Preview Lesson Standards
CS Principles: Computation in Action

This curriculum was developed with funding from Microsoft™.

The CS Principles: Computation in Action curriculum supports the CSTA K–12 Computer Science Standards level 3 courses with advanced topics in exploring real-world problems by applying computational thinking to develop solutions. Projects and learning activities emphasize algorithmic problem solving, team collaboration, ethical computing, and modern collaborative tools and technologies.

The new Next Generation Science Standards™ (NGSS), the next evolution in National Science Standards, include computational thinking standards. The same computational thinking practices underlie the CS Principles course and CS Principles: Computation in Action curriculum. The modules in CS Principles: Computation in Action are inherently cross-disciplinary and can also be adapted to specifically target the NGSS and teach computational thinking lessons in the science classroom or in cross-curricular projects.

The standards addressed in the Preview Lessons of the CS Principles: Computation in Action curriculum are listed below and group according to the Unit number and Lesson name.

Unit 1: Algorithms

Lesson 1: Encoding Pixels as Numbers

CS Principles Objectives:

4.1 An algorithm is a precise sequence of instructions for a process that can be executed by a computer.
   4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]
   4.1.1.a Sequencing, selection, iteration, and recursion are building blocks of algorithms.
   • Sequencing is the application of each step of an algorithm in the order in which the statements are given.
   • Selection uses a Boolean condition to determine which of two parts of an algorithm is used.
   • Iteration is the repetition of a part of an algorithm until a condition is met or for a specified number of times.

4.2 Algorithms are expressed using languages.
   4.2.1 Express an algorithm in a language. [P5]
   4.2.1.a Languages for algorithms include natural language, pseudo code, and visual and textual programming languages.
Natural language and pseudo code describe algorithms so that humans can understand them.

Algorithms described in programming languages can be executed on a computer.

4.4 Algorithms are evaluated analytically and empirically.

4.4.1 Evaluate algorithms analytically and empirically. [P4]

4.4.1.a Algorithms can be evaluated using many criteria, including efficiency, correctness, and clarity.

- Determining an algorithm’s efficiency is usually done by reasoning formally or mathematically about the algorithm.
- Empirical analysis of an algorithm is done by implementing the algorithm and running it on different inputs.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)

- Analyze the representation and trade-offs among various forms of digital information. (L3A:6.CT)

Lesson 2: Sorting Students

CS Principles Objectives:

4.4 Algorithms are evaluated analytically and empirically.

4.4.1 Evaluate algorithms analytically and empirically. [P4]

4.4.1.a Algorithms can be evaluated using many criteria, including efficiency, correctness, and clarity.

- Determining an algorithm’s efficiency is usually done by reasoning formally or mathematically about the algorithm.
- Empirical analysis of an algorithm is done by implementing the algorithm and running it on different inputs.
- The correctness of an algorithm is determined by reasoning formally or mathematically about the algorithm, not by testing an implementation of the algorithm.

4.4.1.b Different correct algorithms for the same problem can have different efficiencies.

- Sometimes more efficient algorithms are more complex.
- Finding a new, more efficient algorithm for a problem helps solve larger instances of the problem.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)
Analyze the representation and trade-offs among various forms of digital information. (L3A:6.CT)

Unit 2: Programming Creativity

Lesson 1: Kodu Game Lab

CS Principles Objectives:

1.1 Computing fosters the creation of artifacts.
   1.1.1 Use computing tools and techniques to create artifacts. [P2]
   1.1.1.a Computing enables people to create digitally — creating knowledge, tools, expressions of ideas, and solutions to problems.
     - Creating digitally requires understanding and using software tools.
     - Creating digitally can be done by combining and modifying existing artifacts or by creating new artifacts.
   1.1.1.b Computing enables people to translate intention into computational artifacts.
     - A computational artifact is created by human conception using software tools.
     - Examples of computational artifacts include digital music, videos, images, documents, and combinations of these such as infographics, presentations, and Web pages.

1.1.2 Collaborate in the creation of computational artifacts. [P6]
   1.1.2.a Collaboration is an essential part of creating computational artifacts.
     - Collaboration facilitates multiple perspectives in developing computational artifacts.
     - A computational artifact can reflect collaborative intent.

1.1.3 Analyze computational artifacts. [P4]
   1.1.3.a A computational artifact can be analyzed for correctness, functionality, and suitability.
     - A computational artifact may have weaknesses, mistakes, or errors depending on the type of artifact. For example, music created by a program may not have an error but may simply be hard to listen to.
     - The functionality of a computational artifact may be related to how it is used or how it is perceived.
     - The suitability (or appropriateness) of a computational artifact may be related to how it is used or how it is perceived.

1.3 Programming is a creative process.
   1.3.1 Use programming as a creative tool. [P2]
   1.3.1.a Programs can be developed for creative expression or to satisfy personal curiosity.
- A program developed for creative expression or to satisfy personal curiosity may have visual, audible, or tactile results; or the program may affect a computer or system without such results.
- Programs developed for creative expression or to satisfy personal curiosity may be developed with different standards or methods than programs developed for widespread distribution.
- A program or the results of running a program may be shared with others.

1.3.1.b Programs can be developed to solve problems, create new knowledge, or help people, organizations, or society
- Programs may be developed specifically with the goal of solving a problem, creating new knowledge, or helping people, organizations, or society; however, the goals may be realized independently of the original purpose of the program.
- Computer programs and the results of running the programs have widespread impact on individuals, organizations, and society.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)
- Work in a team to design and develop a software artifact. (L3A:1.CL)

Lesson 2: Variables - Concussion

4.1 An algorithm is a precise sequence of instructions for a process that can be executed by a computer.
  4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]
  4.1.1.a Sequencing, selection, iteration, and recursion are building blocks of algorithms.
    - Sequencing is the application of each step of an algorithm in the order in which the statements are given.
    - Selection uses a Boolean condition to determine which of two parts of an algorithm is used.
    - Iteration is the repetition of a part of an algorithm until a condition is met or for a specified number of times.
  4.2.1 Express an algorithm in a language. [P5]
  4.2.1.a Languages for algorithms include natural language, pseudocode, and visual and textual programming languages.
    - Natural language and pseudocode describe algorithms so that humans can understand them.
    - Algorithms described in programming languages can be executed on a computer.

4.4 Algorithms are evaluated analytically and empirically.
  4.4.1 Evaluate algorithms analytically and empirically. [P4]
4.4.1.a Algorithms can be evaluated using many criteria, including efficiency, correctness, and clarity.
- Determining an algorithm’s efficiency is usually done by reasoning formally or mathematically about the algorithm.
- Empirical analysis of an algorithm is done by implementing the algorithm and running it on different inputs. The correctness of an algorithm is determined by reasoning formally or mathematically about the algorithm, not by testing an implementation of the algorithm.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)
- Work in a team to design and develop a software artifact. (L3A:1.CL)

Common Core State Standards for Mathematics:

- Create equations in two or more variables to represent relationships between quantities; graph equations on coordinate axes with labels and scales. (A–CED.2)
- Represent constraints by equations or inequalities, and by systems of equations and/or inequalities, and interpret solutions as viable or nonviable options in a modeling context. For example, represent inequalities describing nutritional and cost constraints on combinations of different foods. (A–CED.3)
- Interpret parts of an expression, such as terms, factors, and coefficients. (A–SSE.1a)
- Define appropriate quantities for the purpose of descriptive modeling. (N–Q.2)

Lesson 3: Variables – Concussion

CS Principles Learning Objectives:

- 4.1 An algorithm is a precise sequence of instructions for a process that can be executed by a computer.
- 4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]
- 4.1.1.a Sequencing, selection, iteration, and recursion are building blocks of algorithms.
  - Sequencing is the application of each step of an algorithm in the order in which the statements are given.
Selection uses a Boolean condition to determine which of two parts of an algorithm is used.
Iteration is the repetition of a part of an algorithm until a condition is met or for a specified number of times.

4.2.1 Express an algorithm in a language. [P5]
4.2.1.a Languages for algorithms include natural language, pseudocode, and visual and textual programming languages.
- Natural language and pseudocode describe algorithms so that humans can understand them.
- Algorithms described in programming languages can be executed on a computer.

4.4 Algorithms are evaluated analytically and empirically.
4.4.1 Evaluate algorithms analytically and empirically. [P4]
4.4.1.a Algorithms can be evaluated using many criteria, including efficiency, correctness, and clarity.
- Determining an algorithm’s efficiency is usually done by reasoning formally or mathematically about the algorithm.
- Empirical analysis of an algorithm is done by implementing the algorithm and running it on different inputs. The correctness of an algorithm is determined by reasoning formally or mathematically about the algorithm, not by testing an implementation of the algorithm.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)
- Work in a team to design and develop a software artifact. (L3A:1.CL)

Common Core State Standards for Mathematics:

- Create equations in two or more variables to represent relationships between quantities; graph equations on coordinate axes with labels and scales. (A–CED.2)
- Represent constraints by equations or inequalities, and by systems of equations and/or inequalities, and interpret solutions as viable or nonviable options in a modeling context. For example, represent inequalities describing nutritional and cost constraints on combinations of different foods. (A–CED.3)
- Interpret parts of an expression, such as terms, factors, and coefficients. (A–SSE.1a)
- Define appropriate quantities for the purpose of descriptive modeling. (N–Q.2)
Lesson 4: Variables – Tic Tac Toe

CS Principles Objectives:

5.1 Programs are written to execute algorithms.
  5.1.1 Explain how programs implement algorithms. [P3]
  5.1.1.a Programming instructions are processed during program execution. An intuitive understanding of instruction processing is useful for programming.
    • Programming can be understood by sequential execution using a fetch-execute cycle in a von Neumann architecture.
    • Variables are read and written, as well as initialized and updated.
    • Other execution models and more abstract execution models offer utility in implementing algorithms (e.g., MapReduce).
  5.1.1.b Program execution automates processes.
    • Processes use memory, a central processing unit (CPU), and input and output.
    • A process may execute by itself or with other processes.
    • A process may execute on one or several CPUs.
  5.1.1.c Executable programs increase the scale of problems that can be addressed.
    • Simple algorithms can solve a large set of problems when automated.
    • Improvements in algorithms, hardware, and software increase the kinds of problems and the size of problems solvable by programming.

CSTA Standards:

Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)

Lesson 5: Portfolio Task

CS Principles Learning Objectives:

4.1 An algorithm is a precise sequence of instructions for a process that can be executed by a computer.
  4.1.1 Develop an algorithm designed to be implemented to run on a computer. [P2]
  4.1.1.a Sequencing, selection, iteration, and recursion are building blocks of algorithms.
    • Sequencing is the application of each step of an algorithm in the order in which the statements are given.
    • Selection uses a Boolean condition to determine which of two parts of an algorithm is used.
    • Iteration is the repetition of a part of an algorithm until a condition is met or for a specified number of times.
  4.1.1.b Algorithms can be combined to make new algorithms. Using existing correct algorithms as building blocks for constructing a new algorithm helps ensure the new algorithm is correct.
    • Knowledge of standard algorithms (e.g., search in a list) can help in constructing new algorithms.
4.1.1.c Different algorithms can be developed to solve the same problem.
- Algorithms that solve the same problem can have different efficiencies.
- Developing a new algorithm to solve a problem can yield insight into the problem.

4.2 Algorithms are expressed using languages.
4.2.1 Express an algorithm in a language. [P5]
4.2.1.a Languages for algorithms include natural language, pseudocode, and visual and textual programming languages.
- Natural language and pseudocode describe algorithms so that humans can understand them.
- Algorithms described in programming languages can be executed on a computer.
4.2.1.b Different languages are better suited for expressing different algorithms.
- Natural language may be better than programming languages for expressing algorithms at a high level.
- Some programming languages are designed for specific domains and are better for expressing algorithms in those domains (e.g., Structured Query Language (SQL) for making database queries).
4.2.1.c The language used to express an algorithm can affect characteristics such as clarity or readability but not whether an algorithmic solution exists.
- Every algorithm can be constructed using only sequencing, selection, and iteration.
- Nearly all programming languages are equivalent in terms of being able to express any algorithm.
- Clarity and readability are dependent on the experience of the person reading an algorithm expressed in a language.

4.4 Algorithms are evaluated analytically and empirically.
4.4.1 Evaluate algorithms analytically and empirically. [P4]
4.4.1.a Algorithms can be evaluated using many criteria, including efficiency, correctness, and clarity.
- The correctness of an algorithm is determined by reasoning formally or mathematically about the algorithm, not by testing an implementation of the algorithm.
4.4.1.b Different correct algorithms for the same problem can have different efficiencies.
- Sometimes more efficient algorithms are more complex.
- Finding a new, more efficient algorithm for a problem helps solve larger instances of the problem.

5.1 Programs are written to execute algorithms.
5.1.1 Explain how programs implement algorithms. [P3]
5.1.1.a Programming instructions are processed during program execution.
- An intuitive understanding of instruction processing is useful for programming.
- Variables are read and written, as well as initialized and updated.
5.1.1.c Executable programs increase the scale of problems that can be addressed.
- Simple algorithms can solve a large set of problems when automated.

5.2 Programming is facilitated by appropriate abstractions.
5.2.1 Use abstraction to manage complexity in programs. [P3]
5.2.1.a Functions are reusable programming abstractions.
• A function is a named grouping of programming instructions.
• Functions reduce the complexity of writing and maintaining programs.
• Functions have names and parameters, and may return values.

5.3 Programs are developed and used by people.
5.3.1 Evaluate a program for correctness. [P4]
5.3.1.a Program style affects the determination of program correctness.
• Duplicated code can make it harder to reason about a program.
• Meaningful names for variables and functions help in reasoning about programs.
• Longer code blocks are harder to reason about than shorter code blocks in a program.

5.3.1.b Locating and correcting errors in a program is called debugging the program.
• Knowledge of what a program is supposed to do, including with specific inputs, helps in finding errors.

5.3.1.c Programmers justify and explain a program’s correctness.
• Justification can include a written explanation about how a program meets its specification.
• Correctness of a program depends on correctness of program components, including code blocks and functions.

5.3.1.d Programmers explain how a program functions.
• The functionality of a program is often described by how a user interacts with the program.
• The functionality of a program is best described at a high level by what the program does, not at a lower level of how the program statements work to accomplish this.

5.3.3 Collaborate to solve a problem using programming. [P6]
5.3.3.a Collaboration is an essential part of writing programs to solve problems.
• Collaboration can increase the size and complexity of a program, making it possible to address more problems.
• Collaboration facilitates multiple perspectives in developing ideas for solving problems by programming.

5.3.3.b Collaboration in the iterative development of a program requires different skills than developing a program alone.
• Collaboration can make it easier to find and correct errors when developing programs.
• Collaboration facilitates developing program components independently; thus more quickly developing a program.
• Communication between participants is required.

5.4 Programming uses mathematical and logical concepts.
5.4.1 Employ appropriate mathematical and logical concepts in programming. [P1]
5.4.1.a Numbers and numerical concepts are fundamental to programming.
• Mathematical expressions using arithmetic operators are part of most programming languages.

5.4.1.b Logical concepts and Boolean algebra are fundamental to programming.
• Compound expressions using and, or, and not are part of most programming languages.
• Intuitive and formal reasoning about program components using Boolean concepts helps in developing correct programs.

5.4.1.c Sets and collections are tools for solving computational problems.
Lesson Standards for CS Principles: Computation in Action (Limited Preview)

- Basic operations on collections include adding elements, removing elements, iterating over all elements, and determining whether an element is in a collection.

CSTA Standards:

- Explain how sequence, selection, iteration, and recursion are building blocks of algorithms. (L3A:3.CT)
- Work in a team to design and develop a software artifact. (L3A:1.CL)
- Select appropriate file formats for various types and uses of data. (L3A:6.CP)

Unit 3: Data and Abstraction

Lesson 1: Data Formats

CS Principles Learning Objectives:

3.3 Computational manipulation of information requires consideration of representation, storage, security, and transmission.

3.3.1 Analyze the considerations involved in the computational manipulation of information. [P4]

3.3.1.a There are trade-offs in representing information as digital data.
- There are trade-offs in using lossy and lossless compression techniques for visual and audio information.
- Security concerns engender trade-offs in storing and transmitting information.
- Privacy concerns arise with some data (e.g., with health and financial information).

3.3.1.b Data is stored in many formats depending on its characteristics — such as size and intended use — so that it can be manipulated computationally.
- Data may be digitally archived for future use.
- Storage media affects methods and costs of manipulating data. Reading data and updating data have different requirements for storage.

CSTA Standards:

- Describe how various types of data are stored in a computer system. (L3A:7.CT)
- Analyze the representation and trade-offs among various forms of digital information. (L3A:7.CT)
- Select appropriate file formats for various types and uses of data. (L3A:6.CP)
Identify and select the most appropriate file format based on trade-offs (e.g., accuracy, speed, ease of manipulation). (L3B:3.CD)

Lesson 2: Drawing with Shapes

CS Principles Learning Objectives:

3.3 Computational manipulation of information requires consideration of representation, storage, security, and transmission.
   3.3.1 Analyze the considerations involved in the computational manipulation of information. [P4]
   3.3.1.a There are trade-offs in representing information as digital data.
      • There are trade-offs in using lossy and lossless compression techniques for visual and audio information.
      • Security concerns engender trade-offs in storing and transmitting information.
      • Privacy concerns arise with some data (e.g., with health and financial information).
   3.3.1.b Data is stored in many formats depending on its characteristics — such as size and intended use — so that it can be manipulated computationally.
      • Data may be digitally archived for future use.
      • Storage media affects methods and costs of manipulating data.
      • Reading data and updating data have different requirements for storage.

CSTA Standards:

Analyze the representation and trade-offs among various forms of digital information. (L3A:7.CT)

Common Core State Standards for Mathematics:

Recognize vector quantities as having both magnitude and direction. Represent vector quantities by directed line segments, and use appropriate symbols for vectors and their magnitudes (e.g., \( \mathbf{v} \), \( |\mathbf{v}| \), \( ||\mathbf{v}|| \), \( \mathbf{v} \)). (N-VM.1)

Find the components of a vector by subtracting the coordinates of an initial point from the coordinates of a terminal point. (N-VM.2)

Add and subtract vectors. (N-VM.4)

Multiply a vector by a scalar. (N-VM.5)
Use matrices to represent and manipulate data, e.g., to represent payoffs or incidence relationships in a network. (N-VM.6)

Lesson 3: Connecting Information

CS Principles Objectives:

3.1.1 Use computers to process information to gain insight and knowledge. [P1]
   3.1.1.a Computers can be used to find patterns in, answer questions about, and test hypotheses about digitally represented information.
   • Digital information is filtered and cleaned as part of using computers to process information.
   • Clustering and classification are part of the process of using computers to process information.
   • An iterative and interactive engagement with digital information is part of using computers to process information.
   3.1.1.b Insight and knowledge can be obtained from translating and transforming digitally represented information.
   • Patterns can emerge when data is transformed using computational tools.

3.1.2 Collaborate when processing information to gain insight and knowledge. [P6]
   3.1.2.a Collaboration is an essential part of solving data-driven problems.
   • Collaboration facilitates solving computational problems through multiple perspectives, experiences, and skill sets.
   • Communication between participants working on data-driven problems gives rise to better insights and deeper knowledge.
   3.1.2.b Collaboration in developing hypotheses and questions and in testing hypotheses and answering questions about data helps gain insight and knowledge.
   • Collaboration can take place with participants working closely together, or it can take place with participants using online communities.
   • Managing Big Data collaboratively provides insight and knowledge that cannot be obtained working alone.

3.1.3 Communicate insight and knowledge gained from using computer programs to process information. [P5]
   3.1.3.a Visualizations, notation, and precise language are essential in communicating insight and knowledge gained from data.
   • Visualization tools and software can connect data with communication of information.
   • Tables, diagrams, and other textual displays should be used in communicating insight and knowledge gained from data.

3.2.1 Use computing to facilitate exploration and the discovery of connections in information. [P1]
   3.2.1.a Computing tools facilitate the discovery of connections in information and extracting information and knowledge from data.
   • Search tools are essential tools in finding information.
   • Collaborative filtering is an essential tool in finding information.
• Software tools, including spreadsheets and databases, help in finding information.

3.2.1.b Metadata can increase the effective use of data or a data set by providing additional information about various aspects of that data.

• Image metadata can include time and geolocation, as well as information about the image such as size and color table.

• Standards for metadata facilitate effective use of data depending on the type of data.

CSTA Standards:

Describe how mathematical and statistical functions, sets, and logic are used in computation. (L3A:12.CP)

Analyze data and identify patterns through modeling and simulation. (L3B:9.CT)

Use collaborative tools to communicate with project team members (e.g., discussion threads, wikis, blogs, version control, etc.). (L3A:2.CL)

Deploy various data collection techniques for different types of problems. (L3B:8.CPP)

Lesson 4: Metadata

CS Principles Objectives:

3.2.1 Use computing to facilitate exploration and the discovery of connections in information. [P1]

3.2.1.a Computing tools facilitate the discovery of connections in information and extracting information and knowledge from data.

• Search tools are essential tools in finding information.

• Collaborative filtering is an essential tool in finding information.

• Software tools, including spreadsheets and databases, help in finding information.

3.2.1.b Metadata can increase the effective use of data or a data set by providing additional information about various aspects of that data.

• Image metadata can include time and geolocation, as well as information about the image such as size and color table.

• Standards for metadata facilitate effective use of data depending on the type of data.

CSTA Standards:

Describe how various types of data are stored in a computer system. (L3A:7.CT)
Analyze the representation and trade-offs among various forms of digital information. 
(L3A:7.CT)

Lesson 5: Social Network Computation (parts 1-4)

CS Principles Objectives:

1.1.1 Use computing tools and techniques to create artifacts. [P2]
   1.1.1.a Computing enables people to create digitally — creating knowledge, tools, expressions of ideas, and solutions to problems.
   - Creating digitally requires understanding and using software tools.
   - Creating digitally can be done by combining and modifying existing artifacts or by creating new artifacts.

1.1.1.b Computing enables people to translate intention into computational artifacts.
   - A computational artifact is created by human conception using software tools.
   - Examples of computational artifacts include digital music, videos, images, documents, and combinations of these such as infographics, presentations, and Web pages.

1.1.2 Collaborate in the creation of computational artifacts. [P6]
   1.1.2.a Collaboration is an essential part of creating computational artifacts.
   - Collaboration facilitates multiple perspectives in developing computational artifacts.
   - A computational artifact can reflect collaborative intent.

3.1.2 Collaborate when processing information to gain insight and knowledge. [P6]
   3.1.2.a Collaboration is an essential part of solving data-driven problems.
   - Collaboration facilitates solving computational problems through multiple perspectives, experiences, and skill sets.
   - Communication between participants working on data-driven problems gives rise to better insights and deeper knowledge.

3.1.3 Communicate insight and knowledge gained from using computer programs to process information. [P5]
   3.1.3.a Visualizations, notation, and precise language are essential in communicating insight and knowledge gained from data.
   - Visualization tools and software can connect data with communication of information.
   - Tables, diagrams, and other textual displays should be used in communicating insight and knowledge gained from data.

   3.1.3.b Summaries of insight and knowledge are effective in communicating insights gained from digitally represented information.
• Transforming information is effective in communicating (e.g., scaling in charts and tables).
• Interactivity is an aspect of communicating (e.g., by filtering displayed data or hiding components of what is shown).

7.2.1 Connect computing with innovations in other fields. [P1]

7.2.1.a Computational approaches and data analysis enable innovation.
• Machine learning and data mining have enabled innovation in medicine, business, and science.
• Scientific computing has enabled innovation in science and business.

7.2.1.b Computing enables innovation by providing access to and sharing of information.
• Open access and creative commons have enabled broad access to digital information.
• Open and curated scientific databases, such as GenBank and Ensembl, have benefited scientific researchers.

CSTA Standards:

Use advanced tools to create digital artifacts (e.g., web design, animation, video, multimedia). (L3B:1.CP)

Use collaborative tools to communicate with project team members (e.g., discussion threads, wikis, blogs, version control, etc.). (L3A:2.CL)

Describe how mathematical and statistical functions, sets, and logic are used in computation. (L3A:12.CP)

Analyze data and identify patterns through modeling and simulation. (L3B:9.CT)

Deploy various data collection techniques for different types of problems. (L3B:8.CPP)
Use data analysis to enhance understanding of complex natural and human systems. (L3B:5.CT)

Discuss the impact of computing technology on business and commerce (e.g., automated tracking of goods, automated financial transactions, e-commerce, cloud computing). (L3A:2.CI)

Math Common Core Standards:

Understand statistics as a process for making inferences about population parameters based on a random sample from that population. (S-IC.1)

Use data from a sample survey to estimate a population mean or proportion; develop a margin of error through the use of simulation models for random sampling. (S-IC.4)
Recognize and explain the concepts of conditional probability and independence in everyday language and everyday situations. (S-CP.5)

Unit 4: Internet Impact

Lesson 1: What is a Website

Computer Science Principles

4.2.1 Express an algorithm in a language. [P5]
   4.2.1.b Different languages are better suited for expressing different algorithms.
   • Natural language may be better than programming languages for expressing algorithms at a high level.
   • Some programming languages are designed for specific domains and are better for expressing algorithms in those domains (e.g., Structured Query Language (SQL) for making database queries).

6.1.1 Explain the abstractions in the Internet and how the Internet functions. [P3]

CSTA

Describe a variety of programming languages available to solve problems and develop systems. (L3A:7.CP)

Classify programming languages based on their level and application domain. (L3B:3.CP)

Create and organize Web pages through the use of a variety of web programming design tools. (L3A:1.CP)

Lesson 2: Website Basics

Computer Science Principles

6.1.1 Explain the abstractions in the Internet and how the Internet functions. [P3]
   6.1.1.a The Internet connects devices and networks all over the world.
• An end-to-end architecture facilitates connecting new devices and networks on the Internet.
• Internet protocol (IP) addresses are allocated to each autonomous system (AS) on the Internet.

**6.1.1.b** The Internet and the systems built on it facilitate collaboration.
• Connecting new devices to the Internet is enabled by assignment of an IP number.
• Cloud computing facilitates collaboration.

**6.1.1.c** The Internet is built on evolving standards including those for addresses and names.
• The Domain Name System (DNS) translates names to Internet protocol (IP) addresses.
• IP addresses use Internet protocol version 4 (IPv4) and Internet protocol version 6 (IPv6) to enable routing.

**CSTA**

Compare and contrast client-server and peer-to-peer network strategies. (L3A:7.CD)

Describe how the Internet facilitates global communication. (L3A:9.CD)

Use collaborative tools to communicate with project team members (e.g., discussion threads, wikis, blogs, version control, etc.). (L3A:2.CL)

Create and organize Web pages through the use of a variety of web programming design tools. (L3A:1.CP)

Create and organize Web pages through the use of a variety of web programming design tools. (L3A:1.CP)

**Lesson 3: Internet Protocols**

**CS Principles Objectives:**

**6.1 The Internet is a network of autonomous systems.**

**6.1.1** Explain the abstractions in the Internet and how the Internet functions. [P3]

**6.1.1.a** The Internet connects devices and networks all over the world.
• An end-to-end architecture facilitates connecting new devices and networks on the Internet.
• Internet protocol (IP) addresses are allocated to each autonomous system (AS) on the Internet.

**6.1.1.b** The Internet and the systems built on it facilitate collaboration.
• Connecting new devices to the Internet is enabled by assignment of an IP number.
• Cloud computing facilitates collaboration.

6.1.1.c The Internet is built on evolving standards including those for addresses and names.
• The Domain Name System (DNS) translates names to Internet protocol (IP) addresses.
• IP addresses use Internet protocol version 4 (IPv4) and Internet protocol version 6 (IPv6) to enable routing.
• Standards such as hypertext transfer protocol (HTTP), Internet protocol (IP), and simple mail transfer protocol (SMTP) are developed and overseen by the Internet Engineering Task Force (IETF).

CSTA Standards:

Compare and contrast client-server and peer-to-peer network strategies. (L3A:7.CD)
Describe how the Internet facilitates global communication. (L3A:9.CD)

Lesson 4: Cybersecurity and Encryption

CS Principles Objectives:

6.3 Cybersecurity is an important concern for the Internet and the systems built on it.
   6.3.1 Connect the concern of cybersecurity with the Internet and the systems built on it. [P1]
   6.3.1.c Cryptography is essential to many models of cybersecurity.
   • One-way functions (e.g., factoring the product of two large primes) are used in cryptography.
   • Open standards help ensure cryptography is secure.
   • Encryption methods include symmetric, asymmetric, and public key.

CSTA Standards:

Deploy principles of security by implementing encryption and authentication strategies. (L3B:5.CP)

Explain the principles of security by examining encryption, cryptography, and authentication techniques. (L3A:9.CP)

Lesson 5: Hackers and Hardware

CS Principles Objectives:

6.3 Cybersecurity is an important concern for the Internet and the systems built on it.
6.3.1 Connect the concern of cybersecurity with the Internet and the systems built on it. [P1]
6.3.1.a The trust model of the Internet involves trade-offs.
   • Certificate authorities (CAs) are based on a trust model.
   • The Domain Name System (DNS) was not designed to be completely secure and is vulnerable to attack.
   • Domain Name System Security Extensions (DNSSEC) are protocols that add digital signatures to DNS to provide a layer of security.

CSTA Standards:

Discuss the social and economic implications associated with hacking and software piracy. (L3A:8.CI)

Explore a variety of careers to which computing is central. (L3A:10.CP)

CS Principles Learning Objectives:

3.3 Computational manipulation of information requires consideration of representation, storage, security, and transmission.
   3.3.1 Analyze the considerations involved in the computational manipulation of information. [P4]
   3.3.1.a There are trade-offs in representing information as digital data.
      • There are trade-offs in using lossy and lossless compression techniques for visual and audio information.
      • Security concerns engender trade-offs in storing and transmitting information.
Privacy concerns arise with some data (e.g., with health and financial information).

3.3.1.b Data is stored in many formats depending on its characteristics — such as size and intended use — so that it can be manipulated computationally.

- Data may be digitally archived for future use.
- Storage media affects methods and costs of manipulating data. Reading data and updating data have different requirements for storage.

CSTA Standards:

Describe how various types of data are stored in a computer system. (L3A:7.CT)

Analyze the representation and trade-offs among various forms of digital information. (L3A:7.CT)

Select appropriate file formats for various types and uses of data. (L3A:6.CP)

Identify and select the most appropriate file format based on trade-offs (e.g., accuracy, speed, ease of manipulation). (L3B:3.CD)

CS Principles Learning Objectives:

Learning Objective 16: The student can analyze the considerations involved in the computational manipulation of information. [P4]

- Evaluation of trade-offs involved in the many possible ways to represent digital and non-digital information as digital data.
- Explanation of how data is stored in many formats depending on its characteristics—such as size and intended use—so that it can be manipulated computationally.

CSTA Standards:

Analyze the representation and trade-offs among various forms of digital information. (L3A:7.CT)

Common Core State Standards for Mathematics:

Recognize vector quantities as having both magnitude and direction. Represent vector quantities by directed line segments, and use appropriate symbols for vectors and their magnitudes (e.g., \( \mathbf{v}, |\mathbf{v}|, |\mathbf{v}|, \mathbf{v} \)). (N-VM.1)

Find the components of a vector by subtracting the coordinates of an initial point from the coordinates of a terminal point. (N-VM.2)
Add and subtract vectors. (N-VM.4)

Multiply a vector by a scalar. (N-VM.5)

Use matrices to represent and manipulate data, e.g., to represent payoffs or incidence relationships in a network. (N-VM.6)

Unit 4: The Internet and Impact

Internet Works Lesson

Standards

CS Principles Objectives:

Learning Objective 27: The student can explain the abstractions in the Internet and how the Internet functions. [P3]

a. Explanation of how the Internet connects devices and networks all over the world.
b. Explanation of how the Internet and the systems built on it facilitate collaboration.
c. Description of evolving standards that the Internet is built on, including those for addresses and names.
d. Identification of abstractions in the Internet and how the Internet functions.

CSTA Standards:

Compare and contrast client-server and peer-to-peer network strategies. (L3A:7.CD)

Describe how the Internet facilitates global communication. (L3A:9.CD)

Unit 4: The Internet and Impact

Cyber Security Hackers, and Hardware Lesson

Standards

CS Principles Objectives:
Learning Objective 30: The student can connect the concern of cyber security with the Internet and systems built on it. [P1]
   a. Identification of tradeoffs associated with the trust model of the Internet.
   b. Description of software, hardware, and human components involved in implementing cyber security.

CSTA Standards:

Discuss the social and economic implications associated with hacking and software piracy. (L3A:8.CI)

Explore a variety of careers to which computing is central. (L3A:10.CP)

Unit 4: The Internet and Impact
Cyber Security Encryption Lesson

Standards

CS Principles Objectives:

   Learning Objective 30: The student can connect the concern of cybersecurity with the Internet and systems built on it. [P1]
   c. Explanation of how cryptography is essential to many models of cybersecurity.

CSTA Standards:

Deploy principles of security by implementing encryption and authentication strategies. (L3B:5.CP)

Explain the principles of security by examining encryption, cryptography, and authentication techniques. (L3A:9.CP)