

Think Like A Programmer An Introduction To Creative Problem Solving V Anton Spraul

Computational technologies have been impacting human life for years. Teaching methods must adapt accordingly to provide the next generation with the necessary knowledge to further advance these human-assistive technologies. Teaching Computational Thinking in Primary Education is a crucial resource that examines the impact that instructing with a computational focus can have on future learners. Highlighting relevant topics that include multifaceted skillsets, coding, programming methods, and digital games, this scholarly publication is ideal for educators, academicians, students, and researchers who are interested in discovering how the future of education is being shaped.

A hands-on, problem-based introduction to building algorithms and data structures to solve problems with a computer. Algorithmic Thinking will teach you how to solve challenging programming problems and design your own algorithms. Daniel Zingaro, a master teacher, draws his examples from world-class programming competitions like USACO and IOI. You'll learn how to classify problems, choose data structures, and identify appropriate algorithms. You'll also learn how your choice of data structure, whether a hash table, heap, or tree, can affect runtime and speed up your algorithms; and how to adopt powerful strategies like recursion, dynamic programming, and binary search to solve challenging problems. Line-by-line breakdowns of the code will teach you how to use algorithms and data structures like: • The breadth-first search algorithm to find the optimal way to play a board game or find the best way to translate a book • Dijkstra's algorithm to determine how many mice can exit a maze or the number of fastest routes between two locations • The union-find data structure to answer questions about connections in a social network or determine who are friends or enemies • The heap data structure to determine the amount of money given away in a promotion • The hash-table data structure to determine whether snowflakes are unique or identify compound words in a dictionary NOTE: Each problem in this book is available on a programming-judge website. You'll find the site's URL and problem ID in the description. What's better than a free correctness check?

The education system is constantly growing and developing as more ways to teach and learn are implemented into the classroom. Recently, there has been a growing interest in teaching computational thinking with schools all over the world introducing it to the curriculum due to its ability to allow students to become proficient at problem solving using logic, an essential life skill. In order to provide the best education possible, it is imperative that computational thinking strategies, along with programming skills and the use of robotics in the classroom, be implemented in order for students to achieve maximum thought processing skills and computer competencies. The Research Anthology on Computational Thinking, Programming, and Robotics in the Classroom is an all-encompassing reference book that discusses how computational thinking, programming, and robotics can be used in education as well as the benefits and difficulties of implementing these elements into the classroom. The book includes strategies for preparing educators to teach computational thinking in the classroom as well as design techniques for incorporating these practices into various levels of school curriculum and within a variety of subjects. Covering topics ranging from decomposition to robot learning, this book is ideal for educators, computer scientists, administrators, academicians, students, and anyone interested in learning more about how computational thinking, programming, and robotics can change the current education system.

JavaScript is at the heart of almost every modern Web application, whether it's Google Apps, Twitter, or the newest browser-based game. Though it's simple for beginners to pick up and play with, JavaScript is not a toy—it's a flexible and complex language that can be used to build full-scale applications. Eloquent JavaScript dives into this flourishing language and teaches you to write code that's beautiful and effective. By immersing you in example code and encouraging experimentation right from the start, the author quickly gives you the tools you need to build your own programs. As you follow along with examples like an artificial life simulation and a version of the classic game Sokoban, you'll learn to: -Understand the essential elements of programming: syntax, control, and data -Use object-oriented and functional programming techniques to organize and clarify your programs -Script the browser and make basic Web applications -Work with tools like regular expressions and XMLHttpRequest objects And since programming is an art that's best learned by doing, all example code is available online in an interactive sandbox for you to experiment with. With Eloquent JavaScript as your guide, you can tweak, expand, and modify the author's code, or throw it away and build your own creations from scratch. Before you know it, you'll be fluent in the language of the Web.

"A great book with deep insights into the bridge between programming and the human mind." - Mike Taylor, CGI Your brain responds in a predictable way when it encounters new or difficult tasks. This unique book teaches you concrete techniques rooted in cognitive science that will improve the way you learn and think about code. In The Programmer's Brain: What every programmer needs to know about cognition you will learn: Fast and effective ways to master new programming languages Speed reading skills to quickly comprehend new code Techniques to unravel the meaning of complex code Ways to learn new syntax and keep it memorized Writing code that is easy for others to read Picking the right names for your variables Making your codebase more understandable to newcomers Onboarding new developers to your team Learn how to optimize your brain's natural cognitive processes to read code more easily, write code faster, and pick up new languages in much less time. This book will help you through the confusion you feel when faced with strange and complex code, and explain a codebase in ways that can make a new team member productive in days! Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the technology Take advantage of your brain's natural processes to be a better programmer. Techniques based in cognitive science make it possible to learn new languages faster, improve productivity, reduce the need for code rewrites, and more. This unique book will help you achieve these gains. About the book The Programmer's Brain unlocks the way we think about code. It offers scientifically sound techniques that can radically improve the way you master new technology, comprehend code, and memorize syntax. You'll learn how to benefit from productive struggle and turn confusion into a learning tool. Along the way, you'll discover how to create study resources as you become an expert at teaching yourself and bringing new colleagues up to speed. What's inside Understand how your brain sees code Speed reading skills to learn code quickly Techniques to unravel complex code Tips for making codebases understandable About the reader For programmers who have experience working in more than one language. About the author Dr. Feliienne Hermans is an associate professor at Leiden University in the Netherlands. She has spent the last decade researching programming, how to learn and how to teach it. Table of Contents PART 1 ON READING CODE BETTER 1 Decoding your confusion while coding 2 Speed reading for code 3 How to learn programming syntax quickly 4 How to read complex code PART 2 ON THINKING ABOUT CODE 5 Reaching a deeper understanding of code 6 Getting better at solving programming problems 7 Misconceptions: Bugs in thinking PART 3 ON WRITING BETTER CODE 8 How to get better at naming things 9 Avoiding bad code and cognitive load: Two frameworks 10 Getting better at solving complex problems PART 4 ON COLLABORATING ON CODE 11 The act of writing code 12 Designing and improving larger systems 13 How to onboard new developers

Think Julia

Program Design Solutions for the Bewildered

How Millennials Became the Burnout Generation

What every programmer needs to know about cognition

From Journeyman to Master

Principles Programming Performance

A Modern Introduction to Programming

This book is for anyone who wants to learn computer programming and knows absolutely nothing about it. Of course, if you are wondering whether this book is going to teach you how to create amazing applets or incredible desktop or mobile applications, the answer is "no"-that is a job for other books. So many books out there can teach you those skills in Visual Basic, C#, or Java. Many of them even claim that they can teach you in 24 hours! Don't laugh! They probably can do that, but all of them take one thing for granted-that the reader knows some basics about computer programming. None of those books, unfortunately, bothers to teach you the first thing that a novice programmer needs to learn, which is "Algorithmic Thinking." Algorithmic Thinking involves more than just learning code. It is a problem solving process that involves learning how to code. With 800 pages, and containing more than 300 solved and 400 unsolved exercises, over 450 true/false, 150 multiple choice, and 180 review questions (the solutions and the answers to which can be found on the Internet), this book is ideal for students, teachers, professors, novices or average programmers, or for anyone who wants to start learning or teaching computer programming using the proper conventions and techniques.

It's easier to learn how to program a computer than it has ever been before. Now everyone can learn to write programs for themselves - no previous experience is necessary. Chris Pine takes a thorough, but lighthearted approach that teaches you the fundamentals of computer programming, with a minimum of fuss or bother. Whether you are interested in a new hobby or a new career, this book is your doorway into the world of programming. Computers are everywhere, and being able to program them is more important than it has ever been. But since most books on programming are written for other programmers, it can be hard to break in. At least it used to be. Chris Pine will teach you how to program. You'll learn to use your computer better, to get it to do what you want it to do. Starting with small, simple one-line programs to calculate your age in seconds, you'll see how to write interactive programs, to use APIs to fetch live data from the internet, to rename your photos from your digital camera, and more. You'll learn the same technology used to drive modern dynamic websites and large, professional applications. Whether you are looking for a fun new hobby or are interested in entering the tech world as a professional, this book gives you a solid foundation in programming. Chris teaches the basics, but also shows you how to think like a programmer. You'll learn through tons of examples, and through programming challenges throughout the book. When you finish, you'll know how and where to learn more - you'll be on your way. What You Need: All you need to learn how to program is a computer (Windows, macOS, or Linux) and an internet connection. Chris Pine will lead you through setting set up with the software you will need to start writing programs of your own.

Thoroughly revised for the latest version of C#, this book explains basic concepts in a clear and explicit way that takes very seriously one thing for granted-that the reader knows nothing about computer programming. Addressed to anyone who has no prior programming knowledge or experience, but a desire to learn programming with C#, it teaches the first thing that every novice programmer needs to learn, which is Algorithmic Thinking. Algorithmic Thinking involves more than just learning code. It is a problem-solving process that involves learning how to code. This edition contains all the popular features of the previous edition and adds a significant number of exercises, as well as extensive revisions and updates. Apart from C# 's arrays, it now also covers dictionaries, while a brand new section provides an effective introduction to the next field that a programmer needs to work with, which is Object Oriented Programming (OOP). This book has a class course structure with questions and exercises at the end of each chapter so you can test what you have learned right away and improve your comprehension. With 250 solved and 450 unsolved exercises, 475 true/false, about 150 multiple choice, and 200 review questions and crosswords (the solutions and the answers to which can be found on the Internet), this book is ideal for novices or average programmers, for self-study high school students first-year college or university students teachers professors anyone who wants to start learning or teaching computer programming using the proper conventions and techniques

A revolutionary approach to enhancing productivity, creating flow, and vastly increasing your ability to capture, remember, and benefit from the unprecedented amount of information all around us. For the first time in history, we have instantaneous access to the world's knowledge. There has never been a better time to learn, to contribute, and to improve ourselves. Yet, rather than feeling empowered, we are often left feeling overwhelmed by this constant influx of information. The very knowledge that was supposed to set us free has instead led to the paralyzing stress of believing we'll never know or remember enough. Now, this eye-opening and accessible guide shows how you can easily create your own personal system for knowledge management, otherwise known as a Second Brain. As a trusted and organized digital repository of your most valued ideas, notes, and creative work synced across all your devices and platforms, a Second Brain gives you the confidence to tackle your most important projects and ambitious goals. Discover the full potential of your ideas and translate what you know into more powerful, more meaningful improvements in your work and life by Building a Second Brain.

When programmers list their favorite books, Jon Bentley's collection of programming pearls is commonly included among the classics. Just as natural pearls grow from grains of sand that irritate oysters, programming pearls have grown from real problems that have irritated real programmers. With origins beyond solid engineering, in the realm of insight and creativity, Bentley's pearls offer unique and clever solutions to those nagging problems. Illustrated by programs designed as much for fun as for instruction, the book is filled with lucid and witty descriptions of practical programming techniques and fundamental design principles. It is not at all surprising that Programming Pearls has been so highly valued by programmers at every level of experience. In this revision, the first in 14 years, Bentley has substantially updated his essays to reflect current programming methods and environments. In addition, there are three new essays on testing, debugging, and timing set representations string problems All the original programs have been rewritten, and an equal amount of new code has been generated. Implementations of all the programs, in C or C++, are now available on the Web. What remains the same in this new edition is Bentley's focus on the hard core of programming problems and his delivery of workable solutions to those problems. Whether you are new to Bentley's classic or are revisiting his work for some fresh insight, the book is sure to make your own list of favorites.

Without Even Trying

Think Python

Understanding How We Learn

How Software Works

Learn Python 3 the Hard Way

How to Think Like a Computer Scientist

An Introduction to Creative Problem Solving

*Microsoft's C# ("C sharp") is a modern, object-oriented programming language built from the ground up to exploit the power of XML-based Web services on Microsoft's new .NET platform. With its Visual C++ development system heritage, C# will enable millions of C and C++ developers to use existing skills to rapidly build sophisticated XML-based .NET applications. Why Will Web Developers Switch to C#? ...Because it's the ideal solution for C and C++ programmers who need to combine rapid development with the power to access all the functionality of the Microsoft.NET platform. They want an environment that is completely in sync with emerging Web standards and one that provides easy integration with existing applications. C#.net Web Developer's Guide will enhance developer productivity and help them eliminate programming errors that can lead to increased development costs. This book teaches Web developers to quickly and easily build solutions for the Microsoft .NET platform. Web developers will learn to use C# components to build Web services and applications that are available across the Internet, from any application running on any platform. * Timely coverage of newly released product - programmers and developers are anxious to learn about the new technology * Comes with Syngress' revolutionary wallet-sized CD containing a printable HTML version of the book and all of the source code examples and demos of popular C# upgrade and programming tools*

Get Programming: Learn to code with Python teaches you the basics of computer programming using the Python language. In this exercise-driven book, you'll be doing something on nearly every page as you work through 38 compact lessons and 7 engaging capstone projects. By exploring the crystal-clear illustrations, exercises that check your understanding as you go, and tips for what to try next, you'll start thinking like a programmer in no time. This book works perfectly alongside our video course Get Programming with Python in Motion, available exclusively at Manning.com: www.manning.com/livevideo/get-programming-with-python-in-motion Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. What's Inside Programming skills you can use in any language Learn to code—no experience required Learn Python, the language for beginners Dozens of exercises and examples help you learn by doing About the Reader No prior programming experience needed. Table of Contents LEARNING HOW TO PROGRAM Lesson 1 - Why should you learn how to program? Lesson 2 - Basic principles of learning a programming language UNIT 1 - VARIABLES, TYPES, EXPRESSIONS, AND STATEMENTS Lesson 3 - Introducing Python: a programming language Lesson 4 - Variables and expressions: giving names and values to things Lesson 5 - Object types and statements of code 46 Lesson 6 - Capstone project: your first Python program-convert hours to minutes UNIT 2 - STRINGS, TUPLES, AND INTERACTING WITH THE USER Lesson 7 - Introducing string objects: sequences of characters Lesson 8 - Advanced string operations Lesson 9 - Simple error messages Lesson 10 - Tuple objects: sequences of any kind of object Lesson 11 - Interacting with the user Lesson 12 - Capstone project: name mashup UNIT 3 - MAKING DECISIONS IN YOUR PROGRAMS Lesson 13 - Introducing decisions in programs Lesson 14 - Making more-complicated decisions Lesson 15 - Capstone project: choose your own adventure UNIT 4 - REPEATING TASKS Lesson 16 - Repeating tasks with loops Lesson 17 - Customizing loops Lesson 18 - Repeating tasks while conditions hold Lesson 19 - Capstone project: Scrabble, Art Edition UNIT 5 - ORGANIZING YOUR CODE INTO REUSABLE BLOCKS Lesson 20 - Building programs to last Lesson 21 - Achieving modularity and abstraction with functions Lesson 22 - Advanced operations with functions Lesson 23 - Capstone project: analyze your friends UNIT 6 - WORKING WITH MUTABLE DATA TYPES Lesson 24 - Mutable and immutable objects Lesson 25 - Working with lists Lesson 26 - Advanced operations with lists Lesson 27 - Dictionaries as maps between objects Lesson 28 - Aliasing and copying lists and dictionaries Lesson 29 - Capstone project: document similarity UNIT 7 - MAKING YOUR OWN OBJECT TYPES BY USING OBJECT-ORIENTED PROGRAMMING Lesson 30 - Making your own object types Lesson 31 - Creating a class for an object type Lesson 32 - Working with your own object types Lesson 33 - Customizing classes Lesson 34 - Capstone project: card game UNIT 8 - USING LIBRARIES TO ENHANCE YOUR PROGRAMS Lesson 35 - Useful libraries Lesson 36 - Testing and debugging your programs Lesson 37 - A library for graphical user interfaces Lesson 38 - Capstone project: game of tag Appendix A - Answers to lesson exercises Appendix B - Python cheat sheet Appendix C - Interesting Python libraries

We use software every day to perform all kinds of magical, powerful tasks. It's the force behind stunning CGI graphics, safe online shopping, and speedy Google searches. Software drives the modern world, but its inner workings remain a mystery to many. How Software Works explains how computers perform common-yet-amazing tasks that we take for granted every day. Inside you'll learn: -How data is encrypted -How passwords are used and protected -How computer graphics are created -How video is compressed for streaming and storage -How data is searched (and found) in huge databases -How programs can work together on the same problem without conflict -How data travels over the Internet How Software Works breaks down these processes with patient explanations and intuitive diagrams so that anyone can understand—no technical background is required, and you won't be reading through any code. In plain English, you'll examine the intricate logic behind the technologies you constantly use but never understood. If you've ever wondered what really goes on behind your computer screen, How Software Works will give you fascinating look into the software all around you.

Summary *Get Programming with Go* introduces you to the powerful Go language without confusing jargon or high-level theory. By working through 32 quick-fire lessons, you'll quickly pick up the basics of the innovative Go programming language! Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Go is a small programming language designed by Google to tackle big problems. Large projects mean large teams with people of varying levels of experience. Go offers a small, yet capable, language that can be understood and used by anyone, no matter their experience. About the Book Hobbyists, newcomers, and professionals alike can benefit from a fast, modern language; all you need is the right resource! Get Programming with Go provides a hands-on introduction to Go language fundamentals, serving as a solid foundation for your future programming projects. You'll master Go syntax, work with types and functions, and explore bigger ideas like state and concurrency, with plenty of exercises to lock in what you learn. What's inside Language concepts like slices, interfaces, pointers, and concurrency Seven capstone projects featuring spacefaring gophers, Mars rovers, ciphers, and simulations All examples run in the Go Playground - no installation required! About the Reader This book is for anyone familiar with computer programming, as well as anyone with the desire to learn. About the Author Nathan Youngman organizes the Edmonton Go meetup and is a mentor with Canada Learning Code. Roger Peppé contributes to Go and runs the Newcastle upon Tyne Go meetup. Table of Contents Unit 0 - GETTING STARTED Get ready, get set, Go Unit 1 - IMPERATIVE PROGRAMMING A glorified calculator Loops and branches Variable scope Capstone: Ticket to Mars Unit 2 - TYPES Real numbers Whole numbers Big numbers Multilingual text Converting between types Capstone: The Vigenère cipher Unit 3 - BUILDING BLOCKS Functions Methods First-class functions Capstone: Temperature tables Unit 4 - COLLECTIONS Arrayed in splendor Slices: Windows into arrays A bigger slice The ever-versatile map Capstone: A slice of life Unit 5 - STATE AND BEHAVIOR A little structure Go's got no class Composition and forwarding Interfaces Capstone: Martian animal sanctuary Unit 6 - DOWN THE GOPHER HOLE A few pointers Much ado about nil To err is human Capstone: Sudoku rules Unit 7 - CONCURRENT PROGRAMMING Goroutines and concurrency Concurrent state Capstone: Life on Mars

Tap into the wisdom of experts to learn what every programmer should know, no matter what language you use. With the 97 short and extremely useful tips for programmers in this book, you'll expand your skills by adopting new approaches to old problems, learning appropriate best practices, and honing your craft through sound advice. With contributions from some of the most experienced and respected practitioners in the industry—including Michael Feathers, Pete Goodliffe, Diomidis Spinellis, Cay Horstmann, Verity Stob, and many more—this book contains practical knowledge and principles that you can apply to all kinds of projects. A few of the 97 things you should know: "Code in the Language of the Domain" by Dan North "Write Tests for People" by Gerard Meszaros "Convenience Is Not an -ility" by Gregor Hohpe "Know Your IDE" by Heinz Kabutz "A Message to the Future" by Linda Rising "The Boy Scout Rule" by Robert C. Martin (Uncle Bob) "Beware the Share" by Udi Dahan

Research Anthology on Computational Thinking, Programming, and Robotics in the Classroom

Algorithmic Thinking

A Problem-Based Introduction

C# and Algorithmic Thinking for the Complete Beginner (2nd Edition)

C#.Net Developer's Guide

Everything You Need to Ace Math in One Big Fat Notebook

Get Fit, Feel Better, and Keep Coding

This practical handbook on software project success and survival explains how to confront five important issues involved in all software projects--people, politics, process, project management, and tools.

How to Think Like a Programmer is a bright, accessible, fun read describing the mindset and mental methods of programmers. Anticipating the problems that student's have through the character of Brian the Wildebeest, the slower pace required for this approach is made interesting and engaging by visual impact of hand-drawn sketches, frequent (paper-based) interactivities and the everyday tasks (e.g. coffee making) used as the basis of worked examples.

A back-to-basics guide on coding for absolute beginners, whether adults or children – no prior experience required! Coding is set to change the way we work and the skills we will need in the future. For those who know nothing about coding, getting to grips with the basics is daunting. Too many of the beginner books launch straight into programming techniques but what is really needed is an understanding of the key concepts of coding. Programming then becomes much easier to grasp. This accessible, fun book goes right back to the very basics, teaching central concepts such as loops, data types, pseudocode and calculations without having to learn a single line of code! Using a set of dice, a deck of cards or a pack of dominoes to enjoy fun and straightforward exercises, you will practise key skills such as critical thinking, creativity, logic and problem-solving and begin to think like a coder without even turning on your computer. Once you are equipped with this basic toolkit, Think Like a Coder discusses the basic programmes that are available for beginners, keeping a focus on simple activities that draw analogies with the outside world to make learning easy and fun. Suitable for absolute beginners, adults and children. Designed to be a thorough yet lighthearted introduction for the complete beginner, Think Like a Coder is an essential addition to any keen programmer’s bookshelf.

The real challenge of programming isn't learning a language's syntax—it's learning to creatively solve problems so you can build something great. In this one-of-a-kind text, author V. Anton Spraul breaks down the ways that programmers solve problems and teaches you what other introductory books often ignore: how to Think Like a Programmer. Each chapter tackles a single programming concept, like classes, pointers, and recursion, and open-ended exercises throughout challenge you to apply your knowledge. You'll also learn how to: –Split problems into discrete components to make them easier to solve –Make the most of code reuse with functions, classes, and libraries –Pick the perfect data structure for a particular job –Master more advanced programming tools like recursion and dynamic memory –Organize your thoughts and develop strategies to tackle particular types of problems Although the book’s examples are written in C++, the creative problem-solving concepts they illustrate go beyond any particular language; in fact, they often reach outside the realm of computer science. As the most skillful programmers know, writing great code is a creative art—and the first step in creating your masterpiece is learning to Think Like a Programmer.

Currently used at many colleges, universities, and high schools, this hands-on introduction to computer science is ideal for people with little or no programming experience. The goal of this concise book is not just to teach you Java, but to help you think like a computer scientist. You'll learn how to program—a useful skill by itself—but you'll also discover how to use programming as a means to an end. Authors Allen Downey and Chris Mayfield start with the most basic concepts and gradually move into topics that are more complex, such as recursion and object-oriented programming. Each brief chapter covers the material for one week of a college course and includes exercises to help you practice what you’ve learned. Learn one concept at a time: tackle complex topics in a series of small steps with examples Understand how to formulate problems, think creatively about solutions, and write programs clearly and accurately Determine which development techniques work best for you, and practice the important skill of debugging Learn relationships among input and output, decisions and loops, classes and methods, strings and arrays Work on exercises involving word games, graphics, puzzles, and playing cards

Database

Learn to code with Python

A Proven Method to Organize Your Digital Life and Unlock Your Creative Potential

How to Think Like a Coder

A Visual Guide

Programming Pearls

Computer Science Made Simple

If you want to learn how to program, working with Python is an excellent way to start. This hands-on guide takes you through the language a step at a time, beginning with basic programming concepts before moving on to functions, recursion, data structures, and object-oriented design. This second edition and its supporting code have been updated for Python 3. Through exercises in each chapter, you ' ll try out programming concepts as you learn them. Think Python is ideal for students at the high school or college level, as well as self-learners, home-schooled students, and professionals who need to learn programming basics. Beginners just getting their feet wet will learn how to start with Python in a browser. Start with the basics, including language syntax and semantics Get a clear definition of each programming concept Learn about values, variables, statements, functions, and data structures in a logical progression Discover how to work with files and databases Understand objects, methods, and object-oriented programming Use debugging techniques to fix syntax, runtime, and semantic errors Explore interface design, data structures, and GUI-based programs through case studies

The goal of this book is to teach you to think like a computer scientist. This way of thinking combines some of the best features of mathematics, engineering, and natural science. Like mathematicians, computer scientists use formal languages to denote ideas (specifically computations). Like engineers, they design things, assembling components into systems and evaluating tradeoffs among alternatives. Like scientists, they observe the behavior of complex systems, form hypotheses, and test predictions. The single most important skill for a computer scientist is problem solving. Problem solving means the ability to formulate problems, think creatively about solutions, and express a solution clearly and accurately. As it turns out, the process of learning to program is an excellent opportunity to practice problem-solving skills. That's why this chapter is called, The way of the program. On one level, you will be learning to program, a useful skill by itself. On another level, you will use programming as a means to an end. As we go along, that end will become clearer.

If you ' re just learning how to program, Julia is an excellent JIT-compiled, dynamically typed language with a clean syntax. This hands-on guide uses Julia 1.0 to walk you through programming one step at a time, beginning with basic programming concepts before moving on to more advanced capabilities, such as creating new types and multiple dispatch. Designed from the beginning for high performance, Julia is a general-purpose language ideal for not only numerical analysis and computational science but also web programming and scripting. Through exercises in each chapter, you ' ll try out programming concepts as you learn them. Think Julia is perfect for students at the high school or college level as well as self-learners and professionals who need to learn programming basics. Start with the basics, including language syntax and semantics Get a clear definition of each programming concept Learn about values, variables, statements, functions, and data structures in a logical progression Discover how to work with files and databases Understand types, methods, and multiple dispatch Use debugging techniques to fix syntax, runtime, and semantic errors Explore interface design and data structures through case studies

Learn to Code by Solving Problems is a practical introduction to programming using Python. It uses coding-competition challenges to teach you the mechanics of coding and how to think like a savvy programmer. Computers are capable of solving almost any problem when given the right instructions. That ' s where programming comes in. This beginner ' s book will have you writing Python programs right away. You ' ll solve interesting problems drawn from real coding competitions and build your programming skills as you go. Every chapter presents problems from coding challenge websites, where online judges test your solutions and provide targeted feedback. As you practice using core Python features, functions, and techniques, you ' ll develop a clear understanding of data structures, algorithms, and other programming basics. Bonus exercises invite you to explore new concepts on your own, and multiple-choice questions encourage you to think about how each piece of code works. You ' ll learn how to: • Run Python code, work with strings, and use variables • Write programs that make decisions • Make code more efficient with while and for loops • Use Python sets, lists, and dictionaries to organize, sort, and search data • Design programs using functions and top-down design • Create complete-search algorithms and use Big O notation to design more efficient code By the end of the book, you ' ll not only be proficient in Python, but you ' ll also understand how to think through problems and tackle them with code. Programming languages come and go, but this book gives you the lasting foundation you need to start thinking like a programmer.

You Will Learn Python 3! Zed Shaw has perfected the world ' s best system for learning Python 3. Follow it and you will succeed—just like the millions of beginners Zed has taught to date! You bring the discipline, commitment, and persistence; the author supplies everything else. In Learn Python 3 the Hard Way, you ' ll learn Python by working through 52 brilliantly crafted exercises. Read them. Type their code precisely. (No copying and pasting!) Fix your mistakes. Watch the programs run. As you do, you ' ll learn how a computer works; what good programs look like; and how to read, write, and think about code. Zed then teaches you even more in 5+ hours of video where he shows you how to break, fix, and debug your code—live, as he ' s doing the exercises. Install a complete Python environment Organize and write code Fix and break code Basic mathematics Variables Strings and text Interact with users Work with files Looping and logic Data structures using lists and dictionaries Program design Object-oriented programming Inheritance and composition Modules, classes, and objects Python packaging Automated testing Basic game development Basic web development It ' ll be hard at first. But soon, you ' ll just get it—and that will feel great! This course will reward you for every minute you put into it. Soon, you ' ll know one of the world ' s most powerful, popular programming languages. You ' ll be a Python programmer. This Book Is Perfect For Total beginners with zero programming experience Junior developers who know one or two languages Returning professionals who haven ' t written code in years Seasoned professionals looking for a fast, simple, crash course in Python 3

Think Like a Programmer

Team Geek

Get Programming with Go

97 Things Every Programmer Should Know

Death March

A learner's guide to programming using the Python language

Head First Programming

This book summarizes so many things we need to know as a programmer, from a programmer 's perspective. Starting from the basic technical skills one must acquire, to managerial skills to manage a team of programmers.Emphases are put on the ethics of working as a programmer and as a member of the team. Inside this book you'll find tips on how to learn communication language among your peers, how to talk to non-engineers, and how to deal with difficult people. This book also shows us how to take a break when needed, and how to recognize when to go home, and how to communicate and negotiate with your boss, so that you won't end up working for 50 to 60 hours a week. This is a very good book, one that should be a mandatory for wannabe and professional programmers. If you happened to be a manager who supervises a hive of programmers, this book should provide you with useful insights into their minds and habits.

Thoroughly revised for the latest version of Python, this book explains basic concepts in a clear and explicit way that takes very seriously one thing for granted-that the reader knows nothing about computer programming. Addressed to anyone who has no prior programming knowledge or experience, but a desire to learn programming with Python, it teaches the first thing that every novice programmer needs to learn, which is Algorithmic Thinking. ?lgorithmic Thinking involves more than just learning code. It is a problem-solving process that involves learning how to code. This edition contains all the popular features of the previous edition and adds a significant number of exercises, as well as extensive revisions and updates. Apart from Python's lists, it now also covers dictionaries, while a brand new section provides an effective introduction to the next field that a programmer needs to work with, which is Object Oriented Programming (OOP). This book has a class course structure with questions and exercises at the end of each chapter so you can test what you have learned right away and improve your comprehension. With 250 solved and 450 unsolved exercises, 475 true/false, and 150 multiple choice, and 200 review questions and crosswords (the solutions and the answers to which can be found on the Internet), this book is ideal for novices or average programmers, for self-study high school students first-year college or university students teachers professors anyone who wants to start learning or teaching computer programming using the proper conventions and techniques

Database: Principles Programming Performance provides an introduction to the fundamental principles of database systems. This book focuses on database programming and the relationships between principles, programming, and performance. Organized into 10 chapters, this book begins with an overview of database design principles and presents a comprehensive introduction to the concepts used by a DBA. This text then provides grounding in many abstract concepts of the relational model. Other chapters introduce SQL, describing its capabilities and covering the statements and functions of the programming language. This book provides as well an introduction to Embedded SQL and Dynamic SQL that is sufficiently detailed to enable students to immediately start writing database programs. The final chapter deals with some of the motivations for database systems spanning multiple CPUs, including client-server and distributed transactions. This book is a valuable resource for database administrators, application programmers, specialist users, and end users.

It's the revolutionary math study guide just for middle school students from the brains behind Brain Quest. Everything You Need to Ace Math . . . covers everything to get a student over any math hump: fractions, decimals, and how to multiply and divide them; ratios, proportions, and percentages; geometry; statistics and probability; expressions and equations; and the coordinate plane and functions. The BIG FAT NOTEBOOK™ series is built on a simple and irresistible conceit—borrowing the notes from the smartest kid in class. There are five books in all, and each is the only book you need for each main subject taught in middle school: Math, Science, American History, English Language Arts, and World History. Inside the reader will find every subject's key concepts, easily digested and summarized: Critical ideas highlighted in neon colors. Definitions explained. Doodles that illuminate tricky concepts in marker. Mnemonics for memorable shortcuts. And quizzes to recap it all. The BIG FAT NOTEBOOKS meet Common Core State Standards, Next Generation Science Standards, and state history standards, and are vetted by National and State Teacher of the Year Award-winning teachers. They make learning fun and are the perfect next step for every kid who grew up on Brain Quest.

Printed in full color. To keep doing what you love, you need to maintain your own systems, not just the ones you write code for. Regular exercise and proper nutrition help you learn, remember, concentrate, and be creative--skills critical to doing your job well. Learn how to change your work habits, master exercises that make working at a computer more comfortable, and develop a plan to keep fit, healthy, and sharp for years to come. Small changes to your habits can improve your health--without getting in the way of your work. The Healthy Programmer gives you a daily plan of action that's incremental and iterative just like the software development processes you're used to. Every tip, trick, and best practice is backed up by the advice of doctors, scientists, therapists, nutritionists, and numerous fitness experts. We'll review the latest scientific research to understand how being healthy is good for your body and mind. You'll start by adding a small amount of simple activity to your day--no trips to the gym needed. You'll learn how to mitigate back pain, carpal tunnel syndrome, headaches, and many other common sources of pain. You'll also learn how to refactor your diet to properly fuel your body without gaining weight or feeling hungry. Then, you'll turn the exercises and activities into a pragmatic workout methodology that doesn't interfere with the demands of your job and may actually improve your cognitive skills. You'll also learn the secrets of prominent figures in the software community who turned their health around by making diet and exercise changes. Throughout, you'll track your progress with a "companion iPhone app". Finally, you'll learn how to make your healthy lifestyle pragmatic, attainable, and fun. If you're going to live well, you should enjoy it. Disclaimer This book is intended only as an informative guide for those wishing to know more about health issues. In no way is this book intended to replace, countermand, or conflict with the advice given to you by your own healthcare provider including Physician, Nurse Practitioner, Physician Assistant, Registered Dietician, and other licensed professionals. Keep in mind that results vary from person to person. This book is not intended as a substitute for medical or nutritional advice from a healthcare provider or dietician. Some people have a medical history and/or condition and/or nutritional requirements that warrant individualized recommendations and, in some cases, medications and healthcare surveillance. Do not start, stop, or change medication and dietary recommendations without professional medical and/or Registered Dietician advice. A healthcare provider should be consulted if you are on medication or if there are any symptoms that may require diagnosis or medical attention. Do not change your diet if you are ill, or on medication except under the supervision of a healthcare provider. Neither this, nor any other book or discussion forum is intended to take the place of personalized medical care of treatment provided by your healthcare provider. This book was current as of January, 2013 and as new information becomes available through research, experience, or changes to product contents, some of the data in this book may become invalid. You should seek the most up to date information on your medical care and treatment from your health care professional. The ultimate decision concerning care should be made between you and your healthcare provider. Information in this book is general and is offered with no guarantees on the part of the author, editor or The Pragmatic Programmers, LLC. The author, editors and publisher disclaim all liability in connection with the use of this book.

Teaching Computational Thinking in Primary Education

Collective Wisdom from the Experts

The Healthy Programmer

Learn to Think Like a Programmer

Learn to Think Like a Programmer and Start Writing Code

Geeky Projects for the Curious Programmer

Think Java

Be smarter than your computer If you don't understand computers, you can quickly be left behind in today's fast-paced, machine-dependent society. Computer Science Made Simple offers a straightforward resource for technology novices and advanced techies alike. It clarifies all you need to know, from the basic components of today's computers to using advanced applications. The perfect primer, it explains how it all comes together to make computers work. Topics covered include: * hardware * software * programming * networks * the internet * computer graphics * advanced computer concepts * computers in society Look for these Made Simple titles: Accounting Made Simple Arithmetic Made Simple Astronomy Made Simple Biology Made Simple Bookkeeping Made Simple Business Letters Made Simple Chemistry Made Simple Earth Science Made Simple English Made Simple French Made Simple German Made Simple Inglés Hecho Fácil Investing Made Simple Italian Made Simple Keyboarding Made Simple Latin Made Simple Learning English Made Simple Mathematics Made Simple The Perfect Business Plan Made Simple Philosophy Made Simple Physics Made Simple Psychology Made Simple Sign Language Made Simple Spanish Made Simple Spelling Made Simple Statistics Made Simple Your Small Business Made Simple www.broadway.com

An incendiary examination of burnout in millennials--the cultural shifts that got us here, the pressures that sustain it, and the need for drastic change

Programming isn't just about syntax and assembling code--it's about problem solving, and all good programmers must think creatively to solve problems. Like the best-selling Think Like a Programmer before it (with over 75,000 copies sold worldwide), this Python-based edition will help you transition from reading programs to writing them, in, Python. (No prior programming experience required!) Rather than simply point out solutions to problems, author V. Anton Spraul will get you thinking by exposing you to techniques that will teach you how to solve programming problems on your own. Each chapter covers a single programming concept like data types, control flow, code reuse, recursion, and classes, then a series of Python-based exercises have you put your skills to the test. You'll learn how to: -Break big problems down into simple, manageable steps to build into solutions -Write custom functions to solve new problems -Use a debugger to examine each line of your running program in order to fully understand how it works -Tackle problems strategically by turning each new concept into a problem-solving tool The Python edition of Think Like a Programmer aims squarely at the beginning programmer, with additional chapters on early programming topics such as variables, decisions, and looping. Version: This book is based on Python 3.

Educational practice does not, for the most part, rely on research findings. Instead, there's a preference for relying on our intuitions about what's best for learning. But relying on intuition may be a bad idea for teachers and learners alike. This accessible guide helps teachers to integrate effective, research-backed strategies for learning into their classroom practice. The book explores exactly what constitutes good evidence for effective learning and teaching strategies, how to make evidence-based judgments instead of relying on intuition, and how to apply findings from cognitive psychology directly to the classroom. Including real-life examples and case studies, FAQs, and a wealth of engaging illustrations to explain complex concepts and emphasize key points, the book is divided into four parts: Evidence-based education and the science of learning Basics of human cognitive processes Strategies for effective learning Tips for students, teachers, and parents. Written by "The Learning Scientists" and fully illustrated by Oliver Caviglioli, Understanding How We Learn is a rejuvenating and fresh examination of cognitive psychology's application to education. This is an essential read for all teachers and educational practitioners, designed to convey the concepts of research to the reality of a teacher's classroom.

The official book on the Rust programming language, written by the Rust development team at the Mozilla Foundation, fully updated for Rust 2018. The Rust Programming Language is the official book on Rust: an open source systems programming language that helps you write faster, more reliable software. Rust offers control over low-level details (such as memory usage) in combination with high-level ergonomics, eliminating the hassle traditionally associated with low-level languages. The authors of The Rust Programming Language, members of the Rust Core Team, share their knowledge and experience to show you how to take full advantage of Rust's features--from installation to creating robust and scalable programs. You'll begin with basics like creating functions, choosing data types, and binding variables and then move on to more advanced concepts, such as: • Ownership and borrowing, lifetimes, and traits • Using Rust's memory safety guarantees to build fast, safe programs • Testing, error handling, and effective refactoring • Generics, smart pointers, multithreading, trait objects, and advanced pattern matching • Using Cargo, Rust's built-in package manager, to build, test, and document your code and manage dependencies • How best to use Rust's advanced compiler with compiler-led programming techniques You'll find plenty of code examples throughout the book, as well as three chapters dedicated to building complete projects to test your learning: a number guessing game, a Rust implementation of a command line tool, and a multithreaded server. New to this edition: An extended section on Rust macros, an expanded chapter on modules, and appendixes on Rust development tools and editions.

Get Programming

The Complete Middle School Study Guide

Think Like a Programmer, Python Edition

The Magic Behind Encryption, CGI, Search Engines, and Other Everyday Technologies

How to Think Like a Programmer

The Pragmatic Programmer

What others in the trenches say about The Pragmatic Programmer... "The cool thing about this book is that it's great for keeping the programming process fresh. The book helps you to continue to grow and clearly comes from people who have been there." –Kent Beck, author of Extreme Programming Explained: Embrace Change "I found this book to be a great mix of solid advice and wonderful analogies!" –Martin Fowler, author of Refactoring and UML Distilled "I would buy a copy, read it twice, then tell all my colleagues to run out and grab a copy. This is a book I would never loan because I would worry about it being lost." –Kevin Ruland, Management Science, MSG-Logistics "The wisdom and practical experience of the authors is obvious. The topics presented are relevant and useful... By far its greatest strength for me has been the outstanding analogies–tracer bullets, broken windows, and the fabulous helicopter-based explanation of the need for orthogonality, especially in a crisis situation. I have little doubt that this book will eventually become an excellent source of useful information for journeymen programmers and expert mentors alike." –John Lakos, author of Large-Scale C++ Software Design "This is the sort of book I will buy a dozen copies of when it comes out so I can give it to my clients." –Eric Vought, Software Engineer "Most modern books on software development fail to cover the basics of what makes a great software developer, instead spending their time on syntax or technology where in reality the greatest leverage possible for any software team is in having talented developers who really know their craft well. An excellent book." –Pete McBreen, Independent Consultant "Since reading this book, I have implemented many of the practical suggestions and tips it contains. Across the board, they have saved my company time and money while helping me get my job done quicker! This should be a desktop reference for everyone who works with code for a living." –Jared Richardson, Senior Software Developer, iRenaissance, Inc. "I would like to see this issued to every new employee at my company..." –Chris Cleeland, Senior Software Engineer, Object Computing, Inc. "If I'm putting together a project, it's the authors of this book that I want. . . . And failing that I'd settle for people who've read their book." –Ward Cunningham Straight from the programming trenches, The Pragmatic Programmer cuts through the increasing specialization and technicalities of modern software development to examine the core process--taking a requirement and producing working, maintainable code that delights its users. It covers topics ranging from personal responsibility and career development to architectural techniques for keeping your code flexible and easy to adapt and reuse. Read this book, and you'll learn how to Fight software rot; Avoid the trap of duplicating knowledge; Write flexible, dynamic, and adaptable code; Avoid programming by coincidence; Bullet-proof your code with contracts, assertions, and exceptions; Capture real requirements; Test ruthlessly and effectively; Delight your users; Build teams of pragmatic programmers; and Make your developments more precise with automation. Written as a series of self-contained sections and filled with entertaining anecdotes, thoughtful examples, and interesting analogies, The Pragmatic Programmer illustrates the best practices and major pitfalls of many different aspects of software development. Whether you're a new coder, an experienced programmer, or a manager responsible for software projects, use these lessons daily, and you'll quickly see improvements in personal productivity, accuracy, and job satisfaction. You'll learn skills and develop habits and attitudes that form the foundation for long-term success in your career. You'll become a Pragmatic Programmer.

In a perfect world, software engineers who produce the best code are the most successful. But in our perfectly messy world, success also depends on how you work with people to get your job done. In this highly entertaining book, Brian Fitzpatrick and Ben Collins-Sussman cover basic patterns and anti-patterns for working with other people, teams, and users while trying to develop software. This is valuable information from two respected software engineers whose popular series of talks—including "Working with Poisonous People"—has attracted hundreds of thousands of followers. Writing software is a team sport, and human factors have as much influence on the outcome as technical factors. Even if you've spent decades learning the technical side of programming, this book teaches you about the often-overlooked human component. By learning to collaborate and investing in the "soft skills" of software engineering, you can have a much greater impact for the same amount of effort. Team Geek was named as a Finalist in the 2013 Jolt Awards from Dr. Dobb's Journal. The publication's panel of judges chose five notable books, published during a 12-month period ending June 30, that every serious programmer should read.

Looking for a reliable way to learn how to program on your own, without being overwhelmed by confusing concepts? Head First Programming introduces the core concepts of writing computer programs -- variables, decisions, loops, functions, and objects -- which apply regardless of the programming language. This book offers concrete examples and exercises in the dynamic and versatile Python language to demonstrate and reinforce these concepts. Learn the basic tools to start writing the programs that interest you, and get a better understanding of what software can (and cannot) do. When you're finished, you'll have the necessary foundation to learn any programming language or tackle any software project you choose. With a focus on programming concepts, this book teaches you how to: Understand the core features of all programming languages, including: variables, statements, decisions, loops, expressions, and operators Reuse code with functions Use library code to save time and effort Select the best data structure to manage complex data Write programs that talk to the Web Share your data with other programs Write programs that test themselves and help you avoid embarrassing coding errors We think your time is too valuable to waste struggling with new concepts. Using the latest research in cognitive science and learning theory to craft a multi-sensory learning experience, Head First Programming uses a visually rich format designed for the way your brain works, not a text-heavy approach that puts you to sleep.

Think Like a ProgrammerAn Introduction to Creative Problem SolvingNo Starch Press

Python is a powerful programming language that's easy to learn and fun to play with. But once you've gotten a handle on the basics, what do you do next? Python Playground is a collection of imaginative programming projects that will inspire you to use Python to make art and music, build simulations of real-world phenomena, and interact with hardware like the Arduino and Raspberry Pi. You'll learn to use common Python tools and libraries like numpy, matplotlib, and pygame to do things like: –Generate Spirograph-like patterns using parametric equations and the turtle module –Create music on your computer by simulating frequency overtones –Translate graphical images into ASCII art –Write an autostereogram program that produces 3D images hidden beneath random patterns –Make realistic animations with OpenGL shaders by exploring particle systems, transparency, and billboarding techniques –Construct 3D visualizations using data from CT and MRI scans –Build a laser show that responds to music by hooking up your computer to an Arduino Programming shouldn't be a chore. Have some solid, geeky fun with Python Playground. The projects in this book are compatible with both Python 2 and 3.

Building a Second Brain

Eloquent JavaScript

Visual Basic and Algorithmic Thinking for the Complete Beginner

C# Programming for Absolute Beginners

Learn to Code by Solving Problems

How to Be a Programmer

HT THINK LIKE A COMPUTER SCIEN

Get started using the C# programming language. Based on the author's 15 years of experience teaching beginners, this book provides you with a step-by-step introduction to the principles of programming, or rather, how to think like a programmer. The task-solution approach will get you immersed, with minimum theory and maximum action. What You Will Learn Understand what programming is all about Write simple, but non-trivial, programs Become familiar with basic programming constructs such as statements, types, variables, conditions, and loops Think like a programmer and combine these programming constructs in new ways Get to know C# as a modern, mainstream programming language, and Visual Studio as one of the world's most popular programming tools Who This Book Is For Those with very little or no experience in computer programming, who know how to use a computer, install a program, and navigate the web

A Software Developer's Guide to Working Well with Others

Learn how hardware and software work– and how to make them work for you!

Python and Algorithmic Thinking for the Complete Beginner (2nd Edition)

The Rust Programming Language (Covers Rust 2018)

A Python Programming Primer

Can't Even

The Programmer's Brain