I'm not a bot

You can’t perform that action at this time. How can financial brands set themselves apart through visual storytelling? Our experts explain how.Learn MoreThe Motorsport Images Collections captures events from 1895 to today’s most recent coverage.Discover The CollectionCurated, compelling, and worth your time. Explore our latest gallery of Editors’ Picks.Browse Editors' FavoritesHow can financial brands set themselves apart through visual storytelling? Our experts explain how.Learn MoreThe Motorsport Images Collections captures events from 1895 to today’s most recent coverage.Discover The CollectionCurated, compelling, and worth your time. Explore our latest gallery of Editors’ Picks.Browse Editors' FavoritesHow can financial brands set themselves apart through visual storytelling? Our experts explain how.Learn MoreThe Motorsport Images Collections captures events from 1895 to today’s most recent coverage.Discover The CollectionCurated, compelling, and worth your time. Explore our latest gallery of Editors’ Picks.Browse Editors' Favorites You can’t perform that action at this time. You can’t perform that action at this time. "The great idea of constructive type theory is that there is no distinction between programs and proofs… Theorems are types (specifications), and proofs are programs" (Robert Harper) This is a collection of modern resources on various undergrad level computer science topics, for someone with an interest in theory. Use WorldCat or LibGen if you can't buy these books. You don't have to do everything here, just the topics of interest to you. If you want a condensed version, watch the Great Theoretical Ideas in Computer Science lecture series to see what interests you. On Isaac Newton's iteration method to self-learn geometry: "He bought Descartes' Geometry and read it by himself .. when he was got over 2 or 3 pages he could understand no farther, than he began again and got 3 or 4 pages father till he came to another difficult place, than he began again and advanced farther and continued so doing till he made himself master of the whole without having the least light or instruction from anybody" (King's Cam.,Keynes MS 130.10,fol. 2/v/) Numerous anecdotes exist on studying strategies, like the Feynman method explained here "If you can’t, out loud or on paper, explain the idea without confusion or contradiction, stop and figure it out right there". There's some books that model that method, like Gilbert Strang's Calculus has you reciting back the entire chapter you just read. If you ignore the copious amounts of marketing on his site, Cal Newport has some other interesting anecdotes on studying, such as how he was able to get the best grade in his Discrete Mathematics class, and the rest of the site is full of advice on studying, how to schedule yourself and deliberate practice. My personal advice is to first always get the errata for what you're reading, even course notes sometimes have errata on the author's page, and always take something a little harder than your skill level so then it becomes a research exercise backfilling all the requirements. For example many people want to relearn math they forgot, so they start working through some enormous 1000+ page pre-calculus book and lose interest after the first few chapters. Instead make the goal to learn calculus, and start there. The same goes for learning algorithms, make the goal you want to solve problems in competitive programming or you want to build something impossible and need to learn how to make it possible, now it's a research project to learning fundamental algorithm design techniques that will keep you interested. If you want to learn this material with a community, some anons have started a Discord channel. MIT's the missing semester of your CS education covers a lot of things you'll eventually need to know. Editors such as emacs/vim are also introduced here. The 1980s book The Unix Programming Environment by Kernighan & Pike walks through Awk/Grep/Sed and shell scripting too. I've rewritten this entire section into it's own site, if you have little experience want to try to learn PAPL Programming and Programming Languages from scratch and a few other courses, try the AI workshop I work through Brown's CS019 class from scratch and Tao's Analysis I, plus many other topics. They're poor quality notes but the idea is if you get stuck on something, you can look at them. "So I went to Case, and the Dean of Case says to us, it's a all men's school, "Men, look at, look to the person on your left, and the person on your right. One of you isn't going to be here next year; one of you is going to fail." So I get to Case, and again I'm studying all the time, working really hard on my classes, and so for that I had to be kind of a machine. In high school, our math program wasn't much, and I had never heard of calculus until I got to college. But the calculus book that we had was (in college) was great, and in the back of the book there were supplementary problems that weren't assigned by the teacher. So this was a famous calculus text by a man named George Thomas (second edition), and I mention it especially because it was one of the first books published by Addison-Wesley, and I loved this calculus book so much that later I chose Addison-Wesley to be the publisher of my own book. Our teacher would assign, say, the even numbered problems, or something like that (from the book). I would also do the odd numbered problems. In the back of Thomas's book he had supplementary problems, the teacher didn't assign the supplementary problems; I worked the supplementary problems. I was scared I wouldn't learn calculus, so I worked hard on it, and it turned out that of course it took me longer to solve all these problems than the kids who were only working on what was assigned, at first. But after a year, I could do all of those problems in the same time as my classmates were doing the assigned problems, and after that I could just coast in mathematics, because I'd learned how to solve problems" (Don Knuth) Try the new Math Foundations from Scratch experimental curriculum we'll try and teach ourselves the math needed to try most of the courses here. The reasons I've rewritten this section a dozen+ times is because I've yet to find anything that satisfies the condition of "is this a self learner?" so I'm just going to wing it and write my own. Even if it fails you'll still be able to confidently do any of the discrete math resources here. You only learn math by doing it, so in the beginning it's nice to have few solutions but eventually you should figure it out yourself and you will be better off for it so my (terrible) proof hint walkthrough's will decline as we go but the thing is, you won't need them anymore. This Cornell OCaml course is totally self contained with it's own textbook. You will want to archive the entire course locally using wget. The release code for the assignments wget them here and change URL to /a4/a4.zip, /a3/a3.zip, they range a0.zip - a5.zip (Full Course) CS 3110 Data Structures and Functional Programming The free textbook is essentially the lectures, the notes have recommended chapters for additional books like Real World Ocaml and have exercises. Introduction to Coq, you can extract OCaml from Coq, F* or from Why3 Every spring/fall semester their gitbook is updated, search for the latest version See this talk Effective ML how to properly write interfaces, error handling It is possible in a functional language like ML to do algebra with types, proving two types are isomorphic with the desired properties of reflexivity, symmetry, and transitivity. It's also possible to abstract Lists and Trees into polynomials, as every polynomial looks like a sum of terms. As you will learn in 15-150 Principles of Functional Programming "most functional datastructures have constant time access near the outer layer of their structure, ie: the head of a list or the root of a tree. However, access at some random point inside the structure is typically linear since looking at some element of a list is linear in the length of the list, and looking at some element of a tree is linear in the depth of the tree. Datatype derivatives allow constant time access to the entire structure." Wildberger has a series of intuitive lectures to understand this topic geometrically in the affine space that I highly recommend as no artificial constraints are imposed like an origin (0,0) and some of this is surprisingly advanced you won't find in regular linear algebra courses. (Full Course) CS053 - Coding the Matrix Covers interesting applications like what was shown in Great Theoretical Ideas in Computer Science when you learned how a parity bit works. No formal prerequisites except assumes you know how to do basic proofs. Uses Python. A lot of linear algebra over the complex field This free book Linear Algebra - As an Introduction to Abstract Mathematics from UC Davis has plenty of exercises Terence Tao has a set of notes if you google, they go with the book Linear Algebra by Friedberg, Insel and Spence. Some videos in this playlist introduce Abstract Algebra, walking through it's foundations distinguishing between descriptions, definitions and specifications of abstract algebraic objects. I used to have a course here but the lectures deleted, and reuploads I did were also deleted (you should be archiving everything as you take it), so my recommendation is get Artin's blue book and look on youtube for Harvard's Abstract Algebra course from a few years ago it's still a great course. Try going to the UCSD podcast website and searching for Discrete Math courses many are recorded with slides and screen like this course. Good advice on writing mathematics, introduces LaTeX. Cornell has a similar class with complete lectures notes though if your math background is very weak I recommend working through Chapter 2 and 3 of Terrence Tao's Analysis I to learn to write your own proofs constructing the natural numbers before trying this book. . This book covers most of the topics in courses 15-251 and 15-150, and teaches you to write proofs as programs. If you don't know how to program this book will teach you by modeling math functions in SML, polynomials into lists, sets into types, creating relations and proving their results with programs. This is a crash course in multiple topics such as Probability, Linear Algebra, Modular Arithmetic, Polynomials, Cryptography and Complexity Theory. 15-251 assumes students have completed An Infinite Descent into Pure Mathematics though it is self contained. The best part about this course is old exams with solutions, and extensive notes. "[Computer science] is not really about computers - and it's not about computers in the same sense that physics is not really about particle accelerators, and biology is not about microscopes and Petri dishes...and geometry isn't really about using surveying instruments. Now the reason that we think computer science is about computers is pretty much the same reason that the Egyptians thought geometry was about surveying instruments: when some field is just getting started and you don't really understand it very well, it's very easy to confuse the essence of what you're doing with the tools that you use." (Hal Abelson) This covers architecture from a programmer's perspective, such as how to write cache friendly code/optimizations, assembly, how compilers work, return oriented programming (ROP) to bypass stack protections, the memory hierarchy, and networks. You could read K&R's The C Programming Language for a brief intro, though this course will explain C as you go anyway and fully covers pointers at the assembly language level making it self contained. MIT teaches OS engineering using a rewrite of the sixth edition Unix(v6) similar to the classic Lions' Commentary but in ANSI-C called xv6. The idea is you read the source as you read booklet, to understand the system. There is also a schedule of lecture notes. The latest xv6 source and text are available via: git clone git://github.com/mit-pdos/xv6-public.git git clone git://github.com/mit-pdos/xv6-book.git MIT's 6.004 course covers RISC-V: The free and open RISC Instruction Set Architecture. Take this if you are interested in programming FPGAs or cheap embedded modules. MIT has ported their xv6 book/course to RISC-V. There is also a schedule of lecture notes. THe latest xv6-riscv resources and book: git clone git://github.com/mit-pdos/xv6-riscv.git git clone git://github.com/mit-pdos/xv6-riscv-book.git If you have problems building the book, get the pdf How the computing, economic and sociological worlds are connected. This series of recorded lectures by Cornell professor Nate Foster on functional Network Programming, will cover the modern theory and practice and introduce you to an entire new world of thinking about networks by developing (or using) customized domain specific languages. No background outside of familarity with OCaml/SML-like syntax is needed though the 15-213 course contains a basic introduction to computer networks if you want. I didn't include any standard network class you can easily find books on this. Interesting post on the future of database systems by Andy Pavlo. CMU abandoned Peloton and now is building a new disk-oriented storage prototype for education called BusTub DB. At the end of the semester the grading scripts are publically available. Here are the (poor) notes I took for MIT's 6.148 Web Programming Competition and MIT's 6.170 Software Studio. I did most of these assignments on a phone using Eruda since I wrote this while on lunch breaks. The MIT workshop and 6.170 are actually pretty good, giving you just enough to make a MVP yourself, expecting you to fill in the rest of the details on your own now that you know the basics of express, react, CSS etc. This is probably the best course here that enables you to immediately start freelancing after so you can get paid and then continue to learn more. (Full Course) 15-388 Practical Data Science Recorded lectures are here and archived here Introduction to the 'full stack' of data science analysis: data collection and processing, data visualization and presentation, statistical model building using machine learning, and big data techniques for scaling these methods. Cornell also has a free text with interactive jupyter notebooks (in Python) Jupyter notebook is a literate programming paradigm tool, you can freely write using LaTeX or markdown and run the code cells in any order to not interrupt the flow of documentation. This course teaches the Python libraries, but you can do the assignments in Julia if you want, or experiment in OCaml A draft book Foundations of Data Science The idea for including this came from this post about creating something everyday. (Lecture Notes) 15-322 Introduction to Computer Music Mainly just slides but the books they recommend are extremely thorough if you want to learn manipulating sound with algorithms Nyquist is a Lisp Learn how Autechre made songs like Krib They even perform a 'laptop orchestra' every year (the animations on the screen in the background are from 15-104 Princeton also has a laptop ensemble To learn what a network ensemble is, see the Oxford Handbook to Computer Music on libgen MIT also has several books and a new laptop ensemble course starting fall 2019 There's some more notes around but largely these kinds of courses are fully closed behind credentials/paywalls. A classic introductory computer science book on thinking about the big picture of programs with abstraction: finding general patterns from specific problems and building programs based on these patterns. An applied example of this is the package manager GNU Guix and distro GuixSD, which is a GNU implementation of the NixOS functional software deployment model. Package builds, including entire system builds, are declared in one text file. The resulting software deployment is functional: build inputs go in such as compilers, customizations, environments ect, and a reproducible, immutable build comes out with a hashed identifier meaning you can do roll backs to previous successful builds. A recent GUixSD feature gexp (g-expressions) is a good example what can be achieved through syntactic extensions of the Scheme language. You are more likely to benefit from this book after having some programming experience, but no matter what level of programmer you are you'll still benefit from SICP. How to verify software, and strategies of programming that minimize catastrophe during failure. The Little Prover is a good introduction in determining facts about computer programs. Prof Rob Simmons can be hired as a tutor to teach you Coq, the curriculum follows the Software Foundations book series below. (Book) Verified Functional Algorithms Some recorded lectures by Andrew Appel here Part 3 of the Deep Specifications interactive book series by Andrew Appel, learn by doing Assumes you have read these chapters of Software Foundations Part I: Preface, Basics, Induction, Lists, Poly, Tactics, Logic, IndProp, Maps, (ProofObjects), (IndPrinciples) A good introduction to Dependent Types by Dan Licata is here or The Little Typer book. (Full Course) 15-424 Foundations of Cyber-Physical Systems Course (with recorded lectures) if you're interested in programming drones/space shuttles/robots/cars and other things that cannot fail from avoidable errors. Self contained, will teach you differential eq but assumes you already have some calculus background. Unlike a traditional introduction to algorithms and data structures, this puts an emphasis on thinking about how algorithms can do multiple things at once instead of one at a time and ways to design to exploit these properties. It uses a language based analysis model to estimate complexity for abstractions such as garbage collection. (Book) Parallel and Sequential Algorithms (LibGen has the most recent version) Complete, self-contained book with exercises used for 15-210 Old exams, homework and labs w/solutions are here and done in Standard ML If you enjoyed this try 15-853 Algorithms in the Real World which advances 15-210 material Notably this book on computational geometry, is surprisingly practical "A stroll through intermediate data structures and their associated algorithms, from the point of view of functional programming." Note how pattern matching makes implementing these seemingly complicated structures a much easier task. The typical text for a senior undergrad or graduate course on algorithm core theory and analysis is The Design and Analysis of Algorithms by Kozen (1992). This is an excellent book to study with difficult problems well presented and clearly analyzed by Kozen. If you want a book to practice design, Algorithm Design by Jon Kleinberg and Éva Tardos contains 200+ problems and how to design solutions for them. (Full Course) CS224 Advanced Algorithms Recorded lectures by Prof. Jelani Nelson walking through the design and analysis of online algorithms, linear programming, approximation algorithms, hashing, randomized algorithms. This is a graduate introduction to quantum computation/information theory, from the perspective of theoretical computer science. Topcoder has quantum algorithm bounty's. (Full Course) 15-859BB: Quantum Computation Recorded lectures on YouTube. The prereqs are an undergrad background in complexity theory (15-251 or 15-455), linear algebra, and discrete probability. "90% of the understanding of the quantum circuit model is achieved by reviewing three purely 'classical' topics: classical Boolean circuits; reversible classical circuits; and randomized computation" A series of curated papers on Quantum Computing If this topic interests you, Gerald Jay Sussman & Jack Wisdom's book teaches differential geometry using Scheme as a foundation for quantum field theory or general relativity. NJ Wildberger additionally has a set of introductory lectures. If you've taken 15-213 Computer Systems or 6.004 RISC-V Architecture, and understand graph algorithms you will understand this course (with a lot of extra work), as it covers performance analysis, instruction-level optimizations, caching optimizations, and other techniques for high performance, scalable systems. The course is done in C but the concepts apply to any language. Notice there's a button for "Export Materials ZIP". 18.642 also has an older course on Open CourseWare with lecture videos. You may also find interesting this Quantitative trading summary and this old post on pricing & hedging models. This service matches your skills to people who want to pay you. Jane Street Capital is a finance tech company that hires functional programmers worldwide, you may want to apply there. There are numerous opportunities to apprentice as a security researcher. There are also bounties for writing features available on topcoder, gitcoin or bountysource or (unfortunately 'regionally paid') outfits such as toptal, which tend to use codility tests for filtering candidates. Do your own research on the current state of these outfits, through glassdoor or other employee review sites. A large collection of remote job listing and consulting platforms is here. If you enjoy research look up research programming positions at a university near you. You get hired to a grant funded team of grad students and post-doc researchers. Sometimes they are just hiring for people to build a proper user interface for the doctors accessing a dataset, or you could be doing data visualizations, or statistical computing, or even engineering experimental software. Depending on the work you are doing, they may even write you a letter of recommendation meaning you can use this to apply to a grad school or hustle another programming research position. . Read these slides from A Theorist's ToolKit on how to find research, how to write math in LaTeX, how to give a talk, where to ask on stackexchange etc. The gist of this paper is explained here, CHiTT is a constructivist take on math foundations building an expressive enough model so they can build the next generation of proof assistants but as you can see from the gist it will definitely have other applications. Start with this talk A Functional Programmer's Guide to Homotopy Type Theory with intro to Cubical Type Theory, aka (Cartesian) Cubical Computational Higher Type Theory. This is today regarded as a 'misadventure in formalism' by R.Harper, due to HoTT being insufficiently expressive for programming, but everything you learn in this seminar will be necessary to understand computational higher type theory papers like the magic of paths extending other programs. (Full Course) Homotopy Type Theory Self-contained grad seminar with recorded lectures The self-contained HoTT book contains all the background material needed Thierry Coquand also has seminars in constructive algebra, and univalence Follow the research on Robert Harper's CMU page or see CMU's Principles of Programming Group for more graduate course offerings Watch the recorded lectures at the yearly Oregon Programming Languages Summer School, search through Google Scholar for computational/cubical higher type theory research. I created a seperate AI workshop where eventually we will work through CS4780 or CMU's 10-601. There used to be a course here but it was credentialed and recorded lectures deleted. You can still watch Berkeley's CS188 on Youtube, with Norvig's new 4th edition book out in 2020, and make it more rigorous by taking Sussman's grad course reviewing older papers with unsolved problems or deep ideas, like his own Art of the Propagator paper and implementation. There is also AI readings to explain intelligence from a computational point of view both these are nice compliments to a typical AI course covering interesting ideas. A series of recorded lectures and recommended texts in the Math background for ML introductory courses, though you can also try just taking 36-705 and the book All of Statistics which will teach you probability. There is also a crash course in linear algebra here and a set of notes. You may also enjoy the Vector Boot Camp from Brown. Multivariable calculus Strang has a free book on MIT OCW Calculus with a chapter on vector calculus. Strang now has a big picture of Linear Algebra lectures. (Full Course) 36-705 Intermediate Statistics The author of All of Statistics Larry Wasserman covers Chatpers 1 - 12 from his book and more covering the fundamentals of theoretical statistics in these recorded lectures The lectures assume you already read Chapter 1 - 2 There are many probability lectures around to go with the beginning chapters such as here The quality of these can be changed to 1080p but the audio is terrible, however you can extract the audio easily, and use Audacity (free) to suppress some of the background noise and boost his voice. (Full Course) 10-601 Masters Introduction to ML Recorded lectures and recitations Self contained, assumes you are grad level standing so have familiarity with basic probability and algebra Doesn't cover as much theory as 10-701/715 but that's what Wasserman's 10-702 Advanced Statistical Theory is for if you're interested "This course will teach you basic skills to decide which learning algorithm to use for what problem, code up your own learning algorithm and evaluate and debug it." Anything that doesn't make sense look it up in the Mathematics for Machine Learning free book There is also Cornell's advanced ML class (Full Course) 10-715 Advanced Introduction to ML Most recent version here uses mostly the same recorded lectures. Fast paced curriculum intended to prepare PhD students in the ML grad program to write research papers Some slides on the practical techniques needed for working with large datasets Cornell's CS6780 is more accessible as a more advanced course in ML Same professor of 10-715 now teaches 10-708 for problems with a very large number of attributes and huge datasets (Full Course) 10-701 Introduction to ML Intended for PhD students outside the ML program, more theory than 10-601 Assumes you have this 24.2 math background Some recorded recitations are here Some slides on the practical techniques needed for working with large datasets If you're interested in parallel GPU programming for training see these lectures and notes. (Full Course) 11-785 Introduction to Deep Learning Recorded lectures here to learn to build and tune deep learning models If the playlist is deleted, which is frequently, (use youtube-dl to archive) search YouTube for "CMU 11-785" There are also applied courses and practical challenges/competitions on kaggle.com Some slides on techniques for making deep learning robust to adversarial manipulation Read these slides from A Theorist's ToolKit on how to find research, how to write math in LaTeX, how to give a talk, where to ask on stackexchange etc. Draft book A Graduate Course in Applied Cryptography - Dan Boneh and Victor Shoup Lecture notes from 18-733 Applied Cryptography Read all Daniel J. Bernstein's (and Peter Gutmann's) posts on the IETF Crypto Forum Research Group [Cfrg] archive, it's a master class in modern cryptanalysis and he rips apart bad standards/protocol designs. Read The Art of Computer Programming - Seminumerical Algorithms by Knuth (Vol 2) chapter on Random Numbers. These tests are still used in MIT grad courses. Try them on every library you can find that supposedly generates pseudorandom numbers Read about the proof of the Wireguard protocol, a VPN that uses AEAD_CHACHA20_POLY1305 Follow whatever the PhD students of djb and Tanja Lange are working on Watch the lectures from the 2017 Post-Quantum Crypto Summer School Read the journal of Crypto Engineering (use SciHub proxy) Read a book on Random Graphs there is a connection between Graph Theory and Cryptography Try the Cryptopals challenges Read some cryptocurrency papers, such as Stellar's Consensus algorithm (soon to be used by mobilecoin.com), Fairlayer or the protocol specification for Zcash. You can’t perform that action at this time. Page 2 You can’t perform that action at this time. Freely available programming books View the Project on GitHub EbookFoundation/free-programming-books This list, organized by subject, is for books that cover a programming-related subject in a programming-language agnostic way. Books that cover a specific programming language can be found in the BY PROGRAMMING LANGUAGE list. Index Atariarchives.org - Atariarchives.org makes books, information, and software for Atari and other classic computers available on the Web. Bento Bitsavers.org Code Catalog: A Collection of Code Examples from Prominent Open Source Projects conceptf1.blogspot.com Free Tech Books Goalkicker - Programming Notes for Professionals books IBM Redbooks InfoQ Minibooks InTech: Computer and Information Science JSBooks - directory of free javascript ebooks Learn X in Y minutes Learneroo Resources to Learn Programming Microsoft Guides to Software Microsoft Press: Free E-Books Microsoft Technologies 1, including books on Windows Azure, SharePoint, Visual Studio Guide, Windows phone development, ASP.net, Office365, etc. collection by Eric Ligman Microsoft Technologies 2, including books on Windows Azure, SharePoint, Visual Studio Guide, Windows phone development, ASP.net, etc. collection by Eric Ligman Microsoft Technologies 3, DevOps for ASP.NET Core Developers - Cam Soper, Scott Addie, Colin Dembovsky (PDF) O’Reilly’s Open Books Project Papers we love Red Gate Books Rip Tutorials Stef’s Free Online Smalltalk Books TechBeamers.com TechBooksForFree.com The Definitive C++ Book Guide and List Theassayer.org Tutorials Point Visualgo: Algorithm and Data Structure Visualization - Visualise data structures and algorithms through animation Algorithms & Data Structures A Field Guide To Genetic Programming - Riccardo Poli, William B. Langdon, Nicholas F. McPhee (PDF) (archived) (CC BY-NC-ND) Algorithm Design - Jon Kleinberg, Éva Tardos Algorithmic Graph Theory - David Joyner, Minh Van Nguyen, David Phillips (PDF) (GFDL) Algorithmic Thinking - Donglai Fu Algorithms - Wikibooks Algorithms - Jeff Erickson (PDF) Algorithms, 4th Edition - Robert Sedgewick, Kevin Wayne Algorithms and Automatic Computing Machines (1963) - B. A. Trakhtenbrot Algorithms and Complexity - Herbert S. Wilf (PDF) Algorithms and Data Structures - With Applications to Graphics and Geometry - Jurg Nievergelt, Klaus Hinrichs (PDF) Algorithms Course Materials - Jeff Erickson Algorithms Notes for Professionals - Compiled from StackOverflow Documentation (PDF) Annotated Algorithms in Python: Applications in Physics, Biology, and Finance - Massimo Di Pierro Binary Trees - Nick Parlante (PDF) Data Structures - Aditya CSE (PDF) Data Structures - Wikibooks Data Structures (Into Java) - Paul N. Hilfinger (PDF) Data Structures and Algorithm Analysis in C++ - Clifford A. Shaffer (PDF) Data Structures and Algorithms: Annotated Reference with Examples - G. Barnett, L. Del Tongo (archived) Data Structures Succinctly Part 1, Syncfusion - Robert Horvick Data Structures Succinctly Part 2, Syncfusion - Robert Horvick Elementary Algorithms - Larry Liu Xinyu (PDF) Essential Algorithms - Krzysztof Kowalczyk and Stack Overflow Documentation project (HTML) Foundations of Computer Science - Al Aho, Jeff Ullman Learning Algorithm - Compiled from StackOverflow documentation (PDF) Lectures Notes on Algorithm Analysis and Computational Complexity (Fourth Edition) - Ian Parberry (use form at bottom of license) LEDA: A Platform for Combinatorial and Geometric Computing - K. Mehlhorn, St. Näher Linked List Basics - Nick Parlante (PDF) Linked List Problems - Nick Parlante (PDF) Matters Computational: Ideas, Algorithms, Source Code - Jörg Arndt (PDF) Open Data Structures: An Introduction - Pat Morin Planning Algorithms - Steven M. LaValle Problems on Algorithms (Second Edition) - Ian Parberry (use form at bottom of license) Purely Functional Data Structures (1996) - Chris Okasaki (PDF) (archived) Sequential and parallel sorting algorithms - Hans Werner Lang (HTML) Text Algorithms - Maxime Crochemore, Wojciech Rytter (PDF) The Algorithm Design Manual - Steven S. Skiena (HTML) The Algorithms The Design of Approximation Algorithms - David P. Williamson, David B. Shmoys (PDF) The Great Tree List Recursion Problem - Nick Parlante (PDF) The Kademlia Protocol Succinctly - Marc Clifton Think Complexity (2nd Edition) - Allen B. Downey Artificial Intelligence AI Safety for Fleshy Humans - Nicky Case and Hack Club (in process) (CC BY-NC) Artificial Intelligence, 3rd Edition (1993) - Patrick Henry Winston (PDF) Artificial Intelligence and the Future for Teaching and Learning - Office of Educational Technology (PDF) Artificial Intelligence for a Better Future: An Ecosystem Perspective on the Ethics of AI and Emerging Digital Technologies - Bernd Carsten Stahl (PDF, EPUB) Artificial Intelligence: Foundations of Computational Agents (2010), 1st Edition - David L. Poole, Alan K. Mackworth @ Cambridge University Press (HTML) Artificial Intelligence: Foundations of Computational Agents (2017), 2nd Edition - David L. Poole, Alan K. Mackworth @ Cambridge University Press (HTML, Slides) Clever Algorithms Nature-Inspired Programming Recipes - Jason Brownlee (PDF) (CC BY-NC-SA) Getting Started with Artificial Intelligence , 2nd Edition - Tom Markiewicz, Josh Zheng (PDF) Graph Representational Learning Book - William L. Hamilton Introduction to Autonomous Robots - Nikolaus Correll (PDF) (CC BY-NC-ND) Machine Learning For Dummies®, IBM Limited Edition - Daniel Kirsch, Judith Hurwitz (PDF) On the Path to AI: Law’s prophecies and the conceptual foundations of the machine learning age - Thomas D. Grant, Damon J. Wischik (PDF, EPUB) Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp - Peter Norvig Probabilistic Programming & Bayesian Methods for Hackers - Cam Davidson-Pilon (HTML, Jupyter Notebook) The History of Artificial Intelligence - Chris Smith, Brian McGuire, Ting Huang, Gary Yang (PDF) The Quest for Artificial Intelligence: A History of Ideas and Achievements - Nils J. Nilsson (PDF) Blockchain Bitcoin and Cryptocurrency Technologies - Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, Steven Goldfeder, Jeremy Clark (PDF) Blockchain for Dummies, 2nd IBM Limited Edition - Manav Gupta (PDF) Build a Blockchain from Scratch in Go with gRPC - Volodymyr Prokopyuk chain.courses - James Gan, Rishub Kumar (archived) Getting Started with Enterprise Blockchain: A Guide to Design and Development - Michael Bradley, David Gorman, Matt Lucas, Matthew Golby-Kirk (PDF) Grokking Bitcoin - Kalle Rosenbaum (HTML) (CC BY-NC-SA) IBM Blockchain: The Founder’s Handbook, Third Edition - Antonio Banda, Matthew Hamilton, Eileen Lowry, John Widdifield, et al. (PDF) Learning Bitcoin from the Command Line - Christopher Allen, Shannon Appelcline, et al. (HTML) Mastering Bitcoin - Unlocking digital currencies (2017), 2nd Edition - Andreas M. Antonopoulos (AsciiDoc) (CC BY-NC-ND) Mastering Ethereum (2018), 1st Edition - Andreas M. Antonopoulos, Gavin Wood (AsciiDoc) Mastering the Lightning Network - Andreas M. Antonopoulos, Olaoluwa Osuntokun, Rene Pickhardt (AsciiDoc) Playtime with Hyperledger Composer - Shubham Chadokar (PDF) Cellular Automata Cloud Computing AWS Well-Architected Framework (PDF, HTML) Azure for Architects, Third Edition (PDF) (email address or account required) Azure Functions Succinctly, Syncfusion (PDF, Kindle) (email address requested, not required) Building Serverless Applications with Google Cloud Run - Wietse Venema (PDF) (email address requested, not required) Cloud Computing for Science and Engineering - Ian Foster, Dennis B. Gannon (in process) Cloud Design Patterns Designing Distributed Systems (account required) Distributed Systems 3rd edition - Maarten van Steen, Andrew S. Tanenbaum (email address required) How to optimize storage costs using Amazon S3 - AWS (PDF) Kubernetes Hardening Guidance - NSA, CISA (PDF) Learn Azure in a Month of Lunches - Iain Foulds (PDF) Monitoring Modern Infrastructure (account required) Multi-tenant Applications for the Cloud, 3rd Edition Openstack CERN Admin guide OpenStack Operations Guide Streamline microservice management with Istio Service Mesh (account required) The Developer’s Guide to Azure Competitive Programming Compiler Design An Introduction to GCC - Brian Gough (archived) Basics of Compiler Design (Anniversary Edition) - Torben Mogensen Compiler Design in C (1990) - Allen Holub, Prentice Hall (PDF) Compiler Design: Theory, Tools, and Examples, C/C++ Edition - Seth D. Bergmann (PDF) (archived) Compiler Design: Theory, Tools, and Examples, Java Edition - Seth D. Bergmann (PDF) (archived) Compiling Scala for the Java Virtual Machine - Michel Schinz (PDF) Compiling Techniques (1969) - F.R.A. Hopgood, Macdonald Crafting Interpreters - Bob Nystrom (HTML) EXPL NITC: Build your own Compiler - Murali Krishnan K., students in the Department of Computer Science and Engineering of the Calicut National Institute of Technology (HTML) Implementing Functional Languages: A Tutorial - Simon Peyton Jones, David Lester Introduction to Compilers and Language Design - Douglas Thain (PDF) Let’s Build a Compiler - Jack W. Crenshaw (PDF) Practical and Theoretical Aspects of Compiler Construction (class lectures and slides) The ANTLR Mega Tutorial Computer Organization and Architecture Computer Science Computer Vision Containers CI/CD for Monorepos: Effectively building, testing, and deploying code with monorepos - Pablo Tomas Fernandez Zavalia, Marko Anastasov, SemaphoreCI (PDF, EPUB, Kindle) CI/CD with Docker and Kubernetes Book - Marko Anastasov, Jérôme Petazzoni, Pablo Tom F. Zavalia, SemaphoreCI (PDF, EPUB, Kindle) Docker Jumpstart - Andrew Odewahn Docker Tutorial - Anthony Baire (PDF) (CC BY-NC-ND) Docker Tutorial - Tutorials Point (HTML, PDF) Dotnet Microservices Architecture for Containerized NET Applications - Cesar de la Torre, Bill Wagner, Mike Rousos (PDF) Kubernetes Deployment & Security Patterns - Alex Williams (PDF) Kubernetes for Full-Stack Developers - Jamon Camisso, Hanif Jetha, Katherine Juell (PDF, EPUB) Uncomplicating Kubernetes - Jeferson Fernando Data Science A Programmer’s Guide to Data Mining - Ron Zacharski (Draft) Data Jujitsu: The Art of Turning Data into Product (email address requested, not required) Data Mining Algorithms In R - Wikibooks Data Mining Concepts and Techniques - Jiawei Han, Micheline Kamber, Jian Pei (PDF) (archived) Data Science at the Command Line - Jeroen Janssens Elements of Data Science - Allen B. Downey Feature Engineering and Selection: A Practical Approach for Predictive Models - Max Kuhn, Kjell Johnson Foundations of Data Science - Avrim Blum, John Hopcroft, Ravindran Kannan (PDF) Fundamentals of Data Visualization - Claus O. Wilke (HTML) Hands-On Data Visualization - Jack Dougherty, Ilya Ilyankou (HTML) High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications - John Wright, Yi Ma (PDF) Internet Advertising: An Interplay among Advertisers, Online Publishers, Ad Exchanges and Web Users (PDF) Introduction to Cultural Analytics & Python - Melanie Walsh Introduction to Data Science - Jeffrey Stanton Mining of Massive Datasets - Jure Leskovec, Anand Rajaraman, Jeffrey D. Ullman (PDF) Probability and Statistics with Examples using R - Siva Athreya, Deepayan Sarkar, Steve Tanner (HTML) (in process) School of Data Handbook Statistical inference for data science - Brian Caffo The Ultimate Guide to 12 Dimensionality Reduction Techniques (with Python codes) - Pulkit Sharma Theory and Applications for Advanced Text Mining Database Database Design – 2nd Edition - Adrienne Watt, Nelson Eng @ BCcampus Open Pressbooks (HTML, PDF, EPUB, Kindle) Database Design Succinctly - Joseph D. Booth (HTML, PDF, EPUB, MOBI) Database Explorations - C.J. Date, Hugh Darwen (PDF) Database Fundamentals - Neeraj Sharma et al. (PDF) Database Management Systems Solutions Manual Third Edition - Raghu Ramakrishnan, Johannes Gehrke, Jeff Derstadt, Scott Selikof, and Lin Zhu (PDF) Databases, Types, and The Relational Model: The Third Manifesto - C.J. Date, Hugh Darwen (PDF) Foundations of Databases Readings in Database Systems, 5th Ed. Temporal Database Management - Christian S. Jensen The Theory of Relational Databases Embedded Systems Game Development Graphics Programming 3D Game Shaders For Beginners - David Lettier (Git) (HTML) Blender 3D: Noob to Pro - Wikibooks Blender Manual Computer Graphics from scratch - Gabriel Gambetta (in process) DirectX manual (draft) GPU Gems Graphics Programming Black Book - Michael Abrash (PDF) Introduction to Modern OpenGL - Alexander Overvoorde (HTML, EPUB, PDF) (C++) Introduction to TouchDesigner 099 (Leanpub account or valid email requested) JPEG - Idea and Practice Learn Computer Graphics From Scratch! - Scratchapixel (in process) Learn OpenGL - Joey de Vries Learn OpenGL RS - Benjamin Wasty, et al. (in process) Learning Modern 3D Graphics Programming - Jason L. McKesson (draft) (archived) Notes for a Computer Graphics Programming Course - Steve Cunningham (PDF) OpenGL - Concepts and illustrations Physically Based Rendering, Third Edition: from Theory to Implementation - Matt Pharr, Wenzel Jakob, Greg Humphreys Ray Tracing Gems - Eric Haines, Tomas Akenine-Möller Ray Tracing Gems II - Adam Marrs, Peter Shirley, Ingo Wald Ray Tracing in One Weekend - Peter Shirley (HTML) ShaderX series - Wolfgang Engel Tutorials for modern OpenGL Virtual Reality - Steven M. LaValle WebGL Insights - Patrick Cozzi, et al. Graphical User Interfaces IDE and editors See also ... Emacs Lisp, Regular Expressions Licensing Machine Learning A Brief Introduction to Machine Learning for Engineers - Osvaldo Simeone (PDF) A Brief Introduction to Neural Networks A Comprehensive Guide to Machine Learning - Soroush Nasiriany, Garrett Thomas, William Wang, Alex Yang (PDF) A Course in Machine Learning (PDF) A First Encounter with Machine Learning - Max Welling (PDF) (archived) A Selective Overview of Deep Learning - Fan, Ma, Zhong (PDF) Algorithms for Reinforcement Learning - Csaba Szepesvári (PDF) An Introduction to Statistical Learning - Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani Applied Machine Learning for Tabular Data - Max Kuhn, Kjell Johnson Approaching Almost Any Machine Learning Problem - Abhishek Thakur (PDF) Bayesian Reasoning and Machine Learning Deep Learning - Ian Goodfellow, Yoshua Bengio, Aaron Courville Deep Learning for Coders with Fastai and PyTorch - Jeremy Howard, Sylvain Gugger (Jupyter Notebooks) Dive into Deep Learning Explorations in Parallel Distributed Processing: A Handbook of Models, Programs, and Exercises - James L. McClelland Foundations of Machine Learning, Second Edition - Mehryar Mohri, Afshin Rostamizadeh, Ameet Talwalkar Free and Open Machine Learning - Maikel Mardjan (HTML) Gaussian Processes for Machine Learning - Carl Edward Rasmussen, Christopher K.I. Williams IBM Machine Learning for Dummies - Judith Hurwitz, Daniel Kirsch Information Theory, Inference, and Learning Algorithms - David J.C. MacKay Interpretable Machine Learning - Christoph Molnar Introduction to CNTK Succinctly - James McCaffrey Introduction to Machine Learning - Amnon Shashua Keras Succinctly - James McCaffrey Learn Tensorflow - Jupyter Notebooks Learning Deep Architectures for AI - Yoshua Bengio (PDF) Machine Learning Machine Learning for Beginners - Microsoft Machine Learning for Data Streams - Albert Bifet, Ricard Gavaldà,

Microservices vs. Service-Oriented Architecture - Mark Richards (HTML) Migrating to Cloud-Native Application Architectures (email address requested) (PDF) Naked objects - Richard Pawson (PDF) OAuth - The Big Picture (email address requested) Object-Oriented Reengineering Patterns - S. Demeyer, S. Ducasse, O. Nierstrasz Practicing Domain-Driven Design - Part 1 - Scott Millett (Leanpub account or valid email requested) Reactive Microservices Architecture (email address requested) Reactive Microsystems: The Evolution of Microservices at Scale (email address requested) Refactor Like a Superhero - Alex Bespoyasov Scrum and XP from the Trenches Serverless apps: Architecture, patterns, and Azure implementation Serverless Design Patterns and Best Practices - Brian Zambrano (Packt account required) Shape Up - Stop Running in Circles and Ship Work that Matters - Ryan Singer (PDF) Site Reliability Engineering Software Architecture Patterns (email address requested, not required) Software Engineering for Internet Applications Source Making Design Patterns and UML Test Driven Development, Extensive Tutorial - Grzegorz Gałęzowski The Catalog of Design Patterns The Site Reliability Workbook - Betsy Beyer, Niall Richard Murphy, David K. Rensin, Kent Kawahara, Stephen Thorne Web API Design - Brian Mulloy (PDF) Web Browser Engineering - Pavel Panchekha, Chris Harrelson Working with Web APIs - Launch School Your API Is Bad - Paddy Foran Standards Linux Standard Base UNIX - The POSIX Standard - IEEE Std 1003.1 Theoretical Computer Science Version Control Systems A git Primer - Daniel Miessler A Visual Git Reference - Mark Lodato Conversational Git - Alan Hohn get-git - Arialdo Martini (HTML, PDF, EPUB) git - the simple guide - Roger Dudler (HTML) Git cookbook - Dennis Kaarsemaker (HTML) Git for Computer Scientists - Tommi Virtanen Git From The Bottom Up - J. Wiegley Git Immersion - Jim Weirich (HTML) Git In The Trenches - Peter Savage Git internals - Scott Chacon (PDF) Git Magic - Ben Lynn, et al. (HTML, PDF, EPUB) Git Notes for Professionals - Compiled from StackOverflow Documentation (PDF) Git Pocket Guide - Richard E. Silverman Git Reference - The GitHub team (archived) Git Succinctly, Syncfusion - Ryan Hodson (PDF, Kindle) (email address requested, not required) Git Tutorial - Tutorials Point (HTML, PDF) Git-Tutorial For-Beginners - HubSpot Product Team Git Workflows - Yan Pritzker (archived) Happy Git and GitHub for the useR - Jenny Bryan Hg Init: a Mercurial Tutorial - Joel Spolsky How to Collaborate on GitHub: A mini book about collaborating on GitHub - André J Introduction to Git and Github - Launch School Introduction to Git and Github - Tutorial - Chris Bourke (PDF) Introduction to Git and GitHub eBook - Bobby Iliev (Markdown, PDF) Learn Git - Learn Version Control with Git - Tobias Günther Mercurial: The Definitive Guide - Bryan O'Sullivan Mercurial: The Definitive Guide 2nd edition - Bryan O'Sullivan Pro Git - Scott Chacon, Ben Straub (HTML, PDF, EPUB, Kindle) Pro Git Reedited - Jon Forrest Ry's Git Tutorial - Ryan Hodson (archived) Subversion Version Control - William Nagel (PDF) Think Like (a) Git: A Guide for the Perplexed - Sam Livingston-Gray Version Control with Subversion - Ben Collins-Sussman, Brian W. Fitzpatrick, C. Michael Pilato Web Performance Web Services RESTful Web Services (PDF) Workflow This project is maintained by EbookFoundation Hosted on GitHub Pages — Theme by orderedlist ZDNET