2025 Exploring Computer Science

Program CIP: 11.0101 — Computer and Information Sciences – General

Direct inquiries to:

Project Manager
Research and Curriculum Unit
P.O. Drawer DX
Mississippi State, MS 39762
662.325.2510
helpdesk@rcu.msstate.edu

Program Supervisor
Office of Career and Technical Education
Mississippi Department of Education
P.O. Box 771
Jackson, MS 39205
601.359.3974

Published by:

Office of Career and Technical Education
Mississippi Department of Education
Jackson, MS 39205

Research and Curriculum Unit
Mississippi State University
Mississippi State, MS 39762

The Research and Curriculum Unit (RCU), located in Starkville, as part of Mississippi State University (MSU), was established to foster educational enhancements and innovations. In keeping with the land-grant mission of MSU, the RCU is dedicated to improving the quality of life for Mississippians. The RCU enhances the intellectual and professional development of Mississippi students and educators while applying knowledge and educational research to the lives of the people of the state. The RCU works within the contexts of curriculum development and revision, research, assessment, professional development, and industrial training.

Table of Contents

Acknowledgments

The Exploring Computer Science was presented to the Mississippi State Board of Education on January 16, 2025. The following persons were serving on the state board at the time:

Dr. Lance Evans, State Superintendent of Education, Executive Secretary
Mr. Glen V. East, Chair
Mr. Matt Miller, Vice-Chair
Dr. Ronnie L. McGehee
Mr. Bill Jacobs
Mr. Mike Pruitt
Ms. Mary Werner
Dr. Wendi Barrett
Ms. Billye Jean Stroud
Mr. Matt Mayo
Ms. Kate Riddle, Student Representative
Mr. Crosby Parker, Student Representative

The following Mississippi Department of Education (MDE) and RCU managers and specialists assisted in the development of Exploring Computer Science:

Brett Robinson, the associate state superintendent of the MDE Office of Career and Technical Education (CTE) and Workforce Development, supported the RCU and teachers throughout the development of the framework and supporting materials.
Dr. Louella Mack-Webster, the Multimedia program supervisor of the MDE Office of CTE, supported the RCU and teachers throughout the development of the framework and supporting materials.
Betsey Smith, the director of the RCU, supported RCU staff and teachers throughout the development of this framework and supporting materials.
Courtney McCubbins, the curriculum and assessment manager of the RCU, supported RCU staff and teachers throughout the development of this framework and supporting materials.
Shelly Hollis, assistant director for the Center for Cyber Education at MSU (CCE), supported the RCU and teachers throughout the development of the framework and supporting materials.
Kyle McDill, a project manager with the RCU, researched and co-authored this framework.

Special thanks are extended to the educators who contributed to the development and revision of this framework and supporting materials:

Kessie Key, Rankin County School District, Brandon, MS
Jana Odom, Pascagoula-Gautier School District, Pascagoula, MS
Keelandra Holden, Clarksdale Municipal School District, Clarksdale. MS
Shelly Thompson, Center for Cyber Education, Starkville, MS
Amanda Taylor, Center for Cyber Education, Starkville, MS

Appreciation is expressed to the following professionals who provided guidance and insight throughout the development process:

Jeremy Dowell, Lobaki, Jackson, MS
Micheal Peacock, Lobaki, Jackson, MS

Standards

Standards and alignment crosswalks are referenced in the appendices. Depending on the curriculum, these crosswalks should identify alignment to the standards mentioned below, as well as possible related academic topics as required in the Subject Area Testing Program in Algebra I, Biology I, English II, and U.S. History from 1877, which could be integrated into the content of the units. Mississippi's CTE Exploring Computer Science is aligned to the following standards:

2018 Mississippi College- and Career-Readiness Standards for Computer Science
In an effort to closely align instruction for students who are progressing toward postsecondary study and the workforce, the *2018 Mississippi College- and Career-Readiness Standards (MS CCRS) for Computer Science* includes grade- and course-specific standards for K-12 computer science. Mississippi has adapted these standards from the nationally developed *Computer Science Teachers Association K-12 Computer Science Standards* (Revised 2017).
mdek12.org/OAE/college-and-career-readiness-standards

International Society for Technology in Education Standards (ISTE)
Reprinted with permission from *ISTE Standards for Students* (2016).

College- and Career-Readiness Standards
College- and career-readiness standards emphasize critical thinking, teamwork, and problem-solving skills. Students will learn the skills and abilities demanded by the workforce of today and the future. Mississippi adopted Mississippi College- and Career-Readiness Standards (MCCRS) to provide a consistent, clear understanding of what students are expected to learn and so teachers and parents know what they need to do to help them.
mdek12.org/oae/college-and-career-readiness-standards

Framework for 21st Century Learning
In defining 21st-century learning, the Partnership for 21st Century Skills has embraced key themes and skill areas that represent the essential knowledge for the 21st-century: global awareness; financial, economic, business, and entrepreneurial literacy; civic literacy; health literacy; environmental literacy; learning and innovation skills; information, media, and technology skills; and life and career skills.
battelleforkids.org/networks/p21/frameworks-resources

Preface

Secondary CTE programs in Mississippi face many challenges resulting from sweeping educational reforms at the national and state levels. Schools and teachers are increasingly being held accountable for providing applied learning activities to every student in the classroom. This accountability is measured through increased requirements for mastery and attainment of competency as documented through both formative and summative assessments. This document provides information, tools, and solutions that will aid students, teachers, and schools in creating and implementing applied, interactive, and innovative lessons. Through best practices, alignment with national standards and certifications, community partnerships, and a hands-on, student-centered concept, educators will be able to truly engage students in meaningful and collaborative learning opportunities.

The courses in this document reflect the statutory requirements as found in Section 37-3-49, *Mississippi Code of 1972*, as amended (Section 37-3-46). In addition, this curriculum reflects guidelines imposed by federal and state mandates (Laws, 1988, Ch. 487, §14; Laws, 1991, Ch. 423, §1; Laws, 1992, Ch. 519, §4 eff. from and after July 1, 1992; Strengthening Career and Technical Education for the 21st Century Act, 2019 [Perkins V]; and Every Student Succeeds Act, 2015).

Mississippi Teacher Professional Resources

The following are resources for Mississippi teachers:

Curriculum, Assessment, Professional Learning

Program resources can be found at the RCU's website, rcu.msstate.edu.

Learning Management System: An Online Resource

Learning management system information can be found on the RCU's website, under Professional Learning.

Should you need additional instructions, contact the RCU at 662.325.2510 or helpdesk@rcu.msstate.edu.

Executive Summary

Pathway Description

Exploring Computer Science (ECS) is a survey course that introduces students to the breadth of the computer science field. The course lays a foundation in problem-solving, critical thinking, and algorithmic development and then introduces students to the basics of web development, programming, robotics, data science, and artificial intelligence. Emphasizing hands-on and project-based learning experiences, ECS prepares students for further study and careers in technology-related fields. Students emerge with a professional portfolio, demonstrating proficiency in creating engaging and practical computing solutions for today's technology-driven landscape.

Grade Level and Class Size Recommendations

It is recommended that students enter this program as freshmen. Exceptions to this are district-level decisions based on class size, enrollment numbers, student maturity, and CTE delivery method. This is a classroom-based course. Therefore, a maximum of 25 students is recommended for each class, and only one class with the teacher at a time.

Student Prerequisites

For students to experience success in the program, the following student prerequisites are suggested:

1. C or higher in English (the previous year)
2. C or higher in high school-level math (last course taken or the instructor can specify the level of math instruction needed)
3. Instructor approval and Test of Adult Basic Education (TABE) reading score (eighth grade or higher)

or

1. TABE reading and math score (eighth grade or higher)
2. Instructor approval

or

1. Instructor approval

Assessment

The latest assessment blueprint for the curriculum can be found at https://www.rcu.msstate.edu/curriculum.

Applied Academic Credit

The latest academic credit information can be found at mdek12.org/ese/approved-course-for-the-secondary-schools.

Teacher Licensure

The latest teacher licensure information can be found at https://mdek12.org/cte/licensuretoteach

Professional Learning

If you have specific questions about the content of any training sessions provided, please contact the RCU at 662-325-2510 or helpdesk@rcu.msstate.edu.

Course Outlines

This curriculum consists of one 1-credit course.

Class Name—Course Code: 000283

Unit	Unit Title	Hours
1	Orientation and Ongoing Skills	10
2	Human-Computer Interaction	20
3	Problem-Solving	20
4	Introduction to Programming	25
5	Data Science and Computing	20
6	Artificial Intelligence (AI)	20
7	Physical Computing (hours vary depending on selection)	15
8	Cybersecurity	10
Total		**140**

Career Pathway Outlook

Overview
Computer and information technology employment in the U.S. is projected to grow by 13 percent from 2024 to 2034, much faster than the average for all occupations. These occupations are projected to add about 667,600 new jobs. In Mississippi, employment trends in computer science and other information technology-related fields are expected to see similar or even higher growth due to increased demand for professionals with skills in areas such as cybersecurity, data science, and software development. Graduates of the Exploring Computer Science (ECS) program will be well-positioned to pursue careers in these high-demand fields.

Needs of the Future Workforce
The following data highlights key projected job opportunities in Mississippi from the U.S. Census Bureau, the U.S. Bureau of Labor Statistics (BLS), and the Mississippi Department of Employment Security (MDES).

Table 1.1: Current and Projected Occupation Report

Description	Jobs, 2020	Projected Jobs, 2030	Change (Number)	Change (Percent)	Average Hourly Earnings, 2024
Artificial Intelligence Specialists	350	620	270	77.1%	$46.15
Computer Network Architects	370	390	20	5.4%	$45.13
Computer Network Support Specialists	1,130	1,190	60	5.3%	$27.83
Computer Systems Analysts	2,120	2,190	70	3.3%	$42.19
Data Scientists	330	350	20	6.1%	$34.72
Database Administrators	260	260	0	0%	$39.55
Information Security Analysts	450	470	20	4.4%	$45.95
Network and Computer Systems Admins	1,440	1,500	60	4.2%	$38.96
Software Developers	2,860	2,980	120	4.2%	$43.24
Web Developers	200	210	10	5%	$31.34

Source: Mississippi Department of Employment Security; mdes.ms.gov (2024).

Perkins V Requirements and Academic Infusion
The Exploring Computer Science curriculum meets Perkins V requirements of introducing students to and preparing them for high-skill, high-wage occupations in computer science fields. It also offers students a program of study, including secondary and postsecondary education, and institutions of higher learning courses that will further prepare them for computer science careers. Additionally, this curriculum is integrated with academic college- and career-readiness

standards. Lastly, it focuses on ongoing and meaningful professional development for teachers as well as relationships with industry.

Transition to Postsecondary Education

The latest articulation information for secondary to postsecondary can be found at the Mississippi Community College Board website.

Best Practices

Innovative Instructional Technologies

Classrooms should be equipped with tools that will teach today's digital learners through applicable and modern practices. The goal of Exploring Computer Science educators should be to include teaching strategies that incorporate current technology. To make use of the latest online communication tools—wikis, blogs, podcasts, and social media platforms, for example—the classroom teacher is encouraged to use a learning management system that introduces students to education in an online environment and places more of the responsibility of learning on the student.

Differentiated Instruction

Students learn in a variety of ways, and numerous factors—students' background, emotional health, and circumstances, for example—create unique learners. By providing various teaching and assessment strategies, students with various learning preferences can have more opportunities to succeed.

CTE Student Organizations

Teachers should investigate opportunities to sponsor a student organization. There are several here in Mississippi that will foster the types of learning expected from the Exploring Computer Science curriculum. TSA and FBLA are examples of student organizations with many outlets for computer science. Student organizations provide participants and members with growth opportunities and competitive events. They also open the doors to the world of computer science careers and scholarship opportunities.

Cooperative Learning

Cooperative learning can help students understand topics when independent learning cannot. Therefore, you will see several opportunities in the Exploring Computer Science curriculum for group work. To function in today's workforce, students need to be able to work collaboratively with others and solve problems without excessive conflict. The Exploring Computer Science curriculum provides opportunities for students to work together and help each other complete complex tasks. There are many field experiences within the Exploring Computer Science curriculum that will allow and encourage collaboration with professionals currently in the computer science field.

Professional Organizations

For students:
Future Business Leaders of America
fbla-pbl.org

Technology Student Association
tsaweb.org

For teachers:

Association of Career and Technical Education
acteonline.org

Computer Science Teachers Association
csteachers.org

International Society for Technology in Education
iste.org

Mississippi Educational Computing Association
ms-meca.org

Mississippi Association of Career and Technical Education
mississippiacte.com

Using This Document

Competencies and Suggested Objectives
A competency represents a general concept or performance that students are expected to master as a requirement for satisfactorily completing a unit. Students will be expected to receive instruction on all competencies. The suggested objectives represent the enabling and supporting knowledge and performances that will indicate mastery of the competency at the course level.

Teacher Resources
All teachers should request to be added to the Canvas Resource Guide for their course. For questions or to be added to the guide, send a Help Desk ticket to the RCU by emailing helpdesk@rcu.msstate.edu.

Perkins V Quality Indicators and Enrichment Material
Some of the units may include an enrichment section at the end. This material will greatly enhance the learning experiences of students. If the Exploring Computer Science program is using a national certification, work-based learning, or another measure of accountability that aligns with Perkins V as a quality indicator, this material could very well be assessed on that quality indicator. It is the responsibility of the teacher to ensure all competencies for the selected quality indicator are covered throughout the year.

Unit 1: Orientation and Ongoing Skills

Competencies and Suggested Objectives
1. Identify school policies and safety procedures related to Exploring Computer Science (ECS). DOK1 a. Examine the school handbook, acceptable-use policy for technology, and safety procedures. b. Preview the course outline and its relevance in today's workforce. c. Recognize appropriate safety measures related to technology in the computer lab and online safety.
2. Investigate using an online learning management system (LMS). DOK2 a. Discover online learning environments and how they operate among teachers and students. b. Demonstrate proper email etiquette. c. Participate in online learning methods (discussion boards, student journals, blogs, wikis, and so forth). d. Collaborate with teachers and peers through an online system.
3. Recognize opportunities to participate in student organizations related to technology and computer science. DOK1 a. Identify student organizations available at the school for technology and computer science. b. List student competitions available through each organization.
4. Demonstrate knowledge of 21st-century skills. DOK2 a. Demonstrate effective collaboration and teamwork. b. Demonstrate creativity and imagination. c. Utilize critical thinking through effective reasoning, making judgments and decisions using journaling.
5. Demonstrate effective public speaking skills. DOK2 a. Demonstrate effective communication in groups. b. Demonstrate presentation skills.
6. Explore career opportunities within computer science in programming, cybersecurity, data science, robotics, artificial intelligence, human-computer interaction, and web development. DOK3 a. Identify and research career opportunities in programming, cybersecurity, data science, robotics, AI, and human-computer interaction. b. Describe how career fields use technology in their work. c. Examine the requirements, skills, wages, education, and employment opportunities in computer science career areas.
7. Create and maintain a personal portfolio website to showcase work and projects. DOK2

Note: Safety is to be taught as an ongoing part of the program. Students are required to complete a written safety test with 100% accuracy before entering the shop for lab simulations and projects. This test should be documented in each student's file.

Unit 2: Human-Computer Interaction

Competencies and Suggested Objectives
1. Explain the difference between computers and computing. DOK1 a. Identify characteristics of hardware components and their applications. b. Explain the four characteristics of a computer: input, output, processing, and storage. c. Explain the differences between tasks that can and cannot be accomplished with a computer.
2. Evaluate how the internet works and tools/methods used to navigate it. DOK3 a. Use appropriate tools and methods to execute internet searches. b. Evaluate the reliability of websites and AI responses. c. Define and give examples of the Internet of Things (IoT).
3. Analyze the effects of computing on society within economic, social, and cultural contexts. DOK3 a. Discuss legal, ethical, and security concerns raised by computing innovation. b. Explain the implications of communication as data exchange. • Recognize various forms of communication as data exchange. • Describe the implications of data exchange on social interactions. • Explain how computers are used for communications. • Compare and contrast privacy and access concerns between online versus in-person data exchanges. c. Identify web applications that influence society and education. d. Identify appropriate and inappropriate use of social websites.
4. Explain the basic concepts of cloud computing and its applications. DOK2 a. Define "the cloud" and its major attributes. b. Describe the major benefits of cloud computing.

Unit 3: Problem Solving

Competencies and Suggested Objectives
1. Understand the problem-solving process. DOK2 a. Name and explain the steps in the problem-solving process. b. Solve various problems using the problem-solving process and document each step.
2. Design and interpret algorithms. DOK3 a. Define an algorithm and determine its effectiveness. b. Determine if a given algorithm successfully solves a stated problem. c. Create algorithms that meet specified objectives. d. Summarize the behavior of an algorithm. e. Compare the tradeoffs between different algorithms for solving the same problem. f. Explain the characteristics of problems that an algorithm cannot solve.
3. Demonstrate an understanding of binary numbers. DOK2 a. Explain the connections between binary numbers and computers. b. Count forward and backward in binary. c. Use binary digits to code and decode messages.
4. Understand simple search algorithms. DOK2 a. Illustrate and explain linear and binary search algorithms. b. Explain conditions in which each search would be appropriate.
5. Explain sorting algorithms. DOK2 a. Define sorted and unsorted lists. b. Describe various sorting algorithms and compare them.
6. Describe minimum spanning trees. DOK3 a. Solve minimum spanning tree problems and provide real-world examples (e.g., power grids, gas lines). b. Explain how a minimum spanning tree relates to computer science networks.

Unit 4: Introduction to Programming

Competencies and Suggested Objectives
1. Use appropriate algorithms to solve a problem. DOK3 a. Write steps or flow diagrams to plan solutions to programming problems. b. Write code that is properly sequenced to solve problems.
2. Using a text-based language (i.e., Python, C++), design, code, test, and execute a program corresponding to a set of specifications. DOK2 a. Describe an event-driven program. b. Use industry constructs such as pseudocode and comments to draft a program. c. Apply programming structures such as variables, conditionals, loops, and input/output to create a program.
3. Using a text-based language, locate and correct errors in a program. DOK3 a. Deconstruct programs into smaller components to isolate problems. b. Identify and correct errors in a program written by a student and another by a peer (debug). c. Use AI to create and debug programs. d. Evaluate the reliability of AI in creating and debugging programs. e. Discuss the pros, cons, and ethics of using AI in programming. f. Evaluate a peer's program and provide constructive feedback on accuracy, efficiency, and readability.
4. Compare/contrast at least three programming languages. DOK3 a. Identify the best use of each language and their differences. b. Compare the syntax of each language for the following structures: variables, conditionals, loops, and input/output.
5. Use abstraction to reduce complexity. DOK2 a. Use abstraction to reduce complexity. b. Explain abstraction and provide examples in everyday life.

Unit 5: Data Science and Computing

Competencies and Suggested Objectives
1. Understand big data and its characteristics. a. Understand the complexities of collecting, processing, and managing large data sets. b. Identify real-world applications of big data in various fields. c. Utilize tools to analyze large data sets. d. Draw conclusions about the data set selected. e. Discuss methods of collecting and validating data. f. Collaborate with others to create artifacts (i.e. surveys). g. Understand and discuss data bias.
2. Identify and discuss the considerations that must be made for a large data set to be useful. a. Consider how various data types (numbers, text, dates, etc.) lend themselves to processing. b. Explain how different representations of data can tell different stories. c. Collaborate with others to create, manage, and maintain a large data set.
3. Understand the complexities of collecting, processing, and analyzing data sets. a. Identify the specific variables needed to analyze the data. b. Interpret data and draw conclusions to solve problems. c. Understand the problem-solving process.
4. Define data analytics. a. Discuss what data analytics might involve. b. Compare different analysis techniques and discuss the tradeoffs among them. c. Understand machine learning and how it works.

Unit 6: Artificial Intelligence (AI)

Competencies and Suggested Objectives
1. Summarize artificial intelligence (AI) terms and concepts. DOK1 a. Explain key terminology associated with AI, including weak AI, strong AI, generative AI, artificial general intelligence (AGI), rule-based AI, and context-aware AI. b. Develop an understanding of AI images and narratives. c. Explore the concept of prompt engineering in AI.
2. Explore AI tools and their impact. DOK 3 a. Identify the type of AI being used. (e.g., image recognition, speech recognition, translation, etc.) b. Test various prompts and describe the results. c. Understand how AI is changing different sectors. (e.g., medicine, agriculture, etc.) d. Explore and explain the impact of AI on our society. e. Recognize that future work is changing.
3. Recognize and understand AI data and bias. DOK3 a. Judge algorithmic bias and the effect of bias on individuals and society. b. Examine issues involving privacy and the collection of data.
4. Develop skills in prompt engineering for AI applications. DOK2 a. Understand the basics of prompt engineering and its importance in AI. b. Create effective prompts for AI models to achieve desired outcomes. c. Evaluate the effectiveness of different prompts and refine them for better results.

Choose two Unit 7 options relevant to the local program.

Unit 7a: Physical Computing—Robotics

Competencies and Suggested Objectives
1. Identify the criteria that describe a robot and determine if something is a robot. DOK1 a. Describe how the design of a robot's body affects its behavior. b. Identify the parts and features of a robot (motors, sensors, batteries, buttons/switches, etc.).
2. Build, code, and test a robot that solves a stated problem. DOK3 a. Navigate the programming environment to build and code a robot that performs specific tasks. b. Implement Boolean operators, loops, conditionals, and waits in robot programming to control behavior. c. Debug coding of the robot by testing, identifying, and fixing errors to ensure proper functionality.

Unit 7b: Physical Computing—Microcontrollers

Competencies and Suggested Objectives
1. Identify common microcontroller terms. DOK1 a. Communicate microcontroller terms using multiple formats (e.g., verbally, textually, graphically). b. Identify and label the components of a hands-on or simulation microcontroller from the list below: • Power sources • Inputs • Switches • Push buttons • Sensors • Joysticks and remotes • Outputs • Buzzers • LEDs • LCD modules
2. Use programming to manipulate microcontroller inputs and outputs. DOK2 a. Incorporate the following methods/concepts in the programming: • Different languages (e.g., Scratch, SNAP, Python, etc.) • Logic statements (e.g., if, and, or, not, etc.) • Loops (e.g., for, if, while, etc.)
3. Use a microcontroller for a specified purpose. DOK2 a. Demonstrate the proper use of a microcontroller for a specified purpose. b. Explain how microcontrollers are used to manipulate a robotic system.
4. Add microcontrollers and troubleshooting. DOK 3 a. Describe the use of microcontrollers in physical computing.

Unit 7c: Physical Computing—E-Textiles

Competencies and Suggested Objectives
1. Design, interpret, and evaluate simple circuit diagrams with key components. DOK2 a. Draw a circuit diagram with a simple circuit. b. Identify necessary components of a simple circuit system (power, components, polarity, etc.) c. Peer review one another's circuit diagrams according to a rubric. d. Create a paper greeting card with electronic components (including designing it, crafting it, debugging it, and sharing it with classmates).
2. Construct parallel circuits using conductive materials and switches while applying an iterative design process. DOK3 a. Use conductive thread to sew electronic components. b. Design and create a working parallel circuit with three lights. c. Demonstrate how a switch works to turn electricity flow on and off. d. Use an iterative design process.
3. Implement computational circuits with pre-programmed mini-computers and understand common grounding. DOK3 a. Design a computational circuit using a pre-programmed mini-computer. b. Understand the role of a common ground.

Unit 7d: Physical Computing—Networking

Competencies and Suggested Objectives
1. Explore fundamental networking concepts. DOK2 a. Define what a computer network is and explain its purpose. b. Identify different types of networks (LAN, WAN, WLAN) and their characteristics. c. Explain the concept of IP addresses and their role in networking. d. Describe the basic functions of networking devices such as servers, routers, and switches. e. Demonstrate the ability to set up a simple local network and connect devices on paper, in a virtual setting, or with actual hardware. f. Understand basic network security principles and identify common threats.

Unit 7e: Physical Computing—VR System

Competencies and Suggested Objectives
1. Investigate the process of developing virtual environments. DOK2 a. Create programs using virtual engines such as Unreal (visual) or Unity (text). b. Demonstrate knowledge of key terms such as levels, worlds, scenes, assets, viewport, camera, etc. c. Import assets for use in programming projects. d. Script assets to perform designated tasks within a project. e. Compile and publish completed projects for testing or as finished products.

Unit 8: Cybersecurity

Competencies and Suggested Objectives
1. Define cybersecurity and its importance. DOK 1 a. Differentiate between attackers and defenders. b. Describe types of hacking and the CIA triad (Confidentiality, Integrity, Availability). c. Discuss ethics in cybersecurity.
2. Discuss confidentiality in cybersecurity. DOK 2 a. Explain encryption, passwords, and multifactor authentication. b. Analyze case studies of data breaches.
3. Explain integrity and hashing. DOK 2 a. Describe hashing and its role in maintaining data integrity.
4. Discuss availability considerations. DOK 3 a. Explain backups, DoS (denial-of-service) attacks, disaster plans, and redundancy.
5. Analyze adversary thinking and social engineering. DOK 4 a. Evaluate phishing, insider/outsider threats, and ransomware tactics.

Student Competency Profile

Student's Name: ____

This record is intended to serve as a method of noting student achievement of the competencies in each unit. It can be duplicated for each student, and it can serve as a cumulative record of competencies achieved in the course.

In the blank before each competency, place the date on which the student mastered the competency.

Unit 1: Orientation and Ongoing Skills		
	1.	Identify school policies and safety procedures related to Exploring Computer Science (ECS).
	2.	Investigate using an online learning management system.
	3.	Recognize opportunities to participate in student organizations related to technology and computer science.
	4.	Demonstrate knowledge of 21st-century skills.
	5.	Demonstrate effective public speaking skills.
	6.	Explore career opportunities within computer science in programming, cybersecurity, data science, robotics, artificial intelligence, human-computer interaction, and Web development.
	7.	Create and maintain a personal portfolio website to showcase work and projects.
Unit 2: Human-Computer Interaction		
	1.	Explain the difference between computers and computing.
	2.	Evaluate the results of web searches and the reliability of information found on the internet.
	3.	Analyze the effects of computing on society within economic, social, and cultural contexts.
	4.	Describe features of appropriate data sets for specific problems.
Unit 3: Problem Solving		
	1.	Understand the problem-solving process.
	2.	Design and interpret algorithms.
	3.	Demonstrate an understanding of binary numbers.
	4.	Understand simple search algorithms.
	5.	Explain sorting algorithms.
	6.	Describe minimum spanning trees.
Unit 4: Introduction to Programming		
	1.	Use appropriate algorithms to solve a problem.

	2.	Explain an event-driven program.
	3.	Locate and correct errors in a program.
	4.	Use appropriate programming structures and troubleshooting techniques.
	5.	Use abstraction to reduce complexity.
	6.	Learn the basics of HTML/CSS for web development.
Unit 5: Data Science and Computing		
	1.	Understand data science fundamentals.
	2.	Perform data analysis and visualization.
	3.	Recognize and understand data bias and ethics.
	4.	Incorporate cloud computing concepts in data science projects.
	5.	Apply cloud computing in data science projects.
Unit 6: Artificial Intelligence (AI)		
	1.	Summarize artificial intelligence (AI) terms and concepts.
	2.	Describe how AI is used and its impact.
	3.	Investigate an AI system.
	4.	Recognize and understand AI data and bias.
	5.	Develop skills in prompt engineering for AI applications.
Unit 7a: Physical Computing—Robotics		
	1.	Identify the criteria that describe a robot and determine if something is a robot.
	2.	Build, code, and test a robot that solves a stated problem.
Unit 7b: Physical Computing—Microcontrollers		
	1.	Identify common microcontroller terms.
	2.	Use programming to manipulate microcontroller inputs and outputs.
	3.	Use a microcontroller for a specific purpose.
	4.	Add microcontrollers and troubleshooting.
Unit 7c: Physical Computing— E-Textiles		
	1.	Design, interpret, and evaluate simple circuit diagrams with key components.
	2.	Construct parallel circuits using conductive materials and switches while applying an iterative design process.
	3.	Implement computational circuits with pre-programmed mini-computers and understand common grounding.
Unit 7d: Physical Computing—Networking		
	1.	Explore fundamental networking concepts.
Unit 7e: Physical Computing—VR Systems		
	1.	Investigate the process of developing virtual environments.

Unit 8: Cybersecurity		
	1.	Define cybersecurity and its importance.
	2.	Discuss confidentiality in cybersecurity.
	3.	Explain integrity and hashing.
	4.	Discuss availability considerations in cybersecurity.
	5.	Analyze adversary thinking and social engineering tactics.

Appendix A: Mississippi College- and Career-Readiness Standards for Computer Science

	Units	1	2	3	4	5	6	7	8
Standards									
CS.1.1		X							
CS.2.1			X						
CS.3A.1		X	X						
CS.3A.2			X						
CS.3A.3			X	X					
NL.1.1									X
NL.2.1									X
NL.3.1						X			
NL.3.2				X					
NL.3.3				X					
NL.3.4				X					
DA.1.1				X					
DA.2.1			X						
DA.3A.1			X						
DA.3A.2			X						
AP.1.1						X			
AP.2.1						X			
AP.2.3						X			
AP.3.1						X			
AP.3.2						X			
AP.3.3						X			
AP.3.4						X			
AP.3.5						X			
AP.3.7						X			
AP.3.10						X			
AP.4.1						X			
IC.1.1							X		
IC.2.1		X					X		
IC.3A.1							X		
IC.3A.3							X		
AP.3.11								X	
AP.3.12								X	
AP.3.13								X	
AP.4.2								X	
AP.4.3								X	
DA.4.1									X
DA.4.2									X
DA.4.3									X
CS.1.1		X							
CS.2.1			X						
CS.3A.1		X	X						
CS.3A.2			X						
CS.3A.3			X	X					
NL.1.1									X
NL.2.1									X
NL.3.1						X			
NL.3.2				X					
NL.3.3				X					
NL.3.4				X					
DA.1.1				X					
DA.2.1			X						
DA.3A.1			X						
DA.3A.2			X						

Level 2: GRADES 6-8 - Computing Systems

CS.2 Computing Systems

Conceptual understanding: People interact with a wide variety of computing devices that collect, store, analyze, and act upon information in ways that can affect human capabilities both positively and negatively. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. An understanding of hardware and software is useful when troubleshooting a computing system that does not work as intended.

CS.2.1 Recommend improvements to the design of computing devices based on an analysis of how users interact with the devices. [DEVICES] (P3.3)

The study of human-computer interaction (HCI) can improve the design of devices, including both hardware and software.

a. Students should make recommendations for existing devices (e.g., a laptop, phone, or tablet) or design their own components or interface (e.g., create their own controllers). Teachers can guide students to consider usability through several lenses, including accessibility, ergonomics, and learnability. For example, assistive devices provide capabilities such as scanning written information and converting it to speech.

CS.2.2 Design projects that combine hardware and software components to collect and exchange data. [HARDWARE & SOFTWARE] (P5.1)

Collecting and exchanging data involves input, output, storage, and processing. When possible, students should select the hardware and software components for their project designs by considering factors such as functionality, cost, size, speed, accessibility, and aesthetics.

a. Students will design projects that use both hardware and software to collect and exchange data. For example, components for a mobile app could include an accelerometer, GPS, and speech recognition. The choice of a device that connects wirelessly through a Bluetooth connection versus a physical USB connection involves a tradeoff between mobility and the need for an additional power source for the wireless device.

CS.2.3 Systematically identify and fix problems with computing devices and their components. [TROUBLESHOOTING] (P6.2)

Since a computing device may interact with interconnected devices within a system, problems may not be due to the specific computing device itself but to devices connected to it.

a. Students will use a structured process to troubleshoot problems with computing systems and ensure that potential solutions are not overlooked. Examples of troubleshooting strategies include following a troubleshooting flow diagram, making changes to the software to see if the hardware will work, checking connections and settings, and swapping in working components.

Level 2: GRADES 6-8 - Networks and the Internet

NI.2 Networks and the Internet

Conceptual Understanding: Computing devices typically do not operate in isolation. Networks connect computing devices to share information and resources and are an increasingly integral part of computing. Networks and communication systems provide greater connectivity in the computing world by providing fast, secure communication and facilitating innovation.

NI.2.1 Model the role of protocols in transmitting data across networks and the Internet. [NETWORK COMMUNICATION & ORGANIZATION] (P4.4)

Protocols are rules that define how messages between computers are sent. They determine how quickly and securely information is transmitted across networks and the Internet, as well as how to handle errors in transmission.

a. **Students should model how data is sent using protocols to choose the fastest path, deal with missing information, and securely deliver sensitive data.** For example, students could devise a plan for resending lost information or for interpreting a picture that has missing pieces. The priority at this grade level is understanding the purpose of protocols and how they enable secure and errorless communication. Knowledge of the details of how specific protocols work is not expected.

NI.2.2 Explain how physical and digital security measures protect electronic information. [CYBERSECURITY] (P7.2)

Information that is stored online is vulnerable to unwanted access. Examples of physical security measures to protect data include keeping passwords hidden, locking doors, making backup copies on external storage devices, and erasing a storage device before it is reused. Examples of digital security measures include secure router admin passwords, firewalls that limit access to private networks, and the use of a protocol, such as HTTPS, to ensure secure data transmission.

a. Students will explain how physical and digital security measures protect electronic information.

NI.2.3 Apply multiple methods of encryption to model the secure transmission of information. [CYBERSECURITY] (P4.4)

Encryption can be as simple as letter substitution or as complicated as modern methods used to secure networks and the Internet.

a. Students should encode and decode messages using a variety of encryption methods, and they should understand the different levels of complexity used to hide or secure information. For example, students could secure messages using methods like Caesar cyphers or steganography (i.e., hiding messages inside a picture or other data). They can also model more complicated methods, such as public key encryption, through unplugged activities.

Level 2: GRADES 6-8 - Data and Analysis

DA.2 Data and Analysis

Conceptual Understanding: Computing systems exist to process data. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important. Data is collected and stored so that it can be analyzed to better understand the world and make more accurate predictions.

DA.2.1 Represent data using multiple encoding schemes. [STORAGE] (P4.0)

Data representations occur at multiple levels of abstraction, from the physical storage of bits to the arrangement of information into organized formats (e.g., tables).

a. Students should represent the same data in multiple ways. For example, students could represent the same color using binary, RGB values, hex codes (low-level representations), as well as forms understandable by people, including words, symbols, and digital displays of the color (high-level representations).

DA.2.2 Collect data using computational tools and transform the data to make it more useful and reliable. [COLLECTION, VISUALIZATION, & TRANSFORMATION] (P6.3)

As students continue to build on their ability to organize and present data visually to support a claim, they will need to understand when and how to transform data for this purpose.

a. Students should transform data to remove errors, highlight or expose relationships, and/or make it easier for computers to process. The cleaning of data is an important transformation for ensuring consistent format and reducing noise and errors (e.g., removing irrelevant responses in a survey). An example of a transformation that highlights a relationship is representing males and females as percentages of a whole instead of as individual counts.

DA.2.3 Refine computational models based on the data they have generated. [INFERENCE & MODELS] (P5.3, P4.4)

A model may be a programmed simulation of events or a representation of how various data are related.

a. Students will refine computational models by considering which data points are relevant, how data points relate to each other, and if the data is accurate. For example, students may make a prediction about how far a ball will travel based on a table of data related to the height and angle of a track. The students could then test and refine their model by comparing predicted versus actual results and considering whether other factors are relevant (e.g., size and mass of the ball). Additionally, students could refine game mechanics based on test outcomes in order to make the game more balanced or fair.

Level 2: GRADES 6-8 - Algorithms and Programming

AP.2 Algorithms and Programming

Conceptual understanding: An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Algorithms and programming control all computing systems, empowering people to communicate with the world in new ways and solve compelling problems. The development process to create meaningful and efficient programs involves choosing which information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

AP.2.1 Use flowcharts and/or pseudocode to address complex problems as algorithms. [ALGORITHMS] (P4.4, P4.1)

Complex problems are problems that would be difficult for students to solve computationally.

a. Students will use pseudocode and/or flowcharts to organize and sequence an algorithm that addresses a complex problem, even though they may not actually program the solutions. For example, students might express an algorithm that produces a recommendation for purchasing sneakers based on inputs such as size, colors, brand, comfort, and cost. Testing the algorithm with a wide range of inputs and users allows students to refine their recommendation algorithm and to identify other inputs they may have initially excluded.

AP.2.2 Create clearly named variables that represent different data types and perform operations on their values. [VARIABLES] (P5.1, P5.2)

A variable is like a container with a name, in which the contents may change, but the name (identifier) does not.

a. When planning and developing programs, students should decide when and how to declare and name new variables. Examples of operations include adding points to the score, combining user input with words to make a sentence, changing the size of a picture, or adding a name to a list of people.
b. Students should use naming conventions to improve program readability.

AP.2.3 Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. [CONTROL] (P5.1, P5.2)

Control structures can be combined in many ways. Nested loops are loops placed within loops. Compound conditionals combine two or more conditions in a logical relationship (e.g., using AND, OR, and NOT), and nesting conditionals within one another allows the result of one conditional to lead to another.

a. Students will design and develop programs that combine control structures. For example, when programming an interactive story, students could use a compound conditional within a loop to unlock a door only if a character has a key AND is touching the door.

AP.2.4 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. [MODULARITY] (P3.2)

Decomposition facilitates aspects of program development by allowing students to focus on one piece at a time (e.g., getting input from the user, processing the data, and displaying the result to the user). Decomposition also enables different students to work on different parts at the same time.

a. Students should break down problems into subproblems, which can be further broken down into smaller parts. For example, animations can be decomposed into multiple scenes, which can be developed independently.

AP.2.5 Create procedures with parameters to organize code and make it easier to reuse. [MODULARITY] (P4.1, P4.3)

a. Students will create procedures and/or functions that are used multiple times within a program to repeat groups of instructions. These procedures can be generalized by defining parameters that create different outputs for a wide range of inputs. For example, a procedure to draw a circle involves many instructions, but all of them can be invoked with one instruction, such as "drawCircle." By adding a radius parameter, the user can easily draw circles of different sizes.

AP.2.6 Seek and incorporate feedback from team members and users to refine a solution that meets user needs. [PROGRAM DEVELOPMENT] (P2.3, P1.1)

Development teams that employ user-centered design create solutions (e.g., programs and devices) that can have a large societal impact, such as an app that allows people with speech difficulties to translate hard-to-understand pronunciation into understandable language.

a. Students should begin to seek diverse perspectives throughout the design process to improve their computational artifacts. Considerations of the end user may include usability, accessibility, age-appropriate content, respectful language, user perspective, pronoun use, color contrast, and ease of use.

AP.2.7 Incorporate existing code, media, and libraries into original programs and give attribution. [PROGRAM DEVELOPMENT] (P4.2, P5.2, P7.3)

Building on the work of others enables students to produce more interesting and powerful creations.

a. Students should use portions of code, algorithms, and/or digital media in their own programs and websites. At this level, they may also import libraries and connect to web application program interfaces (APIs). For example, when creating side-scrolling games, students may incorporate portions of code that create a realistic jump movement from another person's game, and they may also import Creative Commons-lessened images to use in the background.
b. Students should give attribution to the original creator's contributions.

AP.2.8 Systematically test and refine programs using a range of test cases. [PROGRAM DEVELOPMENT] (P6.1)

Test cases are created and analyzed to better meet the needs of users and to evaluate whether programs function as intended. At this level, testing should become a deliberate process that is more iterative, systematic, and proactive than at lower levels.

a. Students will test programs by considering potential errors, such as what will happen if a user enters invalid input (e.g., negative numbers and zero instead of positive numbers).

AP.2.9 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. [PROGRAM DEVELOPMENT] (P2.2)

Collaboration is a common and crucial practice in programming development. Often, many individuals and groups work together on the interdependent parts of a project.

a. Students will work collaboratively in groups.
b. Students should assume predefined roles within their teams and manage the project workflow using structured timelines. With teacher guidance, they will begin to create collective goals, expectations, and equitable workloads. For example, students may divide the design stage of a game into planning the storyboard, flowchart, and different parts of the game mechanics. They can then distribute tasks and roles among members of the team and assign deadlines.
c. Students should give attribution to the original creators to acknowledge their contributions.

AP.2.10 Document programs in order to make them easier to follow, test, and debug. [PROGRAM DEVELOPMENT] (P7.2)

Documentation allows creators and others to more easily use and understand a program.

a. Students should provide documentation for end users that explains their artifacts and how they function. For example, students could provide a project overview and clear user instructions.
b. Students should incorporate comments in their product (comments in the code).
c. Students should communicate their process using design documents, flowcharts, and presentations.

Level 2: GRADES 6-8 - Impacts of Computing

IC.2 Impacts of Computing

Conceptual understanding: Computing affects many aspects of the world in both positive and negative ways at local, national, and global levels. Individuals and communities influence computing through their behaviors and cultural and social interactions, and in turn, computing influences new cultural practices. An informed and responsible person should understand the social implications of the digital world, including equity and access to computing.

IC.2.1 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. [CULTURE] (P7.2)

Advancements in computer technology are neither wholly positive nor negative; however, the ways that people use computing technologies have tradeoffs.

a. Students should consider current events related to broad ideas, including privacy, communication, and automation. For example, driverless cars can increase convenience and reduce accidents, but they are also susceptible to hacking. The emerging industry will not only reduce the number of taxi and shared-ride drivers but also create more software engineering and cybersecurity jobs.

IC.2.2 Discuss issues of bias and accessibility in the design of existing technologies. [CULTURE] (P1.2)

a. Students should test and discuss the usability of various technology tools (e.g., apps, games, and devices) with the teacher's guidance. For example, facial recognition software that works better for lighter skin tones was likely developed with a homogeneous testing group and could be improved by sampling a more diverse population. When discussing accessibility, students may notice that allowing a user to change font sizes and colors will not only make an interface usable for people with low vision but also benefit users in various situations, such as in bright daylight or a dark room.

IC.2.3 Collaborate with many contributors through strategies such as crowdsourcing or surveys when creating a computational artifact. [SOCIAL INTERACTIONS] (P2.4, P5.2)

Crowdsourcing is gathering services, ideas, or content from a large group of people, especially from the online community. It can be done at the local level (e.g., classroom or school) or global level (e.g., age-appropriate online communities, like Scratch and Minecraft).

a. Students should collaborate with many contributors. For example, a group of students could combine animations to create a digital community mosaic. They could also solicit feedback from many people through the use of online communities and electronic surveys.

IC.2.4 Describe tradeoffs between allowing information to be public and keeping information private and secure. [SAFETY, LAW, & ETHICS] (P7.2)

Sharing information online can help establish, maintain, and strengthen connections between people. For example, it allows artists and designers to display their talents and reach a broad audience; however, security attacks often start with personal information that is publicly available online. Social engineering is based on tricking people into revealing sensitive information, which can be thwarted by being wary of attacks, such as phishing and spoofing.

a. Students should discuss and describe the benefits and dangers of allowing information to be public or kept private and secure.

Level 3A: GRADES 9-10 - Computing Systems

CS.3A Computing Systems

Conceptual understanding: People interact with a wide variety of computing devices that collect, store, analyze, and act upon information in ways that can affect human capabilities both positively and negatively. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. An understanding of hardware and software is useful when troubleshooting a computing system that does not work as intended.

CS.3A.1 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. [DEVICES] (P4.1)

Computing devices are often integrated with other systems, including biological, mechanical, and social systems. A medical device can be embedded inside a person to monitor and regulate his or her health, a hearing aid (a type of assistive device) can filter out certain frequencies and magnify others, a monitoring device installed in a motor vehicle can track a person's driving patterns and habits, and a facial recognition device can be integrated into a security system to identify a person. The creation of integrated or embedded systems is not an expectation at this level.

a. Students should be able to identify embedded computer systems.
b. Students should describe the types of data and procedures that are included in the embedded system and explain how the implementation details are hidden from the user. For example, a student might select a car stereo and identify the types of data (radio station presets, station name or number, volume level) and procedures (increase volume, store/recall saved station, mute) it includes.

CS.3A.2 Compare levels of abstraction and interactions between application software, system software, and hardware layers. [HARDWARE & SOFTWARE] (P4.1)

At its most basic level, a computer is composed of physical hardware and electrical impulses. Multiple layers of software are built upon the hardware and interact with the layers above and below them to reduce complexity. System software manages a computing device's resources so that software can interact with hardware. System software is used on many different types of devices, such as smart TVs, assistive devices, virtual components, cloud components, and drones. For example, students may explore the progression from voltage to binary signal to logic gates to adders and so on. Knowledge of specific, advanced terms for computer architecture, such as BIOS, kernel, or bus, is not expected at this level.

a. Students should be able to distinguish between hardware and software.
b. Students should be able to describe the purpose of and differences between system software (i.e., operating system) and application software (i.e., word processor).
c. Students should be able to describe how software and hardware interact. For example, text-editing software interacts with the operating system to receive input from the keyboard, convert the input to bits for storage, and interpret the bits as readable text to display on the monitor.

CS.3A.3 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. [TROUBLESHOOTING] (P6.2)

Troubleshooting complex problems involves the use of multiple sources when researching, evaluating, and implementing potential solutions. Troubleshooting also relies on experience, such as when people recognize that a problem is similar to one they have seen before or adapt solutions that have worked in the past. Examples of complex troubleshooting strategies include resolving connectivity problems, adjusting system configurations and settings, ensuring hardware and software compatibility, and transferring data from one device to another.

a. Students should develop guidelines by creating an artifact that conveys systematic troubleshooting strategies (i.e., create a flow chart or a job aid for a help desk employee).

Level 3A: GRADES 9-10 - Networks and the Internet

NI.3A Networks and the Internet

Conceptual understanding: Computing devices typically do not operate in isolation. Networks connect computing devices to share information and resources and are an increasingly integral part of computing. Networks and communication systems provide greater connectivity in computing by providing fast, secure communication, and facilitating innovation.

NI.3A.1 Evaluate the scalability and reliability of networks by describing the relationship between routers, switches, servers, topology, and addressing. [NETWORK COMMUNICATION & ORGANIZATION] (P4.1)

Each device is assigned an address that uniquely identifies it on the network. Routers function by comparing IP addresses to determine the pathways packets should take to reach their destination. Switches function by comparing MAC addresses to determine which computers or network segments will receive frames. Students could use online network simulators to experiment with these factors.

a. Students should be able to define a MAC address – what it is and how it is used.
b. Students should be able to explain what a router and a switch are and how they work inside a network.
c. Students should be able to define what a server is and how it is used in a network.
d. Students should be able to list various types of network topology and explain why each is used.
e. Students should be able to verbally and visually explain how addressing, routers, switches, and servers all work together in a network.

NI.3A.2 Give examples to illustrate how sensitive data can be affected by malware and other attacks. [CYBERSECURITY] (P7.2)

Network security depends on a combination of hardware, software, and practices that control access to data and systems. The needs of users and the sensitivity of data determine the level of security implemented. Potential security problems, such as denial-of-service attacks, ransomware, viruses, worms, spyware, and phishing, present threats to sensitive data.

a. Students should be able to discuss how sensitive data can be affected by malware and other attacks. Students might reflect on case studies or current events in which governments or organizations experienced data leaks or data loss as a result of these types of attacks.

NI.3A.3 Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts. [CYBERSECURITY] (P3.1, 3.3)

Security measures may include physical security tokens, two-factor authentication, and biometric verification. Potential security problems, such as denial-of-service attacks, ransomware, viruses, worms, spyware, and phishing, exemplify why sensitive data should be securely stored and transmitted. The timely and reliable access to data and information services by authorized users, referred to as availability, is ensured through adequate bandwidth, backups, and other measures.

a. Students should understand the different types of security problems and the different types of devices that can be impacted. Potential security problems may include issues such as denial-of-service attacks, ransomware, viruses, worms, spyware, phishing, and social engineering. Some types of devices impacted may include laptops, tablets, cell phones, self-driving cars, ATMs, and others.
b. Students should systematically evaluate different security measures based on efficiency, feasibility, and ethical impacts. Students might address issues such as how efficiency affects feasibility or whether a proposed approach raises ethical concerns.

NI.3A.4 Compare various security measures considering tradeoffs between the usability and security of a computing system. [CYBERSECURITY] (P6.3)

Security measures may include physical security tokens, two-factor authentication, and biometric verification, but choosing security measures involves tradeoffs between the usability and security of the system. The needs of users and the sensitivity of data determine the level of security implemented.

a. Students should be able to explain different types of security measures and discuss the tradeoffs between usability and security. For example, students might discuss computer security policies at the local level that present a tradeoff between usability and security, such as a web filter that prevents access to many educational sites but keeps the campus network safe.

NI.3A.5 Explain tradeoffs when selecting and implementing cybersecurity recommendations. [CYBERSECURITY] (P7.2)

Network security depends on a combination of hardware, software, and practices that control access to data and systems. The needs of users and the sensitivity of data determine the level of security implemented. Every security measure involves tradeoffs between the accessibility and security of the system.

a. Students should be able to describe, justify, and document choices they make using terminology appropriate for the intended audience and purpose. Students could debate issues from the perspective of diverse audiences, including individuals, corporations, privacy advocates, security experts, and the government.

Level 3A: GRADES 9-10 - Data and Analysis

DA.3A Data and Analysis

Conceptual understanding: Computing systems exist to process data. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important. Data is collected and stored so that it can be analyzed to better understand the world and make more accurate predictions.

DA.3A.1 Translate between different bit representations of real-world phenomena, such as characters, numbers, and images. [STORAGE] (P4.1)

a. Students should be able to translate between different bit representations. For example, hexadecimal color codes can be converted to decimal percentages, ASCII/Unicode representation, or binary can be converted to base 10.
b. Students should be able to discuss how data sequences can be interpreted in a variety of formats. For example, text, numbers, sound, and images.

DA.3A.2 Evaluate the tradeoffs in how data elements are organized and where data is stored. [STORAGE] (P3.3)

People make choices about how data elements are organized and where data is stored. These choices affect cost, speed, reliability, accessibility, privacy, and integrity.

a. Students should evaluate whether a chosen solution is most appropriate for a particular problem. Students might consider the cost, speed, reliability, accessibility, privacy, and integrity tradeoffs between storing photo data on a mobile device versus in the cloud.

DA.3A.3 Collect, transform, and organize data to help others better understand a problem. [COLLECTION, VISUALIZATION, & TRANSFORMATION] (P4.4)

People transform, generalize, simplify, and present large data sets in different ways to influence how other people interpret and understand the underlying information. Examples include visualization, aggregation, rearrangement, and application of mathematical operations. People use software tools or programming to create powerful, interactive data visualizations and perform a range of mathematical operations to transform and analyze data.

a. Students should use various data collection techniques for different types of computational problems. For example, user surveys, mobile device GPS, social media data sets, etc.
b. Use computational tools to collect, transform, and organize data to help others better understand a problem.
c. Students should use data analysis to identify significant patterns in data sets.

DA.3A.4 Create and evaluate computational models that represent real-world systems. [INFERENCE & MODELS] (P4.4)

Computational models make predictions about processes or phenomena based on selected data and features. The amount, quality, and diversity of data and the features chosen can affect the quality of a model and the ability to understand a system. Predictions or inferences are tested to validate models.

a. Students should create computational models that simulate real-world systems (e.g., ecosystems, epidemics, spread of disease).
b. Students should analyze and evaluate the ability of models and simulations to formulate, refine, and test hypotheses.

Level 3A: GRADES 9-10 - Algorithms and Programming

AP.3A Algorithms and Programming

Conceptual understanding: An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Algorithms and programming control all computing systems, empowering people to communicate with the world in new ways and solve compelling problems. The development process to create meaningful and efficient programs involves choosing which information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.

AP.3A.1 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests. [ALGORITHMS] (P5.2)

A prototype is a computational artifact that demonstrates the core functionality of a product or process. Prototypes are useful for getting early feedback in the design process and can yield insight into the feasibility of a product. The process of developing computational artifacts embraces both creative expression and the exploration of ideas to create prototypes and solve computational problems.

a. Students create artifacts that are personally relevant or beneficial to their community and beyond. Students should develop artifacts in response to a task or a computational problem that demonstrate the performance, reusability, and ease of implementation of an algorithm.

AP.3A.2 Use lists and functions to simplify solutions, generalizing computational problems instead of repeatedly using simple variables. [VARIABLES] (P4.1)

a. Students should be able to identify common features in multiple segments of code and substitute a single segment that uses lists (arrays) or functions to account for the differences.

AP.3A.3 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made. [CONTROL] (P5.2)

Implementation includes the choice of programming language, which affects the time and effort required to create a program. Readability refers to how clear the program is to other programmers and how it can be improved through documentation. The discussion of performance is limited to a theoretical understanding of execution time and storage requirements; a quantitative analysis is not expected. Control structures at this level may include conditional statements, loops, event handlers, and recursion.

a. Students should be able to justify by explaining the benefits and drawbacks of the selection of specific control structures with regard to implementation, readability, and program performance. For example, students might compare the readability and program performance of iterative and recursive implementations of procedures that calculate the Fibonacci sequence.

AP.3A.4 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. [CONTROL] (P5.2)

In this context, relevant computational artifacts include programs, mobile apps, or Web apps. Events can be user-initiated, such as a button press, or system-initiated, such as a timer firing. At previous levels, students have learned to create and call procedures. Here, students design procedures that are called by events.

a. Students will design procedures that are called by events. Students might create a mobile app that updates a list of nearby points of interest when the device detects that its location has been changed.

AP.3A.5 Decompose problems into smaller components through systematic analysis using constructs such as procedures, modules, and/or objects. [MODULARITY] (P3.2)

a. Students should decompose complex problems into manageable subproblems that could potentially be solved with programs or procedures that already exist. For example, students could create an app to solve a community problem by connecting to an online database through an application programming interface (API).

AP.3A.6 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. [MODULARITY] (P5.2)

Computational artifacts can be created by combining and modifying existing artifacts or by developing new artifacts. Examples of computational artifacts include programs, simulations, visualizations, digital animations, robotic systems, and apps. Complex programs are designed as systems of interacting modules, each with a specific role, coordinate for a common overall purpose. Modules allow for better management of complex tasks. The focus at this level is understanding a program as a system with relationships between modules.

a. Students will create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. The choice of implementation, such as a programming language or a paradigm, may vary. Students could incorporate computer vision libraries to increase the capabilities of a robot or leverage open-source JavaScript libraries to expand the functionality of a Web application.

AP.3A.7 Systematically design and develop programs for broad audiences by incorporating feedback from users. [PROGRAM DEVELOPMENT] (P5.1)

Examples of programs could include games, utilities, and mobile applications. Students at lower levels collect feedback and revise programs.

b. Students should do so through a systematic process that includes feedback from broad audiences. Students might create a user satisfaction survey and brainstorm distribution methods that could yield feedback from a diverse audience, documenting the process they took to incorporate selected feedback in product revisions.

AP.3A.8 Evaluate licenses that limit or restrict the use of computational artifacts when using resources such as libraries. [PROGRAM DEVELOPMENT] (P7.3)

Examples of software licenses include copyright, freeware, and many open-source licensing schemes. At previous levels, students adhered to licensing schemes.

a. Students should consider licensing implications for their own, especially when incorporating libraries and other resources. Students might consider two software libraries that address a similar need, justifying their choice based on the library that has the least restrictive license.

AP.3A.9 Evaluate and refine computational artifacts to make them more usable and accessible. [PROGRAM DEVELOPMENT] (P6.3)

Testing and refinement is the deliberate and iterative process of improving a computational artifact. This process includes debugging (identifying and fixing errors) and comparing actual outcomes to intended outcomes.

a. Students should respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts. For example, students could incorporate feedback from a variety of end users to help guide the size and placement of menus and buttons in a user interface.

AP.3A.10 Design and develop computational artifacts working in team roles using collaborative tools. [PROGRAM DEVELOPMENT] (P2.4)

Collaborative tools could be as complex as a source code version control system or as simple as a collaborative word processor. Team roles in pair programming are driver and navigator, but they could be more specialized in larger teams. As programs grow more complex, the choice of resources that aid program development becomes increasingly important and should be made by the students.

a. Students will work in teams using collaborative tools to design and develop computational artifacts. Students might work as a team to develop a mobile application that addresses a problem relevant to the school or community, selecting appropriate tools to establish and manage the project timeline; design, share, and revise graphical user interface elements; and track planned, in-progress, and completed components.

AP.3A.11 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. [PROGRAM DEVELOPMENT] (P7.2)

Complex programs are designed as systems of interacting modules, each with a specific role, coordinating for a common overall purpose. These modules can be procedures within a program; combinations of data and procedures; or independent, but interrelated, programs. The development of complex programs is aided by resources such as libraries and tools to edit and manage parts of the program.

a. Students will document design decisions using text, graphics, presentations, and/or demonstrations.

Level 3A: GRADES 9-10 - Impacts of Computing

IC.3A Impacts of Computing

Conceptual understanding: Computing affects many aspects of the world in both positive and negative ways at local, national, and global levels. Individuals and communities influence computing through their behaviors and cultural and social interactions, and in turn, computing influences new cultural practices. An informed and responsible person should understand the social implications of the digital world, including equity and access to computing.

IC.3A.1 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices. [CULTURE] (P1.2)

Computing may improve, harm, or maintain practices. Equity deficits, such as minimal exposure to computing, access to education, and training opportunities, are related to larger, systemic problems in society.

a. Students should be able to evaluate the accessibility of a product to a broad group of end users, such as people who lack access to broadband or who have various disabilities.
b. Students should also begin to identify potential biases during the design process to maximize accessibility in product design.

IC.3A.2 Test and refine computational artifacts to reduce bias and equity deficits. [CULTURE] (P1.2)

Biases could include incorrect assumptions developers have made about their user base. Equity deficits include minimal exposure to computing, access to education, and training opportunities.

a. Students should begin to identify potential bias during the design process to maximize accessibility in product design and become aware of professionally accepted accessibility standards to evaluate computational artifacts for accessibility.

IC.3A.3 Demonstrate ways a given algorithm applies to problems across disciplines. [CULTURE] (P3.1)

Computation can share features with disciplines, such as art and music, by algorithmically translating human intention into an artifact.

a. Students should be able to identify real-world problems that span multiple disciplines, such as increasing bike safety with new helmet technology, and that can be solved computationally.

IC.3A.4 Use tools and methods for collaboration on a project to increase connectivity of people in different cultures and career fields. [SOCIAL INTERACTIONS] (P2.4)

Many aspects of society, especially careers, have been affected by the degree of communication afforded by computing. The increased connectivity between people in different cultures and in different career fields has changed the nature and content of many careers.

a. Students should explore different collaborative tools and methods used to solicit input from team members, classmates, and others, such as participation in online forums or local communities. For example, students could compare ways different social media tools could help a team become more cohesive

IC.3A.5 Explain the beneficial and harmful effects that intellectual property laws can have on innovation. [SAFETY, LAW, & ETHICS] (P7.3)

Laws govern many aspects of computing, such as privacy, data, property, information, and identity. These laws can have beneficial and harmful effects, such as expediting or delaying advancements in computing and protecting or infringing upon people's rights. International differences in laws and ethics have implications for computing. For example, laws that mandate the blocking of some file-sharing websites may reduce online piracy but can restrict the right to access information. Firewalls can be used to block harmful viruses and malware but can also be used for media censorship.

a. Students should be aware of intellectual property laws and be able to explain how they are used to protect the interests of innovators and how patent trolls abuse the laws for financial gain.

IC.3A.6 Explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users. [SAFETY, LAW, & ETHICS] (P7.2)

Data can be collected and aggregated across millions of people, even when they are not actively engaging with or physically near the data collection devices. This automated and non-evident collection can raise privacy concerns, such as social media sites mining an account even when the user is not online. Other examples include surveillance video used in a store to track customers for security or information about purchase habits or the monitoring of road traffic to change signals in real time to improve road efficiency without drivers being aware. Methods and devices for collecting data can differ by the amount of storage required, the level of detail collected, and sampling rates.

a. Students should be able to explain the privacy concerns related to the collection and generation of data through automated processes.

IC.3A.7 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. [SAFETY, LAW, & ETHICS] (P7.3)

Laws govern many aspects of computing, such as privacy, data, property, information, and identity. International differences in laws and ethics have implications for computing.

a. Students should evaluate the social and economic implications of privacy in the context of safety, law, or ethics. For example, students might review case studies or current events that present an ethical dilemma when an individual's right to privacy is at odds with the safety, security, or wellbeing of a community.

Appendix B: Framework for 21st Century Learning

	Units	1	2	3	4	5	6	7	8
Standards									
CS1		X		X	X				
CS2		X	X						
CS3		X	X	X					
CS4		X		X					
CS5									
CS6		X	X		X	X	X	X	X
CS7		X			X	X	X	X	X
CS8		X			X				
CS9		X	X	X	X	X	X	X	X
CS10			X		X	X	X	X	X
CS11					X	X	X	X	X
CS12		X		X					
CS13		X			X	X	X	X	X
CS14		X		X					
CS15			X		X		X	X	X
CS16		X	X	X					

CSS1-21st Century Themes

CS1 Global Awareness

a. Using 21st-century skills to understand and address global issues
b. Learning from and working collaboratively with individuals representing diverse cultures, religions, and lifestyles in a spirit of mutual respect and open dialogue in personal, work, and community contexts
c. Understanding other nations and cultures, including the use of non-English languages

CS2 Financial, Economic, Business, and Entrepreneurial Literacy

a. Knowing how to make appropriate personal economic choices
b. Understanding the role of the economy in society
c. Using entrepreneurial skills to enhance workplace productivity and career options

CS3 Civic Literacy

a. Participating effectively in civic life through knowing how to stay informed and understanding governmental processes
b. Exercising the rights and obligations of citizenship at local, state, national, and global levels
c. Understanding the local and global implications of civic decisions

CS4 Health Literacy

1. Obtaining, interpreting, and understanding basic health information and services and using such information and services in ways that enhance health
2. Understanding preventive physical and mental health measures, including proper diet, nutrition, exercise, risk avoidance, and stress reduction
3. Using available information to make appropriate health-related decisions
4. Establishing and monitoring personal and family health goals
5. Understanding national and international public health and safety issues

CS5 Environmental Literacy

1. Demonstrating knowledge and understanding of the environment and the circumstances and conditions affecting it, particularly as relates to air, climate, land, food, energy, water, and ecosystems
2. Demonstrating knowledge and understanding of society's impact on the natural world (e.g., population growth, population development, resource consumption rate, etc.)
3. Investigating and analyzing environmental issues and making accurate conclusions about effective solutions
4. Taking individual and collective action toward addressing environmental challenges (e.g., participating in global actions, designing solutions that inspire action on environmental issues)

CSS2-Learning and Innovation Skills

CS6 Creativity and Innovation
1. Think creatively
2. Work creatively with others
3. Implement innovations

CS7 Critical Thinking and Problem Solving
1. Reason effectively
2. Use systems thinking
3. Make judgments and decisions
4. Solve problems

CS8 Communication and Collaboration
1. Communicate clearly
2. Collaborate with others

CSS3-Information, Media, and Technology Skills

CS9 Information Literacy
1. Access and evaluate information
2. Use and manage information

CS10 Media Literacy
1. Analyze media
2. Create media products

CS11 ICT Literacy
1. Apply technology effectively

CSS4-Life and Career Skills

CS12 Flexibility and Adaptability
1. Adapt to change
2. Be flexible

CS13 Initiative and Self-Direction
1. Manage goals and time
2. Work independently
3. Be self-directed learners

CS14 Social and Cross-Cultural Skills
1. Interact effectively with others

2. Work effectively in diverse teams

CS15 Productivity and Accountability

1. Manage projects
2. Produce results

CS16 Leadership and Responsibility

1. Guide and lead others
2. Be responsible to others

Appendix C: ISTE Standards

	Units	1	2	3	4	5	6	7	8
Standards									
T1		X	X			X			
T2		X							X
T3		X	X	X					
T4				X	X			X	
T5			X	X	X		X		
T6		X							
T7								X	

International Society for Technology in Education (ISTE)

T1 Empowered Learner

Students leverage technology to take an active role in choosing, achieving, and demonstrating competency in their learning goals, as informed by the learning sciences.

a. Articulate and set personal learning goals, develop strategies leveraging technology to achieve them, and reflect on the learning process itself to improve learning outcomes.
b. Build networks and customize their learning environments in ways that support the learning process.
c. Use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
d. Understand the fundamental concepts of technology operations, demonstrate the ability to choose, use, and troubleshoot current technologies, and can transfer their knowledge to explore emerging technologies.

T2 Digital Citizen

Students recognize the rights, responsibilities, and opportunities of living, learning, and working in an interconnected digital world, and they act and model in ways that are safe, legal, and ethical.

a. Cultivate and manage their digital identity and reputation and are aware of the permanence of their actions in the digital world.
b. Engage in positive, safe, legal, and ethical behavior when using technology, including social interactions online or when using networked devices.
c. Demonstrate an understanding of and respect for the rights and obligations of using and sharing intellectual property.
d. Manage their personal data to maintain digital privacy and security and are aware of data-collection technology used to track their navigation online.

T3 Knowledge Constructor

Students critically curate a variety of resources using digital tools to construct knowledge, produce creative artifacts, and make meaningful learning experiences for themselves and others.

a. Plan and employ effective research strategies to locate information and other resources for their intellectual or creative pursuits.
b. Evaluate the accuracy, perspective, credibility, and relevance of information, media, data, or other resources.

c. Curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions.
d. Build knowledge by actively exploring real-world issues and problems, developing ideas and theories, and pursuing answers and solutions.

T4 Innovative Designer

Students use a variety of technologies within a design process to identify and solve problems by creating new, useful, or imaginative solutions.

a. Know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts, or solving authentic problems.
b. Select and use digital tools to plan and manage a design process that considers design constraints and calculated risks.
c. Develop, test, and refine prototypes as part of a cyclical design process.
d. Exhibit a tolerance for ambiguity, perseverance, and the capacity to work with open-ended problems.

T5 Computational Thinker

Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.

a. Formulate problem definitions suited for technology-assisted methods such as data analysis, abstract models, and algorithmic thinking when exploring and finding solutions.
b. Collect data or identify relevant data sets, use digital tools to analyze them, and represent data in various ways to facilitate problem-solving and decision-making.
c. Break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
d. Understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

T6 Creative Communicator

Students communicate clearly and express themselves creatively for a variety of purposes using the platforms, tools, styles, formats, and digital media appropriate to their goals.

a. Choose the appropriate platforms and tools for meeting the desired objectives of their creation or communication.
b. Create original works or responsibly repurpose or remix digital resources into new creations.
c. Communicate complex ideas clearly and effectively by creating or using a variety of digital objects such as visualizations, models, or simulations.
d. Publish or present content that customizes the message and medium for their intended audiences.

T7 Global Collaborator

Students use digital tools to broaden their perspectives and enrich their learning by collaborating with others and working effectively in teams locally and globally.

a. Use digital tools to connect with learners from a variety of backgrounds and cultures, engaging with them in ways that broaden mutual understanding and learning.

b. Use collaborative technologies to work with others, including peers, experts, or community members, to examine issues and problems from multiple viewpoints.
c. Contribute constructively to project teams, assuming various roles and responsibilities to work effectively toward a common goal.
d. Explore local and global issues and use collaborative technologies to work with others to investigate solutions.