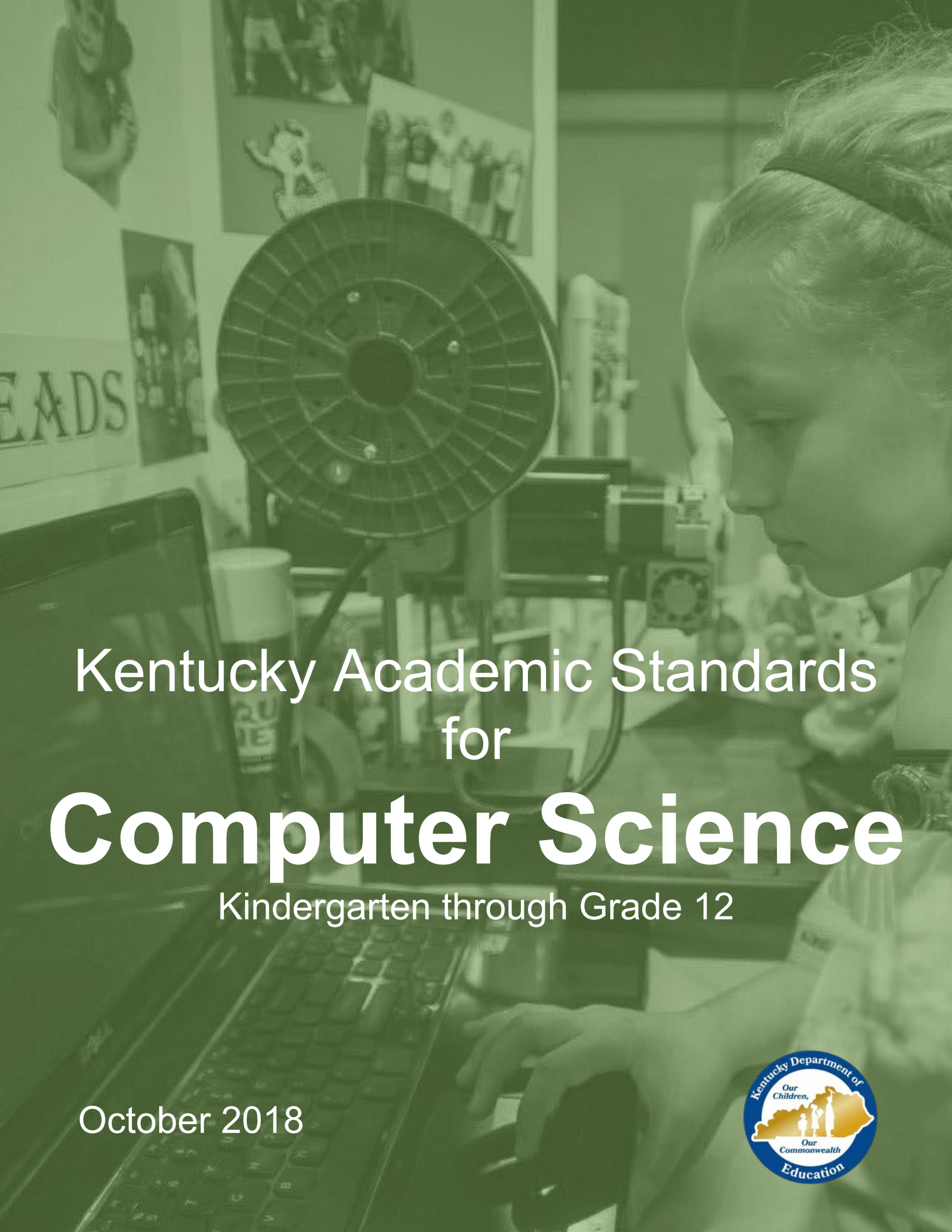


Kentucky Academic Standards for

Computer Science

Kindergarten through Grade 12

October 2018

Table of Contents

Introduction

Background

The United States currently has approximately 494,000 unfilled computing jobs, but only 43,000 computer science graduates to fill those jobs (Promoting Computer Science, 2017). As the need for computer science education increases, Kentucky recognizes the benefit of high quality standards to help all students discover how computing and technology shape the world around them. Therefore, the Kentucky Department of Education (KDE) engaged with state and local partners to develop the *Kentucky Academic Standards for Computer Science* which focus on providing students with opportunities to develop fundamental skills essential to all college and career paths; whereby, stimulating Kentucky's economy and workforce.

Kentucky's Vision for Students

Knowledge about the design and implementation of computer programs to solve problems is not only fundamental to the study of computer science, but also develops skills and dispositions that directly align with the Kentucky Board of Education's (KBE) vision that each and every student is empowered and equipped to pursue a successful future. To equip and empower students, the following capacity and goal statements frame instructional programs in Kentucky schools. The statements and goals were established by the Kentucky Education Reform Act (KERA) of 1990, as found in Kentucky Revised Statute (KRS) 158.645 and KRS 158.6451. All students shall have the opportunity to acquire the following capacities and learning goals:

- Communication skills necessary to function in a complex and changing civilization;
- Knowledge to make economic, social and political choices;
- Understanding of governmental processes as they affect the community, the state and the nation;
- Sufficient self-knowledge and knowledge of their mental health and physical wellness;
- Sufficient grounding in the arts to enable each student to appreciate their cultural and historical heritage;
- Sufficient preparation to choose and pursue their life's work intelligently; and
- Skills to enable students to compete favorably with students in other states and other parts of the world

Furthermore, schools shall:

- Expect a high level of achievement from all students.
- Develop their students' ability to:
    - Use basic communication and mathematics skills for purposes and situations they will encounter throughout their lives;
    - Apply core concepts and principles from mathematics, the sciences, the arts, the humanities, social studies, English/language arts, health, practical living, including physical education, to situations they will encounter throughout their lives;

    - Become self-sufficient individuals;
    - Become responsible members of a family, work group or community as well as an effective participant in community service;
    - Think and solve problems in school situations and in a variety of situations they will encounter in life;
    - Connect and integrate experiences and new knowledge from all subject matter fields with what students have previously learned and build on past learning experiences to acquire new information through various media sources;
    - Express their creative talents and interests in visual arts, music, dance, and dramatic arts.
- Increase student attendance rates.
- Reduce dropout and retention rates.
- Reduce physical and mental health barriers to learning, and
- Be measured on the proportion of students who make a successful transition to work, postsecondary education and the military.

Kentucky law establishes minimum requirements for all students to earn a diploma. However, elective courses are offered based on decisions of local districts and schools. Schools also offered Computer Science courses in the past through Career and Technical Education (CTE) Pathways.

To ensure legal requirements of courses are met, the KDE encourages schools to use the *Model Curriculum Framework* to inform development of curricula related to these courses. The *Model Curriculum Framework* encourages putting the student at the center of planning to ensure that

> …the goal of such a curriculum is to produce students that are ethical citizens in a democratic global society and to help them become self-sufficient individuals who are prepared to succeed in an ever-changing and diverse world. Design and implementation requires professionals to accommodate the needs of each student and focus on supporting the development of the whole child so that all students have equitable access to opportunities and support for maximum academic, emotional, social and physical development.
>
> *(Model Curriculum Framework,* page 19*)*

Legal Basis

The following Kentucky Revised Statutes (KRS) and Kentucky Administrative Regulations (KAR) provide a legal basis for this publication:

KRS 156:160 Promulgation of administrative regulations by the Kentucky Board of Education

> With the advice of the Local Superintendents Advisory Council (LSAC), the KBE shall promulgate administrative regulations establishing standards that public school districts

shall meet in student, program, service and operational performance. These regulations shall comply with the expected outcomes for students and schools set forth in KRS 158:6451.

Writers' Vision Statement

The writing team envisioned standards that would afford students the opportunity to engage in critical thinking, computational thinking and problem-solving through computer science. The writing team wanted standards that would:

- initiate cross-curricular connections to enhance the understanding of computer science skills and concepts;
- establish a continuum of computer science competencies K-12;
- provide opportunities for ALL students to engage in computer science experiences and
- advanced coursework to prepare them for future success; and
- prepare students to address a critical workforce need related to computer science knowledge and skills.

Design Considerations

The writers designed a single set of standards to frame learning opportunities in such a way so as to allow local districts the flexibility to choose the curricular design that best meets the needs of students. The writers chose to organize the standards into five broad concept areas: networks and the internet, data and analysis, algorithms and programming, impacts of computing and computing systems. Standards in each area were written as performance expectations to depict what students must do to demonstrate proficiency.

What is Computer Science Education?

Kentucky defines computer science as an academic discipline encompassing the study of computers and algorithmic processes to include principles, hardware and software designs, applications, networks and the impact on society. The computer science standards focusing on this academic discipline outlined in this document provide foundational opportunities essential to the preparation of students for post-secondary education and careers.

The infusion of technology, computers and digital experiences in our everyday life blurred the lines in terms of how computer science is defined. The K-12 Computer Science Framework recognizes four common terms that have a relationship with computer science education, but do not define it in isolation due to differences in purpose and meaning.

> *Computer science often is confused with the everyday use of computers and computer applications, such as learning how to access the Internet and use digital presentation software. Parents, teachers, students, and local and state administrators share this confusion. The K–12 Computer Science Framework clarifies not only what computer science is but also what students should know and be able to do in computer science from kindergarten to 12th grade. Computer science builds on computer literacy, educational technology, digital citizenship, and information technology. Their differences and relationship with computer science are described below.*
>
> - ***Computer literacy*** *(or digital literacy) refers to the general use of computers and programs (i.e., computer applications) such as productivity software. Examples include performing an Internet search and creating a digital presentation.*
> - ***Educational technology*** *applies computer literacy to school subjects. For example, students in an English class can use a web-based application to collaboratively create, edit, and store an essay online.*
> - ***Digital citizenship*** *refers to the appropriate and responsible use of technology, such as choosing an appropriate password and keeping it secure.*
>
> ***Information technology*** *often overlaps with computer science but is mainly focused on industrial applications of computer science, such as installing and operating software rather than creating it. Information technology professionals often have a background in computer science.*

These aspects of computing are distinguished from computer science because they focus on *using* computer technologies rather than understanding *why* they work and *how* to create those technologies. Knowing why and how computers work (i.e., computer science), provides the basis for a deep understanding of computer use and the relevant rights, responsibilities and applications (K-12 Computer Science Framework Steering Committee, 2016).

Technology Standards vs. Computer Science Standards

Kentucky officially recognized Technology Academic Standards in 2008. While these technology standards complement the *Kentucky Academic Standards for Computer Science*, the resulting competencies are substantially different. The technology standards are broad and should lead students towards competencies that highlight learning with technology. Such as digital literacy skills demonstrating the responsible use of appropriate technology to communicate, solve problems, access, manage, integrate, evaluate and create information to improve learning in all subject areas.

The *Kentucky Academic Standards for Technology*, based on the International Society for Technology in Education (ISTE) Student Standards, provide a framework for integrating technology into all content areas and reflect the basic digital skills required for each student to be competitive in the global economy. Additionally, “demonstrating performance-based

competency in technology" is included as a minimum graduation requirement in Kentucky public schools. For students to attain the required technology competencies, it is essential they have access to technology during the school day at all grade levels. Instruction should provide opportunities for students to gain and demonstrate technology skills that build throughout their K-12 educational careers.

Consulted Partners

Through the *Kentucky Academic Standards for Computer Science* development process many partners were consulted. The following list represents partners who assisted in the drafting of these standards provided valuable research and resources:

- Computer Science Teachers Association (2017). CSTA K-12 Computer Science Standards, Revised 2017. Retrieved from http://www.csteachers.org/standards.
- K–12 Computer Science Framework. (2016). Retrieved from http://www.k12cs.org.
- Southern Regional Education Board. (2016). Bridging the computer science education gap. Retrieved from https://www.sreb.org/publication/bridging-computer-science-education-gap
- Teach Computer Science. (n.d.). Retrieved from https://studio.code.org/courses?view=teacher.

Standards Use and Development

The Kentucky Academic Standards (KAS) are Standards, not Curriculum

The *Kentucky Academic Standards for Computer Science* outline the minimum content standards Kentucky students should learn in elective or integrated courses. The standards address what is to be learned, but do not address how learning experiences are to be designed or what resources should be used.

A standard represents a goal or outcome of an educational program. The standards do not dictate the design of a lesson plan or how units should be organized. The standards establish what students should know and be able to do at the conclusion of a course. The instructional program should emphasize the development of students' abilities to acquire and apply the standards. The curriculum must assure that appropriate accommodations are made for diverse populations of students found within Kentucky schools.

These standards are not a set of instructional or assessment tasks; rather, they are statements of what students should be able to do after instruction. Decisions on how best to help students meet these program goals are left to local school districts and teachers.

Translating the Standards into Curriculum

The KDE does not require specific curriculum or strategies to be used to teach the *Kentucky Academic Standards* (KAS). Local schools and districts choose to meet those minimum required standards using a locally adopted curriculum. As educators implement academic standards, they, along with community members, must guarantee 21st-century readiness that will ensure all learners are transition-ready. To achieve this, Kentucky students need a curriculum designed and structured for a rigorous, relevant and personalized learning experience, including a wide variety of learning opportunities. The *Kentucky Model Curriculum Framework* serves as a resource to help an instructional supervisor, principal and/or teacher leader revisit curriculum planning, offering background information and exercises to generate "future-oriented" thinking while suggesting a process for designing and reviewing the local curriculum.

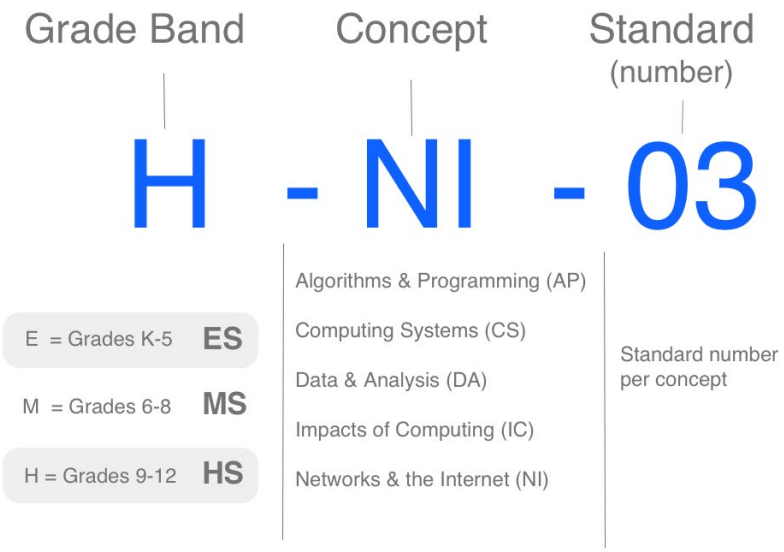
Organization of the Standards

Standards Structure and Identifiers

The Kentucky Academic Standards for Computer Science follow a specific structure.

- **Standard Identifier:** reflects consistent coding for the identification of a standard representing the grade (or grade band), the concept area and the numerated standard number per concept.
- **Grade Band:** identifies the grade band associated with the standard.
- **Concept:** categorizes the standards into five concepts (Computing Systems; Networks and the Internet; Algorithms and Programming; Data and Analysis; and Impacts of Computing).
- **Subconcept:** represents the specific ideas within that concept. Subconcept overviews summarize how learning progresses across multiple grade bands and are used to develop the progression chart (Appendix C).
- **Standard**: outlines what students are expected to know or be able to do.
- **Description**: translates the standard into manageable learning pieces.
- **Grade-by-Grade Indicators:** provide a comprehensive picture of performance expectations for each standard in the K-5 grade band and include thorough descriptions of the exemplary practices and processes.

Standard Identifier (example)



Grade Bands and Grade Level Considerations

The Kentucky Academic Standards for Computer Science are organized in specific grade bands (K-5, 6-8 and 9-12). This organization enables teachers to create grade level or course-specific student expectations derived from the standards. Additionally, connections exist between standards in different grade bands and demonstrate how one concept builds on another to provide vertically aligned learning experiences for students.

- *The Kentucky Academic Standards for Computer Science* represent knowledge and skills that should be modeled through the transition of each grade band (i.e. grade 5, grade 8 and grade 12). While middle school and high school students generally have the opportunity to demonstrate the learning of computer science skills and concepts through dedicated computer science-related courses, students in elementary school may be more likely to learn computer science skills integrated throughout the curriculum in all content areas. Therefore, **grade-by-grade indicators are ONLY included, per standard, for kindergarten through grade 5**.
- Computer science at the middle school level continues to develop students' foundational skills (in problem solving, computational and critical thinking) through the awareness and exploration of computer science-related concepts.
- Computer science at the high school level continues to develop students' foundational understanding of computer science in the world through in-depth learning opportunities, including awareness and exploration activities. Standards marked with a (*) for grades 9-12 represent challenging computer science learning expectations (referred to as Challenge Standards) for students with aspirations toward careers and postsecondary studies in computing disciplines.

Supplementary Materials to the Standards

The final set of the *Kentucky Academic Standards for Computer Science* is the result of educator involvement and public feedback. Short summaries of each of the appendices are listed below.

Appendix A: Glossary of Terms
Disciplinary terms are used throughout the *Kentucky Academic Standards for Computer Science* and its supporting materials. This document provides definitions and descriptions of these terms.

Appendix B: Writing and Review Teams
Background information on the team who wrote the *Kentucky Academic Standards for Computer Science* is included. Additional information includes those who reviewed the standards and/or provided feedback.

Appendix C: Grade Band Progression Chart
The progression chart represents the K-12 *Kentucky Academic Standards for Computer Science* progressions for ALL students, to include all concept areas and subconcepts.

Elementary (K-5)

Kentucky Academic Standards for **Computer Science**

Kentucky Academic Standards (KAS) for Computer Science

Elementary (K-5) Computer Science Standards

Concept: **Networks and The Internet**

Identifier	Standard & Description
	Understand the basic components of how networks operate to protect physical and digital information. Students should be able to articulate that usernames and passwords are used to verify the identity of a person using a computing device or system. Students should use usernames and passwords regularly; be able to show that strong passwords are more secure than weak passwords and longer passwords are stronger than short passwords; indicate that passwords can be made even stronger when numbers and symbols are used as well as letters; and be introduced to the term "complex" as a synonym for "strong."
E-NI-01	Subconcept: Network Communication & Organization <u>*Grade-by-Grade Indicators:*</u> *K - Describe how usernames and passwords protect personal information.* *1 - Demonstrate how to log in and log out of digital device using age appropriate usernames and passwords.* *2 - Describe the characteristics of a strong password.* *3 - Explain the effects of password misuse.* *4 - Explain how acceptable use policies help protect physical devices and digital information.* *5 - Demonstrate an understanding of digital security (i.e. use strong passwords; use usernames; protect personal digital information)*
	Model how information is broken down into smaller pieces (data packets), transmitted over various paths (physical and/or wireless), and reassembled at the destination Computers break down information into smaller pieces called packets, which are sent independently and reassembled at the destination. Students should demonstrate their understanding of this flow of information by: drawing a model of the way packets are transmitted, programming an animation to show how packets are transmitted, or demonstrating this through an analog, non-digital activity which has them act it out in some way (this can also be referred to as an “unplugged activity”).
E-NI-02	Subconcept: Cybersecurity

Identifier	Standard & Description
	Grade-by-Grade Indicators: *K - Not introduced until 4th grade* *1 - Not introduced until 4th grade* *2 - Not introduced until 4th grade* *3 - Not introduced until 4th grade* *4 - Describe how computers break down information.* *5 - Use a model to represent how digital information is sent and received over physical or wireless paths.*

Concept: **Data and Analysis**

Identifier	Standard & Description
E-DA-01	**Appropriately store and modify digital files.** All information stored and processed by a computing device is referred to as data. Data can be images, text documents, audio files, software programs or apps, video files, etc. As students use software to complete tasks on a computing device, they should demonstrate their understand that they are manipulating data. Subconcept: Storage *Grade-by-Grade Indicators:* *K - Open and close digital files with prompting and support.* *1 - Open, close and save digital files with prompting and support.* *2 - Open, close and save digital files.* *3 - Search, modify and delete digital files with prompting and support.* *4 - Search, modify, and delete digital files.* *5 - Convert digital files.*
E-DA-02	**Collect and visually display data using appropriate applications.** The collection and use of data about the world around them is a routine part of life and influences how people live. Students should collect data that they experience in the world around them, then organize the data into two or more visualizations. Data collection and analysis should be cross-curricular and occur throughout the year. Subconcept: Collection, Visualization & Transformation *Grade-by-Grade Indicators:* *K - Not introduced until 1st Grade.* *1 - Ask questions to collect and visually represent data with prompting and support.* *2 - Collect and visually represent data using one digital format with prompting and support.* *3 - Collect and visually represent data in tables or graphical displays using one application or digital format.* *4 - Collect data and determine an appropriate application or format to visually display*

Identifier	Standard & Description
	the data. *5 - Collect and represent data in graphical displays using one or more application to determine the benefits of using more than one visual display type.*
E-DA-03	**Analyze data for trends and relationships** Raw data has little meaning on its own. Students should be able to demonstrate sorting or grouping of data to provide additional clarity and meaning. Organizing data can make interpreting and communicating data to others easier. Data points can be clustered by a number of commonalities. Students should demonstrate understanding that the same data could be manipulated in different ways to emphasize particular aspects or parts of the data set. Raw data should be used to highlight relationships, and to determine different cause and effect relationships. Students can also demonstrate that data can be used to predict things that would happen in the future. Subconcept: Inference & Models *<u>Grade-by-Grade Indicators:</u>* *K - Use observations to describe patterns in organized data with prompting and support.* *1 - Use observations to describe patterns in organized data.* *2 - Use observations to describe patterns that can be predicted in organized data.* *3 - Analyze and interpret data using digital tools.* *4 - Analyze and interpret data to describe patterns using digital tools.* *5 - Represent data in graphical displays and describe cause and effect relationships, communicate ideas or predict outcomes.*

Concept: **Algorithms and Programming**

Identifier	Standard & Description
E-AP-01	**Create, follow, compare and refine algorithms for a task.** Algorithms (step-by-step instructions) are common in many primary classrooms. Just as people use algorithms to complete daily routines, they can program computers to use algorithms to complete different tasks. Algorithms are commonly implemented using a precise language that computers can interpret. Different algorithms can be used to perform the same task. While the end results may be similar, the paths may be different. Students should be able to look at different ways to solve the same task and decide which would be the best solution. Algorithms can be expressed in non-computer languages, including natural language, flowcharts, and pseudocode. Subconcept: Algorithms *<u>Grade-by-Grade Indicators:</u>* *K - Use simple algorithms to complete everyday tasks.* *1 - Create and use simple algorithms to complete everyday tasks.* *2 - Create and use simple algorithms using images, text or visual programming blocks*

Identifier	Standard & Description
	to complete everyday tasks. *3 - Compare two sets of algorithms for the same task to determine the best solution.* *4 - Create and compare two sets of algorithms for the same task to determine the best solution.* *5 - Modify a set of algorithms and discuss how multiple paths can lead to the same solution.*
E-AP-02	**Explore and use variables in a program.** Information in the real world can be represented in computer programs. Additionally, different actions are available for different kinds of information. Students should demonstrate the understanding that variables are not just used for numbers; they can also hold text, including whole sentences (strings) or logical values (true or false). Students should also demonstrate that a variable has a data type and is associated with a data storage location. Subconcept: Variables *<u>Grade-by-Grade Indicators:</u>* *K - Describe ways people represent data.* *1 - Explain how numbers are used to represent data.* *2 - Create a simple model to show how a computer stores information using numbers or symbols.* *3 - Identify ways variables are used in programs.* *4 - Modify or remix an existing program that uses a variable.* *5 - Create a program that uses a variable.*
E-AP-03	**Routinely create programs using a variety of tools to express ideas, address a problem or create an artifact, individually and collaboratively.** Programming is used as a tool to create products that reflect a wide range of interests, including to solve a problem, express and idea or create an artifact. People work together to plan, create and test programs within a context that is relevant to the programmer and those who will use the program. When creating programs, students need to have opportunities to work both individually and with peers. For young learners, collaboration through programming should be encouraged. Student should begin exploring the use of simple sequences and simple loops in Kindergarten and progress to using more complex sequences, loops, events, variables and conditionals by 5th grade. Subconcept: Control *<u>Grade-by-Grade Indicators:</u>* *K - Routinely create simple programs, independently OR collaboratively.* *1 - Routinely create simple programs, independently AND collaboratively.* *2 - Routinely create simple programs with sequences using a variety of tools, independently and collaboratively.* *3 - Routinely create simple programs with sequences or events using a variety of tools, independently and collaboratively.*

Identifier	Standard & Description
	4 - Routinely create simple programs with sequences, events or loops routinely using a variety of tools, independently and collaboratively. *5 - Routinely create simple programs with sequences, events, loops, variables or conditionals routinely using a variety of tools, independently and collaboratively.*
E-AP-04	**Decompose precise steps needed to solve a problem.** Decomposition is the act of breaking down tasks into smaller tasks. Smaller tasks or sub parts (steps that can be broken down into smaller steps) may be broken down even further. The process of decomposition assists in areas of program development by enabling different people to work on different parts at the same time. Students should demonstrate the process of decomposition by enabling different people to work on different parts of program development at the same time. Subconcept: Modularity <u>Grade-by-Grade Indicators:</u> *K - Generate the steps needed to solve a simple problem with prompting and support.* *1 - Generate the steps needed to solve a simple problem.* *2 - Generate and correctly order the steps needed to solve a simple problem.* *3 - Generate and correctly order the steps needed to solve a complex problem.* *4 - Decompose a problem into parts to facilitate program development.* *5 - Decompose a problem into parts and subparts to facilitate program development.*
E-AP-05	**Use a process when creating programs or computational artifacts.** Students demonstrate the use of formal and informal processes for creating computational artifacts or programs include processes to: ask, imagine, plan, create, test and improve, share; or a creative thinking spiral (i.e. imagine, create, play, share, reflect); and design thinking (empathize, define, ideate, prototype, test). Students demonstrate understanding that these processes are iterative: designed for students to cycle through more than once in order to improve or modify the design and reach the best possible result. Subconcept: Modularity <u>Grade-by-Grade Indicators:</u> *K - Use a process when creating simple programs, individually OR collaboratively, with prompting and support.* *1 - Use a process to create simple programs, individually AND collaboratively, with prompting and support.* *2 - Use a process to create simple programs that include sequences.* *3 - Use a process to create programs that include sequences and events.* *4 - Use a process to create programs that includes loops, sequences or events.* *5 - Use a process to create programs that include loops, sequences, events, variables or conditions.*

Identifier	Standard & Description
	Modify, remix or reuse part of an existing program to create a new program, giving attribution to others. The design of a new program often involves existing code or remixing other programs within a community (a group of people who share and provide feedback on another's creations). Students should credit the original creator when remixing a program or when ideas are borrowed and iterated upon. Students should also consider common licenses that place limitations or restrictions on the use of computational artifacts such as images and music downloaded from the Internet. At this stage, attribution should be written in the format required by the teacher and should always be included on any programs shared online.
E-AP-06	Subconcept: Program Development <u>*Grade-by-Grade Indicators:*</u> *K - Not introduced until 3rd grade* *1 - Not introduced until 3rd grade* *2 - Not introduced until 3rd grade* *3 - Modify or add features to an existing program, with prompting and support, to create a new program, giving attribution.* *4 - Modify, remix or reuse parts of an existing program to create a new program, giving attribution.* *5 - Modify, remix, reuse parts or add features to an existing program to create a new program, giving attribution.*
	Document, share and reflect when creating programs using correct terminology. Documentation of the design process allows students and others to better understand a program. In addition, students need to have opportunities to discuss, share and receive feedback from peers and adults when creating and refining projects. Students should be using correct, age-appropriate terminology when sharing their ideas both verbally and written.
E-AP-07	Subconcept: Program Development <u>*Grade-by-Grade Indicators:*</u> *K - Document simple programs, using pictures, in order to share process with others.* *1 - Document simple programs, using pictures, in order to share with others and reflect on the process.* *2 - Document simple programs, with pictures and/or text, to share with others and reflect on the process.* *3 - Document programs and discuss development process with peers.* *4 - Document programs and discuss development process with peers, using correct terminology.* *5 - Document programs using correct terminology and incorporate peer feedback in the development process.*

Identifier	Standard & Description
E-AP-08	**Identify and correct errors in an algorithm.** Debugging is the process of isolating and correcting "bugs" in a program. As part of the debugging process, students demonstrate the importance of determining if the program is fixable (What happened? What was supposed to happen? What does this tell you? Is it fixable?). Students demonstrate use of an iterative process (repeating steps to improve desired result) when programming aids in the detection and isolation of programming errors. Subconcept: Program Development <u>Grade-by-Grade Indicators:</u> *K - Analyze and debug simple algorithms with prompting and support.* *1 - Analyze and debug simple algorithms which includes sequencing.* *2 - Analyze and debug algorithms which includes simple loops.* *3 - Analyze and debug algorithms which includes sequencing and loops.* *4 - Analyze and debug algorithms which includes sequencing, loops and events.* *5 - Analyze and debug algorithms which includes sequencing, loops, events and conditionals.*

Concept: **Impacts of Computing**

Identifier	Standard & Description
E-IC-01	**Discuss how computing has impacted society.** Students demonstrate an understanding that computing technology has positively and negatively changed the way people live and work. Subconcept: Culture <u>Grade-by-Grade Indicators:</u> *K - Make observations to describe ways computing devices are used daily life.* *1 - Describe computing devices used in different careers.* *2 - Demonstrate how some tasks can be completed with or without a computing device.* *3 - Describe how computing technology impacts the way people live, work, and interact.* *4 - Compare and contrast how computing has changed society from the past to the present.* *5 - Describe the positive and negative impacts of computing on society.*
E-IC-02	**Discover how computing devices have affected the way people communicate.** Computing provides the possibility for constantly connected communications. Students

Identifier	Standard & Description
	demonstrate an understanding of communications when connected and disconnected. Students also demonstrate collaboration and the sharing of ideas to allow the benefit of diverse perspectives while also demonstrating collaboration using technology can be synchronous (occurring at the same time) or asynchronous (not occurring at the same time). Subconcept: Social Interactions *Grade-by-Grade Indicators:* *K - Describe different computing devices used for communication.* *1 - Describe ways people can communicate using computing devices.* *2 - Compare similarities and differences between in person and online communications.* *3 - Describe ways in which computing devices could be made more accessible to all users.* *4 - Use online collaborative spaces ethically and safely to work with other students to solve a problem or reach a goal.* *5 - Compare diverse perspectives, synchronously or asynchronously, to improve a project.*
E-IC-03	**Evaluate the relevance and appropriateness of electronic information sources and digital media.** Students should consider who owns digital sources they wish to use. Students should develop an understanding that while technology makes it easy to share digital media and electronic information sources, it is important to follow the rules of using other people's work and give attribution. Knowledge of specific copyright laws are not expected at this level. Subconcept: Safety, Law & Ethics *Grade-by-Grade Indicators:* *K - Describe characteristics of a website, with prompting and support.* *1 - Describe the purpose of different websites, with prompting and support.* *2 - Use and cite sources from approved digital materials.* *3 - Describe the relevance and appropriateness of various electronic information sources and digital media.* *4 - Compare the relevance and appropriateness of various electronic information sources and digital media.* *5 - Use relevant and appropriate electronic information sources and digital media, citing resources, for various tasks.*
E-IC-04	**Understand the importance of proper use of data and information in a computing society.** Online communication facilitates positive and negative interactions; consequently, it is important to protect our data, devices and the information stored on them. Students demonstrate the importance of using data properly including what, how, when, and

Identifier	Standard & Description
	with whom to share. Subconcept: Safety, Law & Ethics *Grade-by-Grade Indicators:* *K - Describe characteristics of private information.* *1 - Identify harmful behaviors when using a connected device.* *2 - Demonstrate appropriate behavior when sending messages online.* *3 - Describe positive qualities of a digital citizen.* *4 - Describe potential strategies to manage and eliminate cyberbullying.* *5 - Understand consequences for sending or receiving inappropriate content.*

Concept: **Computing Systems**

Identifier	Standard & Description
E-CS-01	**Identify, select and operate appropriate software and hardware to perform a variety of tasks and recognize that users have different needs and preferences for the technology they use.** People use computing devices to perform a variety of tasks accurately and quickly. Students should be able to use the appropriate app/program/software for tasks they are required to complete. Students should be able to successfully use designated hardware device(s) for appropriate programs. Subconcept: Devices *Grade-by-Grade Indicators:* *K - Describe ways people use digital devices to perform tasks.* *1 - Use the appropriate device and application or software to complete a given task, with prompting and support.* *2 - Describe and use the appropriate device and application or software to complete a given task.* *3 - Compare and contrast various types and functions of software or applications.* *4 - Describe the capabilities and limitations of various software and applications for a particular use.* *5 - Justify selection of a particular computing device based on a desired application or task.*
E-CS-02	**Identify and describe the function of common physical components of computing systems (hardware) using appropriate terminology.** A computing system is composed of hardware and software. Hardware consists of physical components. While software consists of programs and other operating information used by the computing system or computer. Students should be able to identify and describe the function of external hardware.

Identifier	Standard & Description
	Subconcept: Hardware & Software *<u>Grade-by-Grade Indicators:</u>* *K - Use appropriate terminology to identify basic hardware.* *1 - Use appropriate terminology to identify basic software.* *2 - Describe the function of common hardware and software.* *3 - Compare and contrast features of different digital devices.* *4 - Describe the capabilities and limitations of various digital devices.* *5 - Describe the function of major hardware components of a digital device.*
E-CS-03	**Describe basic hardware and software problems using accurate terminology.** Problems with computing systems have different causes. Students should demonstrate the ability to communicate a computing system problem with accurate terminology and begin to form an understanding of possible causes. Subconcept: Troubleshooting *<u>Grade-by-Grade Indicators:</u>* *K - Identify a simple hardware problem.* *1 - Describe simple hardware and software problems.* *2 - Use observations to distinguish between simple hardware and software problems.* *3 - Demonstrate common troubleshooting strategies to solve simple hardware and software problems.* *4 - Describe the causes of hardware, software and connectivity problems.* *5 - Demonstrate an appropriate response to various error messages and identify the component and/or application causing the error.*

Middle School (6-8)

Kentucky Academic Standards for **Computer Science**

Middle School (6-8) Computer Science Standards

Concept: **Networks and The Internet**

Identifier	Standard & Description
M-NI-01	**Model how different sets of rules (protocols) are used to transmit different types of data across networks and the Internet.** Different protocols, such as TCP/IP, HTTP, FTP, SMTP are used for different types of data. Web traffic uses one protocol (HTTP), while email traffic uses another (SMTP). At this level, the mechanism of how the protocols work is not important. Modeling different protocols could be accomplished using diagrams, analogies, etc. Subconcept: Network Communication & Organization
M-NI-02	**Model how information is disguised using different methods of encryption to secure it during transmission from one point to another.** Encryption helps to secure data so that only the intended recipients can read it. Types of encryption include symmetric and asymmetric encryption which take advantage of keys. Encoding and decoding messages can be modeled through the use of simple letter substitution or through more complicated methods, such as public key encryption. Subconcept: Cybersecurity
M-NI-03	**Explain how physical and digital security practices and measures proactively address the threat of breaches to personal and private data.** 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. Subconcept: Cybersecurity

Concept: **Data and Analysis**

Identifier	Standard & Description
M-DA-01	**Store data using multiple encoding methods.** Data can be stored in multiple formats, from the selection of software packages for text (e.g. txt, rtf, log, docx, etc.), to image representation (jpeg, tiff, gif, png, etc.), to video and sound information (mp3, mpeg-4, mov, etc), and to storage of data into organized formats (e.g., tables). Choosing the most appropriate data storage format for a specific scenario is key to ensuring optimal data accessibility and use. Subconcept: Storage

Identifier	Standard & Description
M-DA-02	**Collect data using computational tools and transform the data to make it more useful and reliable.** Computational tools are used to collect, visualize, and transform data. Appropriate transformation of data helps 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). Subconcept: Collection, Visualization & Transformation
M-DA-3	**Refine computational models based on the data they have generated.** A model may be a programmed simulation of events or a representation of how various data is related. Refining a model involves choosing relevant data points, analyzing how data points relate to each other, and evaluating the accuracy of the data. Subconcept: Inference & Models

Concept: **Algorithms and Programming**

Identifier	Standard & Description
M-AP-01	**Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.** Collaboration is a common and crucial practice in programming development. Program developers often take on varying roles during the design, implementation and review stages of program development, including but not limited to graphic design and code writing. Subconcept: Program Development
M-AP-02	**Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.** Decompose (break down) problems into smaller, more manageable, individual steps. Stepping through the execution of a program is a common practice when debugging and ensuring the accuracy of the program. Subconcept: Modularity
M-AP-03	**Seek and incorporate feedback from team members and users to refine a solution that meets user needs.** Solicit diverse perspectives throughout the design process to improve 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.

Identifier	Standard & Description
	Subconcept: Program Development
M-AP-04	**Create flowcharts and/or pseudocode to address complex problems as algorithms.** A flowchart visually represents an algorithm used to solve a problem. Pseudocode uses a written language. Both are means of logically thinking through a problem before actual programming begins and identify the steps needed to process input(s) to produce the desired output(s). Subconcept: Algorithms
M-AP-05	**Create clearly named variables that represent different data types and perform operations on their values.** A variable is like a container with a name, in which the contents may change, but the name (identifier) does not. When planning and developing programs decide when and how to declare and name new variables. Determine the appropriate type and size of variable to use. Use naming conventions to improve program readability. Subconcept: Variables
M-AP-06	**Create procedures with parameters to organize code and make it easier to reuse.** Create procedures and/or functions that are used multiple times within a program to repeat groups of instructions. Define parameters within the procedure that allow for varying input. Subconcept: Modularity
M-AP-07	**Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.** Control structures can be combined in many ways. Nested loops are loops placed within other loops. Compound conditional statements use two or more conditions (e.g., AND, OR, and NOT) in a logical relationship. Nesting conditionals within one another allows the result of one conditional to lead to another. Subconcept: Control
M-AP-08	**Incorporate existing code, media, and libraries into original programs, and give attribution.** Insert portions of digital media in their own programs and websites. May also import libraries and connect to web application program interfaces (APIs). Subconcept: Program Development

Identifier	Standard & Description
M-AP-09	**Systematically test and refine programs using a range of test cases.** Evaluate whether programs function as intended. Testing should be a deliberate process that is more iterative, systematic, and proactive than at lower levels. Test programs for potential errors. Subconcept: Program Development
M-AP-10	**Document programs in order to make them easier to follow, test, and debug.** Provide documentation for end users that explains their artifacts and how they function. Proper documentation aids in debugging and future program modification. Subconcept: Program Development
M-AP-11	**Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.** Consider licensing implications for their own work, especially when incorporating libraries and other resources. When considering two software libraries that address a similar need, a choice could be justified based on the library that has the least restrictive license. Subconcept: Program Development
M-AP-12	**Develop a process creating a computational artifact that leads to a minimum viable product followed by reflection, analysis, and iteration.** 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. Subconcept: Program Development

Concept: **Impacts of Computing**

Identifier	Standard & Description
M-IC-01	**Discuss issues of bias and accessibility in existing technologies.** Every device has inherent issues that influence usage and adoption across all users. Issues of pricing, production, user access and interface design, reliability, sustainability, and exclusivity all influence existing technologies and how they are used by different populations. Subconcept: Culture
M-IC-02	**Compare the positive & negative effects of computing technologies on society.** The use of technology has reshaped our society in ways not imagined a few short decades ago. Social media, online video, apps, and cloud services have changed the way we interact and those changes should be actively investigated and explored to see the full effects. Subconcept: Culture
M-IC-03	**Collaborate with others using appropriate tools at the local, national, and/or international levels.** Promoting positive experiences in shared work environments is a key component of digital citizenship. Learning to respect the opinions of others and use collaborative spaces to work toward common goals is important to participate effectively in a growing global community. Opportunities for collaboration should be made available to students using different tools and different platforms. Subconcept: Social Interactions
M-IC-04	**Discuss the benefits and consequences of making information either public or private.** How we choose to share information in the digital age has far-reaching implications for the lives of students and the schools they attend. Discussions should take place on what information can and should be shared and how to keep private information private or not accessible online at all. Subconcept: Safety, Law & Ethics

Concept: **Computing Systems**

Identifier	Standard & Description
M-CS-01	**Recommend improvements to the design of computing devices based on an analysis of how users interact with the devices.** The study of human–computer interaction (HCI) can improve the design of devices, addressing both hardware and software. Suggest improvements to existing devices. Design alternative components or interfaces (controllers, graphical interfaces, peripherals, etc.). Evaluate usability based on various metrics, including accessibility, ergonomics, learnability. Subconcept: Devices
M-CS-02	**Design projects that combine hardware and software components to collect and exchange data.** Collecting and exchanging data involves input, output, storage, and processing. Select appropriate hardware and software components for project designs by considering factors such as functionality, cost, size, mobility, speed, accessibility, and aesthetics. Subconcept: Hardware & Software
M-CS-03	**Identify and fix problems with computing devices and their components systematically.** Since a computing device may interact with other interconnected devices within a system, problems may be due to either the specific computing device itself or to devices connected to it. Create and follow detailed, structured processes for troubleshooting problems within computing systems and ensuring that potential solutions are not overlooked. Develop flow diagrams, modify software to test hardware, check connections and settings, and swap in working components. Subconcept: Troubleshooting

High School (9-12)

Kentucky Academic Standards for **Computer Science**

High School (9-12) Computer Science

Networks and the Internet

Identifier	Standard & Descriptor
H-NI-01	**Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, end devices, topology, and addressing.** A computer network consists of a set of computational devices that are interconnected by wired or wireless network mediums to allow for sharing resources and enabling convenient communication. Networks environments are created by switches and communicate to other networks through routers. Devices are assigned unique addresses to identify devices in the network. Students should be able to describe different types of networks, network topologies, networking devices, and how they facilitate or limit the network growth (scalability). Subconcept: Network Communication & Organization
H-NI-02	**Give examples to illustrate how sensitive data can be affected by viruses, malware and other attacks.** Network security depends on a combination of hardware, software, and practices that control access to data and systems. Students should be able to discuss potential network attacks such as denial of service attacks, ransomware, viruses, worms, spyware, phishing, and how these attacks present threats to sensitive data; to discuss real-life examples of such threats; and to understand the safety precautions and good practices to protect privacy when using public and private networks. Subconcept: Cybersecurity
H-NI-03	**Recommend security measures to address various scenarios based on factors such as usability, efficiency, feasibility, and ethical impacts.** Security measures have been developed for the protection of data and network operation. Security measures involve tradeoffs between the usability and security of the system. Students should be able to determine the appropriate level of implemented security, based on the evaluation of the users' needs regarding the usability, proper operation of the network, and sensitivity of data. Subconcept: Cybersecurity
H-NI-04	**Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology). *** The structure and makeup of networks, and network properties affect networks functionality. Students should be able to describe the significant features of networks, network topologies, the speed and capacity of a network. (bandwidth vs. latency), network protocols, and use simulators to experiment with network topologies, functionality, and reliability.

Identifier	Standard & Descriptor
	Subconcept: Network Communication & Organization
H-NI-05	**Compare ways software developers protect devices and information from unauthorized access. *** Because most devices are not secure, choosing to implement security measures involves trade-offs between the usability and security of the system as well as the individual needs of the user. Students should be able to discuss the difference between symmetric and asymmetric encryption and should be able to describe encryption algorithms, hashing and various message authentication methods. Subconcept: Cybersecurity

Data & Analysis

Identifier	Standard & Descriptor
H-DA-01	**Evaluate the tradeoffs in how data elements are organized and where data is stored.*** Students should be able to evaluate the organization and storage of data depending on the type of data (e.g., text, video, audio, image, number, etc.), the storage medium, and the data retrieval needs. Subconcept: Storage
H-DA-02	**Collect data using appropriate data collection tools and techniques to support a claim or to communicate information.** Data can be collected using a variety of technological and non-technological tools and resources. A claim or information should be supported with appropriate data. The techniques for collecting the data depend on the tool used and the purpose of the data. Students should be able to evaluate, recommend and use data collection tools in order to support a claim or provide information to an appropriate population. Subconcept: Collection, Visualization & Transformation
H-DA-03	**Understand and design database structures to optimize search and retrieval.*** Students should be able to design and explain how data can be organized in a database by using data structures (e.g., fields, records, rows, tables, relationships, etc.) to facilitate search and retrieval and to reduce file sizes. Students should know about different database models (e.g., relational, object oriented, distributed, online, etc.). Subconcept: Collection, Visualization & Transformation
H-DA-04	**Explain the privacy concerns related to the collection and generation of data.** Privacy concerns exist wherever personally identifiable information or other sensitive information is collected, stored, used, and finally destroyed or deleted – in digital form or

Identifier	Standard & Descriptor
	otherwise. Information and privacy can be exploited if privacy and other protections are ignored (selling of identifiable information to third parties, background tracking of internet searches). Students should be able to explain that technology allows the collection, use, generation, and possible exploitation of data by private, commercial, and government entities. Subconcept: Collection, Visualization & Transformation
H-DA-05	**Use data analysis tools (e.g. formulas and other software data / statistical tools) to process and transform the data to make it more useful and reliable.** Students should be able to take data and make it meaningful using data analysis tools. Data analysis tools contain advanced features, including formulas and statistical functions, that facilitate proper inference process. Subconcept: Collection, Visualization & Transformation
H-DA-06	**Use data analysis tools and techniques to identify patterns and analyze data represented in complex systems.** Students should be able to use data analysis tools to allow for the extraction of information thus enabling opportunity to identify trends, make discoveries and connections. Subconcept: Inference & Models
H-DA-07	**Create computational models that represent the relationships among different elements of data.** Students should be able to create computational models that represent the structure and inter-dependencies in data, in order to facilitate processing of information, and to identify patterns within data sets. Subconcept: Inference & Models
H-DA-08	**Create interactive data visualizations using software tools to help others better understand real-world phenomena.** Students should be able to use interactive software tools that allow for effective discovery through visualizations by communicating understanding and knowledge from digitally represented sources. Subconcept: Collection, Visualization & Transformation
H-DA-09	**Evaluate the ability of models and simulations to test and support the refinement of hypotheses.*** Students should be able to support and refine hypotheses using computational models. Computational models may differ in functionality and relevance based upon how data is recognized and utilized.

Identifier	Standard & Descriptor
	Subconcept: Inference & Models

Algorithms & Programming

Identifier	Standard & Descriptor
H-AP-01	**Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.** At previous levels, students adhered to licensing schemes. At this level, students should consider and be able to explain licensing implications for their own work, especially when incorporating libraries and other resources. Subconcept: Program Development
H-AP-02	**Use a development process in creating a computational artifact that leads to a minimum viable product followed by reflection, analysis, and iteration.** At previous levels students have developed artifacts in ad-hoc way. By high school, students should be introduced to the discipline of software development. In particular, we focus on analysis, reflection and iteration. When given a problem students should be able to fully explain the problem, consider possible ways to solve it and then apply one of the possible ways. Consideration of possible ways to solve a software problem is termed 'Software Design' and/or 'Analysis'. Once the solution has been implemented and tested, students should reflect on what worked and didn't work during that process. Students will then apply this process again to update the artifact (iteration), continuing to refine the artifact itself while also continuing to improve the process. Subconcept: Program Development
H-AP-03	**Use functions, data structures or objects to simplify solutions, generalizing computational problems instead of repeated use of simple variables.** Students should be able to identify common features in multiple segments of code and substitute a single segment/abstraction (function, data structure or object) to account for the differences. Subconcept: Variables
H-AP-04	**Design and iteratively develop event-driven computational artifacts for practical intent, personal expression, or to address a societal issue.** 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. Students should be able to design and implement procedures that are called by events. Subconcept: Program Development

Identifier	Standard & Descriptor
	Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
H-AP-05	Students should be able to decompose complex problems into manageable sub problems that could potentially be solved with programs or procedures that already exist. Subconcept: Modularity
	Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance and explain the benefits and drawbacks of choices made.
H-AP-06	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 should include conditional statements, loops, and event handlers. Subconcept: Control
	Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
H-AP-07	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. 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, re-usability, and ease of implementation of an algorithm. Subconcept: Algorithms
	Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
H-AP-08	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. 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.

Identifier	Standard & Descriptor
	Subconcept: Program Development
H-AP-09	**Evaluate and refine computational artifacts to make them more usable and accessible using systematic testing and debugging.** 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. Students should respond to the changing needs and expectations of end users and improve the performance, reliability, usability, and accessibility of artifacts. Subconcept: Program Development
H-AP-10	**Systematically design and develop programs for broad audiences by incorporating feedback from users.** Students at lower levels collect feedback and revise programs. At this level, students should do iterations through a systematic process that includes feedback from broad audiences. It is important for students to be able to gather feedback from the audience including peers, teachers and family members to make design decisions based on the feedback. Subconcept: Program Development
H-AP-11	**Design and develop computational artifacts working in team roles using collaborative tools.*** As programs grow more complex, the choice of resources that aid program development becomes increasingly important and should be made by the students. 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. Subconcept: Program Development
H-AP-12	**Describe how artificial intelligence drives many software and physical systems.*** Artificial Intelligence, unlike algorithms, is a mechanism where programs make decisions on what to execute based on its environment similar to how humans make decisions. In the era of big data, artificial intelligence is becoming pervasive and students should be able to describe how AI is used in everyday software systems. Subconcept: Program Development
H-AP-13	**Use and adapt classic algorithms to solve computational problems.*** Students should be able to identify and use well-known algorithms in sorting (e.g., bubble sort, quicksort, merge sort, insertion sort), searching (e.g., linear search, binary search),

Identifier	Standard & Descriptor
	and shortest-path (e.g., Dijkstra's algorithm) problems. Students will also be able to adapt and combine such well-known algorithms to add features that address more complex computational tasks. Subconcept: Algorithms
H-AP-14	**Evaluate algorithms in terms of their efficiency, correctness, and clarity.*** Students should be able to calculate the total number times a loop will be executed given a code snippet, will be able to state whether an algorithm is correct for solving a given problem, and compare/contrast algorithms for clarity and the number of executed operations. Subconcept: Algorithms
H-AP-15	**Compare and contrast fundamental data structures and their uses.*** Students should be able to name the fundamental data structures (array, list, stack, queue and tree) and defend the use of a data structure's use to solve different problems in sorting and searching. Subconcept: Control
H-AP-16	**Illustrate the flow of execution of a recursive algorithm.*** A recursive algorithm is a procedure or function which when implemented in a programming language calls itself. The algorithm solves a smaller problem with each call. Students should be able to identify the termination case and recursive call case in a recursive algorithm and describe the state of the problem space during each call. Subconcept: Algorithms
H-AP-17	**Construct solutions to problems using student-created components, such as procedures, modules and/or objects.*** At this level, students should be regularly implementing programming solutions using some form of structured design with multiple functions/procedures/modules in different files. Object-oriented programming is optional at this level. Problems can be assigned or student-selected. Subconcept: Program Development
H-AP-18	**Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution.*** As students encounter complex, real-world problems that span multiple disciplines or social systems, they should decompose complex problems into manageable sub problems that could potentially be solved with programs or procedures that already exist. Subconcept: Modularity

Identifier	Standard & Descriptor
H-AP-19	**Select and employ an appropriate component or library to facilitate programming solutions.*** Students should be able to use (or, actually reuse) existing code when quality code already exists to accomplish the needed task. Libraries and APIs can be student-created, part of the software development platform or external libraries or APIs selected for their features. Subconcept: Program Development
H-AP-20	**Develop programs for multiple computing platforms.*** Students need to understand the pervasiveness of computing and that computers exist in many different forms. Students should be able to develop software programs that run on a desktop/laptop as well as other IOT and/or mobile devices. Subconcept: Program Development
H-AP-21	**Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project.*** Students need to learn the major tools and skills of software development including version control systems to keep track of all releases (and to back out changes), IDEs to simplify writing and testing of code, documentation of code for code maintenance. Students should be able to explain how these tools are critical to team-developed projects. Group software projects can be assigned or student-selected. Subconcept: Control
H-AP-22	**Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., introducing errors).*** Students should be able to understand and provide examples highlighting that adding functionality to software can introduce new errors, reduce functionality or add an unintended feature. Subconcept: Program Development
H-AP-23	**Evaluate key qualities (including correctness, usability, readability, and efficiency) of a program.*** Given a software program, students should be able to evaluate it in terms of software quality using standard software metrics including readability, usability and efficiency. Subconcept: Program Development
H-AP-24	**Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems.*** Students should be able to explain the difference between a compiled and scripted programming language, defend a choice of a programming language for a certain

Identifier	Standard & Descriptor
	computing device and defend a choice of a language (3rd generation versus 4th generation) for solving different types of problems. Subconcept: Program Development

Impacts of Computing

Identifier	Standard & Descriptor
H-IC-01	**Reduce bias and equity deficits through the design of accessible computational artifacts.** Within the context of computing, one must account for the factors of equity, ethics, access and training in the design of computational artifacts for diverse populations. The ability identify potential bias in one's own work and apply professional practices associated with increasing accessibility in the design of one's own artifacts is a crucial skill in computational development. Subconcept: Culture
H-IC-02	**Evaluate and assess how computing impacts personal, ethical, social, economic, and cultural practices.** Within the context of computing, one must account for the factors of equity, ethics, access and training when developing products for a variety of end users. Computer science requires practitioners to evaluate the accessibility of a product to a global society and assess the implications on that society. Subconcept: Social Interactions
H-IC-03	**Research how computational innovations that have revolutionized aspects of our culture might have evolved from a need to solve a problem.** Computational design can share features across disciplines (i.e. art, music etc.) by translating human intention into an artifact through algorithmic development and the need to solve a problem. Students should conduct research relating to the evolution of a computing innovation from the need to address a perceived need or solve a problem in any discipline or career field. Subconcept: Culture
H-IC-04	**Explain the beneficial and harmful effects that laws governing data (intellectual property, privacy etc.) can have on innovation.** International differences in laws and ethics have implications for computing in a global society (i.e. privacy, data, property, information, and identity). Students should be aware of intellectual property laws and be able to explain how they are used to protect the interests of innovators or abused for financial gain.

Identifier	Standard & Descriptor
	Subconcept: Safety, Law & Ethics
H-IC-05	**Evaluate and design computational artifacts to maximize their benefit to society.*** Within the context of computing, one must account for the factors of equity, security, ethics, access and privacy in the design of computational artifacts for diverse populations. Students should be able to identify potential bias in the work of others and make suggestions in order to make them more beneficial in a diverse society as well as decrease security deficits that could result in harms to culture, society or the economy. Subconcept: Safety, Law & Ethics
H-IC-06	**Evaluate the impact of the digital divide (i.e. inequity of computing access, education and influence) on the development of local communities and society.** Within the context of computing, one must account for the factors of equity, ethics, access and training when developing products for a variety of end users. Students should be able to evaluate the effect the digital divide has and can have on development, innovation and the culture of society. Subconcept: Culture
H-IC-07	**Demonstrate ways computational design (i.e. algorithms, abstractions and analysis) can apply to problems across disciplines.*** Computational design can share features across disciplines (i.e. art, music etc.) by translating human intention into an artifact through algorithmic development and the need to solve a problem. Students should be able to demonstrate how these features are shared across disciplines and how real-world problems can be solved using computational methods. Subconcept: Culture
H-IC-08	**Debate laws and regulations that impact the development and use of software and the protection of privacy.** International differences in laws and ethics have implications for computing in a global society (i.e. privacy, data, property, information, and identity). Students should evaluate case studies or current events which present an ethical dilemma contrasting an individual's right to privacy and the safety, security, or well-being of a community. Subconcept: Safety, Law & Ethics

Computing Systems

Identifier	Standard & Descriptor
H-CS-01	**Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.** Computing devices are often integrated with other systems (biological, mechanical, and social etc.). Students should be able to select an embedded device, identify the types of data and procedures it includes, and explain how the implementation details are hidden from the user. Subconcept: Devices
H-CS-02	**Compare levels of abstraction and interactions between application software, system software and hardware layers.** At its most basic level, computers and computing system are composed of physical hardware with the ability to store, interpret and send bits. Some complex solutions use a multiple layer model where layers of software are built upon the hardware and interact only with the layers above and below them to separate functions and responsibilities in order to reduce system complexity. Students should be able to understand the abstraction in the layer model, and its benefits. Subconcept: Hardware & Software
H-CS-03	**Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.** 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. Students should be able to develop versions of troubleshooting guidelines based upon test cases of their choosing. Subconcept: Troubleshooting
H-CS-04	**Categorize the roles of operating system software.** Students should be able to define an operating system and identify its different roles and characteristics. Subconcept: Hardware & Software
H-CS-05	**Illustrate ways computing systems implement logic, input, and output through hardware components.*** Students should be able to describe, via pictures or prose, the basic von Neumann architecture, CPU and I/O processing and the fetch, decode and execute cycle. Subconcept: Hardware & Software

References

Computer Science Teachers Association. (2017). CSTA Computer Science Standards. Retrieved from https://www.csteachers.org/page/standards

K-12 Computer Science Framework Steering Committee. (2016). K-12 computer science framework. Retrieved from https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf

Kentucky Department of Education. (2015). *Kentucky Academic Standards Technology*. Retrieved from https://education.ky.gov/curriculum/standards/kyacadstand/Documents/Kentucky_Academic_Standards_Technology.pdf

Office of Educational Technology. (2017). *Reimagining the role of technology in education: National Education Technology Plan Update.* US Department of Education. Retrieved from http://tech.ed.gov/netp/

Promote computer science. (2017, December 1). Retrieved from https://code.org/promote/ky

Southern Regional Education Board. (2016). Bridging the computer science education gap. Retrieved from https://www.sreb.org/publication/bridging-computer-science-education-gap

Appendix A: Glossary of Terms

Disciplinary terms are used throughout the *Kentucky Academic Standards for Computer Science* and its supporting materials. The following interactive glossary, hosted by the K12 Computer Science Framework, provides an alphabetical list of key definitions and descriptions of valuable terms. The glossary includes definitions of terms used in the standard statements, the standard descriptions and throughout the framework. These terms are defined for readers of the standards and are not necessarily intended to be used as the exclusive definitions or terms that are seen by students.

Online glossary from K12 Computer Science Framework: https://k12cs.org/glossary/

Appendix B: Writing and Review Committees

The writing team, composed of current computer science or technology-related teachers and education professionals, represented both rural and urban settings – including representation from large, medium and small districts from all regions of the state. While these teachers taught a variety of courses and grade levels throughout their careers, the selected committee members were currently teaching or leading areas related to the standards development process. Additionally, the selected writers served in many roles in their schools, technology or computer science community and a wide variety of professional organizations. To ensure fidelity to the standards, the writing committee provided feedback at all stages of the development process. The writing and review committee members listed below represented Kentucky's best as evidenced by their countless qualifications.

Writing Team Members

Dr. Andrea Peach, Georgetown College
Donnie Piercey, Eminence Independent Schools
Heather Korrell, Hardin County Schools
Dr. Jerzy Jaromczyk, University of Kentucky
Joe Beers, Jessamine County Schools
Joy Neace, Lee County Schools
Kerri Snow, Pikeville Independent Schools
Dr. Leanna Prater, Fayette County Schools
Dr. Maureen Doyle, Norther Kentucky University
Melissa Metcalf, Madision County Schools
Mike Paul, Bardstown Independent Schools
Nikkol Bauer, Henry County Schools
Sandra Hancock, Christian County Schools
Dr. Sarah Bumpas, Jefferson County Schools
Scott Dossett, McCracken County Schools
Scott Horan, Jefferson County Schools
Sean Jackson, Mason County Schools
Shane Jordan, Russell Independent Schools
Stephanie Younger, Pike County Schools
Steven May, Daviess County Schools
Tabetha Cooksey, Cumberland County Schools

Oversight and Feedback Team Members

Aaron Yeiser, Daviess County Schools
Dr. Adel Elmaghraby, University of Louisville
Dr. George Landon, Eastern Kentucky University
Holli McClelland, Fairview Independent Schools
Dr. Joan Mazur, University of Kentucky
Mitch Hawkins, Clay County Schools

Sonja Fischer, Fort Thomas Independent Schools
Tabetha Housekeeper, Scott County Schools
Ruth Ann Gumm, Barren County Schools

Business and Industry Trusted Partners
Ankur Gopal, Interapt
Jena Collins, Apple Inc.
John Allen, Google
Kentucky Computer Science Teachers Association
Mardi Montgomery, Education & Workforce Development Cabinet
Melissa Rowe, Amazon
Monique Morton-Rice, AdvanceKentucky
Nick Such, Awesome Inc.
Sarah Skinner, BrightBytes
Tim Cornett, Microsoft
Vidya Ravichandran, Glowtouch
Dr. Wayne Lewis, Education & Workforce Development Cabinet

Appendix C: Standards Progression Chart

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Networks & the Internet	Network Communication & Organization	**E-NI-01:** Understand the basic components of how networks operate to protect physical and digital information.	**M-NI-01:** Model how different sets of rules (protocols) are used to transmit different types of data across networks and the Internet.	**H-NI-01:** Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, end devices, topology, and addressing.
Networks & the Internet	Network Communication & Organization	function moveForward() { var standard }	function moveForward() { var standard }	**H-NI-04:** Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology). *
Networks & the Internet	Cybersecurity	E-NI-02: Model how information is broken down into smaller pieces (data packets), transmitted over various paths (physical and/or wireless), and reassembled at the destination	M-NI-02: Model how information is disguised using different methods of encryption to secure it during transmission from one point to another.	H-NI-02: Give examples to illustrate how sensitive data can be affected by viruses, malware and other attacks.
Networks & the Internet	Cybersecurity	function moveForward() { var standard }	M-NI-03: Explain how physical and digital security practices and measures proactively address the threat of breaches to personal and private data.	H-NI-03: Recommend security measures to address various scenarios based on factors such as usability, efficiency, feasibility, and ethical impacts.
Networks & the Internet	Cybersecurity	function moveForward() { var standard }	function moveForward() { var standard }	H-NI-05: Compare ways software developers protect devices and information from unauthorized access. *
Data & Analysis	Storage	**E-DA-01:** Appropriately store and modify digital files.	**M-DA-01:** Store data using multiple encoding methods.	**H-DA-01:** Evaluate the tradeoffs in how data elements are organized and where data is stored.*

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Data & Analysis	Collection, Visualization & Transformation	**E-DA-02:** Collect and visually display data using appropriate applications.	**M-DA-02:** Collect data using computational tools and transform the data to make it more useful and reliable.	**H-DA-02:** Collect data using appropriate data collection tools and techniques to support a claim or to communicate information.
Data & Analysis	Collection, Visualization & Transformation	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-03:** Understand and design database structures to optimize search and retrieval.*
Data & Analysis	Collection, Visualization & Transformation	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-04:** Explain the privacy concerns related to the collection and generation of data.
Data & Analysis	Collection, Visualization & Transformation	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-05:** Use data analysis tools (e.g. formulas and other software data / statistical tools) to process and transform the data to make it more useful and reliable.
Data & Analysis	Collection, Visualization & Transformation	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-08:** Create interactive data visualizations using software tools to help others better understand real-world phenomena.
Data & Analysis	Inference & Models	**E-DA-03:** Analyze data for trends and relationships	**M-DA-03:** Refine computational models based on the data they have generated.	**H-DA-06:** Use data analysis tools and techniques to identify patterns and analyze data represented in complex systems.
Data & Analysis	Inference & Models	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-07:** Create computational models that represent the relationships among different elements of data.

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Data & Analysis	Inference & Models	function moveForward() { var standard }	function moveForward() { var standard }	**H-DA-09:** Evaluate the ability of models and simulations to test and support the refinement of hypotheses. *
Algorithms & Programming	Algorithms	**E-AP-01:** Create, follow, compare and refine algorithms for a task.	**M-AP-04:** Create flowcharts and/or pseudocode to address complex problems as algorithms.	**H-AP-07:** Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
Algorithms & Programming	Algorithms	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-13:** Use and adapt classic algorithms to solve computational problems. *
Algorithms & Programming	Algorithms	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-14:** Evaluate algorithms in terms of their efficiency, correctness, and clarity. *
Algorithms & Programming	Algorithms	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-16:** Illustrate the flow of execution of a recursive algorithm. *
Algorithms & Programming	Variables	**E-AP-02:** Explore and use variables in a program.	**M-AP-05:** Create clearly named variables that represent different data types and perform operations on their values.	**H-AP-03:** Use functions, data structures or objects to simplify solutions, generalizing computational problems instead of repeated use of simple variables.
Algorithms & Programming	Control	**E-AP-03:** Routinely create programs using a variety of tools to express ideas, address a problem or create an artifact, individually and collaboratively.	**M-AP-07:** Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.	**H-AP-06:** Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance and explain the benefits and drawbacks of choices made.

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Algorithms & Programming	Control	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-15:** Compare and contrast fundamental data structures and their uses. *
Algorithms & Programming	Control	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-21:** Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project. *
Algorithms & Programming	Modularity	**E-AP-04:** Decompose precise steps needed to solve a problem.	**M-AP-02:** Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs.	**H-AP-05:** Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
Algorithms & Programming	Modularity	**E-AP-05:** Use a process when creating programs or computational artifacts.	**M-AP-06:** Create procedures with parameters to organize code and make it easier to reuse.	**H-AP-18:** Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. *
Algorithms & Programming	Program Development	**E-AP-06:** Modify, remix or reuse part of an existing program to create a new program, giving attribution to others.	**M-AP-01:** Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts.	**H-AP-01:** Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.
Algorithms & Programming	Program Development	**E-AP-07:** Document, share and reflect when creating programs using correct terminology.	**M-AP-12:** Develop a process creating a computational artifact that leads to a minimum viable product followed by reflection, analysis, and iteration.	**H-AP-02:** Use a development process in creating a computational artifact that leads to a minimum viable product followed by reflection, analysis, and iteration.

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Algorithms & Programming	Program Development	**E-AP-08:** Identify and correct errors in an algorithm.	**M-AP-03:** Seek and incorporate feedback from team members and users to refine a solution that meets user needs.	**H-AP-04:** Design and iteratively develop event-driven computational artifacts for practical intent, personal expression, or to address a societal issue.
Algorithms & Programming	Program Development	function moveForward() { var standard }	**M-AP-08:** Incorporate existing code, media, and libraries into original programs, and give attribution.	**H-AP-08:** Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
Algorithms & Programming	Program Development	function moveForward() { var standard }	**M-AP-09:** Systematically test and refine programs using a range of test cases.	**H-AP-09:** Evaluate and refine computational artifacts to make them more usable and accessible using systematic testing and debugging.
Algorithms & Programming	Program Development	function moveForward() { var standard }	**M-AP-10:** Document programs in order to make them easier to follow, test, and debug.	**H-AP-10:** Systematically design and develop programs for broad audiences by incorporating feedback from users.
Algorithms & Programming	Program Development	function moveForward() { var standard }	**M-AP-11:** Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries.	**H-AP-11:** Design and develop computational artifacts working in team roles using collaborative tools. *
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-12:** Describe how artificial intelligence drives many software and physical systems. *

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-17:** Construct solutions to problems using student-created components, such as procedures, modules and/or objects.*
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-19:** Select and employ an appropriate component or library to facilitate programming solutions. *
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-20:** Develop programs for multiple computing platforms. *
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-22:** Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., introducing errors). *
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-23:** Evaluate key qualities (including correctness, usability, readability, and efficiency) of a program. *
Algorithms & Programming	Program Development	function moveForward() { var standard }	function moveForward() { var standard }	**H-AP-24:** Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems. *
Impacts of Computing	Culture	E-IC-01: Discuss how computing has impacted society.	**M-IC-01:** Discuss issues of bias and accessibility in existing technologies.	**H-IC-01:** Reduce bias and equity deficits through the design of accessible computational artifacts.

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Impacts of Computing	Culture	function moveForward() { var standard }	**M-IC-02:** Compare the positive & negative effects of computing technologies on society.	**H-IC-03:** Research how computational innovations that have revolutionized aspects of our culture might have evolved from a need to solve a problem.
Impacts of Computing	Culture	function moveForward() { var standard }	function moveForward() { var standard }	**H-IC-06:** Evaluate the impact of the digital divide (i.e. inequity of computing access, education and influence) on the development of local communities and society.
Impacts of Computing	Culture	function moveForward() { var standard }	function moveForward() { var standard }	**H-IC-07:** Demonstrate ways computational design (i.e. algorithms, abstractions and analysis) can apply to problems across disciplines. *
Impacts of Computing	Social Interactions	**E-IC-02:** Discover how computing devices have affected the way people communicate.	**M-IC-03:** Collaborate with others using appropriate tools at the local, national, and/or international levels.	**H-IC-02:** Evaluate and assess how computing impacts personal, ethical, social, economic, and cultural practices.
Impacts of Computing	Safety, Law & Ethics	**E-IC-03:** Evaluate the relevance and appropriateness of electronic information sources and digital media.	function moveForward() { var standard }	**H-IC-04:** Explain the beneficial and harmful effects that laws governing data (intellectual property, privacy etc.) can have on innovation.
Impacts of Computing	Safety, Law & Ethics	**E-IC-04:** Understand the importance of proper use of data and information in a computing society.	function moveForward() { var standard }	**H-IC-05:** Evaluate and design computational artifacts to maximize their benefit to society. *
Impacts of Computing	Safety, Law & Ethics	function moveForward() { var standard }	M-IC-04: Discuss the benefits and consequences of making information either public or private.	**H-IC-08:** Debate laws and regulations that impact the development and use of software and the protection of privacy.

Progression Chart

Concept	Subconcept	Grades K-5 By the end of Grade 5, students will be able to:	Grades 6-8 By the end of Grade 8, students will be able to:	Grades 9-12 By the end of Grade 12, students will be able to:
Computing Systems	Devices	**E-CS-01:** Select and operate appropriate software and hardware to perform a variety of tasks and recognize that users have different needs and preferences for the technology they use.	**M-CS-01:** Recommend improvements to the design of computing devices based on an analysis of how users interact with the devices.	**H-CS-01:** Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.
Computing Systems	Hardware & Software	**E-CS-02:** Identify and describe the function of common physical components of computing systems (hardware) using appropriate terminology.	**M-CS-02:** Design projects that combine hardware and software components to collect and exchange data.	**H-CS-02:** Compare levels of abstraction and interactions between application software, system software and hardware layers.
Computing Systems	Hardware & Software	function moveForward() { var standard }	function moveForward() { var standard }	**H-CS-04:** Categorize the roles of operating system software.
Computing Systems	Hardware & Software	function moveForward() { var standard }	function moveForward() { var standard }	**H-CS-05:** Illustrate ways computing systems implement logic, input, and output through hardware components. *
Computing Systems	Troubleshooting	**E-CS-03:** Describe basic hardware and software problems using accurate terminology.	**M-CS-03:** Identify and fix problems with computing devices and their components systematically.	**H-CS-03:** Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors.