LOGICAL TECHNIQUES IN COMPUTER SCIENCE
Open Journal

Review

The Importance of Logic for Computer and Electrical Engineers

Laura I. Wilder*; Amitav Ghosh

Department of Artificial Intelligence, MI, USA

Corresponding author
Laura I. Wilder
Department of Artificial Intelligence, MI, USA

Article information
Received: September 13th, 2023; **Revised:** December 22nd, 2023; **Accepted:** January 3rd, 2024; **Published:** January 18th, 2024

Cite this article
Wilder LI, Ghosh A. The importance of logic for computer and electrical engineers. 2024; 2(1). doi: https://doi.org/10.70705/ppp.ltcs.2024.v02.i01.pp1-6

ABSTRACT

Undergraduates majoring in computing disciplines, such as electrical engineering (EE), computer engineering (CE), and computer science (CS), should be required to take a course in logic, and specifically mathematical logic. This is supported by three main points: 1) the integral role of logic in the development of computers, 2) research demonstrating that students who possess strong logical thinking abilities outperform their peers in subjects like mathematics and programming, and 3) the qualities that employers value most in candidates. The authors also think that pupils, particularly those from under-represented groups, would retain more information if they are taught reasoning clearly. The results show that logic education is important for STEM (science, technology, engineering, and mathematics) areas in general, not only computers.

Keywords
Education in the fields of engineering and computer science; Logic; IEEE; Student retention; Attritio.

INTRODUCTION

Aristotle's work was an important step towards formalizing logic, but as Frege and Leibniz complained, logical relations made in natural languages can be vague and ambiguous [2]. As a result, Frege and Leibniz suggested natural languages be replaced by formal languages, a goal which eventually led the development of what is now called mathematical logic. According to logician Jon Barwise, "Modern mathematical logic has its origins in the dream of Leibniz of a universal symbolic calculus which could encompass all mental activity of a logically rigorous nature, in particular, all of mathematics [3]." Significant progress towards Leibniz's dream was made in the 19th century by Augustus De Morgan [4] and George Boole [5], both of whom used mathematics to describe Aristotle's syllogisms [6]. This helped to formalize the field of mathematical logic.

About a century later, Claude Shannon showed the applica- bility of Boole's Laws of Thought [7] to designing electronics in his masters thesis titled "A symbolic analysis of relay and switching circuits" [8]. Mathematician Herman Goldstine con- sidered Shannon's work "one of the most important master's theses ever written, [helping] change digital circuit design from an art to a science [9]." It was Shannon's knowledge of logic that allowed him to establish the fundamentals of digital circuit design over a decade before the invention of the transistor in 1947 [10].Many of the pioneers in electrical engineering and computer science had strong foundations in logic, which is necessary to understand the history of these fields and their subsequent developments. As a result, the authors of this paper propose that teaching logic should form a necessary and foundational part of the undergraduate curriculum in these areas, which are subsequently called the computing fields. Unfortunately, there has been a curricular shift away from deductive logic, to a greater focus on induction, in large part due to the success of machine learning, which is primarily inductive [11]. If students are to understand and contribute to the computing fields, they need to understand topics from logic and mathematical logic; in particular, deductive reasoning, first-order logic, and set theory.

Related Work

A. Computational Thinking

In a 2006 publication, computer scientist Jeannette Wing used the term "computational thinking" to describe an approach to computer science that places greater emphasis on the underlying mental processes than on simple programming [12]. Dijkstra said in 1988 [13] that students shouldn't be able to run their own programs in an introductory programming course "in order to drive home the message that this introductory programming course is primarily a course in formal mathematics." This is similar to his current position. While Di-jkstra's proposal may be too extreme for some (and basic pro-

gramming classes probably have other purposes besides teaching formal mathematics [14]), he does make a reasonable point about the need of sound reasoning while developing software.

Although there is no universally accepted definition of computational thinking, it often encompasses the use of reasoning and concepts like abstraction that are seen to be valuable to computer scientists [15, 16]. Education in the computer disciplines should prioritise teaching students the mental processes required for jobs like programming rather than focussing solely on teaching them how to code, as Wing has argued. Logic classes, especially those focussing on mathematical topics like set theory and first-order logic[3], can thus be seen as teaching explicit topics relevant to computational thinking.

B. Mathematical Thinking

First, what is mathematical thinking? It's the process of reasoning about mathematical problems using logic, specifically mathematical logic. A good grasp of engineering mathematics necessitates familiarity with several of these areas, including set theory as it pertains to linear algebra [19] and deductive reasoning as it relates to proof theory [18]. Students must be taught to think mathematically since mathematics is fundamental to several computer science specializations, including machine learning, communication, statistics, and signal processing. To illustrate the point, "mathematical thinking is a vital skill needed in electrical engineering [20]" is the conclusion drawn from a 2010 article by Lauri Hietalahti that outlines many instances in which mathematical reasoning is applicable to the resolution of power system problems.

2) Logic and Proofs in Mathematics: Assigning proofs is a typical approach to teaching mathematical thinking and logic. According to mathematician Susanna Epp, students in computer science (and, by implication, engineering) are expected to "operate in a mathematically sophisticated environment...[and enhance their] logical reasoning [21]" because of the proof-writing assignments that are assigned to them. More so, "The activity of constructing and reasoning about programs in not all that different from the activity of con-structing and reasoning about proofs [22]." (Reeves, 2017).

Students will not profit from proof writing unless they have a firm grasp of the mathematics and logical concepts that underpin valid and invalid proofs. Sadly, "very few have an intuitive understanding of [logical] reasoning principles [21]." This was noted by Epp when instructing students in beginning proof writing for computer science and mathematics. According to her, the problems faced by her students were more severe than she had anticipated. The lack of effort put into proofreading their work nearly overwhelmed me. The majority of the time, all they could muster up was a scattershot set of computations and some poorly chosen words and phrases that didn't add anything to their arguments. The type of abstract mathematical reasoning that I was attempting to instill in my pupils was utterly foreign to them since their language and logical worldview was so different from mine. [21]

Epp reasoned that teaching students how to reason formally would provide them with the groundwork they needed to competently write and comprehend proofs. Once students grasp the deductive concepts required to do proofs and the relevance of proofs to their educational goals, teaching proof writing can be a potential strategy to further enhance logical thinking, according to the authors of this research.

3) Mathematical Thinking and Problem Solving: [23] One of the most important skills for aspiring engineers and designers is the ability to think mathematically and solve problems. Indeed, mathematician Paul Halmos asserted that "problems and solutions [24]" constitute mathematics' essence. Among the requirements for accreditation, both the Engineering Council and the Accreditation Board for Engineering and Technology (ABET) mention the ability to solve problems [25]. In addition, several engineering education coalitions were set up by the National Science Foundation (NSF) to deal with issues such fostering mathematical thinking, which includes open-ended problem solving, and creating a curriculum based on fundamental principles [26]. The purpose of computer science curricula should be to students to solve problems, and mathematical thinking and thus mathematical logic can help students solve problems, then such education should include topics from mathematical logic that can improve their thinking and help them understand the mathematics required for their major.

LOGIC IN THE COMPUTING FIELDS

A. Logic in the History of Computing

Many of the most influential figures in the field of com- puting were trained in and contributed to mathematical logic. For instance, Alonzo Church, an important figure in computer science, was a founding editor of the Journal of Symbolic Logic, publishing "A Bibliography of Symbolic Logic" in 1936 [27] and "An Introduction to Mathematical Logic" in 1956 [28]. His doctoral student, Alan Turing, developed ideas around algorithms and computation with what is now called Turing machines, getting his PhD in 1938 with a dissertation titled "Systems of Logic Based on Ordinals" [29]. Some believe Turing's 1936 work [30] was directly inspired by logician Kurt Godel's 1931 incompleteness theorems [31, 32]. Another of Church's doctoral students was Stephen Cole Kleene, who also studied mathematical logic. Kleene's thesis was titled "A Theory of Positive Integers in Formal Logic

[33]". Kleene's pioneered the branch of mathematical logic called recursion theory, and helped to form basis of what are now regular expressions. Much of Kleene's syntax, such as the star, is still used by computer scientists when pattern matching [34].

John Von Neumann, before his work on computer architec- ture [35] also did work in mathematical logic. In 1923, he published a paper on the construction of the natural numbers

[36] and his 1925 dissertation produced an axiomatization of set theory [37]. Fellow physicist and mathematician Freeman Dyson attributed Von Neuman's incredible success to his "unique gift as a mathematician was to transform problems in all areas of mathematics into problems of logic" [38].

B. Logic's Lasting Legacy

Many modern ideas in the computing fields are also inspired by ideas in logic. Edgar Codd based the relational model for database systems on set theory and propositional logic [39]. Ole-Johan Dahl, one of the fathers of object oriented program- ming (along with Kristen

Nygaard), taught formal methods for 30 years, believing "computer science students should know the principles of program reasoning, and that this would make them better programmers even without performing detailed verification [40]." Stephen Cook, who formalized the notions of polynomial time and NP-completeness [41] (independently discovered by Leonid Levin [42]) also published in proposi- tional calculus [43] and the logic of proofs [44].

One can see from their work that these individuals were able to make contributions to computing due, at least in part, to their understanding of logic. Furthermore, since logic was instrumental to the development of computing and is needed for understanding its historical and current concepts, the authors of this paper believe it should play a critical role in the computing curriculum.

I. EMPIRICAL SUPPORT FOR TEACHING LOGIC

Students' performance in computer-related professions, including programming, is correlated with their level of proficiency in logical and mathematical reasoning, according to the research. pupils' computational thinking abilities were predicted by their mathematics and logical thinking abilities, according to a research that examined Turkish high school pupils. Another study, performed on college freshmen, found a significant correlation between logical thinking skills and programming achievement, concluding that "Learners with strong logical thinking skills can program more efficiently than those without [46]."

Despite these associations, the idea that studying logic in depth may enhance one's ability to think logically was less apparent until quite recently. A research conducted in 1936 on teenage males indicated that pupils who received an hour of logic training weekly for three months had better exam results, indicating that teaching logic can enhance critical thinking. In contrast, "deductive reasoning is not likely to be improved by training on standard logic [48]." This conclusion was drawn from a 1986 research that examined students' performance on the renowned Wason selection problem [47]. The high degree to which success rates are task-specific has led some studies to cast doubt on the validity of the Wason selection task as a measure of reasoning [49]. If true, this suggests

[48] is premature, if not outright wrong, to conclude deductive reasoning cannot improved with instruction in logic.

Still, Cheng's findings are informative in that they suggest students may not initially see the relevance of learning logic to their desired educational goals without proper motivation. Ed- ucational research on task value suggests students' educational outcomes are influenced by how important and relevant they perceive course material [50]. In other words, the effectiveness of teaching logic may depend on the extent students believe learning logic will help their academic or professional career. This is discussed this more in Section VI.

More recent work has supported the hypothesis that logical reasoning can be improved. A 2016 study by Attridge et. al found a significant increase in the abstract reasoning skills of students who had some previous experience in logic [51]. A 2022 study found that teaching students concepts from propo- sitional logic, such as deductive arguments and Venn and Euler diagrams, improved their logical reasoning skills [52]. The authors conclude the explicit instruction of logic will improve students' performance on tasks in the computing fields such as programming, algorithmic reasoning, and hardware design.

II. THE IMPORTANCE OF LOGIC FOR INDUSTRY

A. Logic and Hardware Design

Understanding logic is not just essential for those wanting to become research scientists in academia - it is also important for students pursuing a career in industry. If the curriculum of computing fields is to mirror what is expected of recent graduates, particular attention must be paid to the role of elec- tronic design automation (EDA) which performs many tasks previously done by hardware engineers. Gone are the days when computer engineers explicitly place and route transistors on a PCB (printed circuit board). Instead hardware designers routinely use hardware description languages (HDLs), such asVerilog or VHDL to describe the desired behavior or logic of the device in question [53, 54]. HDL is sent to a logic synthesizer [55], where a netlist is generated. In some cases, the netlist is placed and routed onto a Field Programmable Gate Array (FPGA), or it is turned into masks which are etched for Application Specific Integrated Circuits (ASICs). This increased design abstraction means strong logical thinking skills are more important than they ever have been [20, 56, 57], since computers can arrange and route the hardware using optimization-based approaches once the behavior is specified. As a result, being able to formally and symbolically describe the desired logical behavior of a device is a necessary skill for those who want a career in hardware design and computer architecture.

B. Logic and Software Design

1) Logic and Programming: Strong mathematical logic skills are also important for those seeking employment design- ing software. This is because a computer, in the end, is a device built to follow a set of explicit instructions. Programming thus involves conveying what these instructions are in a precise manner. Since all errors in software can be traced back to human error and poor thought, it is important for programmers to be able to think rigorously and formally enough to avoid costly, and in some cases deadly, errors [13]. For a particularly harrowing incident of a fatal software bug, see the infamous Therac-25 [58]. Such errors can be avoided when programmers write code that is provably and verifiably correct.

Aswith hardware, there is increasing abstraction in software, in part due to the proliferation and adoption of high level and object oriented programming languages such as Python, Javascript, and C# [59]. These higher level languages remove many of the underlying hardware concepts such as pointers and registers, allowing more development time to be spent de- scribing the desired behavior of a program than implementing it. Furthermore, due to advances in compiler technology, less development time is needed to test different implementations of the same algorithm, meaning more time can be spent writing algorithms than writing code.

This is not to say discussions of the underlying hardware or software mechanisms should be removed or de-emphasized from the curriculum. Instead instructors should focus students' efforts and energy learning skills that transfer well to their future careers. The authors of this paper believe understanding logic is key for reasoning about algorithms and programming, and will clearly transfer to what students will be expected to do. In essence, logical thinking is much more future proof than specific programming languages, hardware, or packages which could be irrelevant in a few years.

2) Programming, Logic, and the Role of AI: Artificial Intelligence based programming tools, including large produced are correct [62]. While AI may can be used as a tool to help programmers with mundane or repetitive tasks, authors propose programmers with strong logic and deductive reasoning skills need not fear replacement by AI.

A. Additional Benefits of Learning Logic

There may be additional benefits to learning logic beyond improving technical skills. One such area is writing: while many engineering students avoid writing classes [63] (perhaps even choosing engineering due to a preference for numbers over words), one book suggests many professional engineers spend over 40% of their working time writing [64]. Teaching students to write by emphasizing valid arguments and avoiding fallacies offers a clear path for improving the structure and clarity of writing [65]. Further, because the same principles of valid argumentation are those that produce valid hardware and software, a parallel can be made between clean hardware, clean code, and clean writing. In fact, many of deficiencies found in engineering report writing, such as inappropriate narrative and poor organization [66], can be framed as logical fallacies such as non-sequiturs. Once engineers can identify these issues, they can work towards eliminating them, improv- ing the quality of their writing.

TEACHING LOGIC

A. Emphasizing the Role of Logic

Currently, topics from logic may be introduced in discrete math courses, often required for computer science majors, and digital design courses, usually required for electrical engineering and computer engineering majors. For instance, some discrete mathematics textbooks introduce propositional logic and set theory in the context of proofs [67], and many digital design textbooks introduce logical connectives and truth tables in the form of logic gates such as NANDS and XORs [55, 68].

One potential pitfall of this method of teaching logic is students may fail to see the connection between the underlying logical principles in in different courses and contexts. This creates a challenge to learning transfer [69] because these topics are only taught in the context of a single course or field of study. For instance the disjunction is fundamental the com- puting fields, but the different names, notations, and courses it is presented in may prevent students from recognizing they can leverage skills and experience they have already developed.

As a result, the authors of this paper suggest introduc- ing topics such as first order logic simultaneously with the different contexts it applies to when it is introduced. For example, while teaching about propositions and logical con- nectives, instructors can show how these connectives are used in proofs, circuit design, HDL, machine code, and high level programming languages. This allows students to see the wide applicability of logic while they are learning it, increasing task value [50]. This is important because a common reason students choose engineering or computer science is so they can design and build things. Introducing logic alongside different applications gives students practice seeing the relevance of theory to practical design work. The above suggestion is consistent with suggestions in the literature for mitigating the common complaint that engineering contains too much math and theory: [70], for instance, recommends adopting a problem-based learning approach and include practice along- side theory (not as a replacement for it).

B. What to Teach and When?

The authors of this paper suggest teaching logic by be- ginning with Boolean Algebra, including the Law of the Excluded Middle and the Law of Non-Contradiction [71]. A connection can made between Boolean Algebra and the behavior of semiconductor devices such as transistors. From here, logical connectives such as disjunctions and conjunctions can be introduced, teaching students to represent compound logical statements on paper, with circuit diagrams, and in pro- gramming languages. Here, students can be reminded that the voltage along a wire or a bit stored in memory can represent whether a given proposition is true, and can practice design work in hardware and software with these basic building blocks. From here, set theory can be introduced, allowing for quantification and completing first-order logic [3]. Further topics that can be included involve that of state and memory, allowing for sequential logic [55] and a discussion of finite state machines [67].LOGIC AND ATTRITION

A concern for the computing fields is attrition, the phe- nomenon of students "dropping out" of engineering and com- puter science to another field [72]. Attrition rates in electrical engineering and computer engineering are typically higher than all other engineering fields, with often more than half dropping out [73, 74]. Computer science is similar; one study found a 38% attrition rate in computer science, with this number varying by institution from about 30% to 60% [75, 76] These numbers are particularly pronounced for minorities in STEM, such as Black and Hispanic students [77, 78].

One survey found that the number one factor students cited for not completing an engineering degree was that they were "not performing academically [79]." Another survey found that the top reason students gave for not completing an engineering degree was coursework related, which included comments of classes being too theoretical, too hard, or having too much math [70]. With this in mind, it is perhaps unsurprising that a common explanation for the high attrition rates in computer science is "poor math skills and problem solving abilities [80]." Given that the computing fields tend to be more math and theory intensive than other engineering disciplines, this helps to explain their higher attrition rates.

Teaching mathematical logic can develop students' analyti- cal thinking skills, thus preparing them for the mathematical rigor of their future courses. Furthermore, the authors of this paper suggest teaching logic will particularly help first generation and minority students, who often enter college dis- proportionately unprepared. One study suggests that the lower average Math SAT scores of minority students [77] Engineering courses, and unpreparedness leads to attrition, one would expect to observe (as is observed empirically) higher attrition rates for minority students. This is compounded by the fact that many instructors may fail to distinguish between poor talent and poor preparation, prematurely discouraging minority students from continuing in the computing fields [72]. If electrical engineering and computer science departments are committed to improving retention, they should ensure all students have the prerequisite skills expected in their coursework. The authors of this paper argue that logic and mathematical logic will teach many of these required skills, and will be especially beneficial for students unaccustomed to the mathematical sophistication demanded by their major.

FUTURE WORK

It is remarkable that no studies have investigated whether students who take a mathematical logic course have better outcomes in the computer disciplines, given the significance of logic to these areas. Therefore, it is imperative that future research not only verify this broad concept, but also zero in on specific areas of mathematical logic that have been shown to boost performance in other classes. Classes in digital design, real analysis, linear algebra, and other proof-based mathematics, as well as algorithms and other discrete mathematics, have obvious applications to mathematical logic. Maybe deductive logic is more useful when developing compilers, while set theory or modal logic is more crucial for understanding probability. The fact that mathematical logic is typically reserved for philosophy departments makes it difficult to get statistics on this area as students may not see it on course listings or have it counted toward their degree requirements. Therefore, students can only enroll in this type of subject during their last year of high school, or whenever their schedule permits. Regrettably, this might prevent students from utilizing these talents in mathematics and theory classes taken by sophomores and juniors, when they would be most useful. In addition, individuals who are already well-prepared, such those whose high school provided advanced placement (AP) courses that counted toward college credit, may be the ones who enroll in and benefit most from a free elective logic course. Substantially widening the gap between well-prepared and underprepared pupils may result from a curriculum that does not clearly incorporate reasoning. Course in mathematical logic improves their outcomes in these fields. Future work should thus not only test this general hypothesis, but also focus on testing which topics from mathematical logic improves student success in different courses. Courses with clear relevance to mathematical logic are digital design courses, proof-based mathematics courses such as real analysis and linear algebra, and courses with material from discrete mathematics such as algorithms. Perhaps it is the case that set theory or modal logic is important for under- standing probability while deductive logic is more important for building compilers. One of the challenges to collecting data on this subject is that mathematical logic is often only offered by philosophy departments, and thus may not be listed or count towards a student's credits for graduation. As a result, students may only take such a class if and when their schedule allows, which may be their junior or senior years. Unfortunately, this means they may miss out on being able to use these skills when they could be the most helpful, such as in sophomore and junior-level mathematics and theory courses. Furthermore, relegating logic to a free elective means the students who are more likely to take such a course (or understand its relevance) may be those who are already better prepared, such as those who attended a high school that offered advanced placement (AP) courses to count for college credit. Failing to include logic into the curriculum explicitly may thus inadvertently worsen the gap between better and poorer prepared students.

CONCLUSION

The significance of logic in computer science and electrical engineering courses has been emphasized in this work. It is not a matter of whether logic should be taught, but rather how it should be taught, in order to increase problem solving and design abilities, according to the authors, who point to the history of computers and the skills that are anticipated of future engineers. Therefore, task value and students' perceptions of the relevance of learning logic to their future aspirations should receive special emphasis. Students in computer science and related subjects have it made easy to show how logic is relevant to their studies because of the obvious.

REFERENCES

[1] This is taken from The Organon by Aristotle. In 1853, Charles G. Bohn was written.

[2] From the spring of 2024 edition of The Stanford Encyclopaedia of Philosophy, edited by E. N. Zalta and U. Nodelman, comes "Classical Logic" by S. Shapiro and T. Kouri Kissel. This article was published by the Stanford University Metaphysics Research Lab. World Wide Web. You may see the source at: https://plato.stanford.edu/archives/spr2024/en tries/logic-classical/..

The Mathematical Logic Handbook, edited by J. Barwise [3]. Published by Elsevier in March of 1982.

Among A. D. Morgan's many works on logic is "On the Syllogism." In September 2019, Routledge published the textbook.

G. Boole's The Mathematical Approach to Logic, [5]. Publish your work independently with CreateSpace in 1847.

Journal of Philosophy article "Aristotle's Prior Analytics and Boole's Laws of Thought" by J. Corcoran, published in December 2003 by Taylor & Francis, delves into the connection between Aristotle's analytics and Boole's logic. This eprint may be found at this URL: https://doi.org/10.1080/01445340310001604707. World Wide Web. The publication may be accessed online at this URL: https://doi.org/10.1080/0144534031 0001604707.

G. Boole's 1854 publication, The Laws of Thought. Released in 1911 by the Open Court Publishing Business.

In the December 1938 edition of Electrical Engineering, volume 57, number 12, pages 713-723, C. E. Shannon presented a symbolic analysis of relay and switching circuits. This journal was an official component of the IEEE conference on electrical engineering. World Wide Web. Click here for additional information: https://ie eexplore.ieee.org/abstract/document/6431064?casa tok en=scCHdlbSI48AAAAA:0of06MtaJoP5LzSJixmOt4p5LWmAXFbUKrPDXcaWgSC0Rrsf6QZsJ64l-hnSGhRVqufJ678W.

[9] From Pascal to von Neumann: The Computer, edited by H. H. Goldstine. University Press of Princeton, 1993.

[10] "A history of the invention of the transistor and where it will lead us," James Brinkman, David Haggan, and William Troutman Volume 32, Issue 12, Pages 1858–1865, Published in December 1997 The publication in question is the IEEE Journal of Solid-State Circuits. World Wide Web. Link to the article: https://ieeexplore.ieee.org/abstract/document/643644?casa token=n5rDzECU4c8AAAAA:Iisq a4V BMztoyl5lUcY-oLkJcldY-bZ nme8NExBYHjBkoy6o6w VJ3MHc7XaGycK9flTBcQH

[11] W. Pietsch's Things to Think About When Developing an Inductive Theory for the Study of Data Science. Springer Nature, December 20, 2021.

This is cited in [12] the 2006 ACM Communications journal article "Computational thinking" by J. M. Wing (volume 49, number 3, pages 33–35).

"On the cruelty of really teaching computing science," E. W. Dijkstra penned in 1989 for the Communications of the ACM, volume 32, number 12, pages 1398-1404.

"A debate on teaching computing science" is an essay that spans pages 1397–1414 in the December 1989 issue of the Communications of the ACM, volume 32, number 12. World Wide Web. View the full article at: https://dl.acm.org/doi/10.1145/76380.76381.

[15] "Computational thinking: the developing definition," by C. Selby and J. Woollard, 2013, page count: 6. Southampton University Press published this work. World Wide Web. This information is retrieved from the following URL: https://eprints.soton.ac.uk/356481/ Computational Thinking, written by P. J. Denning and M. Tedre. [16]. Cambridge, MA: MIT Press, May 2019.

[17] Edited by S. Sentence, E. Barendsen, and C. Schulte, Computer Science Education: Perspectives on Classroom Instruction and Student Performances. Publish by Bloomsbury in March 2018.

In July 2019, the third edition of How to Prove It was published by Cambridge University Press, written by D. J. Velleman.

The 2008 edition of G. Strang's "Linear Algebra and Its Applications" has significant revisions and updates. Published in 2006 by Thomson, Brooks/Cole, this book has the ID q9CaAAAACAAJ.

In [20], "Is mathematical logic necessary in electrical engineering?" Mathematical Modelling in Civil Engineering, written by L. Hietalahti of the Technical University of Civil Engineering and published in 2010 in the Scientific Bulletin, covers pages 12-22.

This passage is taken from an essay called "The Role of Logic in Teaching Proof" that was published in December 2003 by Taylor & Francis in The American Mathematical Monthly. Go to https://doi.org/10.1080/00029890.2003.11920029. That's the title of the article. [On the internet]. In order to access this publication, one may use the following DOI: 10.1080/00029890.2003.11920029.

[22] is a Principles of Computer Science (CS), This was written in 1990 by S. Reeves.

K. Stacey posed the question, "What is mathematical thinking and why is it important?" in the 2006 lesson study that dealt with mathematical thinking. Included in the APEC project's status report was this collaborative study on new approaches to mathematics education that take into account students' cultural backgrounds (II).

[24]Publication of P. R. Halmos's "The Heart of Mathematics" in The American Mathematical Monthly's August 1980 edition was notable. World Wide Web. The article may be accessed online at: https://www.tandfonline.com/doi/abs/10.1080/00029890.1980.11995081.

[25] The Anwar and Richards (2018) compare EC and ABET certification criteria in the American Society of Civil Engineers' Journal of Professional Issues in Engineering Education and Practice. Page 06018001 is where you may find the article.

[26] "The future of electrical and computer engineering education" (vol. 46, no. 4, pp. 467-476), an article published in the IEEE Transactions on Education in November 2003 by Berry, DiPiazza, and Sauer. World Wide Web.

[27]A reprint version of Introduction to Mathematical Logic was published in October 1996 by Princeton University Press in Chichester, West Sussex.

There is an online version of Section 0 of A. Turing's "Systems of Logic Based on Ordinals (1938)" in The Essential Turing, edited by B. J. Copeland and released by Oxford University Press in September 2004, which addresses this matter. World Wide Web. Visit https://doi.org/10.1093/oso/9780198250791.003.0007 to access the publication.

[29]This article, "On Computable Numbers, with an Application to the Entscheidungsproblem," written by A. M. Turing and published in 1937 in the Proceedings of the London Mathematical Society (vol. s2-42, no. 1, pp. 230-265), may be seen online at this URL: https://onlinelibrary.wiley.com/doi/pdf/10.1112/plms/s4.230. Additional information may be found at https://onlinelibrary.wiley. com/doi/abs/10.1112/plms/s2-42.1.230.

[30]"Über formal unentscheidbare Sa¨tze der Principia Mathematica und verwandter Systeme I," in Monatshefte fu«r Mathematik und Physik, vol. 38, no. 1, pp. 173-198, December 1931, by K. Gödel. The web resource with this information is: https://doi.org/10.1007/BF01700692.

In their essay dated [31], B. J. Copeland and Z. Fan asked, "Did Turing Stand on Go¨del's Shoulders?" Issue 4, page 308–319, volume 44, The Mathematical Intelligencer, December 20, 2022. Download it and see it online at: https://doi.org/10.1007/s00283-022-10177-y.

[32]In 1935, the Johns Hopkins University Press released an essay titled "A Theory of Positive Integers in Formal Logic. Part I" which was published in the American Journal of Mathematics, volume 57, number 1, pages 153-173. The publication is available online at: https://www.jstor.org/stable/2372027.

[33]That book contains the essay "Representation of Events in Nerve Nets and Finite Automata" as well as the rest of the book. Pages 3-42, published by Princeton University Press in February 2016. This information was retrieved from the following URL: https://www.degruyter.com/document/doi/10.1515/9781400882618-002/pdf?licenseType=restricted.

"First draft of a report on the EDVAC," written by J. Von Neumann and published in the 1993 conference proceedings of the IEEE Annals of the History of Computing (vol. 15, no. 4, pp. 27-75), may be found on page 34. This information may be found online at: https://ieeexplore.ieee.org/abstract/document/ 238389?casa token=lYnkYwx3kBMAAAAA:5bushafZyuOTxkX-WE2 iHDa2s0DE8gpGDciz14JUR1Io9fWY -DTI2f-dgyshmD5vOt5D9FrX for example....

[35]Section scientiarum mathemati-carum, Acta Litterarum ac Scientiarum Regiae Universitatis Hungari-cae Francisco-Josephinae, volume 1, pages 199-208, 1923.