openstax™

Introduction to Computer Science

Introduction to Computer Science

SENIOR CONTRIBUTING AUTHOR

Dr. Jean-Claude Franchitti, NYU Courant Institute

openstax™

OpenStax
Rice University
6100 Main Street MS-375
Houston, Texas 77005

To learn more about OpenStax, visit https://openstax.org.
Individual print copies and bulk orders can be purchased through our website.

COLOR PAPERBACK BOOK ISBN-13	**978-1-711471-83-9**
B&W PAPERBACK BOOK ISBN-13	**978-1-711471-82-2**
DIGITAL VERSION ISBN-13	**978-1-961584-58-7**
ORIGINAL PUBLICATION YEAR	**2024**

1 2 3 4 5 6 7 8 9 10 CJP 24

OpenStax

OpenStax provides free, peer-reviewed, openly licensed textbooks for introductory college and Advanced Placement® courses and low-cost, personalized courseware that helps students learn. A nonprofit ed tech initiative based at Rice University, we're committed to helping students access the tools they need to complete their courses and meet their educational goals.

Rice University

OpenStax is an initiative of Rice University. As a leading research university with a distinctive commitment to undergraduate education, Rice University aspires to path-breaking research, unsurpassed teaching, and contributions to the betterment of our world. It seeks to fulfill this mission by cultivating a diverse community of learning and discovery that produces leaders across the spectrum of human endeavor.

Philanthropic Support

OpenStax is grateful for the generous philanthropic partners who advance our mission to improve educational access and learning for everyone. To see the impact of our supporter community and our most updated list of partners, please visit openstax.org/foundation.

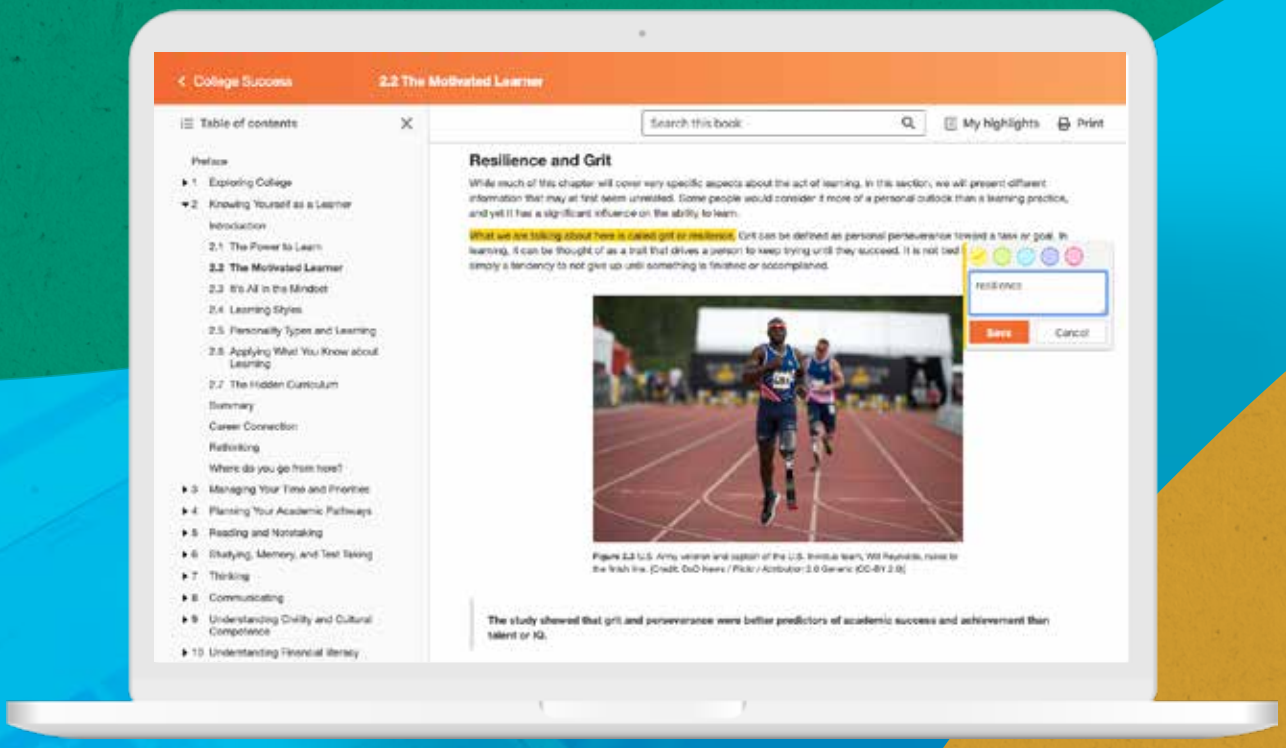
Arnold Ventures
Chan Zuckerberg Initiative
Chegg, Inc.
Arthur and Carlyse Ciocca Charitable Foundation
Digital Promise
Ann and John Doerr
Bill & Melinda Gates Foundation
Girard Foundation
Google Inc.
The William and Flora Hewlett Foundation
The Hewlett-Packard Company
Intel Inc.
Rusty and John Jaggers
The Calvin K. Kazanjian Economics Foundation
Charles Koch Foundation
Leon Lowenstein Foundation, Inc.
The Maxfield Foundation
Burt and Deedee McMurtry
Michelson 20MM Foundation
National Science Foundation
The Open Society Foundations
Jumee Yhu and David E. Park III
Brian D. Patterson USA-International Foundation
The Bill and Stephanie Sick Fund
Steven L. Smith & Diana T. Go
Stand Together
Robin and Sandy Stuart Foundation
The Stuart Family Foundation
Tammy and Guillermo Treviño
Valhalla Charitable Foundation
White Star Education Foundation
Schmidt Futures
William Marsh Rice University

openstax™

Study where you want, what you want, when you want.

When you access your book in our web view, you can use our new online **highlighting and note-taking** features to create your own study guides.

Our books are free and flexible, forever.
Get started at **openstax.org/details/books/introduction-computer-science**



Access. The future of education.
openstax.org

RICE

CONTENTS

Preface 1

PART 1 PROBLEM SOLVING AND ALGORITHMS

1 Introduction to Computer Science 9

Introduction 9
1.1 Computer Science 10
1.2 Computer Science across the Disciplines 20
1.3 Computer Science and the Future of Society 25
Chapter Review 33

2 Computational Thinking and Design Reusability 39

Introduction 39
2.1 Computational Thinking 40
2.2 Architecting Solutions with Adaptive Design Reuse in Mind 53
2.3 Evolving Architectures into Useable Products 73
Chapter Review 83

3 Data Structures and Algorithms 91

Introduction 91
3.1 Introduction to Data Structures and Algorithms 91
3.2 Algorithm Design and Discovery 100
3.3 Formal Properties of Algorithms 107
3.4 Algorithmic Paradigms 113
3.5 Sample Algorithms by Problem 119
3.6 Computer Science Theory 127
Chapter Review 131

PART 2 REALIZATIONS OF ALGORITHMS

4 Linguistic Realization of Algorithms: Low-Level Programming Languages 145

Introduction 145
4.1 Models of Computation 146
4.2 Building C Programs 158
4.3 Parallel Programming Models 175
4.4 Applications of Programming Models 181
Chapter Review 184

5 Hardware Realizations of Algorithms: Computer Systems Design 195

Introduction 195

5.1 Computer Systems Organization 196
5.2 Computer Levels of Abstraction 200
5.3 Machine-Level Information Representation 208
5.4 Machine-Level Program Representation 214
5.5 Memory Hierarchy 221
5.6 Processor Architectures 230
Chapter Review 234

6 Infrastructure Abstraction Layer: Operating Systems 243

Introduction 243
6.1 What Is an Operating System? 244
6.2 Fundamental OS Concepts 250
6.3 Processes and Concurrency 262
6.4 Memory Management 272
6.5 File Systems 279
6.6 Reliability and Security 284
Chapter Review 290

PART 3 DESIGNING AND DEVELOPING SOFTWARE SOLUTIONS

7 High-Level Programming Languages 303

Introduction 303
7.1 Programming Language Foundations 304
7.2 Programming Language Constructs 317
7.3 Alternative Programming Models 337
7.4 Programming Language Implementation 346
Chapter Review 353

8 Data Management 365

Introduction 365
8.1 Data Management Focus 366
8.2 Data Management Systems 372
8.3 Relational Database Management Systems 378
8.4 Nonrelational Database Management Systems 396
8.5 Data Warehousing, Data Lakes, and Business Intelligence 403
8.6 Data Management for Shallow and Deep Learning Applications 408
8.7 Informatics and Data Management 418
Chapter Review 420

9 Software Engineering 435

Introduction 435
9.1 Software Engineering Fundamentals 436
9.2 Software Engineering Process 446

9.3 Special Topics 474
Chapter Review 496

10 Enterprise and Solution Architectures Management 507

Introduction 507
10.1 Patterns Management 508
10.2 Enterprise Architecture Management Frameworks 516
10.3 Solution Architecture Management 551
Chapter Review 556

PART 4 BUILDING MODERN END-TO-END SOLUTIONS TO BUSINESS AND SOCIAL PROBLEMS

11 Web Applications Development 565

Introduction 565
11.1 Modern Web Applications Architectures 566
11.2 Sample Responsive WAD with Bootstrap and Django 584
11.3 Sample Responsive WAD with Bootstrap/React and Node 610
11.4 Sample Responsive WAD with Bootstrap/React and Django 625
11.5 Sample Native WAD with React Native and Node or Django 630
11.6 Sample Ethereum Blockchain Web 2.0/Web 3.0 Application 643
Chapter Review 656

12 Cloud-Native Applications Development 665

Introduction 665
12.1 Introduction to Cloud-Native Applications 666
12.2 Cloud-Based and Cloud-Native Applications Deployment Technologies 690
12.3 Example PaaS and FaaS Deployments of Cloud-Native Applications 711
Chapter Review 750

13 Hybrid Multicloud Digital Solutions Development 761

Introduction 761
13.1 Hybrid Multicloud Solutions and Cloud Mashups 762
13.2 Big Cloud IaaS Mainstream Capabilities 766
13.3 Big Cloud PaaS Mainstream Capabilities 774
13.4 Towards Intelligent Autonomous Networked Super Systems 790
Chapter Review 805

PART 5 HUMAN-CENTERED RESPONSIBLE COMPUTING

14 Cyber Resources Qualities and Cyber Computing Governance 817

Introduction 817
14.1 Cyber Resources Management Frameworks 818
14.2 Cybersecurity Deep Dive 839

14.3 Governing the Use of Cyber Resources 899
Chapter Review 906

A **Appendix A: Network Design Application of Algorithms 917**

Index 923

Preface

About OpenStax

OpenStax is part of Rice University, which is a 501(c)(3) nonprofit charitable corporation. As an educational initiative, it's our mission to improve educational access and learning for everyone. Through our partnerships with philanthropic organizations and our alliance with other educational resource companies, we're breaking down the most common barriers to learning. Because we believe that everyone should and can have access to knowledge.

About OpenStax Resources

Customization

Because our books are openly licensed, you are free to use the entire book or select only the sections that are most relevant to the needs of your course. Feel free to remix the content by assigning your students certain chapters and sections in your syllabus, in the order that you prefer. You can even provide a direct link in your syllabus to the sections in the web view of your book.

Instructors also have the option of creating a customized version of their OpenStax book. Visit the Instructor Resources section of your book page on OpenStax.org for more information.

Art Attribution

In *Introduction to Computer Science,* art contains attribution to its title, creator or rights holder, host platform, and license within the caption. Because the art is openly licensed, anyone may reuse the art as long as they provide the same attribution to its original source.

Errata

All OpenStax textbooks undergo a rigorous review process. However, like any professional-grade textbook, errors sometimes occur. In addition, the wide range of evidence, standards, practices, data, and legal circumstances in computer science change frequently, and portions of the text may become out of date. Since our books are web-based, we can make updates periodically when deemed pedagogically necessary. If you have a correction to suggest, submit it through the link on your book page on OpenStax.org. Subject matter experts review all errata suggestions. OpenStax is committed to remaining transparent about all updates, so you will also find a list of past and pending errata changes on your book page on OpenStax.org.

Format

You can access this textbook for free in web view or PDF through OpenStax.org, and for a low cost in print. The web view is the recommended format because it is the most accessible—including being WCAG 2.2 AA compliant – and most current. Print versions are available for individual purchase, or they may be ordered through your campus bookstore.

About *Introduction to Computer Science*

Introduction to Computer Science provides a comprehensive foundation in core computer science concepts and principles, aligning with the scope and sequence of most introductory computer science courses. The textbook serves as an engaging entry point for students pursuing diverse fields of study and employment, including computer science, business, engineering, data science, social sciences, and related disciplines. By addressing a broad learner audience—ranging from computer science majors to non-majors—the book offers a thorough introduction to computational thinking and its applications across multiple domains.

Introduction to Computer Science is designed to be both interactive and practical, focusing on real-world applications that showcase how core computer science concepts can be used to solve complex problems. Students will explore foundational topics, such as algorithms, data structures, computer systems organization, and software development, using an array of engaging, hands-on activities. The textbook integrates meaningful learning experiences through chapter-based scenarios, problem-solving exercises, and project-based assessments that encourage students to apply what they learn in authentic contexts.

Features such as embedded coding exercises, industry insights, and explorations of emerging technology trends provide a holistic approach to learning that extends beyond theory. With a forward-looking perspective, Introduction to Computer Science prepares students to engage with advanced topics in computer science, such as machine learning, cybersecurity, and cloud computing, ensuring they have a solid foundation for continued study and future professional success.

Coverage and Scope

The authors and contributors consulted with other educators and industry professionals from a range of institutions and organizations in order to ensure that *Introduction to Computer Science* meets the diverse needs of both computer science majors and non-majors. The book is structured into five main parts, each focusing on critical areas of the discipline:

- **Part 1: Problem Solving and Algorithms** This section introduces students to the foundations of computer science, focusing on computational thinking, problem-solving techniques, and algorithm design. Topics include data structures, formal properties of algorithms, and algorithmic paradigms. Through practical examples and exercises, students will develop the skills needed to construct and analyze algorithms and understand their applications across various domains.
- **Part 2: Realizations of Algorithms** In this part, students explore how algorithms are realized in hardware and software, starting with low-level programming languages and moving into hardware design and computer systems organization. Students will learn about models of computation, machine-level representation, processor architectures, and memory hierarchy. This foundational understanding enables students to see the connections between abstract algorithms and their physical implementations.
- **Part 3: Designing and Developing Software Solutions** This section covers the principles of software development, high-level programming languages, and data management. Students will learn the fundamentals of software engineering and gain hands-on experience with both relational and non-relational database systems. Emphasis is placed on designing robust software solutions and managing complex data structures, ensuring students are well-prepared for future roles in software development and engineering.
- **Part 4: Building Modern End-to-End Solutions to Business and Social Problems** Students apply their knowledge to design and build web and cloud-native applications. This part includes examples of modern web architectures, responsive design techniques, and cloud-based solutions using PaaS and FaaS technologies. Additionally, students will explore the development of hybrid multi-cloud digital solutions, providing them with experience in addressing complex business and social challenges using modern computing technologies.
- **Part 5: Human-Centered Responsible Computing** The final section delves into the ethical and societal implications of computing. Topics include cybersecurity, governance of cyber resources, and responsible computing practices. Students will learn to navigate the complexities of cybersecurity and governance while considering the broader impacts of technology on society.

Each core concept is designed to build on the previous one, ensuring a coherent learning experience that provides students with a clear view of the field. The book's approach enables students to not only understand the principles of computer science but also see how they can be applied to address practical, real-world problems.

Pedagogical Foundation and Features

The *Introduction to Computer Science* textbook is designed to engage students through a combination of practical, real-world applications and thought-provoking scenarios that promote critical thinking and a deeper understanding of core concepts. The pedagogical approach is centered on making computer science relevant and accessible for students from diverse backgrounds, whether they are pursuing a degree in computer science or exploring how computational thinking can be applied to their respective fields. To support this vision, the textbook incorporates several key features:

- **Concepts in Practice** features present how computer science concepts are applied in real-world contexts by both professionals and non-professionals. Each box profiles personas and practical applications that demonstrate how core topics, such as algorithms, data management, and software engineering, are utilized across various industries. The purpose is to inspire students—particularly non-majors—by showing them the value of computer science in solving everyday challenges and to foster a greater appreciation for the discipline.
- **Global Issues in Technology** features help students think globally about the societal impact of technology. These boxes highlight how technology affects communities and economies around the world and may introduce topics such as digital equity, environmental sustainability, and global data security. Students are encouraged to consider the broader implications of technological advancements and to think critically about their potential to drive positive change or create new challenges in global contexts.
- **Industry Spotlight** boxes focus on specific industry challenges and how technology progress or application can help solve them. *Industry Spotlight* features introduce students to various sectors—such as healthcare, finance, education, and law—providing a glimpse into how computer science can drive innovation and efficiency. By connecting theoretical concepts to industry-specific problems, these features encourage students to explore the wide-ranging applications of computer science and understand its value across different fields.
- **Link to Learning** features provide a very brief introduction to online resources—videos, interactives, collections, maps, and other engaging resources that are pertinent to students' exploration of the topic at hand.
- **Technology in Everyday Life** features connect computer science principles to students' personal experiences and the world around them. These boxes explore how technology intersects with daily life or current events, making computer science concepts more relatable and relevant. Some features may prompt students to think creatively and propose their own ideas for applying computer science solutions to familiar scenarios.
- **Think It Through** scenarios present students with thought-provoking dilemmas or complex problems related to the use of technology. Students are asked to reflect on ethical questions, problem-solving strategies, and real-world decision-making processes. These features emphasize that not all problems have straightforward answers and encourage students to weigh the pros and cons of different approaches. By navigating these scenarios, students learn to develop judgment skills that are crucial in business and technology environments.

Overall, these features are integrated throughout the material to foster active learning, critical thinking, and an appreciation for the practical applications of computer science. By connecting theory to practice and encouraging students to explore real-world issues, *Introduction to Computer Science* provides a meaningful and supportive learning experience that equips students with the knowledge and skills necessary for success in their academic and professional journeys.

Answers to Questions in the Book

The end-of-chapter Review, Conceptual Questions, Practice Exercises, Problem Sets, Thought Provokers, and Labs are intended for homework assignments or classroom discussion; thus, student-facing answers are not provided in the book. Answers and sample answers are provided in the Instructor Answer Guide, for

instructors to share with students at their discretion, as is standard for such resources.

About the Author

Senior Contributing Author

Senior contributing author: Dr. Jean-Claude Franchitti

Dr. Jean-Claude Franchitti is a Clinical Associate Professor of Computer Science and the Associate Director of Graduate Studies for the CS Master's program in Information Systems at NYU Courant Institute. He earned his M.S. in Electrical Engineering (1985) and his M.S. and PhD. in Computer Science from the University of Colorado at Boulder (1988, 1993). He is the founder and CEO of Archemy, Inc., and has over 40 years of experience in a myriad of industries, and over 30 years of teaching and corporate training experience. He held executive positions in large US-based corporations and leading business technology consulting firms such as Computer Sciences Corporation. He has been involved in many large business technology strategy and modernization projects and has a proven record of delivering large scale business solutions. He was the original designer and developer of jcrew.com and the suite of products now known as IBM InfoSphere DataStage. He also created the Agile Enterprise Architecture Management (AEAM) methodology and a corresponding framework that are patented components of the Archemy business evolution platform. He has developed partnerships with many companies at New York University to incubate new methodologies, transition research into business solutions, help recruit skilled graduates, and increase the companies' visibility. Dr. Franchitti has been a reviewer member on several industry standards committees including OMG, ODMG, and X3H2. Dr. Franchitti taught at CU-Boulder, Denver University, Columbia University, NYU SCPS, before joining NYU Courant Institute in 1997. He has extensive experience with corporate training and developed and delivered training and mentoring programs for the top corporate education providers. He conducted research as part of several NSF- and DARPA-funded research programs.

Dr. Franchitti's teaching and research interests include machine learning, artificial intelligence, data management systems and software engineering with an emphasis on large-scale software architectures and business solutions. He has published articles in numerous refereed publications including the Proceedings of Third Int. Conf. on Cooperative Information Systems, Proceedings of the Sixth International Workshop on Persistent Object Systems, and the 16th International Conference on Software Engineering. Dr. Franchitti received an award for Outstanding Service from NYU's School of Continuing and Professional Studies.

Contributing Authors

Amal Alhosban, University of Michigan–Flint

Mark Buckler, Grand Canyon University

Joanna Gilberti, Archemy, Inc.

Scott Gray, Nashua Community College

Matthew Hertz, University at Buffalo

Andrew Hurd, Empire State University

Kevin Lin, University of Washington

Sai Mukkavilli, Georgia Southwestern State University

Phuc (Brian) Nguyen, UC Irvine

Shahab Tayeb, Fresno State

Zdeněk Troníček, Tarleton State University

Kevin Wortman, Cal State Fullerton

Mohamed Zahran, New York University

Reviewers

Reni Abraham, Houston Community College

Shakil Akhtar, Clayton State University

Kiavash Bahreini, Florida International University

Tammie Bolling, Pellissippi State Community College

Phillip Bradford, UConn

Quiana Bradshaw, Campbellsville University

Christopher Bunton, Austin Community College

Komal Chhibber, South Mountain Community College

Chen-Fu Chiang, SUNY Polytechnic Institute

Gabriel de la Cruz, North Idaho College

Gabriel Ferrer, Hendrix College

David Fogarty, New York University

Yuexing Hao, Massachusetts Institute of Technology

Nazli Hardy, Millersville University

Angela Heath, Baptist Health System

Rania Hodhod, Columbus State University

Benita Hubbard, Southern New Hampshire University

Sumit Jha, Florida International University

Mohamed E. Khalefa, SUNY Old Westbury

Steven Ko, Simon Fraser University

Jessica Kong, University of Washington

Alex Krasnok, Florida International University

Blaise Liffick, Millersville University

Sue McCrory, Missouri State University

Morgan McKie, Florida International University

Sandeep Mitra, SUNY Brockport

Mourya Narasareddygari, Rider University

Saty Raghavachary, University of Southern California

Muhammad Rahman, Clayton State University

Amir Rahmati, Stony Brook University

Caryl Rahn, Florida International University

Jerry Reed, Valencia College

Jordan Ringenberg, University of Findlay

Eman Saleh, University of Georgia

Vincent Sanchez, Florida International University

John Schriner, Queensborough Community College

Tiffanie R. Smith, Lincoln University

Hann So, De Anza College

Jayesh Soni, Florida International University

Derrick Stevens, Mohawk Valley Community College

Kathleen Tamerlano, Cuyahoga Community College

Chintan Thakkar, Rasmussen University

Jingnan Xie, Millersville University

Ning Xie, Florida International University

Additional Resources

Student and Instructor Resources

We have compiled additional resources for both students and instructors, including Getting Started Guides, an instructor's answer guide, test bank, and image slides. Instructor resources require a verified instructor account, which you can apply for when you log in or create your account on OpenStax.org. Take advantage of these resources to supplement your OpenStax book.

Instructor's answer guide. Each component of the instructor's guide is designed to provide maximum guidance for delivering the content in an interesting and dynamic manner.

Test bank. With hundreds of assessment items, instructors can customize tests to support a variety of course objectives. The test bank includes review questions (multiple-choice, identification, fill-in-the-blank, true/false), short answer questions, and long answer questions to assess students on a variety of levels. The test bank is available in Word format.

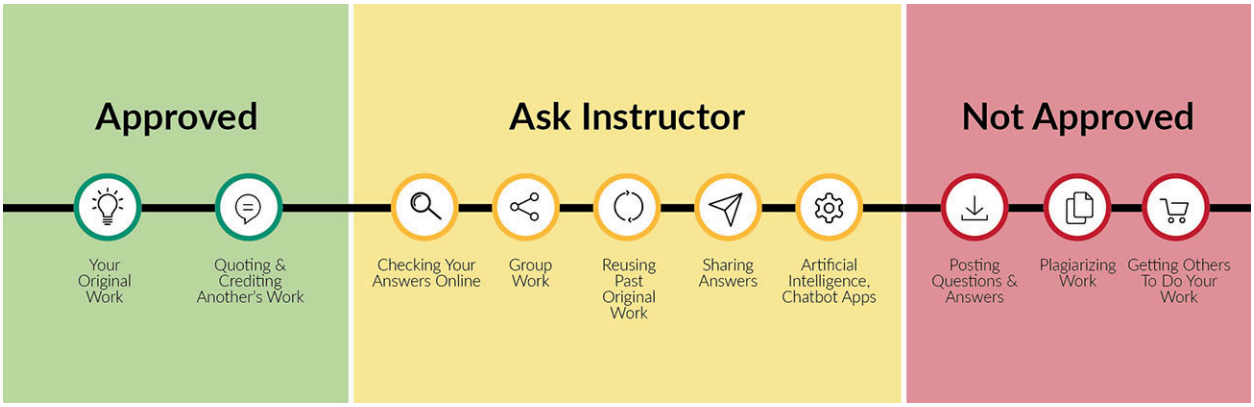
PowerPoint lecture slides. The PowerPoint slides provide learning objectives, images and descriptions, feature focuses, and discussion questions as a starting place for instructors to build their lectures.

Academic Integrity

Academic integrity builds trust, understanding, equity, and genuine learning. While students may encounter significant challenges in their courses and their lives, doing their own work and maintaining a high degree of authenticity will result in meaningful outcomes that will extend far beyond their college career. Faculty, administrators, resource providers, and students should work together to maintain a fair and positive experience.

We realize that students benefit when academic integrity ground rules are established early in the course. To

that end, OpenStax has created an interactive to aid with academic integrity discussions in your course.



Visit our academic integrity slider (https://view.genial.ly/61e08a7af6db870d591078c1/interactive-image-defining-academic-integrity-interactive-slider). Click and drag icons along the continuum to align these practices with your institution and course policies. You may then include the graphic on your syllabus, present it in your first course meeting, or create a handout for students. (attribution: Copyright Rice University, OpenStax, under CC BY 4.0 license)

At OpenStax we are also developing resources supporting authentic learning experiences and assessment. Please visit this book's page for updates. For an in-depth review of academic integrity strategies, we highly recommend visiting the International Center of Academic Integrity (ICAI) website at https://academicintegrity.org/ (https://academicintegrity.org/).

Community Hubs

OpenStax partners with the Institute for the Study of Knowledge Management in Education (ISKME) to offer Community Hubs on OER Commons—a platform for instructors to share community-created resources that support OpenStax books, free of charge. Through our Community Hubs, instructors can upload their own materials or download resources to use in their own courses, including additional ancillaries, teaching material, multimedia, and relevant course content. We encourage instructors to join the hubs for the subjects most relevant to your teaching and research as an opportunity both to enrich your courses and to engage with other faculty. To reach the Community Hubs, visit www.oercommons.org/hubs/openstax.

Technology partners

As allies in making high-quality learning materials accessible, our technology partners offer optional low-cost tools that are integrated with OpenStax books. To access the technology options for your text, visit your book page on OpenStax.org.

1 Introduction to Computer Science

Figure 1.1 Computing is everywhere, affecting everyone, for better and for worse. (credit: modification of "Whereas design is expansive, engineering is narrowing" by Jessie Huynh/Critically Conscious Computing, CC0)

Chapter Outline

1.1 Computer Science
1.2 Computer Science across the Disciplines
1.3 Computer Science and the Future of Society

Introduction

This textbook will introduce you to the exciting and complex world of computer science. In this chapter, you'll review the history of computer science, learn about its use in different fields, and explore how computer science will impact the future of society. Computer science is a powerful tool, and computer scientists have used their vast knowledge of technology to create and implement technology that has transformed societies around the world.

This book will also introduce the computational thinking aspects of problem-solving and analytical thinking that enable the study of algorithms, which are step-by-step instructions for solving specific problems or carrying out computations. Therefore, this book also covers algorithms and their realization via programming languages, computer systems architectures, networks, and operating systems. The book subsequently delves into computer science areas that enable the design and development of software solutions using high-level programming languages (i.e., coding languages designed to be more intuitive for humans), architectural styles and related models, data management systems, and software engineering. Finally, the book demonstrates how to leverage computer science realizations and areas to build modern end-to-end solutions to business and social problems. In particular, the book focuses on modern web applications development, cloud-native applications development, and hybrid Cloud/on-premise digital solutions. The various chapters emphasize how to achieve software solution qualities such as performance and scalability. The last chapter explains how to secure software applications and their applications in the context of various cyber threats. It also explains how to make the right decisions about using computers and information in society to navigate social, ethical, economic, and political issues that could result from the misuse of technology. To conclude this textbook, we'll introduce you to cybersecurity and help you understand why responsible computing is essential to promote

ethical behavior in computer science. The book is designed to help students grasp the full meaning of computer science as a tool that can help them think, build meaningful solutions to complex problems, and motivate their careers in information technology (IT).

You're already familiar with computer science. Whenever you use a laptop, tablet, cell phone, credit card reader, and other technology, you interact with items made possible by computer science. Computer science is a challenging field, and the outputs of computer science offer many benefits for society. At the same time, we have to be cautious about how we use computer science to ensure it impacts society in ethical ways. To help you understand this, the next section will explain how computer science came to be and discuss the field's potential.

1.1 Computer Science

Learning Objectives

By the end of this section, you will be able to:

- Discuss the history that led to the creation of computer science as a field
- Define computer science
- Assess what computer science can do, as well as what it should not do

The field of **computer science (CS)** is the study of **computing**, which includes all phenomena related to computers, such as the Internet. With foundations in engineering and mathematics, computer science focuses on studying algorithms. An **algorithm** is a sequence of precise instructions that enables computing. This includes components computers use to process information. By studying and applying algorithms, computer science creates applications and solutions that impact all areas of society. For example, computer science developed the programs that enable online shopping, texting with friends, streaming music, and other technological processes.

While computers are common today, they weren't always this pervasive. For those whose lives have been shaped by computer technology, it can sometimes seem like computer technology is ahistorical: computing often focuses on rapid innovation and improvement, wasting no time looking back and reflecting on the past. Yet the foundations of computer science defined over 50, and as much as 100, years ago very much shape what is possible with computing today.

The Early History of Computing

The first computing devices were not at all like the computers we know today. They were physical calculation devices such as the abacus, which first appeared in many societies across the world thousands of years ago. They allowed people to tally, count, or add numbers (Figure 1.2). Today, abaci are still used in some situations, such as helping small children learn basic arithmetic, keeping score in games, and as a calculating tool for people with visual impairments. However, abaci are not common today because of the invention of number systems such as the Arabic number system (0, 1, 2, 3, . . .), which included zero and place values that cannot be computed with abaci. The concept of an algorithm was also invented around this time. Algorithms use inputs and a finite number of steps to carry out arithmetic operations like addition, subtraction, multiplication, and division, and produce outputs used in computing. Today's computers still rely on the same foundations of numbers, calculations, and algorithms, except at the scale of billions of numbers and billions of calculations per second.

To introduce a concrete example of an algorithm, let us consider binary search algorithm, which is used to locate a number in a sorted array of integers efficiently. The algorithm operates by repeatedly dividing the search interval in half to perform the search. If the number being searched is less than the integer in the middle of the interval, the interval is narrowed to the lower half. In the alternative, the interval is narrowed to the upper half. The algorithm repeatedly checks until the number is found or the interval is empty.