Kian Agheli
Professor Richard Peters
English 101 – 41477
December 11 2023
Research Paper – Final Draft

The Impact of the C Programming Language on The Past Fifty Years of Computing

The C Programming Language was initially developed between 1969 and 1979. It was further refined when standardized by the American National Standards Institute in 1989. In the twenty years between birth and standardization, C became the dominant programming language, shaping the environments it entered and the programmers who used it. In the time between ANSI standardization and now, the majority of computing systems have been designed to leverage C.

A programming language is a format for expressing logic. The programming language that a computer natively understands, called an "instruction set," is composed of minuscule operations. Across all kinds of digital computers, these allow for logical evaluation, performing arithmetic, management of information, and general hardware control (Intel 23), (Arm 21), and (Waterman 19).

These instructions are expressed in binary digits, commonly contracted to "bits" (Mackenzie 80). Unlike a decimal digit, which may be a whole value between 0 and 9, a bit may be either 0 or 1. In digital computing's infancy, series of these were manually "toggled in" (Post 83), using physical switches and buttons. Save for theories put on paper, such as the works of Ada Lovelace (Menabrea 1842) and Alan Turing (Turing 36), this was the first form of computer programming.

As hardware and software both advanced, programmers became able to write programs textually. They would physically interact with a teleprinter, essentially a typewriter which would send key strokes to and print output from a computer. They would digitally interact with software to manage and store teleprinter input, naturally named "text editors." Then, the stored input from the teleprinter would be passed to a program called an "assembler," to further interpret the saved input. The textual representations of computer instructions, such as "NOT A," would be encoded as real computer instructions, such as "00100001 01101010" (Nelson 63), (Ritchie 99), (Thompson 71a), (Thompson 71b), (Thompson 71c). The text which programmers would provide to assemblers came to be called "assembly language." (IBM 92).

Assembly language directly correlates to a computer's instruction set, meaning that the programmer must textually manage each operation in a program. Doing so is less tedious than flipping switches hundreds of times over, but still tedious (Nather 83), (Post 83). Computer scientists found that a computer may instead be provided a text which vaguely describes the intended logic of a program. Though the computer has been provided less information, it may have enough produce a reasonable translation of the abstract program to its own instruction set. As opposed to an assembler, the software which performs this kind of interpretation is called a "compiler" (Aho 14).

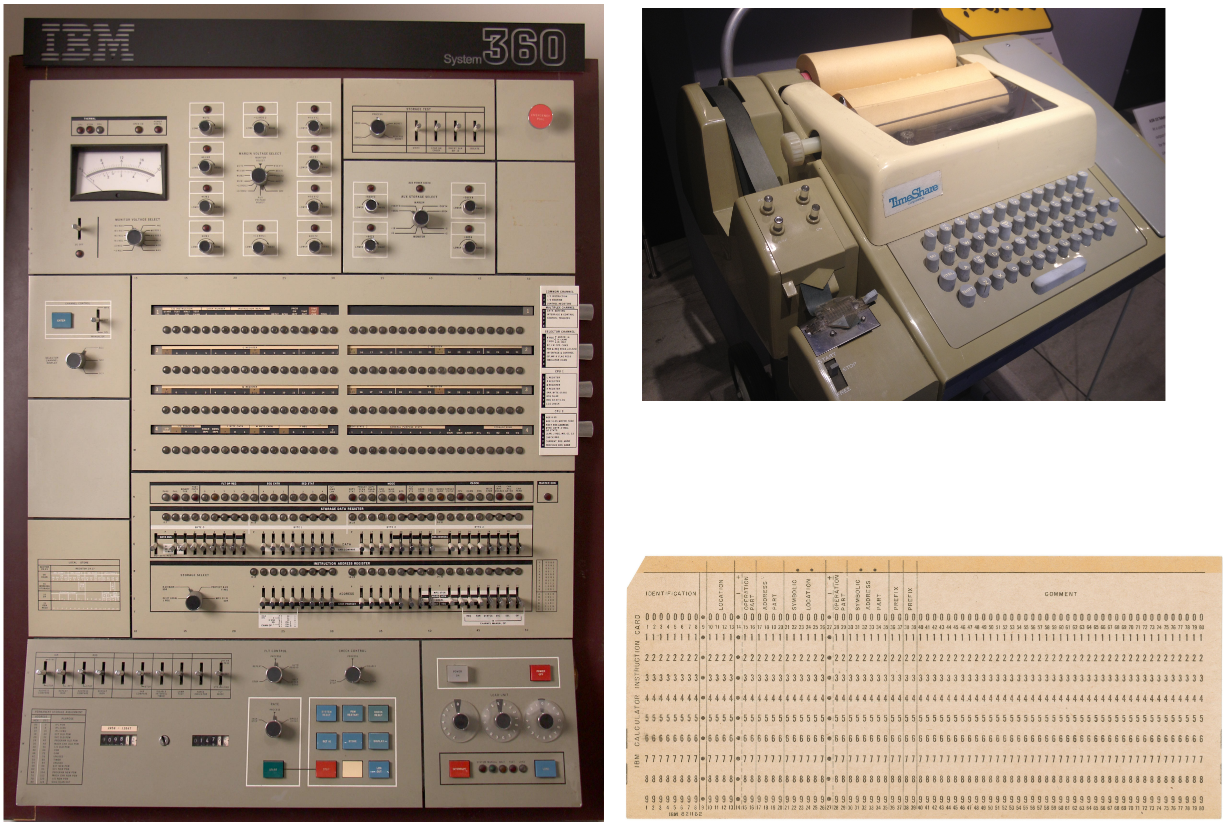

Figure 1. Left: the control panel of an early digital computer, designed to be programmed through the toggling of several switches (Shirriff 19). Top right: A Teletype-branded teleprinter (Reinhold 14). Bottom right: A punch card, at a time the most economical method of storing program data (Jones 22).

Compiling a program from an abstract — "high-level" — language is comparable to translating expressions from one human language to another: it may be done, loosely (Aho 14). With this loss in accuracy comes two benefits: programs may be written in less time, and structured to read closer to human language; and programs may be interpreted with similar logical meanings on architecturally different computers. These benefits are so strong that the vast majority of today's software is written in high-level programming languages (TIOBE 23), (Backus 56).

C was among the languages developed to transition from programming in assembly to a high-level language (Backus 56), (Ritchie 93). It began its development at Bell Laboratories, a subsidiary of AT&T, "as a system implementation language for the nascent Unix operating system" (Ritchie 93). Unix was a set of software to enable a singular computer to run multiple programs concurrently — as opposed to one at a time. Further, it fulfilled the basic infrastructure needed to make hardware accessible in software (Ritchie 74), (Ritchie 84).

```c
#include <stdio.h>

int
main()
{
	puts("Hello.");
	return 0;
}
```

```
	.data
msg:	.asciz "Hello.\n"
	len = . - msg - 1
	.text
	.global _start
_start:
	pushl	$len
	pushl	$msg
	pushl	$1
	movl	$4, %eax
	call	do_syscall
	addl	$12, %esp
	pushl	$0
	movl	$1, %eax
	call	do_syscall
do_syscall:
	int	$0x80
	ret
```

Figure 2. Left: A trivial program written in C for printing “Hello.” to the screen. Written by the author. Right: A program written in Assembly for the same purpose. It assumes a specific version of modern Unix and a PC-architecture computer. Adapted from (Sommers 00).

Unix and C were designed around each-other. C was designed to leverage the hardware abstractions provided by Unix, and much of Unix itself came to be written in C (Ritchie 93). This bond allowed C and Unix to form the first “portable” programming environment. In such an environment, programs written for one set of hardware may run largely or wholly unmodified on another set of hardware (Johnson 78). Unix and C — and many programs written in C to run on top of Unix — could then be ported to a new machine with relative ease. “[T]hough originally unplanned,” Unix’s authors discovered it “to be possible to produce an operating system and set of software that runs on several machines and whose expression in source code is, except for a few modules, identical on each machine” (Ritchie 76).

Despite C and Unix forming a revolutionary product, they were distributed liberally to universities. AT&T was not oblivious to C’s value, however their hand was forced. C and Unix were to be distributed liberally, or not distributed at all.

> AT&T held a government-sanctioned monopoly on the US telephone system. The terms of AT&T's agreement with the US government prevented it from selling software, which meant that it could not sell UNIX as a product. Instead, beginning in 1974 with Fifth Edition, and especially with Sixth Edition, AT&T licensed UNIX for use in universities for a nominal distribution fee. … AT&T's release of UNIX into universities greatly contributed to the popularity and use of the operating system, and by 1977, UNIX was running at some 500 sites, including 125 universities in the United States and several other countries. (Kerrisk 10)

Due to C and Unix's bonded success, in tandem with an affordable price, C became the preferred programming language in education.

This liberty in distribution led to several derivative versions of Unix. At University of California, Berkeley, Unix was modified and re-written over several years. It became considered by its authors as an independent, compatible operating system. By 1983, the system had become a substantial superset of Unix. The "Berkeley Software Distribution" itself became a licensed product, eventually the basis for "[s]everal commercial OSes [operating systems]" (Salus 94a).

Come 1988, independent recreations of Unix had been released, under totally free terms. They were largely compatible with Unix, but contained no actual Unix source code. These could be distributed without restriction. This was in stark contrast to AT&T's selective licensing to universities. Following legal dispute with AT&T, Berkeley Software Distribution itself became free to distribute, spawning the BSD family of operating systems (Salus 94b), (OpenBSD 23), (NetBSD 23), (FreeBSD 23). As Unix became both feature-rich and free to license, its influence spread further. By necessity, where Unix and its derivatives went, C went too.

As C became increasingly popular outside of Bell Labs, "it was clear that C needed formal standardization" (Ritchie 93). The language informally documented did not wholly match the language in use (Kernighan 78) (Ritchie 75). As of 1989, C became formally standardized by the American National Standards Institute (ANSI 89), (X3J11 98). This formal standard became integral to the continued growth of C's popularity. "[T]he incipient use of C in projects subject to commercial and government contract meant that the imprimatur of an official standard was important" (Ritchie 93). Separately, C's sister project, Unix, became standardized (IEEE 07), (Harbour 01), (Stallman 11).

Thanks in large part to their standardization, Unix and C have been implemented and re-implemented several times. Modern versions of Unix continue to be designed in C (Linux 23), (OpenBSD 23), (NetBSD 23), (FreeBSD 23), (Illumos 23). On computers which provide services en masse — public access web sites; e-mail servers — these are go-to choices (StatCounter 23a), (Fortune 23). The small computers that manage home internet connections often use the same or similar infrastructure

(Dunn 18), (OpenWRT 23). In essence, Unix's descendants enable the internet as we know it today, and they themselves are enabled by C.

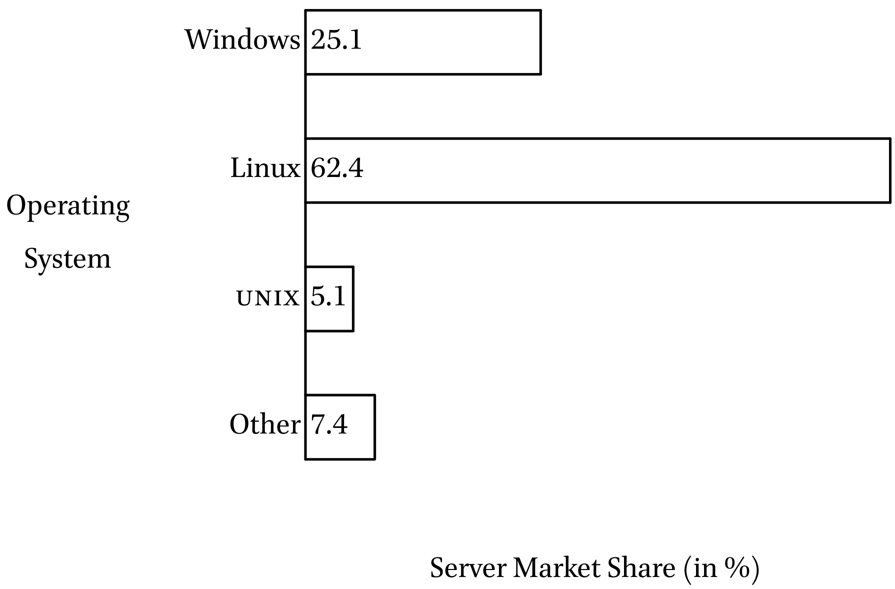

Figure 3. Market share of operating systems for server use (Fortune 23). "Linux" is a specific derivative of Unix. "UNIX," as written in the provided data, generally refers to BSD-derived Unix. The distinction is arguably made due to Linux's overwhelming popularity.

More generally, C and Unix have become integral to commercial software development. Among today's most affluent software companies, Apple (Nicas 22) has been using C-derived languages since their acquisition of NeXTSTEP in 1997 (Singh 06). NeXTSTEP was a BSD-derived operating system. It was designed in a super-set of C named Objective-C. At the time of the acquisition, Apple's Macintosh line of computers were in need of new system software. They proceeded to use NeXTSTEP — implemented in C and borrowing portions of Unix — as the basis for their advancement (Reisinger 16), (Singh 06), (Singh 03), (Edwards 20). Years after the acquisition, it became the basis for the software behind the iPhone (Apple 23a), (Garling 12).

Before acquiring NeXTSTEP, Apple was near bankruptcy. NeXTSTEP saved them, and they have since flourished. In particular, the 2007 release of the iPhone earned Apple enormous success (Martins 23). Today, the iPhone itself is now uncontroversially called one of the most revolutionary products of our time (Leswing 19), (Eadicicco 17). It's not a stretch to say that C was and continues to be integral to that success. Accordingly, Apple has invested heavily in their C-based infrastructure, funding the development of a new set of compiler software (Treat 05), (Apple 12), (Apple 23b).

Microsoft, a company which genuinely competes with Apple for "World's Most Valuable Company" (Klebnikov 21), has similar ties to C. Their premier software product, Windows (Ward 23), has been written in C since 1995 (Microsoft 14). In the quarter-century since then, Microsoft has maintained a compatible software environment. Aside from other features (Microsoft 23a), this allows years-old programs to run largely or wholly unmodified on Microsoft's latest iterations of Windows (Finck 20), (Microsoft 22).

Individual programs, lesser in scope than operating systems, are often written in C because of its ubiquity. Programmers intending to write accessible, performant software resort to it. Notable examples include Google's Chrome, the most popular web browser (StatCounter 23b); ffmpeg, an exceedingly popular video processing solution (Melanson 11), (Maki 20), (Larabel 15); SQLite, a set of software included in a majority of consumer computers (SQLite 22); and OpenSSH, the networked computer log-in utility bundled with most of today's laptops (Microsoft 23b), (Loder 22). Programs originally written in languages other than C, but that need portability, may be machine-translated to C. This has been most notably done with TeX (TUG 23), a program for digital document typesetting, popularly used for academic papers (CTAN 23), (Beeton 18).

C's portability is the result of serious effort. Aside from the work of standards committees, C has been wedged into environments far different from Unix. For the sake of porting programs written in C — "a good deal of interesting software" as early as 1976 (Ritchie 76), subsets of Unix would be implemented on other operating systems.

As Microsoft's operating systems took hold in the home computer market, the desire to port C programs to them took hold just as firmly. For this purpose, Unix was first grafted on top of MS-DOS beginning in 1989 (Zaretskii 99). As MS-DOS was replaced by Windows, that work was carried over, resulting in a multitude of projects. Most prominently, Cygwin allows a large portion of standards-compliant Unix programs to run without modification. It maps Unix features to Windows features, and those Unix features which have no analogue are emulated (Cygwin 22). MinGW is a separate but related project to port modern, publicly-availably C compilers to Windows (MinGW 23).

These projects enabled programmers to write programs to run on all three of the popular software platforms — Apple's BSD-based Macintosh, IBM's PC and derivatives running Microsoft's Windows, and the hardware-agnostic Unix servers (Reimer 05). These projects became integral to the furthering of Windows's software library. Microsoft decided to further the effort for portability of C, by means of the WSL project. While the previous methods require Unix software to be wholly re-compiled for Windows, Microsoft's solution provides a full Unix environment. (Microsoft 23c), (Microsoft 23d).

Language	*Year*							
	2023	2018	2013	2008	2003	1998	1993	1988
Python	1	4	8	6	11	24	22	-
C	2	2	1	2	2	1	1	1
C++	3	3	4	3	3	2	2	4
Java	4	1	2	1	1	18	-	-
C#	5	5	5	8	9	-	-	-
JavaScript	6	8	9	9	8	21	-	-
Visual Basic	7	18	-	-	-	-	-	-
PHP	8	7	6	5	6	-	-	-
SQL	9	88	-	-	7	-	-	-
Assembly Language	10	13	-	-	-	-	-	-
Ada	24	31	22	20	16	16	7	3
Objective-C	27	12	3	41	48	-	-	-
Lisp	30	29	14	17	15	10	8	2
(Visual) Basic	-	-	7	4	5	3	3	7

Figure 4. Estimated and averaged programming language popularity over the course of a given year, from 1988 through 2023 (TIOBE 23). "1" marks the first most popular, "30" marks the 30th most popular. "(Visual) Basic" and "Visual Basic" are intentionally distinct. The former refers to a now split family of languages. The latter refers to a specific language from that split.

C's influence has permeated the evolution of programming languages. The high-level programming languages which rival C's popularity (TIOBE 23) borrow heavily from its appearance and semantics. Briefly analyzing the table above, it is only these languages which have popularly flourished for the past two decades. C++ is a direct derivative of C. Java resembles C closely, to the point at which valid C is nearly valid Java. Python and C# each take basic syntax from C. JavaScript is intended to be C for the web browser (Miller 22), (Ezick 02). These are the notable market competitors. Evidently, C shaped the market for 50 years past its inception.

```
while   return   break   continue
 if      else     for
```

Figure 5. Key words with similar semantics in C, Python, C++, Java, C#, and JavaScript (Microsoft 23e), (Microsoft 21), (Mozilla 23), (Python 23). Despite being distinct languages, some of their semantics are similar.

C's popularity has warranted continued extension and standardization. Though the 1989 standard remains de-facto, C has been continually standardized by the International Organization for Standardization and the International Electrotechnical Commission (WG14 21). C has been grown from a singular language to a family of languages. This growth blurs the lines between C and its contemporaries, bringing modern considerations into the old language's design.

As opposed to languages distinct from C, newly standardized versions of C have the benefit of its existing software infrastructure. That infrastructure has been honed over half of a century. The authors of languages operating in the same market as C have a lofty goal: to beat 50 years of pre-built software, documentation, and compiler improvement. Particularly, programs written in C are as fast and efficient as high-level programs may be. If there are better ways of generating code than modern C compilers perform, we don't know of them. By comparison, most other languages and their compilers are grossly inefficient. (Pereira 17).

1. C, Pascal, Go
2. Rust, C++, Fortran
3. Ada
4. Java, Chapel, Lisp, Ocaml
5. Swift, Haskell, C#
6. Dart, F#, Racket, Hack, PHP
7. JavaScript, Ruby, Python
8. TypeScript, Erlang
9. Lua, JRuby, Perl

Figure 6. The measured energy, time, and memory efficiency of programming languages, separated into distinct tiers (Pereira 17). C is in the top tier, and the first in its class.

C is everywhere. It's the basis for software in home computers, cell phones, Wi-Fi routers, and web servers. Those categories arguably span the majority of computing systems people interact with today. Despite C's ubiquity, it is not trivial. The time before C's standardization had no similar language. The time since has produced derivatives, but no replacement. Before the advent of these derivatives, C had become so universal as to be a "lingua franca" among programming languages (Armstrong 14). Modern and popular competition remains structurally and syntactically similar.

The popularity of programming languages derived from C is comparable to the popularity of natural languages derived from Latin. In each case, the derivatives are unique, but similar enough to be structurally and sometimes literally compatible. The gaping hole in that comparison is that, quite unlike Latin, C is very much a living language. No computing system is complete without it.

Works Cited

(ANSI 89) American National Standard Institute; International Organization for Standardization, *American National Standard for Programming Languages – C; ANSI/ISO 9899-1990.* International Organization for Standardization, 1990.

(Aho 14) Aho, Alfred V.; Lam, Monica S.; Sethi, Ravi; Ullman, Jeffrey D. *Compilers: Principles, Techniques, and Tools Second Edition.* Pearson Education Limited, 2014.

(Apple 12) Apple, "LLVM Compiler Overview." Apple, 13 December 2012, developer.apple.com/library/archive/documentation/CompilerTools/Conceptual/LLVMCompilerOverview/index.html. Accessed 14 November 2023.

(Apple 23a) Apple, "xnu." Github, 26 September 2023, github.com/apple-oss-distributions/xnu/commit/1031c584a5e37aff177559b9f69dbd3c8c3fd30a. Accessed 13 November 2023.

(Apple 23b) Apple, "LLVM / Clang." Apple, 2023, opensource.apple.com/projects/llvm-clang. Accessed 14 November 2023.

(Arm 21) Arm Limited. *Arm® A64 Instruction Set Architecture Armv8, for Armv8-A architecture profile,* p. 2. Arm Limited, March 2021, documentation-service.arm.com/static/606ef2575e70d934bc69e1bf. Accessed 11 November 2023.

(Armstrong 14) Joe Armstrong. "The Mess We're In." Strange Loop Conference via YouTube, 19 September 2014, https://www.youtube.com/watch?v=lKXe3HUG2l4&t=1149

(Backus 56) J.W. Backus, R.J. Beeber, S. Best, R. Goldberg, H.L. Herrick, R.A. Hughes, L.B. Mitchell, R.A. Nelson, R. Nutt, D. Sayre, P.B. Sheridan, H. Stern, I. Ziller, "The FORTRAN Automatic Coding System for the IBM 704 EDPM: Programmer's Reference Manual." IBM, 15 October 1956. Re-published at archive.computerhistory.org/resources/text/Fortran/102649787.05.01.acc.pdf. Accessed 8 December 2023.

(Beeton 18) Barbara Beeton, Karl Berry, David Walden. "TeX: A Branch in Desktop Publishing Evolution, Part 1," in *Annals of the History of Computing,* vol.40, no.3, pp. 78–93. July–September 2018. Peer-reviewed manuscript published at www.walden-family.com/ieee/texhistory.html. Accessed 9 December 2023.

(CTAN 23) Comprehensive TeX Archive Network. "What are TeX and its friends?" 2023, ctan.org/tex. Accessed 9 December 2023.

(Cygwin 22) Cygwin authors. *Cygwin User's Guide,*, pp. 1–2. 27 January 2022, cygwin.com/cygwin-ug-net/cygwin-ug-net.pdf. Accessed 9 December 2023.

(Dunn 18) John E. Dunn. "Most home routers lack simple Linux OS hardening security." Sophos Ltd, 20 December 2018, web.archive.org/web/20230322194141/https://nakedsecurity.sophos.com/2018/12/20/most-home-routers-lack-simple-linux-os-hardening-security. Accessed 8 December 2023.

(Eadicicco 17) Lisa Eadicicco, "This Is Why the iPhone Upended the Tech Industry." TIME, 29 June 2017, time.com/4837176/iphone-10th-anniversary/. Accessed 13 November 2023.

(Edwards 20) Benj Edwards. "Before Mac OS X: What Was NeXTSTEP, and Why Did People Love It?" How-To Geek, 7 November 2020, www.howtogeek.com/698532/before-mac-os-x-what-was-nextstep-and-why-did-people-love-it. Accessed 10 December 2023.

(Ezick 02) James Ezick. "Java and C similarities, by example (not a complete list)." Cornell University, Spring 2002, www.cs.cornell.edu/courses/cs202/2002sp/JavaCcomparison.html. Accessed 9 December 2023.

(Finck 20) Colin Finck. "Writing Win32 apps like it's 2020: Introduction." ENLYZE GmbH, 30 July 2020, building.enlyze.com/posts/writing-win32-apps-like-its-2020-part-1. Accessed 8 December 2023.

(Fortune 23) Fortune Business Insights. *Server Operating System Market Volume, Share & COVID-19 Impact Analysis, By Operating System* (*Windows, Linux, UNIX, and Others*)*, By Virtualization Status* (*Virtual Machine, Physical, and Virtualized*)*, By Subscription Model* (*Non-paid Subscription and Paid Subscription*)*, By Enterprise Type* (*Large Enterprises and Small & Medium Enterprises*)*, and Regional Forecast, 2023-2030.* Fortune Business Insights, May 2023, www.fortunebusinessinsights.com/server-operating-system-market-106601. Accessed 8 December 2023.

(FreeBSD 23) freebsd.org. Accessed 15 November 2023.

(Garling 12) Caleb Garling, "iPhone Coding Language Now World's Third Most Popular." WIRED, 9 July 2012, www.wired.com/2012/07/apple-objective-c. Accessed 14 November 2023.

(Google 23a) Google LLC. "Set up for Android Development." Google LLC, 2023, source.android.com/docs/setup/about. Accessed December 2023.

(Google 23b) The Chromium Authors. "Developer Information for Chrome OS Devices." Google LLC, 2023, www.chromium.org/chromium-os/developer-information-for-chrome-os-devices. Accessed 9 December 2023.

(Harbour 01) Michael González Harbour, "REAL-TIME POSIX: AN OVERVIEW." Universidad de Cantabria Departamento de Electrónica, 16 March 2001, www.cs.unc.edu/˜anderson/teach/comp790/papers/posix-rt.pdf. Accessed 13 November 2023.

(IBM 92) IBM. *High Level Assembler for* Z/OS & Z/VM & Z/VSE *Language Reference Version 1 Release 6*. IBM, 1992, https://publibz.boulder.ibm.com/epubs/pdf/asmr1021.pdf. Accessed 11 November 2023.

(IEEE 07) IEEE, *Draft Standard for Information Technology—Portable Operating System Interface* (*POSIX*®). IEEE, 2007, www.open-std.org/jtc1/sc22/open/n4217.pdf. Accessed 13 November 2023.

(Illumos 23) illumos.org. Accessed 8 December 2023.

(Intel 23) Intel Corporation. *Intel® 64 and IA-32 Architectures Software Developer's Manual Combined Volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D, and 4.* Intel Corporation, September 2023, cdrdv2.intel.com/v1/dl/getContent/671200, Volume 1 pp. 5-5. Accessed 11 November 2023.

(Johnson 78) S. C. Johnson, Dennis M. Ritchie, "Portability of C Programs and the UNIX System" in *The Bell System Techinical Journal*, Vol. 57, No. 6, Part 2, July–August 1978, pp. 2021-2048. Re-published at www.bell-labs.com/usr/dmr/www/portpap.pdf. Accessed 8 December 2023.

(Jones 22) Douglas W. Jones. "Punched Cards for Computer Programs." University of Iowa Department of Computer Science, 5 December 2022, homepage.divms.uiowa.edu/˜jones/cards/collection/i-program.html. Date derived from image meta data. Specifically, from "821162IBM701code.jpg." Accessed 10 December 2023.

(Kernighan 78) Brian W. Kernighan, Dennis M. Ritchie, *The C Programming Language.* Prentice Hall, 1978.

(Kernighan 88) Brian W. Kernighan; Dennis M. Ritchie, *The C Programming Language, Second Edition.* Prentice Hall, 1988.

(Kerrisk 10) Michael Kerrisk, *The Linux Programming Interface.* No Starch Press, 2010. pp 3–5.

(Klebnikov 21) Sergei Klebnikov. “Microsoft is Now The World’s Most Valuable Company After Apple Falls On Earnings.” Forbes Media, 29 October 2021, www.forbes.com/sites/sergeiklebnikov/2021/10/29/microsoft-is-now-the-worlds-most-valuable-company-after-apple-falls-on-earnings. Accessed 8 December 2023.

(Larabel 15) Michael Larabel. “Firefox Enables FFmpeg Support By Default.” Phoronix Media, 15 November 2015, www.phoronix.com/news/Firefox-FFmpeg-Default. Accessed 9 December 2023.

(Leswing 19) Kif Leswing, “The iPhone decade: How Apple’s phone created and destroyed industries and changed the world.” CNBC, 16 December 2019, www.cnbc.com/2019/12/16/apples-iphone-created-industries-and-changed-the-world-this-decade.html. Accessed 13 November 2023.

(Linux 23) kernel.org. Accessed 15 November 2023.

(Loder 22) Chip Loder. “How to use SSH for secure connections in macOS.” Quiller Media, 15 December 2022, appleinsider.com/inside/macos/tips/how-to-use-ssh-for-secure-connections-in-macos. Accessed 9 December 2023.

(Mackenzie 80) Charles E Mackenzie, *Coded Character Sets, History and Development.* Addison-Wesley Publishing Company, 1980, archive.org/download/mackenzie-coded-char-sets/Mackenzie_CodedCharSets_text.pdf, pp. 12. Accessed 11 November 2023.

(Mahoney 16) Michael S. Mahoney, *An Oral History of Unix.* 1 January 2016, gromnitsky.users.sourceforge.net/lit/an-oral-history-of-unix/book.pdf. Accessed 12 November 2023.

(Maki 20) J.N. Maki, D. Gruel, C. McKinney, M.A. Ravine, M. Morales, D. Lee, R. Willson, D. Copley-Woods, M. Valvo, T. Goodsall, J. McGuire, R.G. Sellar, J.A. Schaffner, M.A. Caplinger, J.M. Shamah, A.E. Johnson, H. Ansari, K. Singh, T. Litwin, R. Deen, A. Culver, N. Ruoff, D. Petrizzo, D. Kessler, C. Basset, T. Estlin, F. Alibay, A. Nelessen, S. Algermissen. "The Mars 2020 Engineering Cameras and Microphone on the Perseverance Rover: A Next-Generation Imaging System for Mars Exploration," excerpted from *The NASA Mars 2020 Mission: Seeking Signs of Ancient Life and Preparing for Sample Return*, p. 37. Springer, 24 November 2020, www.ncbi.nlm.nih.gov/pmc/articles/PMC7686239. Accessed 9 December 2023.

(Martins 23) Daniel Martins, unnamed contributor. "Apple In The 1990s: Why It Nearly Went Bankrupt." The Arena Group Holdings, 10 February 2023, www.thestreet.com/apple/news/apple-in-the-1990s-why-it-nearly-went-bankrupt. Accessed 10 December 2023.

(Melanson 11) Mike Melanson. "Google's YouTube Uses FFmpeg." 8 February 2011, multimedia.cx/eggs/googles-youtube-uses-ffmpeg. Accessed 9 December 2023.

(Menabrea 1842) L. F. Menabrea, Ada Augusta Lovelace, *Sketch of The Analytical Engine Invented by Charles Babbage* in *Bibliothèque Universelle de Genève*, October 1842, No. 82. Re-published at www.fourmilab.ch/babbage/sketch.html. Accessed 12 November 2023.

(Microsoft 14) Microsoft, "Windows NT System Overview." Microsoft, 20 February 2014, learn.microsoft.com/en-us/previous-versions/cc767881(v=technet.10). Accessed 14 November 2023.

(Microsoft 21) Microsoft. "C Keywords." Microsoft, 20 September 2021, learn.microsoft.com/en-us/cpp/c-language/c-keywords. Accessed 10 December 2023.

(Microsoft 22) Microsoft, "Get Started with Win32 and C++." Microsoft, 27 January 2022, learn.microsoft.com/en-us/windows/win32/learnwin32/learn-to-program-for-windows. Accessed 14 November 2023.

(Microsoft 23a) Microsoft. "Make older apps or programs compatible with Windows." Microsoft, 2023, support.microsoft.com/en-us/windows/make-older-apps-or-programs-compatible-with-windows-783d6dd7-b439-bdb0-0490-54eea0f45938. Accessed 8 December 2023.

(Microsoft 23b) Microsoft. "Tutorial: SSH in Windows Terminal." Microsoft, 28 September 2023, learn.microsoft.com/en-us/windows/terminal/tutorials/ssh.

(Microsoft 23c) Microsoft. "How to install Linux on Windows with WSL." Microsoft, 28 October 2023, learn.microsoft.com/en-us/windows/wsl/install. Accessed 10 December 2023.

(Microsoft 23d) Microsoft. "WSL2-Linux-Kernel." Github, 6 October 2023, github.com/microsoft/WSL2-Linux-Kernel. Accessed 10 December 2023.

(Microsoft 23e) Microsoft. "C# Keywords." Microsoft, 22 April 2023, learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords. Accessed 10 December 2023.

(Miller 22) Stephan Miller. "How Are C, C++, C#, and Objective-C Different?" Codecademy, 4 March 2022, www.codecademy.com/resources/blog/c-vs-cplusplus-vs-csharp-vs-objective-c. Accessed 9 December 2023.

(MinGW 23) www.mingw-w64.org. Accessed 10 December 2023.

(Mozilla 23) Mozilla. "[JavaScript's] Lexical grammar." Mozilla, 2023, developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Lexical_grammar#keywords. Accessed 10 December 2023.

(Nather 83) Ed Nather, "The story of Mel," in *The Jargon File*. 21 May 1983, stuff.mit.edu/afs/sipb/user/marc/stuff.athena/jargon/jargon2910.ascii.gz. Accessed 11 November 2023.

(Nelson 63) R. A. Nelson; K. M Lovitt, "History of Teletype Development." Teletype Corporation, October 1963. Re-published at web.archive.org/web/20201105231651/http://www.thocp.net/hardware/history_of_teletype_development_.htm. Accessed 11 November 2023.

(NetBSD 23) netbsd.org. Accessed 15 November 2023.

(Nicas 22) Jack Nicas, "Apple Becomes First Company to Hit $3 Trillion Market Value." The New York Times, 3 January 2022, www.nytimes.com/2022/01/03/technology/apple-3-trillion-market-value.html. Accessed 13 November 2023.

(OpenBSD 23) openbsd.org. Accessed 15 November 2023.

(OpenWRT 23) OpenWRT. "OpenWRT Hardware List." OpenWRT, 2023, openwrt.org/docs/techref/hardware/list. Accessed 8 December 2023.

(Pereira 17) Rui Pereira, Marco Couto, Francisco Ribeiro, Rui Rua, Jácome Cunha, João Paulo Fernandes, João Saraiva. “Energy Efficiency across Programming Languages: How Do Energy, Time, and Memory Relate?” Association for Computing Machinery, 23–24 October 2017, greenlab.di.uminho.pt/wp-content/uploads/2017/10/sleFinal.pdf. Accessed 11 December 2023.

(Post 83) Ed Post, “Real Programmer’s Don’t Use Pascal.” *Datamation,* Volume 29, Number 7, July 1983. Re-published digitally at www.usm.uni-muenchen.de/~hoffmann/roff/tmp/rpdup.pdf; www.pbm.com/~lindahl/real.programmers.html; www.ecb.torontomu.ca/~elf/hack/realmen.html. Accessed 11 November 2023.

(Python 23) Python Software Foundation. *The Python Language Reference,* “Lexical analysis.” Python Software Foundation, 2023, docs.python.org/3/reference/lexical_analysis.html#identifiers. Accessed 10 December 2023.

(Reimer 05) Jeremy Reimer. “Total share: 30 years of personal computer market share figures.” WIRED Media Group, 14 December 2005, arstechnica.com/features/2005/12/total-share. Accessed 11 December 2023.

(Reinhold 06) Arnold Reinhold. “Punch card from a typical Fortran program.” Wikimedia Commons, 11 May 2006,

commons.wikimedia.org/wiki/File:FortranCardPROJ039.agr.jpg. Accessed 10 December 2023.

(Reinhold 14) Arnold Reinhold. “Teletype Corporation ASR-33 on display at the Computer History Museum.” Wikimedia Commons, 2 February 2014, commons.wikimedia.org/wiki/File:ASR-33_at_CHM.agr.jpg. Accessed 10 December 2023.

(Reisinger 16) Don Reisinger. “Steve Jobs Sold NeXT to Apple 20 Years Ago Today.” Fortune Media IP Limited, 20 December 2016, fortune.com/2016/12/20/apple-next-anniversary. Accessed 10 December 2023.

(Ritchie 74) Dennis M. Ritchie, Ken Thompson, Eric A. Brewer, “The Unix Time-Sharing System” in *Communications of the ACM,* July 1974, Volume 17, Number 7. Re-published at landley.net/history/mirror/unix/unix.pdf. Accessed 12 November 2023.

(Ritchie 75) Dennis M. Ritchie, *C Reference Manual.* Bell Laboratories, May 1975. Re-published at doc.cat-v.org/unix/v6/operating-systems-lecture-notes/v6/doc/c.ps and www.tuhs.org/cgi-bin/utree.pl?file=V6/usr/doc/c.

(Ritchie 76) Dennis M. Ritchie, "The UNIX Time-sharing System—A Retrospective." Bell Laboratories, January 1977, www.bell-labs.com/usr/dmr/www/retro.pdf. Accessed 8 December 2023.

(Ritchie 84) Dennis M. Ritchie, "The Evolution of the Unix Time-sharing System" in *AT&T Bell Laboratories Technical Journal 63 No. 6 Part 2*, October 1984, pp. 1577-1593. Re-published at read.seas.harvard.edu/~kohler/class/aosref/ritchie84evolution.pdf. Accessed 12 November 2023.

(Ritchie 93) Dennis M. Ritchie, "The Development of the C Language." Association for Computing Machinery, April 1993, www.bell-labs.com/usr/dmr/www/chist.pdf. Accessed 11 November 2023.

(Ritchie 99) Dennis M. Ritchie, "Unix Programmer's Manual, November 3, 1971." Bell Laboratories, 11 January 1999, www.bell-labs.com/usr/dmr/www/1stEdman.html.

(SQLite 22) SQLite. "Most Widely Deployed and Used Database Engine." SQLite, 8 January 2022, www.sqlite.org/mostdeployed.html. Accessed 9 December 2023.

(Salus 94a) Peter H. Salus. *A Quarter Century of UNIX.* Addison-Wesley Publishing, 1994, p. 171.

(Salus 94b) Peter H. Salus. *A Quarter Century of UNIX.* Addison-Wesley Publishing, 1994, pp. 222–225.

(Shirriff 19) Ken Shirriff. "Iconic consoles of the IBM System/360 mainframes, 55 years old." April 2019, www.righto.com/2019/04/iconic-consoles-of-ibm-system360.html. Accessed 10 December 2023.

(Singh 03) Amit Singh. "What is Mac OS X?" December 2003, web.archive.org/web/20120514135706/http://osxbook.com/book/bonus/ancient/whatismacosx/history.html. Accessed 10 December 2023.

(Singh 06) Amit Singh, *Mac OS X Internals: A Systems Approach.* Addison Wesley, 19 June 2006.

(Sommers 00) Thoman M. Sommers. "Frequently Asked Questions for FreeBSD 2.X, 3.X and 4.X." 2000, web.archive.org/web/20001027104103/http://home.ptd.net/~tms2/hello.html. Accessed 9 December 2023.

(Stallman 11) Richard Stallman, "The origin of the name POSIX." 11 May 2011, stallman.org/articles/posix.html. Accessed 13 November 2023.

(StatCounter 23a) StatCounter. "Operating System Market Share Worldwide - November 2023." StatCounter, December 2023, gs.statcounter.com/os-market-share#monthly-202211-202311. Accessed 8 December 2023.

(StatCounter 23b) StatCounter. "Browser Market Share Worldwide - November 2023." StatCounter, December 2023, https://gs.statcounter.com/browser-market-share/. Accessed 9 December 2023.

(TIOBE 23) TIOBE Software BV. "TIOBE Index for November 2023." TIOBE Software BV, December 2023, www.tiobe.com/tiobe-index. Accessed 9 December 2023.

(TUG 23) TeX User's Group. "Web2c." February 2022, tug.org/texinfohtml/web2c.html. Accessed 10 December 2023.

(Thompson 05) Ken Thompson; John Mashey; Yan Rosenshteyn, "Thompson, Ken oral history." Computer History Museum, 8 February 2005, computerhistory.org/collections/catalog/102657921. Accessed 20 October 2023.

(Thompson 71a) Ken Thompson, Dennis M. Ritchie, "Section 1 Part 1" in *Unix Programmer's Manual.* Bell Laboratories, 3 November 1971, www.bell-labs.com/usr/dmr/www/man11.pdf.

(Thompson 71b) Ken Thompson, Dennis M. Ritchie, "Section 1 Part 2" in *Unix Programmer's Manual.* Bell Laboratories, 3 November 1971, www.bell-labs.com/usr/dmr/www/man12.pdf.

(Thompson 71c) Ken Thompson, Dennis M. Ritchie, "Introduction" in *Unix Programmer's Manual.* Bell Laboratories, 3 November 1971, www.bell-labs.com/usr/dmr/www/manintro.pdf.

(Treat 05) Adam Treat, "Qt4-preview-feedback Archive, February 2005 mkspecs and patches for LLVM compile of Qt4." 19 February 2005, http://lists.trolltech.com/qt4-preview-feedback/2005-02/msg00691.html. Accessed 14 November 2023.

(Turing 36) Alan M. Turing, "On Computable Numbers, With An Application to the Entscheidungsproblem." 12 November 1936. Re-published at www.cs.virginia.edu/~robins/Turing_Paper_1936.pdf. Accessed 12 November 2023.

(VCF 19) Vintage Computer Federation; Ken Thompson; Brian W. Kernighan, "Ken Thompson interviewed by Brian Kernighan at VCF East 2019." YouTube, 6 May 2019, youtube.com/watch?v=EY6q5dv_B-o. Accessed 20 October 2023.

(WG14 07) Joint Technical Committee 1, Subcommittee 22, Working Group 14, ISO/IEC 9899:TC3 *Committee Draft* (*C99 with Technical Corrigendum 1, 2, and 3*). ISO/IEC, 7 September 2007, www.open-std.org/JTC1/SC22/WG14/www/docs/n1256.pdf. Accessed 11 November 2023.

(WG14 21) WG14. "ISO/IEC JTC1/SC22/WG14 - C." ISO/IEC, 25 November 2021, www.open-std.org/jtc1/sc22/wg14. Accessed 10 December 2023.

(Ward 23) Keith Ward, "A Brief History of Microsoft Windows." Lifewire, 9 February 2023, www.lifewire.com/brief-history-of-microsoft-windows-3507078. Accessed 14 November 2023.

(Waterman 19) Andrew Waterman; Krste Asanović, *The RISC-V Instruction Set Manual Volume I: Unprivileged ISA*. RISC-V International, 13 December 2019, riscv.org/wp-content/uploads/2019/12/riscv-spec-20191213.pdf, pp. 18. Accessed 11 November 2023.

(X3J11 98) X3J11, *Rationale for International Standard - Programming Language - C.* ANSI, 22 Jan 1998, www.open-std.org/JTC1/SC22/WG14/www/docs/n802.pdf. Accessed 11 November 2023.

(Zaretskii 99) Eli Zaretskii. "The DJGPP Project." July 1999, www.delorie.com/djgpp/doc/eli-m17n99.html. Accessed 9 December 2023.