De Programmatica *Ipsum*

DE PROGRAMMATICA IPSUM

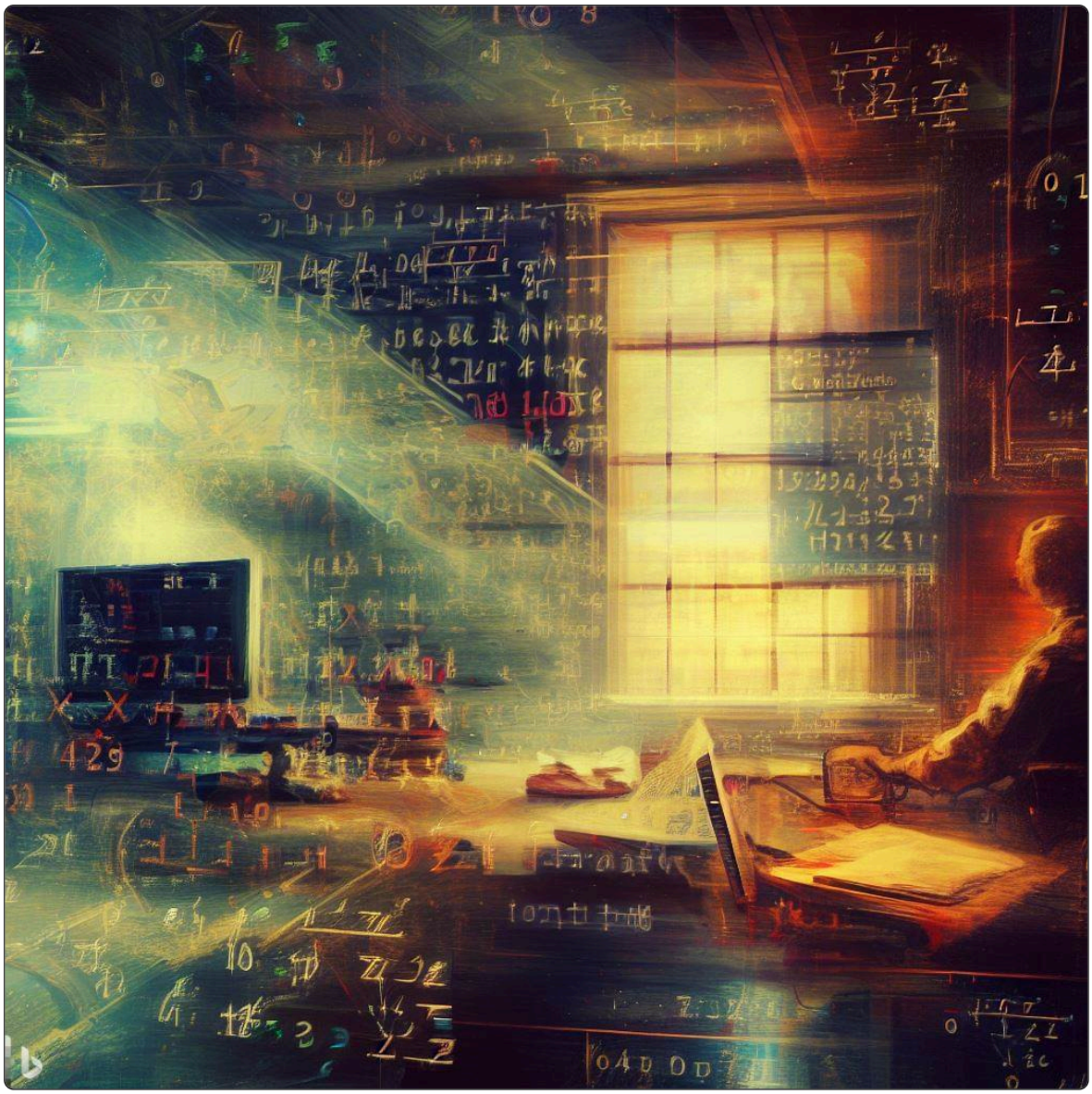
Issue 055: Mathematics

April 3rd, 2023

Table of Contents

Issue 055: Mathematics 5
In Praise Of Mathematics 9
Nemean 19
Guillermo Martínez & Gustavo Piñeiro 23

Issue 055: Mathematics



April 3rd, 2023

Welcome to the fifty-fifth issue of *De Programmatica Ipsum*, about *Mathematics*.

In this edition:

- We explore the intricate relationship between computers and mathematics[1].
- In the Library section[2], we review "Gödel ∀ (para todos)"[3] by Martínez and Piñeiro.
- In our Vidéothèque section[4], we learn how to calculate inverse square roots[5] in 1990's hardware.

For the first time in this magazine, not only we cover a non-English book in the Library section, but this month we have used images generated by the Bing Images Creator[6] to decorate our articles, instead of pictures from Unsplash[7].

The author dedicates this issue to the two high-school teachers who sparked his lifelong interest in math: Elisa Quastler in Argentina, and Gilbert Elia in Switzerland.

We opened an account on Mastodon last year: follow us at [8] to be notified of new releases!

We would also like to thank our patrons who generously contribute every month (or have contributed in the past) to our work and help us run this magazine. Thank you so much! In alphabetical order: Adam Guest, Adrian Tineo Cabello, Benjamin Sheldon, Christopher Nascone, Jean-Paul de Vooght, Patryk Matuszewski, Paul Hudson, Quico Moya, Roger Turner, and Szymon Licau.

Enjoy this issue! Please subscribe to our free newsletter[9] to stay updated about new releases, share the articles on social media, or contribute[10] if you would like to support our work.

Cover image by Bing Image Creator[11]; prompt: "The Intersection of Mathematical Equations and Computers in a Painting by Vermeer."

References

[1] https://deprogrammaticaipsum.com/in-praise-of-mathematics/
[2] https://deprogrammaticaipsum.com/category/library/
[3] https://deprogrammaticaipsum.com/guillermo-martinez-gustavo-pineiro/
[4] https://deprogrammaticaipsum.com/category/videotheque/
[5] https://deprogrammaticaipsum.com/nemean/
[6] https://www.bing.com/images/create/
[7] https://unsplash.com/
[8] https://mas.to/@deprogrammaticaipsum
[9] https://deprogrammaticaipsum.com/newsletter/
[10] https://deprogrammaticaipsum.com/contribute/
[11] https://www.bing.com/images/create/the-intersection-of-mathematical-equations-and-com/641c2535a05b4e43ab7182f4c5c2c512?FORM=GUH2CR

In Praise Of Mathematics

By Adrian Kosmaczewski

How much math knowledge does a programmer need to know to fulfill their job? How deep should the mathematical knowledge of full-stack engineers, computer security experts, or large language model engineers be? The spectrum of possible

answers to these questions is too large to be useful. Perhaps the eternal answer "it depends" is enough?

The original conundrum in the previous paragraph is related to the fact that the word "programmers" points, at the time of this writing, to quite an eclectic group of human beings. There are many different programmers in 2023, and spoiler alert, they do not all require the same levels of knowledge of mathematics. Fifty years ago, however, the word "programmers" used to refer to a much more coherent and uniform set of skills.

Precisely, let us travel back in time to 1973. By that time, the first volume of the defining work of our craft, "The Art of Computer Programming"[1] had been available in bookstores since 1968, and its first chapter literally consisted (consists) of a 100-something page long introduction to various mathematical concepts. Induction, logarithms, series, matrices, elementary number theory, permutations and factorials, Fibonacci numbers, are some of the subjects exposed in those beautifully typeset pages. Not only are the concepts introduced, but they are also accompanied by their respective exercises. For Donald Knuth, the answer to the opening question of this article is a resounding "yes;" you do need to know about Stirling's approximation[2] to understand what will come after. Even though we are still waiting for the final volumes of Dr. Knuth's work, we are certain that math will be a required skill to understand them.

Such a treatment was not new back then. The first volume of Richard Feynman's legendary "Lectures on Physics,"[3] dedicated to "mainly mechanics, radiation, and heat" is sparsely populated with chapters dedicated entirely to purely mathematical subjects: in particular probabilities, vectors, and algebra. The second volume, about "mainly electromagnetism and matter" has chapters about vector differentials, vector integral calculus, and tensors; fundamental concepts to, at least, understand the notation used to represent Maxwell's four equations[4] of electromagnetism. Maybe the inclusion of those chapters is one of the key reasons for the timeless success of Dr. Feynman's work.

Some more recent authors share this point of view. Christopher Bishop[5]'s groundbreaking 2006 book "Pattern Recognition and Machine Learning,"[6] arguably one of the triggers of the current popularity of machine learning, is quite literally

a book about applied mathematics, diving into probabilities, linear algebra, inference, neural networks, sampling, Markov models[7], and combinatorics. And quite rightfully so; if your objective is to find a job as an engineer at OpenAI[8], knowing a thing or two about eigenvalues and eigenvectors[9] is definitely going to be useful.

A different pattern appears in the outstanding "Structure and Interpretation of Computer Programs"[10] by Abelson and Sussman, where the use of a functional language like Scheme ties the book to a closer relationship with mathematics. This is easily seen in the structure of this work: there are no chapters dedicated to the subject of mathematics *per se,* but mathematical subjects are used all over the place as a mechanism to teach computation: functions and lambda calculus of course, differential calculus, polynomials, Monte Carlo simulations, numerical methods, and various algebras.

Let us look at the problem from the other side: what do mathematicians say about Computer Science? A lot, as it turns out; take for example the massive "Princeton Companion to Mathematics"[11] edited by Timothy Gowers, June Barrow-Green, and Imre Leader. On page 106 we can find a dissertation about algorithms by Jean-Luc Chabert; on page 575, another about computational complexity by Oded Goldreich and Avi Wigderson. Finally, an article about computational number theory by Carl Pomerance on page 348. Not to forget the insolubility of the halting problem on page 706, and the short biographies of Kurt Gödel on page 819 and Alan Turing on page 821.

Not all authors give the same importance to math in programming. Some pragmatic books about the subject, such as Steve McConnell[12]'s "Code Complete" or Tymann & Reynold's "Schaum's Outline of Principles of Computer Science"[13] have no reference nor explanation of math subjects in them whatsoever. The same is true of many graduate-level textbooks used daily in universities worldwide.

What is, then, the proper level of math required to lead a decent career as a programmer these days? Setting aside for a moment the upcoming maelstrom of AI-powered tools such as GitHub Copilot X[14], this author thinks that a high-school level of arithmetic, plane geometry (useful for writing games[15]), linear algebra (in particular vectors, and matrices), classic two-variable calculus (functions, duh!), set theory (useful for object-oriented programming[16]), and floating-point arithmetic[17]

(always useful), are the bare minimum things one needs to know when writing code for a living. Depending on their specialization, of course, programmers will have to master various other skills, ranging from finance, to numerical recipes[18], to number theory, to type theory, to even maybe quantum physics; nothing that smart people cannot learn, if not at university, through self-study.

Weaponizing Math

But there is a more profound issue at hand, however. Mathematics is today the subject used by universities to "filter the chaff" out of the thousands of applicants to various degrees ranging from humanities to science. This author remembers the dramatically reduced number of students in his second year of economics studies in Buenos Aires, after the killer subjects of Analysis 1 and Algebra 1 had performed their sacrifice rituals. Using Mathematics in this way is a rather unfortunate fact of our world, and in the opinion of this author, one of the root causes of the impossible dialogue[19] between management and engineering.

Math has been weaponized into a discrimination tool, and the resentment caused into hordes of unsuspecting victims is felt by technicians in many different fields, including programming. This is, simply put, a tragedy.

We have reached a tipping point as a species, where the teaching of mathematics must change dramatically to provide humans with the tools required to survive *and* protect our planet at the same time. We cannot afford to continue laughing at scientists just because we hated math back in high school. Such a behavior constitutes in itself an extinction-level mechanism.

But how can we teach math differently? According to Bill Gates[20], fond of throwing technology at problems, we should use artificial intelligence[21] for that.

> *In the United States, the best opportunity for reducing inequity is to improve education, particularly making sure that students succeed at math. The evidence shows that having basic math skills sets students up for success, no matter what career they choose. But achievement in math is going down across the country, especially for Black, Latino, and low-income students. AI can help turn that trend around.*

The thing is, right now, ChatGPT happily asserts[22] that `π == 4` with a straight face, so this whole "AI solution" does not sound like the best idea at this point in time, Mr. Gates.

Think Different

Let us evaluate two different options to solve the issue of teaching math: one based on philosophy, and the other based on history. Regular readers of this magazine should not be surprised in learning that this magazine is fond of throwing humanities and science at problems, instead of mere technology.

The first approach has been described in a short but influential volume published at Flammarion in 2015, called "Éloge des Mathématiques"[23] by French philosopher and mathematician Alain Badiou[24], whose title inspired this article. In this book, Badiou, also author of "L'Être et l'Événement," proposes a return to the roots, and to a pedagogy linking mathematics and philosophy more closely.

> *On a ainsi d'un côté une mathématique inventive et créatrice, confinée dans un monde intellectuel extrêmement dense et international mais fortement aristocratique, et de l'autre côté une sorte de diffusion scolaire et universitaire des mathématiques, dont à mon avis l'usage est de plus en plus obscur ou incertain. (...) Les mathématiques devraient absolument être considérées, non pas simplement comme une discipline scolaire chargée de sélectionner ceux qui vont être ingénieur ou ministre, mais comme quelque chose qui possède un intérêt extraordinaire en soi-même.*

For those not speaking French and too lazy to use Deepl[25] to translate the text above, the idea consists of bringing back a sense of wonder around mathematics. Maybe by having more mathemagicians[26] in classrooms. Maybe by taking a few storytelling cues from Kurzgesagt[27]. And yes, philosophy has a massive place in that equation (pun intended.)

Mathematicians and philosophers often came together, as shown by the examples of Russell[28], Poincaré[29], Dedekind[30], Descartes[31], Leibniz[32], Spinoza[33], Πυθαγόρας[34], Penrose[35], Gödel[36], Cantor[37], and countless others. Mathematicians can also be excellent writers of fiction or popularization works, like the following

South American figures: Júlio César de Mello e Souza[38] (also known as Malba Tahan[39]), Adrián Paenza[40], and Guillermo Martínez[41].

The second approach consists of a historical perspective on mathematics, one that this author has already described previously[42] in this magazine. Instead of teaching mathematics as a dull sequence of subjects, one after the other, why not tell the story of those who created it?

> *Regarding the latter, I still have the book[43] written by my Analysis teachers, Ernst Hairer[44] and Gerhard Wanner[45]. They taught us analysis following history, from the Renaissance to the 19th century. A fantastic idea, even though I had to take the exams three times until I passed. It was hard (oh yes) but very, very enjoyable.*

Beyond Math

Bertrand Russell, co-author of the *Principia Mathematica*[46] that would ultimately trigger[47] Gödel to formulate his incompleteness[48] theorems, has a lot to teach to future generations[49] beyond formulas:

> *In this world, which is getting more and more closely interconnected, we have to learn to tolerate each other. We have to learn to put up with the fact, that some people say things that we don't like. We can only live together in that way. And if we are to live together and not to die together, we must learn a kind of charity and a kind of tolerance, which is absolutely vital to the continuation of human life on this planet.*

Mathematics is the primary tool we use, not only to build software, but to build every single thing that constitutes our modern world. Raising it to a level where it coexists with history[50], art[51], or ethics[52], will undoubtedly horrify the insufferable pragmatists populating our capitalist world, continuously conflating usefulness or applicability with importance.

So be it. Education should be far more than preparing employees to be crunched[53] by a machine. It should raise our perspectives and make us dream of a better place

for all; not just for a small minority of vocal billionaires. Or at least, a better kind of mathematical education should avoid future generations the dreadful feeling of yet another mandatory, dull, dry, early Monday-morning math class.

Cover picture by Bing Image Creator[54]; prompt: "In Praise Of Mathematics Style Klimt"

References

[1] https://deprogrammaticaipsum.com/the-art-of-the-art-of-computer-programming/
[2] https://en.wikipedia.org/wiki/Stirling%27s_approximation
[3] https://www.feynmanlectures.caltech.edu/
[4] https://en.wikipedia.org/wiki/Maxwell%27s_equations
[5] https://en.wikipedia.org/wiki/Christopher_Bishop
[6] https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/
[7] https://en.wikipedia.org/wiki/Markov_model
[8] https://openai.com/
[9] https://en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors
[10] https://mitpress.mit.edu/9780262510875/structure-and-interpretation-of-computer-programs/
[11] https://press.princeton.edu/books/hardcover/9780691118802/the-princeton-companion-to-mathematics
[12] https://deprogrammaticaipsum.com/steve-mcconnell/
[13] https://dl.acm.org/doi/book/10.5555/1203622
[14] https://github.blog/2023-03-22-github-copilot-x-the-ai-powered-developer-experience/
[15] https://deprogrammaticaipsum.com/insert-coin/
[16] https://deprogrammaticaipsum.com/the-hype-cycle-of-oop/
[17] https://dl.acm.org/doi/10.1145/103162.103163
[18] http://numerical.recipes/
[19] https://deprogrammaticaipsum.com/the-impossible-dialogue/
[20] https://www.gatesnotes.com/The-Age-of-AI-Has-Begun
[21] https://deprogrammaticaipsum.com/open-letter-to-a-future-ai/
[22] https://mastodon.social/@Cdespinosa/110092792044177610
[23] https://editions.flammarion.com/eloge-des-mathematiques/9782081395930
[24] https://en.wikipedia.org/wiki/Alain_Badiou
[25] https://www.deepl.com/translator
[26] https://en.wikipedia.org/wiki/Mathemagician
[27] https://www.youtube.com/@kurzgesagt/videos
[28] https://en.wikipedia.org/wiki/Bertrand_Russell
[29] https://en.wikipedia.org/wiki/Henri_Poincar%C3%A9
[30] https://en.wikipedia.org/wiki/Richard_Dedekind
[31] https://en.wikipedia.org/wiki/Ren%C3%A9_Descartes
[32] https://en.wikipedia.org/wiki/Gottfried_Wilhelm_Leibniz
[33] https://en.wikipedia.org/wiki/Baruch_Spinoza
[34] https://en.wikipedia.org/wiki/Pythagoras
[35] https://en.wikipedia.org/wiki/Roger_Penrose
[36] https://en.wikipedia.org/wiki/Kurt_G%C3%B6del
[37] https://en.wikipedia.org/wiki/Georg_Cantor

[38] https://en.wikipedia.org/wiki/J%C3%BAlio_C%C3%A9sar_de_Mello_e_Souza
[39] https://en.wikipedia.org/wiki/Malba_Tahan
[40] https://en.wikipedia.org/wiki/Adri%C3%A1n_Paenza
[41] https://deprogrammaticaipsum.com/guillermo-martinez-gustavo-pineiro/
[42] https://deprogrammaticaipsum.com/teacher-leave-this-kid-alone/
[43] https://link.springer.com/book/10.1007/978-0-387-77036-9
[44] https://en.wikipedia.org/wiki/Ernst_Hairer
[45] https://en.wikipedia.org/wiki/Gerhard_Wanner
[46] https://en.wikipedia.org/wiki/Principia_Mathematica
[47] https://www.cambridge.org/core/journals/bulletin-of-symbolic-logic/article/russell-and-godel/0C347F171985A54395CDA89D31C4A59A
[48] https://plato.stanford.edu/entries/goedel-incompleteness/
[49] https://www.youtube.com/watch?v=ihaB8AFOhZo
[50] https://deprogrammaticaipsum.com/history-repeating/
[51] https://deprogrammaticaipsum.com/a-brief-history-of-programming-artists/
[52] https://deprogrammaticaipsum.com/primum-non-nocere/
[53] https://deprogrammaticaipsum.com/business-as-unusual/
[54] https://www.bing.com/images/create/in-praise-of-mathematics-style-klimt/641eb70ca0c44d899ad97f4d3b0ee6e4?FORM=GENCRE

Nemean

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y  = number;
    i  = * ( long * ) &y;                       // evil floating point bit hack
    i  = 0x5f3759df - ( i >> 1 );               // what the fuck?
    y  = * ( float * ) &i;
    y  = y * ( threehalfs - ( x2 * y * y ) );   // 1st iteration
//  y  = y * ( threehalfs - ( x2 * y * y ) );   // 2nd iteration, can be removed

    return y;
}
```

By Adrian Kosmaczewski

One of my favorite hobbies is called recreational mathematics[1]. This is the kind of revelation that I can only offer in the pages of this magazine and other select locations, feeling confident and hopeful that there is a more receptive public here than, say, at a Christmas dinner conversation or at the pub.

This is why I found fascinating the recently discovered aperiodic monotile[2] solving the so-called Einstein's problem[3] (and Rob Pike found it fascinating[4] too). I learned about this quite inefficient formula returning the nth prime[5]. I marveled at Tupper's self-referential formula that draws itself[6] when plotted on a graph. I discovered a curious integral[7] whose result is equal to the inverse of e. I delighted in the infinite curiosities around Fibonacci numbers[8]. I watched in awe Arthur Benjamin[9] correctly guessing dates and squaring five-digit numbers with fishes and cookies. And so on and so forth.

This is also the reason why I subscribed to various channels on YouTube with such content: Math with Janine[10], Veritasium[11], Michael Penn[12], Numberphile[13], Stand-up Maths[14], Jeffrey Caplan[15], Mathemaniac[16], Quanta Magazine[17], The Math Sorcerer[18], the greatly missed PBS Infinite Series[19], the outstanding Mathologer[20], the even more incredible 3Blue1Brown[21], and Nemean[22].

Nemean features only four videos at the time of this writing, but with excellent content and production. Among those videos, we find the subject of this month's Vidéothèque article, the explanation of the Fast Inverse Square Root[23] algorithm.

The original algorithm was discovered in the source code of Quake III Arena,[24] released to the public in August 2005[25]. (We've talked about id software and their coding prowess just two months ago[26].) It is used to normalize vectors used in 3D rendering, an operation repeated *ad nauseam* in real time as players navigate through the game. This operation is related to Pythagoras' theorem, but the code features an embedded "magic" `0x5f3759df` number in it. What is that constant for?

As a short summary of the movie, suffice to say that while additions and multiplications are fast to execute, square roots are not. Quite the opposite, actually, particularly in late 1990s hardware. The trick of this algorithm consists in working directly with floating-point number representations, bit shifts, and logarithms, to reach a "good enough" result; ideally with an error lower than 1%.

And lo and behold, this algorithm does the trick.

The video divides the explanations in three parts: "Evil bit hack," "What the fuck," and "Newton iteration." Understanding how this all works together requires some math and computer science background: the IEEE 754[27] standard, memory address access tricks in C (to read a float as if it were an int and vice versa), logarithms, and Newton's method[28].

A Wikipedia page[29] provides even more information about this algorithm, which, to be honest, was not invented by Carmack or anyone else at id software (spoiler alert: it existed since the 1980s.) Nevertheless, this video provides an excellent introduction to the subject, and more importantly, a peek at the complexities of working with real numbers in a real computer (pun intended) in a context such as games, where speed and efficiency are as crucial for success as visual design.

The real problem surfaces when such optimizations are used in non performance-sensitive contexts, like line-of-business applications. As one of the commenters below the video said,

> *Id's performance tuning was something that us programmers would nerd out about, and I'd often write my code to similar standards. Curious coworkers would sometimes encounter some of my "performance-tuned code" and be like "uhh, why did you do this like that?" I'd explain all about efficiencies and they'd just kind of roll their eyes and be like "okaaayy."*

Knowledge of these hacks is useful, but context is king. Decades later, the good old `sqrt()` function in `<math.h>` provides excellent performance in our multicore world; let us keep such hacks as historic curiosities, and let us make sure that our code is readable by all of our coworkers.

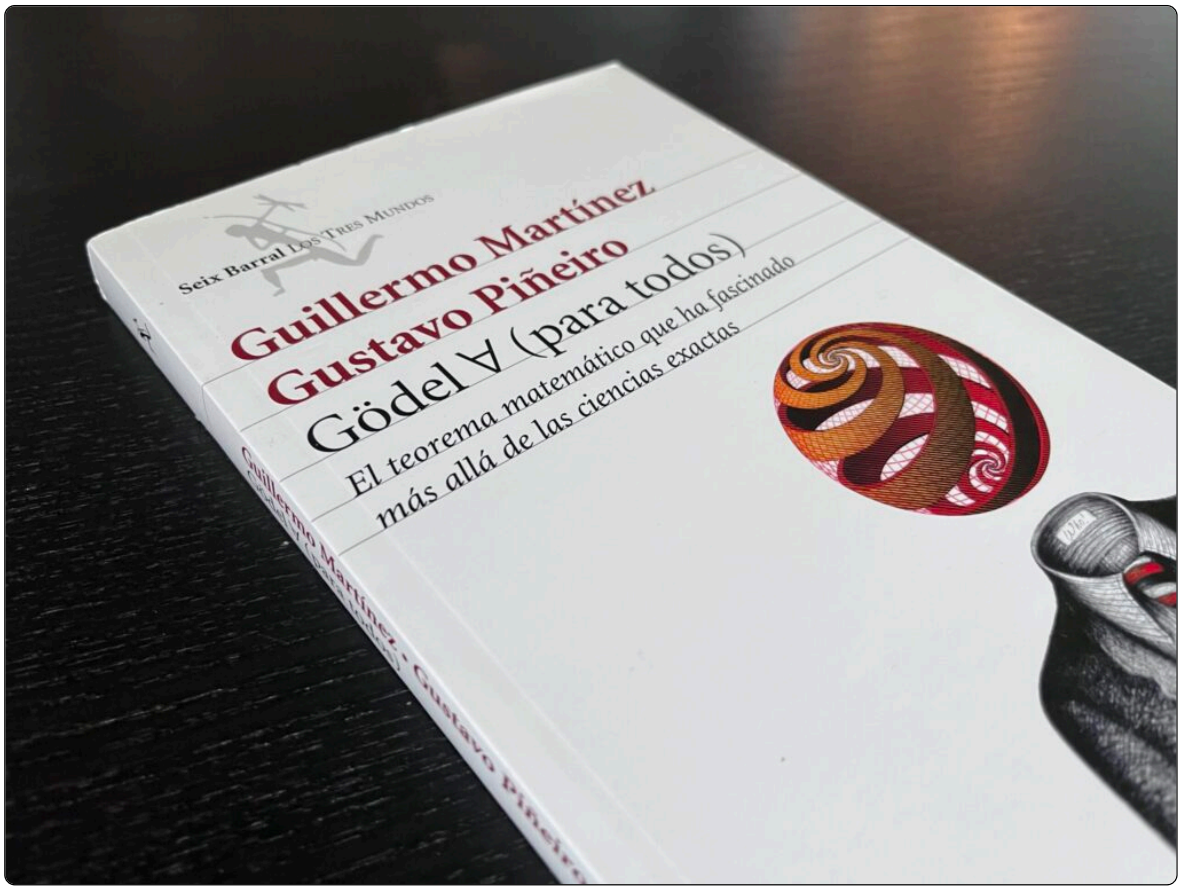
And please remember *not* to sprinkle your code with magic constants such as `0x5f3759df`. Please.

Cover snapshot by the author.

References

[1] https://en.wikipedia.org/wiki/Recreational_mathematics
[2] https://cs.uwaterloo.ca/~csk/hat/
[3] https://en.wikipedia.org/wiki/Einstein_problem
[4] https://hachyderm.io/@robpike/110122727704803762
[5] https://www.semanticscholar.org/paper/On-Formulae-for-the-Nth-Prime-Number-Willans/98172ccd9a2c1e3052d21eb4a3393e5663cc9f09
[6] https://en.wikipedia.org/wiki/Tupper%27s_self-referential_formula
[7] https://twitter.com/fermatslibrary/status/1392833267573870593
[8] https://akos.ma/blog/fibonacci/
[9] https://www.ted.com/talks/arthur_benjamin_a_performance_of_mathemagic
[10] https://www.youtube.com/@mathwithjanine
[11] https://www.youtube.com/@veritasium
[12] https://www.youtube.com/@MichaelPennMath
[13] https://www.youtube.com/c/numberphile/videos
[14] https://www.youtube.com/@standupmaths
[15] https://www.youtube.com/@jeffreykaplan1
[16] https://www.youtube.com/@mathemaniac
[17] https://www.youtube.com/@QuantaScienceChannel
[18] https://www.youtube.com/@TheMathSorcerer
[19] https://www.youtube.com/@pbsinfiniteseries
[20] https://www.youtube.com/@Mathologer
[21] https://www.youtube.com/@3blue1brown
[22] https://www.youtube.com/@Nemean
[23] https://www.youtube.com/watch?v=p8u_k2LIZyo
[24] https://en.wikipedia.org/wiki/Quake_III_Arena
[25] https://games.slashdot.org/story/05/08/20/1329236/quake-3-arena-source-gpled
[26] https://deprogrammaticaipsum.com/john-romero/
[27] https://en.wikipedia.org/wiki/IEEE_754
[28] https://en.wikipedia.org/wiki/Newton%27s_method
[29] https://en.wikipedia.org/wiki/Fast_inverse_square_root

Guillermo Martínez & Gustavo Piñeiro



By Adrian Kosmaczewski

So far, in this Library section, we have only covered books in English. We have already discussed[1] the hegemony of this language, and we think it is important to challenge it; so today we break the mould and introduce a book originally published in Spanish in 2009, "Gödel ∀ (para todos)"[2] by Argentine mathematicians Guillermo Martínez[3] and Gustavo Piñeiro, the former also a renowned fiction[4] author.

The name of Kurt Gödel often[5] appears in the pages of this magazine, and with reason. After all, the thought process that led to modern computing has Gödel as a major milestone and can be clumsily summarized as follows: Cantor ⇒ Hilbert ⇒ Russell ⇒ Gödel ⇒ Church ⇒ Turing ⇒ Von Neumann. This outstanding sequence of names happened in human history in a terrifyingly short span of time: roughly between 1880 and 1960. This means that in roughly 80 years, we went from Victorian-era steampunk[6] machines to the IBM 1401[7] and FORTRAN[8]. The speed of development of the computer (first as a concept and then as an industry) has no precedents in the history of Mankind.

It is hard to fathom the level of change experienced by society in such a short amount of time. We are used nowadays to change, or at least that is what John Kotter[9] would like us to believe. The thing is, confusion and charlatans are well-known byproducts in situations of great change happening in a short period of time. As a result, Gödel's incompleteness theorems[10] got snatched and abused by philosophers, psychoanalysts, fiction writers, gurus, and pretty much anyone who aimed for a seat in the pantheon of LSD-fueled New Age[11] pocket philosophy.

And this is where Martínez and Piñeiro's book steps in to provide some much-needed context. Theirs is not only an explanation of Gödel's theorem, its origins and structure down to its demonstration, or even its meaning; but also its impact in pop culture. This is where the book shines; highlighting all the occasions (and all the celebrities) that took the word "incompleteness" and turned it into something that it is not. The use of the mathematical sign "∀" in the title gives a clear idea of the target audience: this book is for everyone, or at least all those interested in understanding how the hype machine works in our psyche.

Let us enumerate some culprits guilty of extrapolating Gödel's ideas into riskier fields, with more or less acumen: the psychoanalyst Jacques Lacan[12], the philosopher and politician Régis Debray[13], the philosopher Gilles Deleuze[14], the Bulgarian-French semiotician Юлия Стоянова Кръстева[15], and the architect Paul Virilio[16]. To be honest, Gödel was not the only one undergoing such treatment; almost all big ideas of the twentieth century, including relativity and quantum physics, suffered from similar fates.

The book fulfills a double role; not only it constitutes a refreshing and correct description of Gödel's incompleteness theorems themselves, it also debunks claims and folklore built upon decades of misinterpretation. We need more books like this, in many other fields.

Many more books have been written about Gödel's incompleteness theorems; come to mind at least four. First, "Теорема Гёделя о неполноте" by Влади́мир Андре́евич Успе́нский[17], published by the venerable Mir Publishers[18] in 1982. Then, "Gödel's Proof"[19] by Ernest Nagel and James R. Newman, originally published in 1958 and reprinted in 2001 with an introduction by Douglas Hofstadter. Of course, "Gödel, Escher, Bach" from the same Hofstadter, reviewed by Graham in a previous article[20] in this magazine. Finally, Roger Penrose's 1989 book "The Emperor's New Mind"[21] also contains important discussions about Gödel's work (in particular a section in chapter 4) and rightfully belongs to this list.

But nothing beats going to the source. If you are drawn to this subject, you can read an English translation of the paper[22] Gödel originally published at the *Monatshefte für Mathematik und Physik*[23], or watch a recent video about it on Veritasium[24], to understand its substance a bit better. Meanwhile, let us hope that an English translation of Martínez and Piñeiro's book (and blog[25]) will be available soon for those non-Spanish-speaking readers of this magazine.

Cover photo by the author.

References

[1] https://deprogrammaticaipsum.com/the-winner-takes-it-all/
[2] https://www.goodreads.com/book/show/7788548-g-del
[3] https://en.wikipedia.org/wiki/Guillermo_Mart%C3%ADnez_(writer)
[4] https://en.wikipedia.org/wiki/The_Oxford_Murders_(novel)
[5] https://deprogrammaticaipsum.com/william-aspray/
[6] https://en.wikipedia.org/wiki/Steampunk
[7] https://en.wikipedia.org/wiki/IBM_1401
[8] https://en.wikipedia.org/wiki/Fortran
[9] https://www.kotterinc.com/methodology/8-steps/
[10] https://en.wikipedia.org/wiki/G%C3%B6del%27s_incompleteness_theorems
[11] https://en.wikipedia.org/wiki/New_Age
[12] https://en.wikipedia.org/wiki/Jacques_Lacan
[13] https://en.wikipedia.org/wiki/R%C3%A9gis_Debray
[14] https://en.wikipedia.org/wiki/Gilles_Deleuze
[15] https://en.wikipedia.org/wiki/Julia_Kristeva
[16] https://en.wikipedia.org/wiki/Paul_Virilio
[17] https://en.wikipedia.org/wiki/Vladimir_Andreyevich_Uspensky
[18] https://en.wikipedia.org/wiki/Mir_Publishers
[19] https://www.taylorfrancis.com/books/mono/10.4324/9780203715222/godel-proof-ernest-nagel-james-newman
[20] https://deprogrammaticaipsum.com/douglas-hofstadter/
[21] https://en.wikipedia.org/wiki/The_Emperor%27s_New_Mind
[22] https://homepages.uc.edu/~martinj/History_of_Logic/Godel/Godel%20%E2%80%93%20On%20Formally%20Undecidable%20Propositions%20of%20Principia%20Mathematica%201931.pdf
[23] https://en.wikipedia.org/wiki/Monatshefte_f%C3%BCr_Mathematik
[24] https://www.youtube.com/watch?v=HeQX2HjkcNo
[25] https://godelparatodos.blogspot.com/