LOCK, SO IT LOOKS LIKE YOU'RE YELLING ALL THE TIME.

These traits are similar to those of **B1FF**[39], who originated as a parody of naive **BBS**[39] users. Occasionally, this sort of distortion may be used as heavy sarcasm by a real hacker, as in:

```
> I got X windows running under Linux!

d00d!  u R an \1337 hax0r
```

The only practice resembling this in actual hacker usage is the substitution of a dollar sign of 's' in names of products or service felt to be excessively expensive, e.g., Compu$serve, Micro$oft.

For further discussion of the pirate-board subculture, see **lamer**[50], **elite**[42], **leech**[50], **poser**[54], **cracker**[40], and especially **warez d00dz**[61]. Although cracker slang (which they call leetspeak, out of a corruption of the word "elite") is distinct from hacker jargon, some leetspeak terms are piped from hacker fargon and then

transformed according to the above rules. For example, newbie became "n00b". Recently, leetspeak has become mainstream and mutated into a further form called LOLSPEAK, which is generally derived from shorthand used in text messaging (SMS). Again, these are distinct from hacker slang, though some have produced an unholy hybrid known as LOLCODE (as well as lolbash), an esoteric programming language which uses LOLSPEAK in its syntax.

Part II

How to Use the Lexicon

Pronunciation Guide

Pronunciation keys are provided in the jargon listings for all entries that are neither dictionary words pronounced as in standard English nor obvious compounds thereof. Slashes bracket phonetic pronunciations, which are to be interpreted using the following conventions:

1. Syllables are hyphen-separated, except that an accent or back-accent follows each accented syllable (the back-accent marks a secondary accent in some words of four or more syllables). If no accent is given, the word is pronounced with equal accentuation on all syllables (this is common for abbreviations).

2. Consonants are pronounced as in American English. The letter 'g' is always hard (as in "got" rather than "giant"); 'ch' is soft ("church" rather than "chemist"). The letter 'j' is the sound that occurs twice in "judge". The letter 's' is always as in "pass", never a z sound. The digraph 'kh' is the guttural of "loch" or "l'chaim". The digraph 'gh' is the aspirated g+h of "bughouse" or "ragheap" (rare in English).

3. Uppercase letters are pronounced as their English letter names; thus (for example) /H-L-L/ is equivalent to /aych el el/. /Z/ may be pronounced /zee/ or /zed/ depending on your local dialect.

4. Vowels are represented as follows:

   back, that
   father, palm (see note)
   far, mark
   flaw, caught
   bake, rain
   less, men
   easy, ski
   their, software
   trip, hit
   life, sky
   block, stock (see note)
   flow, sew
   loot, through
   more, door
   out, how
   boy, coin
   but, some
   put, foot
   yet, young
   few, chew
   /oo/ with optional fronting as in 'news' (/nooz/ or /nyooz/)

The glyph /*/ is used for the 'schwa' sound of unstressed or occluded vowels (the one that is often written with an upside-down 'e'). The schwa vowel is omitted in syllables containing vocalic r, l, m or n; that is, in 'kitten' and 'color' would be rendered /kit'n/ and /kuhl'r/, not /kit'*n/ and /kuhl'*r/.

Note that the above table reflects mainly distinctions found in standard American English (that is, the neutral dialect spoken by TV network announcers and typical of educated speech in the Upper Midwest, Chicago, Minneapolis/St. Paul and Philadelphia). However, we separate /o/ from /ah/, which tend to merge in standard American. This may help readers accustomed to accents resembling British Received Pronunciation.

The intent of this sceme is to permit as many readers as possible to map the pronunciations into their local dialect by ignoring some subset of the distinctions we make. Speakers of British RP, for example, can smash terminal /r/ and all unstressed vowels. Speakers of many varieties of southern American will automatically map /o/ to /aw/; and so forth. (Standard American makes a good reference dialect for this purpose because it has crisp consonants and more vowel distinctions than other major dialects, and tends to retain distinctions between unstressed vowels. It also happens to be what your editor speaks.)

Entries with a pronunciation of '//' are written-only usages. (No, Unix weenies, this does not mean 'pronounce like previous pronunciaton'!)

Other Lexicon Conventions

Entries are sorted in case-blind ASCII collation order (rather than the letter-by-letter order ignoring interword spacing common in mainstream dictionaries), except that all entries beginning with nonalphabetic characters are sorted after Z. The case-blindness is a feature, not a bug.

The beginning of each entry is marked by a colon (:) at the left margin. This convention helps out tools like hypertext browsers that benefit from knowing where entry boundaries are, but aren't as context-sensitive as humans.

In pure ASCII renderings of the Jargon File, you will see {} used to bracket words which themselves have entries in the File. This isn't done all the time for every such word, but it is done everywhere that a reminder seems useful that the term has a jargon meaning and one might wish to refer to its entry.

In this all-ASCII version, headwords for topic entries are distinguished from those for ordinary entries by being followed by "::" rather than ":"; similarly, references are surrounded by "{{" and "}}".

Defining instances of terms and phrases appear in 'slanted type'. A defining instance is one which occurs near to or as part of an explanation of it.

Prefixed ** is used as linguists do; to mark examples of incorrect usage.

We follow the 'logical' quoting convention described in the Writing Style section above. In addition, we reserve double quotes for actual excerpts of text or (sometimes invented) speech. Scare quoets (which mark a word being used in a nonstandard way), and philosopher's quotes (which turn an utterance into the string of letters or words that name it) are both rendered with single quotes.

References such as malloc(3) and patch(1) are to Unix facilities (some of which, such as patch(1) are actually freeware distributed over Usenet). The Unix manuals use foo(n) to refer to item foo in section (n) of the manual, where n=1 is utilities, n=2 is system calls, n=3 is C library routines, n=6 is games, and n=8 (where present) is system administration utilities. Sections 4, 5, and 7 of the manuals have changed roles frequently and in any case are not referred to in any of the entries.

Various abbreviations used frequently in the lexicon are summarized here:

abbrev. abbreviation

adj. adjective

adv. adverb

alt. alternate

cav. caveat

conj. conjunction

esp. especially

excl. exclamation

imp. imperative

interj. interjection

n. noun

obs. obsolete

pl. plural

pref. prefix

prob. probably

prov. proverbial

quant. quantifier

suff. suffix

syn. synonym (or synonymous with)

v. verb (may be transitive or intransitive)

var. variant

vi. intransitive verb

vt. transitive verb

Where alternate spellings or pronunciations are given, alt. separates two possibilities with nearly equal distribution, while var. prefixes one that is markedly less common than the primary.

Where a term can be attributed to a particular subculture or is known to have originated there, we have tried to so indicate. Here is a list of abbreviations used in etymologies:

Amateur Packet Radio A technical culture of ham-radio sites using AX.25 and TCP/IP for wide-area networking and BBS systems.

Berkeley University of California at Berkeley

BBN Bolt, Beranek & Newman

Cambridge the university in England (not the city in Massachusetts where MIT happens to be located!)

CMU Carnegie-Mellon University

Commodore Commodore Business Machines

DEC The Digital Equipment Corporation (now Compaq).

Fairchild The Fairchild Instruments Palo Alto development group

FidoNet See the **FidoNet**[43] entry

IBM International Business Machines

MIT Massachusetts Institute of Technology; esp. the legendary MIT AI Lab culture of roughly 1971 to 1983 and its feeder groups, including the Tech Model Railroad Club

NRL Naval Research Laboratories

NYU New York University

OED The Oxford English Dictionary

Purdue Purdue University

SAIL Stanford Artificial Intelligence Laboratory (at Stanford University)

SI From Systéme International, the name for the standard conventions of metric nomenclature used in the sciences

Stanford Stanford University

Sun Sun Microsystems

TMRC Some MITisms go back as far as the Tech Model Railroad Club (TMRC) at MIT c. 1960. Material marked TMRC is from "An Abridged Dictionary of the TMRC Language", originally compiled by Pete Samson in 1959

UCLA University of California at Los Angeles

UK the United Kingtom (England, Wales, Scotland, Northern Ireland)

Usenet See the **Usenet**[59] entry

WPI Worcester Polytechnic Institute, site of a very active community of PDP-10 hackers during the 1970s

WWW The World-Wide-Web.

XEROC PARC XEROX's Palo Alto Research Center, site of much pioneering research in user interface design and networking

Yale Yale University

Some other etymology abbreviatons such as **Unix**[59] and **PDP-10**[54] refer to technical cultures surrounding specific operating sytems, processors, or other environments. The fact that a term is labelled with any one

of these abbreviations does not necessarily mean its use is confined to that culture. In particular, many terms labelled 'MIT' and 'Stanford' are in quite general use. We have tried to give some indication of the distribution of speakers in the usage notes; however, a number of factors mentioned in the introduction conspire to make these indications less definite than might be desirable.

A few new definitions attached to entries are marked [proposed]. These are usually generalizations suggested by editors or Usenet respondents in the process of commenting on previous definitinos of those entries. These are not represented as established jargon.

Format for New Entries

You can mail submissions for the Jargon File to `yashtulsyan@gmail.com`.

We welcome new jargon, and corrections to or amplifications of existing entries. You can improve your submission's chances of being included by adding background information on user population and years of currency. References to actual usage via URLs and/or DejaNews pointers are particularly welcomed.

All contributions and suggestions about the Jargon File will be considered donations to be placed in the public domain as part of this File, and may be used in subsequent paper editions. Submissions may be edited for accuracy, clarity and concision.

We are looked to expand the File's range of technical specialties covered. There are doubtless rich veins of jargon yet untapped in the scientific computing, graphics, and networking hacker communities; also in numerical analysis, computer architectures and VLSI design, language design, and many other related fields. Send us your jargon!

We are not interested in straight technical terms explained by textbooks or technical dictionaries unless an entry illuminates 'underground' meanings or aspects not covered by official histories. We are also not interested in 'joke' entries – there is a lot of humor in the file but it most flow naturally out of the explanations of what hackers do and how they think. Furthermore, we will emphatically reject all entries used to promote a political or religious agenda unless it's justified by a large proportion of hackers (e.g. the Free Software movement will get into the Jargon File, but libertarianism, which does not account for a majority of hackers, would not). This is one of the reasons why version 5.0.0 was forked from ESR-2003.

It is OK to submit items of jargon you have originated if they have spread to the point of being used by people who are not personally acquainted with you. We prefer items to be attested by independent submission from two different sites.

Please send URLs for materials related to the entries, so we can enrich the File's link structure.

The Jargon File will be regularly maintained and made available for browsing on the World Wide Web, and will include a version number. Read it, pass it around, contribute – this is your monument!

The Jargon Lexicon

ad-hockery

A

B

B1FF

barf

bboard

BBS

beam

bignum 1 (sense 1) 2 (sense 2) 3 (sense 3)

BNF

boxen

brain-damaged

burble

C

chomp

Commonwealth Hackish

core

cracker

crock

cruft

cuspy

D

Datamation

DEC

droid

E

elegant adj.

[common; from mathematical usage] Combining simplicity, power, and a certain ineffable grace of design. Higher praise than 'clever', 'winning', or even **cuspy**[40].

The French aviator, adventurer, and author Antoine de Saint-Exupéry, probably best known for his classic children's book "The Little Prince", was also an aircraft designer. He gave us perhaps the best definition of engineering elegance when he said "A designer knows he has achieved perfection not when there is nothing left to add, but when there is nothing left to take away."

elite

email

F

FidoNet

flamage

flame

flavorful

foo

foobar

frob /frob/

1 n.

[MIT; very common] The **TMRC**[58] definition was "FROB = a protruding arm or trunnion"; by metaphoric extension, a 'frob' is any random small thing, an object that you can comfortably hold in one hand; something you can frob (sense 2). See **frobnitz**[43]. 2 vt. Abbreviated form of **frobnicate**[43]. from the **MUD**[51] world A command on some MUDs that changes a player's experience level (this can be used to make wizards); also, to request **wizard**[61] privileges on the 'professional courtesy' grounds that one is a wizard elsewhere. The command is actually 'frobnicate' but is universally abbreviated to the shorter form.

frobnicate

frobnitz

FTP /F-T-P/, not /fit'ip/

techspeak n. The File Transfer Protocol for transmitting files between systems on the Internet. 1 vt. To **beam**[39] a file using the File Transfer Protocol. 2 Sometimes used as a generic even for file transfers not using **FTP**[43]. · "Lemme get a copy of *Wuthering Heights* ftp'd from uunet."

G

Gosperism

grep

H

hack

HCF /H-C-F/ n.

Mnemonic for 'Halt and Catch Fire', any of several undocumented and semi-mythical machine instructions with destructive side-effects, supposedly included for test purposes on several well-known architectures going as far back as the IBM 360. The MC6800 microprocessor was the first for which an HFC opcode became widely known. This instruction caused the processor to **toggle**[58] a subset of the bus lines as rapidly as it could; in some configurations this could actually cause lines to burn up. Compare **killer poke**[48].

> Once upon a time there was a machine called the Compucolor II. It was made by Intecolor, a terminal manufacturer, and was deliberately crippled to keep it from competing with their terminals. They apparently were able to save about $50 in hardware by giving the software more control over it, and cutting safety margins. The file system required al files be contiguous on disk: when you deleted a file it compressed the rest of the disk, using video RAM as a temporary buffer. Amusing, maybe. But not a very nice machine.
>
> OK, if you executed the following BASIC program, the display would vanish, it would make weird "singing" noises, and a burned smell would be immediately obvious. Turning off the power switch didn't have any effect: you have to pull the plug. Whether it would actually catch fire, I was too chicken to find out.
>
> The program?
>
> ```
> FOR I = 0 TO 255:  OUT 6, I: NEXT I
> ```
>
> Anyone know what this did, really?
>
> (da Silva, Peter, 1989: Real life HCF code in this message *alt.folklore.computers*, 14 December)

high moby

holy wars

I

ITS /I-T-S/ n.

1 Incompatible Time-sharing System, an influential though highly idiosyncratic operating system written for PDP-6s and PDP-10s at MIT and long used at the MIT AI Lab. Much AI-hacker jargon derives from ITS folklore, and to have been 'an ITS hacker' qualifies one instantly as an old-timer of the most venerable sort. ITS pioneered many important innovations, including transparent file sharing between machines and terminal-independent I/O. After about 1982, most actual work was shifted to newer machines, with the remaining ITS boxes run essentially as a hobby and service to the hacker community. The shutdown of the lab's last ITS machine in May 1990 marked the end of an era and sent old-timer hackers into mourning nationwide (see **high moby**[45]) 2 A mythical image of operating-system perfection worshiped by a bizarre, fervent retro-cult of old-time hackers and ex-users (see **troglodyte (sense 2)**[58]). ITS worshipers manage somehow to continue believing that an OS maintained by assembly-language hand-hacking that supported only monocase 6-character filenames in one directory per account remains superior to today's state of commercial art (their venom against **Unix**[59] is particularly intense). See also **holy wars**[45], **Weenix**[61].

J

K

killer poke

kludge /klooj/

kluge /klooj/

[from the German 'klug', clever; poss. Polish 'klucz' (a key, a hint, a mainpoint)] 1 n. A Rube Goldberg (or Heath Robinson) device, whether in hardware or software. 2 n. A clever programming trick intended to solve a particularly nasty case in an expedient, if not clear, manner. Often used to repair bugs. Often involves **ad-hockery**[37] and verges on being a **crock**[40]. 3 n. Something that works for the wrong reason. 4 vt. To insert a kludge into a program. "I've kluged this routine to get around that weird bug, but there's probably a better way." WPI n. A feature that is implemented in a **rude**[56] manner.

Nowadays this term is often encountered in the variant spelling 'kludge'. Reports from **old fart**[64]s are consistent that 'kluge' was the original spelling, reported around computers as far back as the mid-1950s and, at that time, exclusively of *hardware* kluges. In 1947, the "New York Folklore Quarterly" reported a classic shaggy-dog story 'Murgatroyd the Kluge Maker' then current in the Armed Forces, in which a 'kluge' was a complex and puzzling artifact with a trivial function. Other sources report that 'kluge' was common Navy slang in the WWII era for any piece of electronics that worked well on shore but consistently failed at sea.

However, there is a reason to believe this slang use may be a decade older. Several respondents have connected it to the brand name of a device called a "Kluge paper feeder", an adjunct to mechanical printing presses. Legend has it that the Kluge feeder was designed before small, cheap electric motors and control electronics; it relied on a fiendishly complex assortment of cams, belts, and linkages to both power and synchronize all its operations from one motive driveshaft. It was accordingly temperamental, subject to frequent breakdowns, and devilishly difficult to repair – but oh, so clever! People who tell this story also aver that 'Kludge' was the name of a design engineer.

There is in fact a Brandtjen & Kluge Inc., an old family business that manufactures printing equipment - interestingly, their name is pronounced /kloo'gee/! Henry Brandtjen, president of the firm, told me (ESR, 1994) that his company was co-founded by his father and an engineer named Kluge /kloo'gee/, who built and co-designed the original Kluge automatic feeder in 1919. Mr. Brandtjen claims, however, that this was a *simple* device (with only four cams); he says he has no idea how the myth of its complexity took hold.

TMRC[58] and the MIT hacker culture of the early '60s seems to have developed in a milieu that remembered and still used some WWII military slang (see also **foobar**[43]). It seems likely that 'kluge' came to MIT via alumni of the many military electronics projects that had been located in Cambridge (many in MIT's venerable Building 20, in which **TMRC**[58] is also located) during the war.

The variant 'kludge' was apparently popularized by the **Datamation**[41] article mentioned above; it was titled "How to Design a Kludge" (February 1962, pp. 30, 31). This spelling was probably imported from Great Britain, where **kludge**[48] has an independent history (though this fact was largely unknown to hackers on either side of the Atlantic before a mid-1993 debate in the Usenet group *alt.folklore.computers* over the First

and Second Edition versions of this entry; everybody used to think **kludge**[48] was just a mutation of **kluge**[48]). It now appears that the British, having forgotten the etymology of their own 'kludge' when 'kluge' crossed the Atlantic, repaid the U.S. by lobbing the 'kludge' orthography in the other direction and confusing their American cousins' spelling!

The result of this history is a tangle. Many younger U.S. hackers pronounce the word as /klooj/ but spell it, incorrectly for its meaning and pronunciation, as 'kludge'. (Phonetically, consider huge, centrifuge, and deluge as opposed to sludge, judge, budge, and fudge. Whatever its failings in other areas, English spelling is perfectly consistent about this distinction.) British hackers mostly learned /kluhj/ orally, use it in a restricted negative sense and are at least consistent. European hackers have mostly learned the word from written American sources and tend to pronounce it /kluhj/ but use the wider American spelling!

Some observers consider this mess appropriate in view of the word's meaning.

L

lamer

leech

lose

lossage

M

management

meeces

moby /moh'bee/

[MIT: seems to have been in use among model railroad fans years ago. Derived from Melville's "Moby Dick" (some say from 'Moby Pickle'). Now common.] 1 adj. Large, immense, complex, impressive. · "A Saturn V rocket is a truly moby frob." · "Some MIT undergrads pulled off a moby hack at the Harvard-Yale game." (See **Appendix A**[65] for discussion.) 2 n. obs. The maximum address space of a machine (see below). For a 680[234]0 or VAX or most modern 32-bit architectures, it is 4,294,967,296 8-bit bytes (4 gigabytes). 3 A title or address (never of third-person reference), usually used to show admiration, respect, and/or friendliness to a completent hacker. · "Greetings, moby Dave. How's that address-book thing for the Mac going?" 4 adj. In backgammon, doubles the dice, as in 'moby sixes', 'moby ones', etc. Compare this with **bignum (sense 3)**[39]: double sixes are both bignums and moby sixes, but moby ones are not bignums (the use of 'moby' to describe double ones is sarcastic). Standard emphatic forms: 'Moby foo', 'moby win', 'moby loss'. 'Foby moo': a spoonerism due to Richard Greenblatt. 5 The largest available unit of something which is available in discrete increments. Thus, ordering a "moby Coke" at the local fast-food joint is not just a request for a large Coke, it's an explicit request for the largest size they sell.

This term entered hackerdom with the Fabritek 256K memory added to the MIT AI PDP-6 machine, which was considered unimaginably huge when it was installed in the 1960s (at a time when a more typical memory size for a timesharing system was 72 kilobytes). Thus, a moby is classically 256K 36-bit words, the size of a PDP-6 or PDP-10 moby. Back when address registers were narrow the term was more generally useful, because when a computer had virtual memory mapping, it might actually have more physical memory attached to it than any one program could access directly. One could then say · "This computer has 6 mobies" meaning that the ratio of physical memory to address space is 6, without having to say specifically how much memory there actually is. That in turn implied that the computer could timeshare six 'full-sized' programs without having to swap programs between memory and disk.

Nowadays the low cost of processor logic means that address spaces are usually larger than the most physical memory you can cram onto a machine, so most systems have much *less* space than one theoretical 'native' moby of **core**[40]. Also, more modern memory-management techniques (esp. paging) make the 'moby count' less significant. However, there is one series of widely-used chips for which the term could stand to be revived – the Intel 8088 and 80286 with their incredibly **brain-damaged**[39] segmented-memory designs. On these, a 'moby' would be the 1-megabyte address span of a segment/offset pair (by coincidence, a PDP-10 moby was exactly 1 megabyte of 9-bit-bytes).

mu

MUD

N

newsgroup

NIL

O

P

PDP-10

PHB

phreaking /freek'ing/ n.

[from 'phone phreak'] 1 The art and science of

cracking the phone network (so as, for example, to make free long-distance calls). 2 By extension, security-cracking in any other context (especially, but not exclusively, on communications networks) see (**cracking**[54]).

At one time phreaking was a semi-respectable activity among hackers; there was a gentleman's agreement that phreaking as an intellectual game and a form of exploration was OK, but serious theft of services was taboo. There was significant crossover between the hacker community and the hard-core phone phreaks who ran semi-underground networks of their own through such media as the legendary *TAP Newsletter*. This ethos began to break down in the mid-1980s as wider dissemination of the techniques put them in the hands of less responsible phreaks. Around the same time, changes in the phone network made old-style technical ingenuity less effective as a way of hacking it, so phreaking came to depend more on overtly criminal acts such as stealing phone-card numbers. The crimes and punishments of gangs like the '414 group' turned that game very ugly. A few old-time hackers still phreak casually just to keep their hand in, but most these days have hardly even heard of 'blue boxes' or any of the other paraphernalia of the greatest phreaks of yore.

NOT IN THE JARGON FILE

NOT IN THE JARGON FILE

The *TAP Newsletter* seems to have previously been called the *YIPL Newsletter* or the *Youth International Party Line Newsletter*. The first issue, accessible at `<http://artofhacking.com/tap/yipl/live/aoh_yipl01.htm>`, was published in June 1971.

pointy-haired

poser

Q

R

random

rude

S

T

T

talk mode

TMRC

toggle

troglodyte 1 (sense 1) 2 (sense 2)

TWENEX /twe'neks/ n.

The TOPS-20 operating system by **DEC**[41] – the second proprietary OS for the PDP-10 – preferred by most PDP-10 hackers over TOPS-10 (that is, by those who were not **ITS**[46] or **WAITS**[61] partisans). TOPS-20 began in 1969 as Bolt, Beranek & Newman's TENEX operating sysetm using special paging hardware. By the early 1970s, almost all of the systems on the ARPANET ran TENEX. DEC purchased the rights to TENEX from BBN and began work to make it their own. The first in-house code name for the operating system was VIROS (VIRtual memory Operating System); when customers started asking questions, the name was changed to SNARK so DEC could truthfully deny that there was any project called VIROS. When the name SNARK became known, the name was priefly reversed to become KRANS; this was quickly abandoned when someone objected that 'krans' meant 'funeral wreath' in Swedish (though some Swedish speakers have since said it means simply 'wreath'; this part of the story may be apocryphal). Ultimately DEC picked up TOPS-20 as the name of the operating system, and it was as TOPS-20 that it was marketed. The hacker community, mindful of its origins, quickly dubbed it TWENEX (a contraction of 'twenty TENEX'), even though by this point very little of the original TENEX code remained (analogously to the differences between AT&T V6 Unix and BSD). DEC people cringed when they heard "TWENEX", but the term caught on nevertheless (the written abbreviation '20x' was also used). TWENEX was successful and very popular; in fact, there was a period in the early 1980s when it commanded as fervent a culture of partisans as Unix or ITS – but DEC's decision to scrap all the internal rivals to the VAX architecture and its relatively stodgy VMS OS killed the DEC-20 and put a sad end to TWENEX's brief day in the sun. DEC attempted to convince TOPS-20 users to convert to **VMS**[60], but instead, by the late 1980s, most of the TOPS-20 hackers had migrated to Unix.

U

Unix

Usenet /yoos'net/ or /yooz'net/ n.

[from 'Users Network'; the original spelling was USENET, but the mixed-case form is now widely preferred] A distributed **bboard**[39] (bulletin board) system supported mainly by Unix hachines. Originally implemented in 1979-1980 by Steve Bellovin, Jim Ellis, Tom Truscott, and Steve daniel at Duke University, it has swiftly grown to become international in scope and is now probably the largest decentralized information utility in existence. As of early 1996, it hosts over 10,000 **newsgroup**[52]s and an average of over 500 megabytes (the equivalent of several thousand paper pages) of new technical articles, news, discussion, chatter, and **flamage**[43] every day (and that leaves out the graphics. . .).

By the year the Internet hit the mainstream (1994) the original UUCP transport for Usenet was fading out of use (see **UUCPNET**[59]) - almost all Usenet connections were over Internet links. A lot of newbies and journalists began to refer to "Internet newsgroups" as though Usenet was and always had been just another Internet service. This ignorance greatly annoys experienced Usenetters. After the advent of the World Wide Web, Web-based email clients (such as Gmail and Yahoo! mail), as well as online groups (Google Groups, though they also contain Usenet archives), Usenet has fallen out of common usage, though some hackers still u se it. The signal-to-noise ratio is still pretty low, many being spam posts from crackers and befuddled Google Groups users who think that Usenet was always just another Google service.

UUCPNET

V

VAXen

vi

VMS

W

WAITS

warez d00dz

Weenix

win

wizard

wtf [Not present in Jargon File.]

> In the Altos 68000, one of the sagans of 68000 Unix boxes that flooded the market back around '83, the ROM loader had one syntax error message:
>
> <beep> WTF?
>
> I mentioned it to a tech support type once when I called about something else and they had an answer ready - it meant "With Trace Flag?".
>
> Yeah, sure.
>
> (Nuchia, Steve, 1989: DEC's sense of humor *alt.folklore.computers*, 14 December)

X

Y

Z

old fart

Stupids

suit n.

1 Ugly and uncomfortable 'business clothing' often worn by non-hackers. Invariably worn with a 'tie', a strangulation device that partially cuts off the blood supply to the brain. It is thought that this explains much about the behavior of suit-wearers. Compare **droid**[41]. 2 A person who habitually wears suits, as distinct from a techie or hacker. See **pointy-haired**[54], **burble**[39], **management**[51], **Stupids**[64], **SNAFU principle**[64], **PHB**[54], and **brain-damaged**[39].

SNAFU principle

Appendix A

Some AI Koans

These are some of the funnies examples of a genre of jokes told at the MIT AI Lab about various noted hackers. The original koans were composed by Danny Hillis, who would later found Connection Machines, Inc. In reading these, it is at least useful to know that Minsky, Sussman, and Drescher are AI researchers of note, that Tom Knight was one of the Lisp machine's principal designers, and that David Moon wrote much of Lisp Machine Lisp.

* * *

A novice was trying to fix a broken Lisp machine by turning the power off and on.

Knight, seeing what the student was doing, spoke sternly: "You cannot fix a machine by just power-cycling it with no understanding of what is goign wrong."

Knight turned the machine off and on.

The machine worked.

* * *

One day a student came to Moon and said: "I understand how to make a better garbage collector. We must keep a reference count of the pointers to each cons."

Moon patiently told the student the following story:

"One day a student came to Moon and said: 'I understand how to make a better garbage collector. . . '
"

[Ed. note: Pure reference-count garbage collectors have problems with circular structures that point to themselves.]

* * *

In the days when Sussman was a novice, Minsky once came to him as he sat hacking at the PDP-6.

"What are you doing?", asked Minsky.

"I am training a randomly wired neural net to play Tic-Tac-Toe" Sussman replied.

"Why is the net wired randomly?", asked Minstky.

"I do not want it to have any preconceptions of how to play", Sussman said.

Minsky then shut his eyes.

"Why do you close your eyes?", Sussman asked his teacher.

"So that the room will be empty."

At that moment, Sussman was enlightened.

* * *

A disciple of another sect once came to Drescher as he was eating his morning meal.

"I would like to give you this personality test", said the outsider, "because I want you to be happy."

Drescher took the paper that was offered him and put it in the toaster, saying: "I wish the toaster to be happy, too."

Appendix B

Appendix C

Bibliography