Mississippi Secondary Curriculum Frameworks in Career and Technical Education, Middle School

2020 Exploring Computer Science

Program CIP: 11.0101 — Computer and Information Sciences – General

Direct inquiries to

Instructional Design Specialist
Research and Curriculum Unit
P.O. Drawer DX
Mississippi State, MS 39762
662.325.2510

Program Coordinator
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 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 (ECS) curriculum was presented to the Mississippi Board of Education on June 11, 2020. The following persons are serving on the state board at the time:

Dr. Carey M. Wright, state superintendent of education
Dr. Jason S. Dean, chair
Mr. Buddy Bailey, vice chair
Ms. Rosemary G. Aultman
Dr. Karen Elam
Dr. John R. Kelly
Ms. Nancy Collins
Ms. Brittany Rye
Mr. Sean Suggs
Mr. Omar G. Jamil, Student Representative

The following Mississippi Department of Education (MDE) and RCU managers and specialists assisted in the development of the ECS curriculum:

Wendy Clemons, executive director for the Office of Secondary Education, supported the RCU and the teachers throughout the development of the framework and supporting materials.
Dr. Aimee Brown, state director for Career and Technical Education (CTE), supported the RCU and the teachers throughout the development of the framework and supporting materials.
Shelly Hollis, assistant director for the RCU Center for Cyber Education (CCE), researched and authored this framework. shelly.hollis@rcu.msstate.edu
Heather McCormick, project coordinator for the RCU CCE, researched and authored this framework. heather.mccormick@rcu.msstate.edu

Also, special thanks are extended to the teachers who contributed teaching and assessment materials that are included in the framework and supporting materials:

Corey Burt, Lafayette High School, Oxford
Brittany Cohen, North Panola Vocational Complex, Como
Angie Davis, Pontotoc High School, Pontotoc
April Dill, Millsaps Career and Technical Center, Starkville
Abbie Hendley, Oak Grove High School, Hattiesburg
Jana Odom, Career and Technical Institute, Pascagoula
Mario Rios, Greenville High School, Greenville

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

Shanta Villanueva, TSA and STEM program coordinator for the MDE Office of CTE and Workforce Development
Betsey Smith, director for the RCU
Brad Skelton, curriculum manager for the RCU
Melissa Luckett, instructional design specialist for the RCU

Standards

Standards and alignment crosswalks are referenced in the appendices. Mississippi's CTE ECS curriculum 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.org

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. *21 Framework Definitions* (2015).
p21.org/storage/documents/docs/p21_framework_definitions_new_logo_2015.pdf

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 course 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 at the RCU's website, under Professional Learning.

Should you need additional instructions, call the RCU at 662.325.2510.

Executive Summary

Program Description
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.

Applied Academic Credit
The latest academic credit information can be found at mdek12.org/Accred/AAS. Once there, click the "Mississippi Public School Accountability Standards Year" tab. Review the appendices for graduation options and superscript information regarding specific programs receiving academic credit.

Grade Level and Class Size Recommendations
It is recommended that students enter this program as a ninth grader or higher. Classes may contain mixed grade levels if allowed by district policy. The classroom and lab should be designed to accommodate a maximum of 24 students. Class size should be determined by the number of operational computers in the lab. Each student needs access to their computer to be successful in this course.

Teacher Licensure
The latest teacher licensure information can be found at mdek12.org/OTL/OEL.

Professional Learning
If you have specific questions about the content of any of training sessions provided, please contact the RCU at 662.325.2510.

Course Outline

Exploring Computer Science—Course Code 000283

Unit	Unit Name	Hours
1	Orientation and Ongoing Skills	10
2	Human Computer Interaction	20
3	Problem Solving	20
4	Web Design	25
5	Introduction to Programing	25
6	Artificial Intelligence	15
7	Robotics	25
Total		140

Note: This chart simply shows a listing of the units to be covered and not necessarily a required order of delivery.

Research Synopsis

Overview

ECS is an instructional program that lays a foundation in problem solving and critical thinking and introduces individuals to a variety of computer science content areas including programming, data science, Web development, artificial intelligence, and robotics. Students will also be exposed to the societal impacts of computer science and ethical issues surrounding the field. Computer science impacts the daily life of all citizens and influences every career field. This foundational course will provide students with skills to navigate technology in their personal life and give them the basic tools needed for success in the future workforce.

Needs of the Future Workforce

The computing industry is a rapidly growing and ever-changing field. Students will learn basic skills that will serve as the foundation of their knowledge. The workforce will require them to use these skills and adapt them to various specialties. As seen in Table 1.1, the computing workforce is equally competitive as it is abundant in opportunities for upward mobility.

Table 1.1: Current and Projected Occupation Report

Description	Jobs, 2010	Projected Jobs, 2020	Change (Number)	Change (Percent)	Total Projected Avg. Annual Job Openings	Average Hourly Earning
Computer Systems Analysts	1,540	1,690	150	9.7%	45	$29.09
Computer Programmers	1,430	1,400	-30	-2.1%	35	$33.83
Software Developers, Applications	600	720	120	20%	20	$45.42
Software Developers, Systems Software	240	280	40	16.7%	5	$41.32
Database Administrators	410	500	90	22%	15	$34.27
Network and Computer Systems Administrators	2,150	2,570	420	19.5%	80	$434.41
Computer Support Specialists	2,450	2,730	280	11.4%	90	$20.87
Information Security Analysist, Web Developers, and Computer Network Architects	290	330	40	13.8%	10	$32.06
Computer and Information Systems Managers	1,140	1,270	130	11.4%	30	$42.66

Computer Operators	600	540	-60	-10%	5	$17.71
Executive Secretaries and Administrative Assistants	7,130	7,140	10	0.1%	95	$16.86
First-Line Supervisors/Managers of Office and Administrative Support Workers	10,250	11,000	750	7.3%	350	$20.35
General and Operations Managers	17,260	16,710	-550	-3.2%	320	$29.42
Word Processors and Typists	850	740	-110	-12.9%	5	$12.55
Desktop Publishers	80	70	-10	-12.5%	0	$18.55

Source: Mississippi Department of Employment Security; www.mdes.ms.gov (2018).

Perkins V Requirements

The ECS curriculum meets Perkins V requirements of introducing students to and preparing them for high-skill, high-wage occupations. It also offers students a program of study including secondary, postsecondary, and institutions of higher learning courses that will prepare them for occupations. Additionally, the ECS curriculum is integrated with the Common Core State Standards (CCSS) and the 2018 Mississippi College- and Career-Readiness Standards for Computer Science. Lastly, the Cyber Foundations curriculum focuses on ongoing and meaningful professional development for teachers as well as relationships with industry.

Curriculum Content: Summary of Standards

The standards included in the ECS curriculum reflect state and national standards. The curriculum aligns with the 2018 Mississippi College- and Career-Readiness Standards for Computer Science, *Framework for 21st Century Learning*, and the standards for the International Society for Technology in Education. Aligning the curriculum content to these standards will result in students who are highly skilled, well rounded, more academically proficient, and more likely to be successful in community colleges, institutions of higher learning, and the workforce.

Academic Infusion

ECS is aligned to the Mississippi College and Career Readiness Standards. The Mississippi College and Career Readiness Standards are aligned with college and work expectations and include rigorous content and application of knowledge through high-order thinking skills. This applied approach to learning academic skills has long been the practice in career and technical education and brings relevance and enhances and reinforces these academic skills. Throughout the curriculum, students will be required to perform calculations and use strategic and critical thinking skills to solve real-world problems.

Transition to Postsecondary Education
The latest articulation information for secondary to postsecondary can be found at the Mississippi Community College Board website, mccb.edu.

Best Practices
Innovative Instructional Technologies
Classrooms should be equipped with tools that will teach today's digital learners through applicable and modern practices. The ECS curriculum includes teaching strategies that incorporate current technology. Each classroom should incorporate a teacher desktop or laptop and student computers in a networked environment. It is suggested that each classroom be equipped with an interactive white board and projector that will intensify the interaction between students and teachers during class. Teachers are encouraged to make use of the latest online communication tools, including wikis, blogs, and podcasts. They are also encouraged to teach using a learning management system that introduces students to education in an online environment and places 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—create unique learners. By providing various teaching and assessment strategies, students with various learning preferences can have more opportunity to succeed.

Career and Technical Education Student Organizations
Teachers are required to investigate and charter one of the many student organizations available to students. The suggested organizations for this course are Technology Student Association (TSA) and Future Business Leaders of America (FBLA). Contact information for these organizations is listed under "Professional Organizations" in this document.

Conclusions
Based on the previous information, the ECS curriculum will be filled with opportunities to develop workforce skills. Widely used teaching strategies—collaborative learning, problem-based learning, and demonstration, for example—will also be included. These will help to prepare students for the hands-on, team-based environment they will likely experience upon entering the workforce. The curriculum document will be updated regularly to reflect the needs of the information and communication technology workforce.

Professional Organizations

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

Technology Student Association
tsaweb.org

For teachers:
Mississippi Educational Computing Association
ms-meca.org

Mississippi Association of Career and Technical Education
mississippiacte.com

Mississippi Business Education Association
ms-mbea.com

Computer Science Teachers Association
csteachers.org

Association of Career and Technical Education
acteonline.org

International Society for Technology in Education
iste.org

Using This Document

Suggested Time on Task
This section indicates an estimated number of clock hours of instruction that should be required to teach the competencies and objectives of the unit. A minimum of 140 hours of instruction is required for each Carnegie unit credit. The curriculum framework should account for approximately 75-80% of the time in the course. The remaining percentage of class time will include instruction in nontested material, review for end-of-course testing, and special projects.

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.

Integrated Academic Topics, 21st Century Skills and Information and Communication Technology Literacy Standards, ACT College Readiness Standards, and Technology Standards for Students
This section identifies 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 are integrated into the content of the unit. Research-based teaching strategies also incorporate ACT College Readiness standards. This section also identifies the 21st century skills and information and communication technology literacy skills. In addition, national technology standards for students associated with the competencies and suggested objectives for the unit are also identified.

References
The teacher resources document contains references, lesson ideas, websites, teaching and assessment strategies, scenarios, skills to master, and other resources divided up by unit. This document will be updated periodically by RCU staff. Please check it regularly for new information in each unit. If you have something you would like to add, or a question about something in the document, simply contact the RCU and ask for the instructional design specialist (IDS) for your program or email the IDS directly. The teacher resource document can be downloaded at rcu.msstate.edu/Curriculum/CurriculumDownload.aspx.

Enrichment Material
Many of the units include an enrichment section at the end. This section of material will not be tested on the Mississippi Career Planning and Assessment System (MS-CPAS), however it will greatly enhance the learning experiences for the students. It is suggested to use the enrichment material when needed or desired by the teacher and if time allows in the class.

Unit 1: Orientation and Ongoing Skills

Competencies and Suggested Objectives
1. Identify school policies and safety procedures related to ECS. DOK 1 a. Examine the school handbook, the acceptable-use policy for technology, and other safety procedures for building-level situations. 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. Use an online learning management system. DOK 1 a. Discover online learning environments and how they operate among teachers and students. b. Demonstrate proper email etiquette. c. Participate in online learning methods, such as 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. DOK 1 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. DOK 2 a. Demonstrate effective collaboration and teamwork. b. Demonstrate creativity and imagination. • Use journaling as a means of drafting creative and imaginative work/ideas. c. Utilize critical thinking through effective reasoning, making judgements, and decisions. • Use journaling as a means of expressing critical thinking. d. Execute problem solving techniques.
5. Demonstrate effective public speaking skills. DOK 2 a. Demonstrate effective communication in groups. b. Demonstrate presentation skills.
6. Explore career opportunities within computer science in the specialty areas of programming, cybersecurity, data science, robotics, artificial intelligence, human computer interaction, and Web development. DOK 3 a. Research career opportunities for employment in each of the specialty areas listed above. b. Examine the requirements, skills, wages, education, and employment opportunities in each of the specialty areas listed above. c. Describe how at least one of the specialty areas listed above is used in a career field outside of computer science (e.g., automotive, health care, or fashion design).
Note: The content from this unit should be reinforced throughout the program.

Unit 2: Human Computer Interaction

Competencies and Suggested Objectives

1. Explain the difference between computers and computing. DOK 1
    a. Analyze the characteristics of hardware components to determine the applications for which they can be used.
        - Explain and give examples of the concepts of a computer and computing.
        - Identify and describe the major components of a computer.
        - Define the purpose of each major component of a computer.
    b. Describe 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.
    d. Explain the concept of a computer program.

2. Evaluate the results of Web searches and the reliability of information found on the internet. DOK 3
    a. Use appropriate tools and methods to execute internet searches that yield requested data.
    b. Perform searches and explain how to refine searches to retrieve better information.
    c. Identify resources for finding information in addition for ranking based search engines.
    d. Differentiate between ranking based search engines and social bookmarking (collaborative) search engines.
    e. Define and give examples of Web 2.0 applications.
    f. Develop and use a rubric to evaluate the reliability of websites.

3. Analyze the effects of computing on society within economic, social, and cultural contexts. DOK 3
    a. Communicate legal and ethical concerns raised by computing innovation.
    b. Explain the implications of communication as data exchange.
        - Explain how computers are used for communications.
        - Recognize various forms of communication as data exchange.
        - Describe the implications of data exchange on social interactions.
        - Consider privacy of data that they create.
    c. Explain basic security issues on the internet.
    d. Identify Web applications that influence society and education.
    e. Identify appropriate vs. inappropriate uses of social websites.

4. Describe the features of appropriate data sets for specific problems. DOK 1
    a. Explain how different representations of data can tell different stories.
    b. Recognize data is an incomplete record of reality.
    c. Understand how data can be used to make a case or discovery to solve or identify a problem.

d. Understand the complexities of collecting, processing, and analyzing sets of data.
    - Identify research questions that could be answered or stories that could be told from data.
    - Collaborate to develop a method for data collection.
    - Collect data.
    - Identify issues related to the data collection process (what can and can't be captured in data).
    - Explain aggregation of data.

e. Consider how various types of data (numbers, text, dates, etc.) lend themselves to processing.

5. Explain how computers can be used as tools for analyzing and visualizing data, modeling, and design. DOK 1

a. Identify mathematical connections (symmetry, coordinates/coordinate planes, patterns, iterations, etc.).

b. Use a variety of visualization tools and methods.

Unit 3: Problem Solving

Competencies and Suggested Objectives
1. Understand the problem-solving process. DOK 1 a. Name and explain the steps in the problem-solving process. ● Identify and document each step of the problem-solving process for a variety of scenarios. b. Solve a variety of problems by applying the problem-solving process. c. Express a solution using standard design tools (e.g., draw a diagram or picture, make a systemic list, divide and conquer, find the pattern, or guess and check).
2. Design and interpret algorithms. DOK 1 a. Define an algorithm. 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 cannot be solved by an algorithm.
3. Demonstrate an understanding of binary. DOK 2 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. DOK 1 a. Illustrate a linear search algorithm. b. Illustrate a binary search algorithm. c. Explain conditions in which each search would be appropriate.
5. Explain sorting algorithms. DOK 1 a. Define sorted and unsorted lists. b. Describe various sorting algorithms. c. Compare various sorting algorithms.
6. Describe minimal spanning trees. DOK 2 a. Solve a minimal spanning tree. b. List examples of minimal spanning trees in society (power grids, gas lines, brain, etc.). c. Explain how a minimal spanning tree relates to computer science networks

Unit 4: Web Design

Competencies and Suggested Objectives
1. Construct Web pages to address specified objectives. DOK 2 a. Select appropriate techniques when creating Web pages. b. Use abstraction to separate style from content in Web page design and development. c. Describe the use of a website with appropriate documentation d. Create a storyboard.
2. Design a Web page using HTML. DOK 4 a. Navigate an HTML editor. b. Create an HTML page with a title and body. c. Create an HTML page with paragraph tags, headings, line breaks, emphasized text, images, and horizontal lines. d. Identify standard image resolution for the Web (72 dpi). e. Resize and crop images for the Web. f. Identify and differentiate between various image formats (JPEG, GIF, PNG, etc.). g. Understand intellectual property and creative copyrights. h. Revise design based on peer review and testing.
3. Style a Web page using CSS. DOK 4 a. Create inline styles with CSS. b. Add inline styles to a Web page. c. Create an internal style sheet with CSS. d. Create a Web page that uses an internal style sheet. e. Create an HTML page that links to a separate CSS file. f. Use HTML tags and CSS styling elements to separate style from structure.
4. Create a multipage website using HTML and CSS as a final project. DOK 4 a. Create a storyboard for a multipage website. b. Create an HTML page that includes hyperlinks. c. Use table, row, and column tagging in an HTML page. d. Add CSS styling to an HTML table. e. Use ordered and unordered list tagging in an HTML page. f. Add CSS styling to an HTML list. g. Use grid elements in CSS div placement. h. Add a menu to an HTML page. i. Add layout styles to a Web page. j. Revise the design based on peer review and testing. k. Present the final project and discuss design decisions.

Unit 5: Introduction to Programming

Competencies and Suggested Objectives
a. Use appropriate algorithms to solve a problem. DOK 2 a. Use written steps or a flow diagram to plan a solution to a programming problem. b. Write code to solve a problem. c. Write code that is properly sequenced.
2. Design, code, test, and execute a program that corresponds to a set of specifications. DOK 4 a. Explain an event-driven program b. Create programs with practical, personal, and/or societal intent. c. Explain how a specific program functions (explaining pseudo code, comments within program, final presentation, etc.).
3. Apply appropriate use of programming structures. DOK 4 a. Define the following terms: • Variables • Conditionals • Loops (iteration) • Navigation (moving, timing) • Dialog (showing text, timing) • Event-driven programming (when, broadcast, user input, etc.) b. Demonstrate proper use of the following terms within a program • Variables • Conditionals • Loops (iteration) • Navigation (moving, timing) • Dialog (showing text, timing) • Event driven programming (when, broadcast, user input, etc.)
4. Locate and correct errors in a program. DOK 2 a. Deconstruct the program into smaller components to isolate a problem. b. Identify errors in a program written by the student and one written by another person (debug). c. Identify components of the software to identify where/how errors are displayed. d. Justify the correctness of a program. e. Evaluate a peer's program and provide constructive feedback.
5. Use abstraction to reduce complexity. DOK 2 a. Explain abstraction. b. Identify examples of abstraction in everyday life.

c. Modify a program to utilize user-defined code to represent more detailed tasks commands.
6. Create a program as a final project that incorporates unit objectives as a summative assessment. DOK 4
Note: Competencies and objectives can be covered in any order and many will be covered simultaneously.

Unit 6: Artificial Intelligence (AI)

Competencies and Suggested Objectives
1. Summarize AI terms and concepts. DOK 1 a. Explain key terminology associated with the field of AI. b. Develop an understanding of AI images and narratives.
2. Describe how AI is used and its impact. DOK 1 a. Identify the type of AI being used (image recognition, speech recognition, translation, etc.). b. Gain an understanding of how AI is changing different sectors (medicine, agriculture, etc.). c. Explore and explain the impact of AI for our society. d. Recognize the future of work is changing.
3. Investigate an AI system. DOK 3 a. Examine how a neural network functions. b. Build, train, and test an AI prototype system. c. Develop an understanding of datasets.
4. Recognize and understand AI data and bias. DOK 1 a. Judge algorithmic bias and the effect of bias on individuals and society. b. Examine issues around privacy and the collection of data.

Unit 7: Robotics

Competencies and Suggested Objectives
1. Identify the criteria that describes a robot and determines if something is a robot. DOK 1 a. Describe how the design of a robot's body affects its behavior. b. Identify the parts and features of a robot.
2. Build, code, and test a robot that solves a stated problem. DOK 4 a. Navigate the programming environment for a robot. b. Use sequential code to program a robot to perform various tasks c. Understand the following programming concepts as they relate to robotics: • Boolean operators • Loops and interrupts • Conditionals (if, then, else) • Waits d. Build (if applicable) and program a robot that uses input and output devices (sensors, actions, etc.) e. Match the actions of the robot to the corresponding parts of the program. f. Describe the tradeoffs of different ways to program a robot to achieve a goal. g. Debug code by testing.

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.
	2.	Use 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 the specialty areas of programming, cybersecurity, data science, robotics, artificial intelligence, human computer interaction, and Web development.
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.
	5.	Explain how computers can be used as tools for analyzing and visualizing data, modeling, and design.
Unit 3: Problem-Solving		
	1.	Describe the problem-solving process.
	2.	Design and interpret algorithms.
	3.	Demonstrate an understanding of binary.
	4.	Understand simple search algorithms.
	5.	Explain sorting algorithms.
	6.	Describe minimal spanning trees.
Unit 4: Web Design		
	1.	Construct Web pages to address specified objectives.

	2.	Design a Web page using HTML.
	3.	Style a Web page using CSS.
	4.	Create a multipage website using HTML and CSS as a final project.
Unit 5: Introduction to Programming		
	1.	Use appropriate algorithms to solve a problem.
	2.	Design, code, test, and execute a program that corresponds to a set of specifications.
	3.	Apply appropriate use of programming structures.
	4.	Locate and correct errors in a program.
	5.	Use abstraction to reduce complexity.
	6.	Create a program as a final project that incorporates unit objectives as a summative assessment.
Unit 6: Artificial Intelligence (AI)		
	1.	Summarize 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.
Unit 7: Robotics		
	1.	Identify the criteria that describes a robot and determine if something is a robot.
	2.	Build, code, and test a robot that solves a stated problem.

Source: *Miss. Code Ann. §§ 37-1-3 and 37-31-103*

Appendix: 2018 Mississippi College- and Career-Readiness Standards for Computer Science

Computer Science (CS) Crosswalk for Exploring Computer Science							
	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6	Unit 7
CS Standards							
CS.2.1							
CS.2.2							
CS.2.3							
CS.3A.1					X	X	
CS.3A.2		X			X	X	X
CS.3A.3							
NI.2.1							
NI.2.2							
NI.2.3							
NI.3A.1							
NI.3A.2		X					
NI.3A.3							
NI.3A.4							
NI.3A.5							
DA.2.1							
DA.2.2							
DA.2.3							
DA.3A.1		X	X				
DA.3A.2		X	X				
DA.3A.3		X					
DA.3A.4		X					
AP.2.1			X	X	X		
AP.2.2					X		
AP.2.3							
AP.2.4							
AP.2.5							
AP.2.6							
AP.2.7							
AP.2.8							
AP.2.9							
AP.2.10							
AP.3A.1			X	X	X		
AP.3A.2					X		
AP.3A.3				X			X
AP.3A.4				X	X		
AP.3A.5				X	X		
AP.3A.6							
AP.3A.7				X			
AP.3A.8							
AP.3A.9				X	X		X
AP.3A.10				X			
AP.3A.11				X	X		X
IC.2.1							
IC.2.2							
IC.2.3							
IC.2.4							
IC.3A.1		X				X	
IC.3A.2				X		X	
IC.3A.3		X				X	
IC.3A.4							
IC.3A.5							
IC.3A.6		X				X	
IC.3A.7						X	

Level 2: GRADES 6-8 - Computing Systems

Computing Systems (CS.2)

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.

CS.2.1a ***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.

CS.2.2a ***Students will design projects that use both hardware and software to collect and exchange data.*** *For example, components for a mobile app could include 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.

CS.2.3a ***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 software to see if hardware will work, checking connections and settings, and swapping in working components.*

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

Networks and the Internet (NI.2)

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.

NI.2.1a ***Students should model how data is sent using protocols to choose the fastest path, to deal with missing information, and to deliver sensitive data securely.*** *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.

NI.2.2a ***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.

NI.2.3a ***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

Data and Analysis (DA.2)

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).

DA.2.1a ***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.

DA.2.2a ***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 is related.

DA.2.3a ***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

Algorithms and Programming (AP.2)

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.

AP.2.1a ***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.

AP.2.2a ***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.*

AP.2.2b ***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.

AP.2.3a ***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.

AP.2.4a ***Students should break down problems into subproblems, which can be further broken down to 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)

AP.2.5a ***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.

AP.2.6a ***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.

AP.2.7a ***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 a 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.*

AP.2.7b ***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.

AP.2.8a ***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 on the interdependent parts of a project together.

AP.2.9a ***Students will work collaboratively in groups.***

AP.2.9b ***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.*

AP.2.9c ***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.

AP.2.10a ***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.*

AP.2.10b ***Students should incorporate comments in their product (comments in the code).***

AP.2.10c ***Students should communicate their process using design documents, flowcharts, and presentations.***

Level 2: GRADES 6-8 - Impacts of Computing

Impacts of Computing (IC.2)

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.

IC.2.1a ***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)

IC.2.2a ***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 benefits 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).

IC.2.3a ***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 though 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 and can be thwarted by being wary of attacks, such as phishing and spoofing.

IC.2.4a ***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

Computing Systems (CS.3A)

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.

CS.3A.1a ***Students should be able to identify embedded computer systems.***

CS.31.1b ***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, 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.

CS.3A.2a ***Students should be able to distinguish between hardware and software.***

CS.3A.2b ***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).***

CS.3A.2c ***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.

CS.3A.3a ***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

Networks and the Internet (NI.3A)

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.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.

NI.3A.1a ***Students should be able to define a MAC address – what is it and how it is used.***

NI.3A.1b ***Students should be able to explain what a router and a switch are and how they work inside a network.***

NI.3A.1c ***Students should be able to define what a server is and how it is used in a network.***

NI.3A.1d ***Students should be able to list various types of network topology and explain why each is used.***

NI.3A.1e ***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.

NI.3A.2a ***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.

NI.3A.3a ***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.*

NI.3A.3b ***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.

NI.3A.4a ***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 in place 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.

NI.3A.5a ***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 government.*

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

Data and Analysis (DA.3A)

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)

DA.3A.1a ***Students should be able to translate between different bit representations.*** *For example, convert hexadecimal color codes to decimal percentages, ASCII/Unicode representation, or converting binary to base 10.*

DA.3A.1b ***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.

DA.3A.2a ***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.

DA.3A.3a ***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.*

DA.3A.3b ***Use computational tools to collect, transform, and organize data to help others better understand a problem.***

DA.3A.3c ***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 phenomenon 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 ability to understand a system. Predictions or inferences are tested to validate models.

DA.3A.4a ***Students should create computational models that simulate real-world systems (e.g., ecosystems, epidemics, spread of disease).***

DA.3A.4b ***Students should analyze and evaluate the ability of models and simulations to formulate, refine, and test hypothesis.***

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

Algorithms and Programming (AP.3A)

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.

AP.3A.1a ***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)

AP.3A.2a ***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 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.

AP.3A.3a ***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.

AP.3A.4a ***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)

AP.3A.5a ***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, coordinating 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.

AP.3A.6a ***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 programming language or 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.

AP.3A.1b ***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 use of computational artifacts when using resources such as libraries. [PROGRAM DEVELOPMENT] (P7.3)

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

AP.3A.8a ***Students should consider licensing implications for their own work, 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.

AP.3A.9a ***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 source code version control system or as simple as a collaborative word processor. Team roles in pair programming are driver and navigator but 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.

AP.3A.10a ***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.

AP.3A.11a ***Students will document design decisions using text, graphics, presentations, and/or demonstrations.***

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

Impacts of Computing (IC.3A)

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.

IC.3A.1a ***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.***

IC.3A.2b ***Students should also begin to identify potential bias 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.

IC.3A.2a ***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.

IC.3A.3a ***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.

IC.3A.4a ***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 examples, 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.

IC.3A.5a ***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 nonevident 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, level of detail collected, and sampling rates.

IC.3A.6a ***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.

IC.3A.7a ***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.*