2020 COMPUTER SCIENCE

WYOMING CONTENT & PERFORMANCE STANDARDS (WYCPS)

WITH PERFORMANCE LEVEL DESCRIPTORS (PLDS)

BOOT UP WYOMING

Jillian Balow, Superintendent of Public Instruction

WDE Facilitators - Laurie Hernandez, Director of Standards & Assessment, Barb Marquer, Standards Supervisor, Brian Cole, Consultant, Catherine Palmer, Consultant, and Alicia Wilson, Consultant

Content Standards Effective - February 4, 2020
Performance Standards Effective – April 7, 2021

TO BE FULLY IMPLEMENTED IN DISTRICTS BY THE BEGINNING OF SCHOOL YEAR 2022-23

2020 Wyoming Computer Science Content Standards & PLDs

ACKNOWLEDGMENT: The State Board of Education thanks the Computer Science Standards Review Committee and the Wyoming Department of Education for their work on this new set of standards.

INTRODUCTION: The Wyoming Computer Science Content and Performance Standards (WYCPS) were developed in accordance with Wyoming State Statute W.S. 21-2- 304(c). In the years 2018-2019 Wyoming Computer Science Content Standards were developed collaboratively through the contributions of the Computer Science Standards Review Committee (CSSRC) which included Wyoming parents, educators, and community members, as well as business members from across the state and nation. The committee's work was informed and guided by initial public input through community forums, as well as input solicited from specific stakeholder groups. Additional appendices and teacher resources, created by the CSSRC, are also available on the WDE Computer Science website at edu.wyoming.gov/standards.

In 2020, twelve educators from the original committee convened to review the Content Standards and identify those of which should be the most critical for **all students** to demonstrate mastery. They then reviewed and refined, as necessary, the Proficient PLD statement to become the Computer Science Performance Standards. **All of the Content Standards are critically important to build upon the skills students need to master this content; however, the Performance Standards are the essential standards by which a student's performance is measured to ensure they are proficient on (have mastered) the content. Teachers should provide extra focus, targeted supports, and offer multiple opportunities to demonstrate** student understanding (mastery) of the content through the District Assessment System (DAS). In the secondary levels, only students electing to take a course aligned to the Computer Science Standards would need to be assessed.

RATIONALE: The committee's (CSSRC) vision is that every student in every school has the opportunity to learn computer science. We believe that computing is fundamental to understanding and participating in an increasingly technological society, and it is essential for every Wyoming student to learn as part of a modern education. We see computer science as a subject that provides students with a critical lens for interpreting the world around them and challenges them to explore how computing and technology can expand Wyoming's impact on the world.

The standards we (CSSRC) present here provide the necessary foundation for local school district decisions about curriculum, instruction, and assessment. Implementation of these standards will better prepare Wyoming high school graduates for the rigors of college and/or career. In turn, Wyoming employers will be able to hire workers with a strong foundation in Computer Science—both in specific content areas and in critical thinking and inquiry-based problem solving.

The Computer Science Standards are written in grade bands (K-2, 3-5, 6-8, and 9-12). The committee (CSSRC) determined the standard to be met by the end of the grade band and also provides suggested progressions, which can be found on *Appendix D: Teacher Resource Progression Document*. (See all Appendices on pg. 4) **Districts have local flexibility on mapping the Computer Science Content Standards to their curriculum and instruction; however, all students must have multiple opportunities to demonstrate proficiency on the Performance Standards by the end of the grade band.**

In grades 9-12, the committee provides level 1 and level 2 standards. Level 1 standards include introductory skills. The level 2 standards are intended for students who wish to advance their study of Computer Science.

COMPUTER SCIENCE: Computer Science is the study of computing principles, design, and applications (hardware & software); the creation, access, and use of information through algorithms and problem solving, and the impact of computing on society.

COMPUTATIONAL THINKING: Computational thinking is a necessary and meaningful 21st century skill. Computational thinking is defined as the

2020 Wyoming Computer Science Content Standards & PLDs

thought processes involved in formulating a problem and expressing its solutions in such a way that a computer (human or machine) can effectively carry them out. Computational thinking develops into competencies in problem solving, critical thinking, productivity, and creativity. Over time, engaging in computational thought builds a student's capacity to persevere, work efficiently, gain confidence, recognize and resolve ambiguity, generalize concepts, and communicate effectively. In order to adapt to global advancements in technology, students will need to use their computational thinking skills to formulate, articulate, and discuss solutions in a meaningful manner.

ORGANIZATION OF THE STANDARDS: (with terminology**)**
Standard Code: Grade.Domain.Subconcept.Standard#
Key: 2.CS.D.01 = **2**nd Grade.**C**omputing **S**ystems.**D**evices.Standard #**1**

DOMAIN: The core concepts to be studied in computer science are as follows: 1) Computing Systems; 2) Networks and the Internet; 3) Data and Analysis; 4) Algorithms and Programming; and 5) Impacts of Computing.

CONTENT STANDARDS define the content and skills students are expected to know and be able to do by the end-of-the grade band. They are built foundationally and then in learning progressions. They do not dictate what methodology or instructional materials should be used, nor how the material is delivered. In grades 9- 12, the computer science standards are organized into 2 levels. Mostly, Level 1 is intended to represent the introductory level while Level 2 reaches a deeper level.

 = Plugged in
This symbol designates when a standard may require hardware, software, or both in order to fully address the intent of the benchmark.

PERFORMANCE STANDARDS (PS), identified with the code in blue highlight and an asterisk (*), are the standards all students are expected to learn and be assessed on through the district assessment system by the end-of-the grade band. They specify the specific degree of understanding or demonstration of the knowledge and/or skill for a particular content standard. As such, it employs clear action verbs and describes "how good is good enough." Districts are expected to give students multiple opportunities to demonstrate proficiency on the Performance Standards through the District Assessment System (DAS) and provide appropriate supports for student success. **Teachers should provide extra focus, targeted supports, and offer multiple opportunities to demonstrate** student understanding (mastery). In the secondary level, only students electing to take a course aligned to these standards need to be assessed in the DAS.

PS Snapshot (# of Content Standards tied to PS)
The PS are also listed on last 2 pages of this document

Grade K-2 (7 out of 18)	Grade 3-5 (10 out of 23)	6-8 MS (8 out of 25)	HS Level 1 (9 out of 35)	HS Level 2 (7 out of 28)
2.CS.HS.01	5.CS.HS.01	8.CS.HS.01	L1.NI.C.03	L2.NI.NCO.01
2.NI.C.01	5.NI.C.01	8.NI.C.01	L1.DA.CVT.01	L2.DA.CVT.02
2.AP.A.01	5.DA.S.01	8.DA.CVT.01	L1.AP.A.01	L2.AP.A.03
2.AP.C.01	5.DA.IM.01	8.AP.V.01	L1.AP.C.03	L2.AP.M.01
2.AP.M.01	5.AP.A.01	8.AP.C.01	L1.AP.M.01	L2.AP.PD.04
2.AP.PD.03	5.AP.V.01	8.AP.M.01	L1.AP.PD.01	L2.AP.PD.05
2.IC.SI.01	5.AP.C.01	8.AP.PD.04	L1.AP.PD.04	L2.IC.C.02
	5.AP.PD.03	8.IC.SLE.01	L1.IC.SLE.03	
	5.IC.SI.02		L1.IC.SLE.04	
	5.IC.SLE.01			

PERFORMANCE LEVEL DESCRIPTORS (PLDs) describe the performance expectations of students for each of the four (4) performance level categories: advanced, proficient, basic, and below basic. These are a description of what students within each performance level are expected to know and be able to do. All PLDs are found within this document.

2020 Wyoming Computer Science Content Standards & PLDs

COMPUTER SCIENCE (CS) PRACTICES: There are seven (7) CS Practices that can be used as the standards and benchmarks are taught and measured. The seven (7) CS Practices are listed below and are more deeply explored in **Appendix A**: Descriptions of the CS Practices. (See Appendices below)

Practice 1. Fostering an Inclusive Computing Culture
Practice 2. Collaborating Around Computing
Practice 3. Recognizing and Defining Computational Problems
Practice 4. Developing and Using Abstractions
Practice 5. Creating Computational Artifacts
Practice 6. Testing and Refining Computational Artifacts
Practice 7. Communicating About Computing

APPENDICES: found at edu.wyoming.gov/standards

- APPENDIX A: DESCRIPTIONS OF COMPUTER SCIENCE (CS) PRACTICES
- ~~APPENDIX B: PERFORMANCE LEVEL DESCRIPTORS (PLDs)~~ **NOTE:** This is now replaced within this document.
- APPENDIX C: GLOSSARY
- APPENDIX D: TEACHER RESOURCE PROGRESSION DOCUMENT
- APPENDIX E: ADMINISTRATOR K-12 CS STANDARDS OVERVIEW
- APPENDIX F: WYOMING DIGITAL LEARNING GUIDELINES (based on the 2016 ISTE Standards for Students)

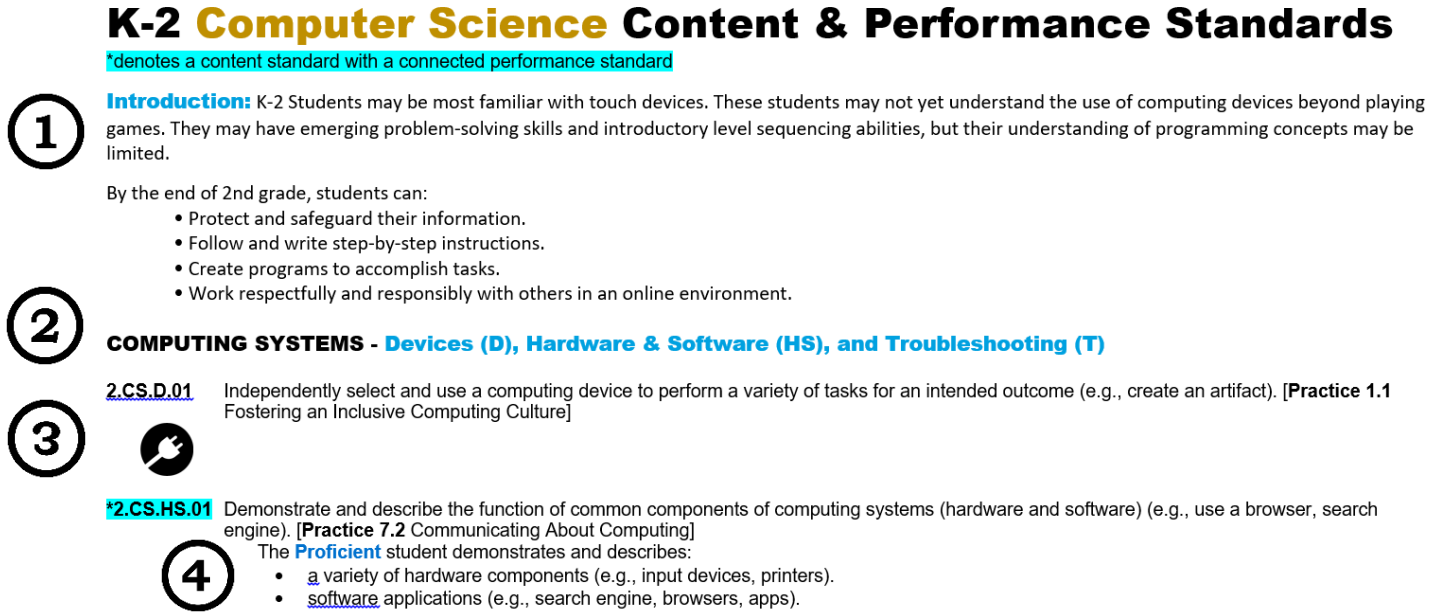
RESOURCES/REFERENCES:

- K-12 Computer Science Framework, (2016). Retrieved from http://k12cs.org/. [Ch. 5 Practices].
- Computer Science Teachers Association (CSTA), (2017). Retrieved from http://www.csteachers.org/page/standards.

HOW TO READ THIS DOCUMENT

Each grade band has 4 main sections:

1) Introduction describing the expectation by the end of the grade-band
2) Domain (Core Concepts) and Subconcepts
3) Content Standard with correlating CS Practice
4) Performance Standard – The Proficient statement that accompanies certain Content Standards that are denoted with an asterisk (*) and highlighted in blue



K-2 Computer Science Content & Performance Standards

*denotes a content standard with a connected performance standard

1

Introduction: K-2 Students may be most familiar with touch devices. These students may not yet understand the use of computing devices beyond playing games. They may have emerging problem-solving skills and introductory level sequencing abilities, but their understanding of programming concepts may be limited.

By the end of 2nd grade, students can:

- Protect and safeguard their information.
- Follow and write step-by-step instructions.
- Create programs to accomplish tasks.
- Work respectfully and responsibly with others in an online environment.

2

COMPUTING SYSTEMS - Devices (D), Hardware & Software (HS), and Troubleshooting (T)

3

2.CS.D.01 Independently select and use a computing device to perform a variety of tasks for an intended outcome (e.g., create an artifact). [**Practice 1.1** Fostering an Inclusive Computing Culture]

*2.CS.HS.01 Demonstrate and describe the function of common components of computing systems (hardware and software) (e.g., use a browser, search engine). [**Practice 7.2** Communicating About Computing]

4

The **Proficient** student demonstrates and describes:

- a variety of hardware components (e.g., input devices, printers).
- software applications (e.g., search engine, browsers, apps).

K-2 Computer Science Content Standards & PLDs

*denotes a content standard with a connected performance standard

Introduction: K-2 Students may be most familiar with touch devices. These students may not yet understand the use of computing devices beyond playing games. They may have emerging problem-solving skills and introductory level sequencing abilities, but their understanding of programming concepts may be limited.

By the end of 2nd grade, students can:

- Protect and safeguard their information.
- Follow and write step-by-step instructions.
- Create programs to accomplish tasks.
- Work respectfully and responsibly with others in an online environment.

COMPUTING SYSTEMS - Devices (D), Hardware & Software (HS), and Troubleshooting (T)

2.CS.D.01 Independently select and use a computing device to perform a variety of tasks for an intended outcome (e.g., create an artifact). [**Practice 1.1** Fostering an Inclusive Computing Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., can recognize capabilities of multiple devices and can perform similar tasks with them).
The **Proficient** student regularly uses a computing device to independently:

- Power on and off devices.
- Authenticate, when applicable.
- Open appropriate programs.
- Complete assignments or teacher led activities.

The **Basic** student, with guidance, uses a computing device to complete assignments or teacher led activities.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***2.CS.HS.01** Demonstrate and describe the function of common components of computing systems (hardware and software) (e.g., use a browser, search engine). [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., justifies hardware and software choices).
The **Proficient** student demonstrates and describes:

- a variety of hardware components (e.g., input devices, printers).
- software applications (e.g., search engine, browsers, apps).

The **Basic** student with guidance:

- Identifies hardware components and software applications.
- Utilizes hardware components and software applications

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.CS.T.01 Recognize computing systems might not work as expected and identify and effectively communicate simple hardware or software problems and implement solutions (e.g., app or program is not working as expected, no sound is coming from the device, caps lock turned on) and discuss problems with peers and adults. [**Practice 6.2** Testing and Refining Computational Artifacts**; Practice 7.2** Communicating About Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., helps others in troubleshooting issues, can troubleshoot more complex issues like connectivity or advanced software features).

The **Proficient** student:

- Recognizes that computing systems may not work as expected.
- Identifies and effectively communicates simple hardware and software problems.
- Implements solutions to simple hardware or software issues.

The **Basic** student:

- Can recognize that computing systems may not work as expected.
- With guidance, identifies and effectively communicates simple hardware and software problems.
- With guidance, implements solutions to simple hardware or software issues.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

NETWORK & THE INTERNET - Network, Communication, & Organization (NCO) and Cybersecurity (C)

2.NI.NCO.01 Identify and describe that computing devices can be connected in a variety of ways (e.g., Bluetooth, Wi-Fi, home and school networks, the internet). [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., evaluates the appropriateness of different connectivity options for a variety of tasks).

The **Proficient** student:

- Can identify that computing devices can be connected in a variety of ways.
- Can describe different connectivity options (e.g., Bluetooth, internet).

The **Basic** student:

- Can identify that computing devices can be connected in a variety of ways.
- With guidance, can describe a connectivity option (e.g., Wi-Fi or Bluetooth).

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***2.NI.C.01** Explain what authentication factors (e.g., login) are, why we use them, and apply authentication to protect devices and information (personal and private) from unauthorized access. [**Practice 7.3** Communication About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., can compare authentication methods, one factor versus two factors).

The **Proficient** student:

- Explains what authentication factors (e.g., login) are and why we use them.
- Models use of authentication factors to appropriate apps and devices.

The **Basic** student identifies what authentication factors are and, with guidance, applies authentication factors to appropriate apps and devices.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

DATA ANALYSIS – Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

2.DA.S.01 With guidance, develop and modify an organizational structure by creating, copying, moving, and deleting files and folders. [**Practice 4.2** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., can independently create organizational structure).

The **Proficient** student, with guidance, develops and modifies an organizational structure by:

- Creating folders.
- Copying existing folders and files.
- Moving existing folders and files.
- Deleting folders and files.

The **Basic** student, while working with a computing device and with guidance:

- Locates existing files.
- Opens existing files.
- Modifies existing files.
- Saves changes to a file.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.DA.CVT.01 With guidance, collect data and independently present the same data in various visual formats. [**Practice 4.4** Developing and Using Abstractions; **Practice 7.1** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., independently creates and presents their own data sets).

The **Proficient** student:

- With guidance, creates a data set, **and**
- Independently presents that data in multiple formats (e.g., as a table and graph or as a table and chart).

The **Basic** student:

- With guidance, creates a data set, **and**

- Independently presents that data in multiple formats (e.g., as a table and graph or as a table and chart).

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.DA.IM.01 With guidance, interpret data and present it in a chart or graph (visualization) in order to make a prediction, with or without a computing device. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., independently perform any of the proficient student steps).

The **Proficient** student with guidance:

- Interprets data.
- Presents it in a chart or graph (visualization).
- Makes a prediction based on the data.

The **Basic** student with guidance:

- Interprets data, **and**
- Presents it in a chart or graph (visualization).

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

ALGORITHMS & PROGRAMMING – Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

***2.AP.A.01** With guidance, identify and model daily processes by creating and following algorithms (sets of step-by- step instructions) to complete tasks (e.g., verbally, kinesthetically, with robot devices, or a programming language). [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., independently creates algorithms via one or more of the following techniques: verbally, kinesthetically, with robot devices, or a programing language).

The **Proficient** student, with guidance:

- Follows algorithms to complete tasks.
- Creates algorithms to complete tasks.

The **Basic** student with guidance, follows algorithms to complete tasks.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.AP.V.01 Model the way programs store and manipulate data by using numbers or other symbols to represent information (e.g., thumbs up/down as representations of yes/no, arrows when writing algorithms to represent direction, or encode and decode words using numbers, pictographs, or other symbols to represent letters or words). [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., creates complex expressions with symbols).

The **Proficient** student:

- Uses symbols to represent information.
- Understands that inferred meanings of the symbols can change or can represent missing information.
- Creates expressions with symbols to convey data, information, or processes.

The **Basic** student with guidance:

- Uses symbols to represent information.
- Understands that inferred meanings of the symbols can change or can represent missing information.
- Creates expressions with symbols to convey data, information, or processes.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***2.AP.C.01** With guidance, independently and collaboratively create programs to accomplish tasks using a programming language, robot device, or unplugged activity that includes sequencing, conditionals, and repetition. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., independently creates programs that include sequencing, conditionals, and repition).

The **Proficient** student, with guidance:

- Individually creates programs that include sequencing, conditionals, and repetition.
- Collaboratively creates programs that include sequencing, conditionals, and repetition.

The **Basic** student with guidance, create:

- Programs that include sequencing, conditionals, or repetition.
- Tasks that include sequencing, conditionals, or repetition.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***2.AP.M.01** Using grade appropriate content and complexity, decompose (breakdown) the steps needed to solve a problem into a precise sequence of instructions (e.g., develop a set of instructions on how to play your favorite game). [**Practice 3.2** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., independently creates programs that include sequencing, conditionals, and repetition).

The **Proficient** student, independently:

- Decomposes a problem.
- Creates a sequence of instructions to solve that problem.

The **Basic** student with guidance:

- Decomposes a problem.
- Creates a precise sequence of instructions to solve that problem.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.AP.PD 01 through 04 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., demonstrates the development process on different platforms, languages, or mediums).
The **Basic** student with guidance, demonstrates program development by:

- Creating a plan for a program.
- Writing the program.
- Giving credit for the resources used.
- Debugging the program.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

2.AP.PD.01 Develop plans that describe a program's sequence of events, goals, and expected outcomes. [**Practice 5.1** Creating Computational Artifacts**;** **Practice 7.2** Communicating About Culture]

The **Proficient** student demonstrates program development by creating a plan for a program.

2.AP.PD.02 Give credit to ideas, creations, and solutions of others while writing and developing programs. [**Practice 7.3** Communicating About Computing]

The **Proficient** student gives credit for the resources used.

***2.AP.PD.03** Independently and collaboratively debug (identify and fix errors) programs using a programming language. [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Proficient** student debugs using a programming language:

- Independently.
- Collaboratively.

2.AP.PD.04 Use correct terminology (debug, program input/output, code) to explain the development of a program or an algorithm (e.g., in an unplugged activity, hands on manipulatives, or a programming language). [**Practice 7.2** Communicating About Computing]

The **Proficient** student uses correct terminology to explain the development of a program or an algorithm.

IMPACTS OF COMPUTING – Culture (C) and Social Interactions (SI)

2.IC.C.01 Describe how people use different types of technologies in their daily work and personal lives. [**Practice 3.1** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., identifies and describes the potential impacts of different technologies).
The **Proficient** student describes how people use different types of technologies (e.g., cell phones, computers) in their daily work and personal lives.
The **Basic** student identifies how people use different types of technologies (e.g., cell phones, computers) in their daily work and personal lives.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***2.IC.SI.01** Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [**Practice 2.1** Collaborating Around Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student, independently:

- Makes appropriate choices when participating in an online community.
- Interacts respectfully with others.
- Reports inappropriate behavior to an adult.

The **Basic** student with guidance:

- Makes appropriate choices when participating in an online community.
- Identifies inappropriate behavior and reporting procedures.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

3-5 Computer Science Content Standards & PLDs

*denotes a content standard with a connected performance standard

Introduction: Throughout grades 3-5, students engage in creative applications of Computer Science concepts and practices introduced in K-2. By the end of fifth grade, students will build upon their previous understanding of algorithms, programming (coding), networks, and the Internet. In addition, students will create, modify, and troubleshoot increasingly complex programs for a variety of purposes. Students will be able to explain cultural, social, and ethical impacts of computing.

By the end of 5th grade, students can:

- Model how information is translated, transmitted, and processed.
- Identify and implement strategies for protecting personal information.
- Justify the format and location for storage.
- Create and modify (remix) programs through an iterative process.
- Develop, test, and refine digital artifacts.
- Work respectfully and responsibly with others in an online environment and discuss the social impact of violating intellectual property rights.

COMPUTING SYSTEMS - Devices (D), Hardware & Software (HS), and Troubleshooting (T)

5.CS.D.01 Independently, describe how internal and external parts of computing devices function to form a system. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., demonstrates on different types of devices).
The **Proficient** student independently describes with few to no errors how internal and external parts of computing devices function to form a system.
The **Basic** student with guidance, describes with some errors how internal and external parts of computing devices function to form a system.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***5.CS.HS.01** Model how information is translated, transmitted, and processed in order to follow through hardware and software to accomplish tasks. [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., compare and contrast different devices).
The **Proficient** student creates a simple example of how a computing device takes input, stores information, processes input and information, and provides output (e.g., click volume button and the sound level increases).
The **Basic** student partially models how information is translated, transmitted, and processed in order to flow through hardware and software to accomplish tasks.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.CS.T.01 Identify hardware and software problems that may occur during everyday use, then develop, apply, and explain strategies for solving these problems. [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., develops a troubleshooting guide, helps others with troubleshooting issues efficiently, suggests preventative measures).
The **Proficient** student accurately:

- Identifies hardware and software problems that may occur during everyday use.
- Develops, applies, and explains strategies for solving identified problems, when applicable.

The **Basic** student partially:

- Identifies hardware and software problems that may occur during everyday use.
- Attempts to solve identified problems, when applicable.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

NETWORK & THE INTERNET - Network, Communication, & Organization (NCO) and Cybersecurity (C)

5.NI.NCO.01 Model and explain how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the internet, and reassembled at the destination. [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., compares and contrasts different connection types).
The **Proficient** student accurately models and explains how information is:

- broken down into smaller pieces.
- transmitted as packets through multiple devices over networks and the internet.
- reassembled at the destination.

The **Basic** student partially models and explains how information is:

- broken down into smaller pieces, **and**
- transmitted as packets through multiple devices over networks and the internet, **and/or**
- reassembled at the destination.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***5.NI.C.01** Discuss real-world cybersecurity problems and identify and implement appropriate strategies for how personal information can be protected. [**Practice 3.1** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., compares and contrasts a variety of approaches to authentication, evaluates current practices).
The **Proficient** student:

- Describes, with specificity, real-world cybersecurity problems.
- Discusses with peers using a variety of methods (e.g., blog, discussion post, oral) the personal consequences of inappropriate use.
- Identifies and implements appropriate strategies for how personal information can be protected.

The **Basic** student:

- Generally discusses real-world cybersecurity problems, **and/or**

- Identifies appropriate strategies for how personal information can be protected.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

DATA ANALYSIS – Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

***5.DA.S.01** Justify the format and location for storing data based on sharing requirements and the type of information (e.g., images, videos, text). [**Practice 4.2** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., compares and contrasts a variety of approaches to authentication, evaluates current practices). demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., determines the best file type for a given purpose, suggests strategies to solve a problem, creates a document in a variety of formats, converts files).
The **Proficient** student explains/justifies why a particular location or format is appropriate for the data they are creating or using.
The **Basic** student describes the format, location, sharing requirements, or the type of information when storing data.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.DA.CVT.01 Organize and present collected data to highlight relationships and support a claim. [**Practice 7.1** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., helps others organize collected data, suggests improvements on how to organize collected data to a specific audience).
The **Proficient** student organizes and presents collected data to:

- highlight comparisons.
- highlight relationships.
- support a claim.

The **Basic** student organizes and presents collected data.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***5.DA.IM.01** Use data to highlight or propose relationships, predict outcomes, or communicate an idea. [**Practice 7.1** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., proposes alternative models, proposes additional factors that could affect a relationship).
The **Proficient** student independently:

- Makes a statement based on a data set to describe a relationship, **and then**
- Predicts an outcome **or** communicates an idea.

The **Basic** student with guidance, uses data to:

- Highlight relationships, **and/or**
- Communicate an idea.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

ALGORITHMS & PROGRAMMING – Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

***5.AP.A.01** Using grade appropriate content and complexity, compare and refine multiple algorithms for the same task and determine which is the most appropriate. [**Practice 3.3** Recognizing and Defining Computational Problems] [**Practice 6.3** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., develop alternative algorithms).

The **Proficient** student:

- Compares and refines multiple algorithms for the same task.
- Determines which algorithm is the most appropriate for the same task.

The **Basic** student compares simple algorithms for the same task.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***5.AP.V.01** Using grade appropriate content and complexity, create programs that use variables to store and modify data. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., uses a variety of variable types).

The **Proficient** student creates programs that use variables to:

- Store data.
- Modify data.

The **Basic** student modifies programs that use variables to:

- Store data.
- Modify data.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***5.AP.C.01** Using grade appropriate content and complexity, create programs that include sequences, events, loops, and conditionals, both individually and collaboratively. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., incorporating nested loops and complex conditionals).

The **Proficient** student:

- Independently, creates programs that include combinations of sequences, events, loops, and conditionals.
- Collaboratively, creates programs that include combinations of sequences, events, loops, and conditionals.

The **Basic** student:

- Independently, creates programs that include sequences and events.
- Collaboratively, creates programs that include sequences and events.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.AP.M 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., helps others modify code, incorporates portions of multiple programs).
The **Basic** student:

- Decomposes (breaks down) problems into smaller, manageable sub-problems to facilitate the program development process, **and/or**
- Modifies, remixes, or incorporates portions of an existing program into one's own work.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.AP.M.01 Using grade appropriate content and complexity, decompose (break down) problems into smaller, manageable sub-problems to facilitate the program development process. [**Practice 3.2** Recognizing and Defining Computational Problems]

The **Proficient** student decomposes (breaks down) problems into smaller, manageable sub-problems to facilitate the program development process.

5.AP.M.02 Using grade appropriate content and complexity, modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features. [**Practice 5.3** Creating Computational Artifacts]

The **Proficient** student modifies, remixes, or incorporates portions of an existing program into one's own work to develop something new or add more advanced features.

5.AP.PD 01 through 05 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard. By way of examples:

- Justifies their own copyright on their work;
- Explains the different types of copyrights and the process of getting permission;
- Provides guidance to other students when testing and debugging a program or algorithm;
- Proposes alternatives and justifies why they went with their current code.

The **Basic** student:

- Observes intellectual property rights and gives appropriate credit when creating or remixing programs, and
- Uses an iterative process to plan the development of a program by including other perspectives and considers user preferences, and/or
- Tests and debugs (identify and fix errors) a program or algorithm to ensure it runs as intended, and/or
- Describes choices made during program development using code comments, presentations, and demonstrations, and/or
- With teacher guidance, performs varying roles when collaborating with peers during the design, implementation, and review stages of program development.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.AP.PD.01 Use an iterative process to plan the development of a program by including others' perspectives and considering user preferences. [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Proficient** student independently uses an iterative process to plan the development of a program by including other perspectives and considers user preferences.

5.AP.PD.02 Using grade appropriate content and complexity, observe intellectual property rights and give appropriate credit when creating or remixing programs. [**Practice 5.2** Creating Computational Artifacts] [**Practice 7.3** Communicating About Computing]

The **Proficient** student observes intellectual property rights and gives appropriate credit when creating or remixing programs.

***5.AP.PD.03** Using grade appropriate content and complexity, test and debug (i.e., identify and fix errors) a program or algorithm to ensure it runs as intended. [**Practice 6.1 & 6.2** Testing and Refining Computational Artifacts]

The **Proficient** student independently tests and debugs a program or algorithm to ensure it runs as intended.

5.AP.PD.04 Using grade appropriate content and complexity, describe choices made during program development using code comments, presentations, and demonstrations. [**Practice 7.2** Communicating About Computing]

The **Proficient** student independently describes choices made during program development using code comments, presentations, and demonstrations.

5.AP.PD.05 Using grade appropriate content and complexity, with teacher guidance, perform varying roles when collaborating with peers during the design, implementation, and review stages of program development. [**Practice 2.2** Collaborating Around Computing]

The **Proficient** student, with teacher guidance, performs varying roles when collaborating with peers during the design, implementation, and review stages of program development.

IMPACTS OF COMPUTING – Culture (C) and Social Interactions (SI)

5.IC.C 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., makes and justifies predictions based on historical patterns, incorporates multiple forms of accessibility in one artifact).

The **Basic** student:

- Gives examples of how computing technologies have changed the world, **and/or**
- Expresses how technologies interact with cultural practices, **and/or**
- Tests digital artifacts or devices for accessibility and usability for diverse end users.

The **Below Basic** student:

- Gives examples of how computing technologies have changed the world, **and/or**
- Expresses how technologies interact with cultural practices, **and/or**
- Tests digital artifacts or devices for accessibility and usability for diverse end users.

5.IC.C.01 Give examples and explain how computing technologies have changed the world and express how those technologies influence and are influenced by cultural practices. [**Practice 3.1** Recognizing and Defining Computational Problems]

The **Proficient** student:

- Gives examples and explains how computing technologies have changed the world.
- Expresses how technologies influence and are influenced by cultural practices.

5.IC.C.02 Develop, test, and refine digital artifacts or devices to improve accessibility and usability for diverse end users. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Proficient** student tests digital artifacts or devices for accessibility and usability for diverse end users.

5.IC.SI 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., creates resources that models or explains to peers how to participate in online communities or independently uses video conferencing tools or other online collaborative spaces, such as blogs, wikis, forums, or website comments, to gather feedback from individuals and groups).
The **Basic** student:

- practices grade-level appropriate behavior and responsibilities while participating in an online community.
- identifies and reports inappropriate behavior, when applicable.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

5.IC.SI.01 Seek diverse perspectives for the purpose of improving computational artifacts. [**Practice 1.1** Fostering an Inclusive Computing Culture]

The **Proficient** student seeks diverse perspectives for the purpose of improving computational artifacts.

***5.IC.SI.02** Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [**Practice 2.1** Collaborating Around Computing]

The **Proficient** student practices grade-level appropriate behavior and responsibilities while participating in an online community.

***5.IC.SLE.01** Recognize and appropriately use public domain and creative commons media and discuss the social impact of violating intellectual property rights. [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., explain the process of contributing to a public domain or creative commons media, create and use a custom intellectual property rights system used by members of the class).
The **Proficient** student:

- Appropriately uses public domain and creative commons media.
- Discusses the social impact of violating intellectual property rights.

The **Basic** student identifies types of digital data that may have intellectual property rights that prevent copying or require attribution.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

6-8 Computer Science Content Standards & PLDs

*denotes a content standard with a connected performance standard

Introduction: Throughout grades 6-8, students continue to develop their understanding of algorithms and programming (coding). Students work collaboratively and independently to create and modify increasingly complex programs for a variety of purposes introduced in grades 3-5.

By the end of 8th grade, students can:

- Systematically identify, recommend, resolve, and document increasingly complex software and hardware problems with computing devices and their components.
- Model the role of protocols in transmitting data across networks and the internet.
- Critique physical and digital procedures that could be implemented to protect electronic data/information.
- Use and refine computational tools to transform collected data in order to make it more useful and reliable.
- Create flowcharts and pseudocode to design algorithms to solve complex problems.
- Create clearly named variables that represent different data types and perform operations on their values.
- Design and iteratively develop programs that combine control structures, including nested loops and compound conditionals.
- Decompose problems into parts to facilitate the design, implementation, and review of programs.
- Create procedures with parameters to organize code and make it easier to reuse.
- Seek and incorporate feedback from team members and users to refine a solution to a problem.
- Describe impacts associated with computing technologies that affect people's everyday activities and career options along with issues of bias and accessibility in the design of technologies.
- Practice grade-level appropriate behavior and responsibilities while participating in an online community, including identifying and reporting inappropriate behavior.
- Describe tradeoffs between allowing information to be public and keeping information private and secure.
- Discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent.

COMPUTING SYSTEMS - Devices (D), Hardware & Software (HS), and Troubleshooting (T)

8.CS.D.01 Recommend improvements to the design of computing devices based on an analysis of how a variety of users interact with the device. [**Practice 3.3** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., recommend improvements to the design in more than one area (input, output, processing, storage) or group (special populations).

The **Proficient** student:

- Analyzes the needs of the users.
- Recommends improvements to the design of computing devices based on that analysis.

The **Basic** student:
- Understands the needs of the users, but is unable to analyze, **and/or**
- Describes the parts of computing devices, but cannot recommend improvements to the design.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.CS.HS.01** Design and refine a project that combines hardware and software components to collect and exchange data. [**Practice 5.1** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., design a project that combines hardware and software components to collect and exchange data that affects the world around them, refine a project multiple times that combines hardware and software components to collect and exchange data to address real world usage).

The **Proficient** student:
- Designs a project that combines hardware and software components to collect and exchange data.
- Refines a project that combines hardware and software components to collect and exchange data.

The **Basic** student:
- Describes how hardware and software components collect and exchange data, but cannot design a project, **and/or**
- Creates a project that combines hardware and software components to collect and exchange data but cannot refine.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.CS.T.01 Systematically identify, resolve, and document increasingly complex software and hardware problems with computing devices and their components. [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., systematically assists others with hardware or software problems, creates a detailed troubleshooting document or tutorial, comes up with novel solutions).

The **Proficient** student can systematically:
- Identify software problems with computing devices and their components.
- Identify hardware problems with computing devices and their components.
- Resolve software problems with computing devices and their components.
- Resolve hardware problems with computing devices and their components.
- Document software problems with computing devices and their components.
- Document hardware problems with computing devices and their components.

The **Basic** student can do **some** of the following:
- Identify software problems with computing devices and their components,
- Identify hardware problems with computing devices and their components,
- Resolve software problems with computing devices and their components,
- Resolve hardware problems with computing devices and their components,
- Document software problems with computing devices and their components, **or**

- Document hardware problems with computing devices and their components.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

NETWORK & THE INTERNET - Network, Communication, & Organization (NCO) and Cybersecurity (C)

8.NI.NCO.01 Model the role of protocols in transmitting data across networks and the internet (e.g., explain protocols and their importance to data transmission; model how packets are broken down into smaller pieces and how they are delivered). [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., research and compare/contrast multiple network protocols).
The **Proficient** student models the role of protocols in transmitting data across networks and the internet.
The **Basic** student:

- Identifies protocols used in transmitting data across networks and the internet, **and/or**
- Explains the role of protocols in transmitting data across networks and the internet.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.NI.C 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., explain the impacts of hacking, ransomware, scams, and ethical/legal concerns; compare the advantages and disadvantages of multiple methods of encryption to model the secure transmission of information).
The **Basic** student:

- Lists physical and digital procedures that could be implemented to protect electronic data/ information, **and/or**
- Describes multiple methods of encryption used to secure data.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.NI.C.01** Critique physical and digital procedures that could be implemented to protect electronic data/information. [**Practice 7.3** Communicating About Computing]

The **Proficient** student critiques physical and digital procedures that could be implemented to protect electronic data/information.

8.NI.C.02 Apply multiple methods of encryption to model the secure transmission of data. [**Practice 4.4** Developing and Using Abstractions]

The **Proficient** student applies multiple methods of encryption to model the secure transmission of data.

DATA ANALYSIS – Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

8.DA.S.01 Represent data using multiple encoding schemes (e.g., ASCII, binary). [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., convert data between multiple encoding schemes; ASCII to binary, hex to rgb).
The **Proficient** student represents data using multiple encoding schemes.
The **Basic** student recognizes data is stored in multiple encoding schemes.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.DA.CVT.01** Using computational tools, transform collected data to make it more useful and reliable. [**Practice 6.3** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., error checking input during data collection process, export data to another format).
The **Proficient** student uses computational tools to:

- Transform data to improve reliability by removing errors.
- Highlight or expose relationships in the data.

The **Basic** student:

- Explores a variety of computational tools and the content of their data.
- Uses computational tools to collect data.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.DA.IM.01 Refine computational models based on generated data. [**Practice 4.4** Developing and Using Abstractions] [**Practice 5.3** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., make multiple refinements).
The **Proficient** student refines computational models based on generated data.
The **Basic** student:

- Uses models and simulations to formulate, refine, and test hypotheses, **and/or**
- Tests and analyzes the effects of changing variables while using computational models.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

ALGORITHMS & PROGRAMMING – Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

8.AP.A.01 Create flowcharts and pseudocode to design algorithms to solve complex problems. [**Practice 4.1 & 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., design algorithms to solve complex problems in multiple ways and determine and use the most effective planning tool).

The **Proficient** student:

- Creates flowcharts to design algorithms to solve complex problems.
- Writes pseudocode to design algorithms to solve complex problems.

The **Basic** student:

- Uses flowcharts to modify existing algorithms, **and/or**
- uses pseudocode to modify existing algorithms, **and/or**
- uses natural language to modify existing algorithms.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.AP.V.01** Using grade appropriate content and complexity, create clearly named variables that represent different data types and perform operations on their values. [**Practice 5.1 & 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., explain types of errors that can occur if improper data types are used in operations, understand structures or classes can contain multiple data types).

The **Proficient** student:

- Creates clearly named variables.
- Creates variables that represent different data types.
- Performs operations on the values of variables.

The **Basic** student:

- Recognizes that variables can represent different data types, **and/or**
- can create a variable, **and/or**
- can perform operations on the values of variables.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.AP.C.01** Using grade appropriate content and complexity, design and iteratively develop programs that combine control structures, including nested loops and compound conditionals. [**Practice 5.1 & 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., multiple examples of nested loops and compound conditions in a program, evidence of efficient code, clear documentation).

The **Proficient** student designs and iteratively develops programs that include:

- nested loops.
- compound conditionals.

The **Basic** student designs and iteratively develops programs that:

- use simple loops.
- use simple conditionals.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.AP.M 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., create procedures with multiple parameters and/or return values).

The **Basic** student:

- Recognizes the inefficiency of repetition in programming, **and/or**
- recognizes the organizational, readability and labor-saving advantages of code reuse.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***8.AP.M.01** Using grade appropriate content and complexity, decompose problems and sub-problems into parts to facilitate the design, implementation, and review of programs. [**Practice 3.2** Recognizing and Defining Computational Problems]

The **Proficient** student decomposes problems and sub-problems into parts for the design, implantation, and review of programs.

8.AP.M.02 Using grade appropriate content and complexity, create procedures with parameters to organize code and make it easier to reuse. [**Practice 4.1 & 4.3** Developing and Using Abstractions]

The **Proficient** student creates procedures with parameters.

8.AP.PD 01 through 04 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., seek open source libraries to include in their program, seek feedback from a wide audience).
The **Basic** student:

- Recognizes the advantage of using existing code.
- Recognizes reasons for testing and refining programs.
- Recognizes the advantage of documenting programs.
- Recognizes the role of using feedback.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.AP.PD.01 Using grade appropriate content and complexity, seek and incorporate feedback from team members and users to refine a solution to a problem. [**Practice 1.1** Fostering an Inclusive Computing Culture] [**Practice 2.3** Collaborating Around Computing]

The **Proficient** student seeks and incorporates feedback.

8.AP.PD.02 Incorporate existing code, media, and libraries into original programs of increasing complexity and give attribution. [**Practice 4.2** Developing and Using Abstractions] [**Practice 5.2** Creating Computational Artifacts] [**Practice 7.3** Communicating About Computing]

The **Proficient** student incorporates existing code, media, and libraries into original programs.

8.AP.PD.03 Systematically test and refine programs using a range of test cases. [**Practice 6.1** Testing and Refining Computational Artifacts]

The **Proficient** student systematically tests and refines programs.

***8.AP.PD.04** Using grade appropriate content and complexity, document programs in order to make them easier to follow, test, and debug. [**Practice 7.2** Communicating About Computing]

The **Proficient** student documents programs in order to make them easier to follow, test, and debug.

8.AP.PD.05 Distribute tasks and maintain a project timeline when collaboratively developing computational artifacts. [**Practice 2.2** Collaborating Around Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., adjust the timeline and redistribute tasks to meet the deadline).
The **Proficient** student, when collaboratively developing computational artifacts:

- Distributes tasks.
- Maintains a project timeline.

The **Basic** student, using a pre-written computational artifact:

- Identifies the project timeline tasks necessary for program development.
- Breaks down tasks and follows an individual timeline when developing a computational artifact.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

IMPACTS OF COMPUTING – Culture (C) and Social Interactions (SI)

8.IC.C 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., devises solutions to solve issues of bias in accessibility, reduce negative impacts of computing technology in everyday life).
The **Basic** student:

- Lists computing technologies that affect people's everyday activities, **and/or**
- lists computing technologies that affect people's career options, **and/or**
- identifies an accessibility issue related to technology.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.IC.C.01 Describe impacts associated with computing technologies that affect people's everyday activities and career options. [**Practice 7.2** Communicating About Computing]

The **Proficient** student:

- Describes impacts associated with computing technologies that affect people's everyday activities.
- Describes impacts associated with computing technologies that affect people's career options.

8.IC.C.02 Describe issues of bias and accessibility in the design of technologies. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Proficient** student describes issues of bias and accessibility in the design of technologies.

8.IC.SI 01 through 02 - PLDs other than Proficient are clustered as follows:

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., moderate, model appropriate behavior, and facilitate discussions in an online community).
The **Basic** student:

- Collaborates with peers using a tool in an attempt to create a computational artifact.
- Intermittently collaborates and behaves within an online community.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

8.IC.SI.01 Using grade appropriate content and complexity, collaborate using tools to connect with peers when creating a computational artifact. [**Practice 2.4** Collaborating Around Computing] [**Practice 5.2** Creating Computational Artifacts]

The **Proficient** student:

- Collaborates using tools to connect with peers when creating a computational artifact.
- Practices grade-level appropriate behavior and responsibilities while participating in an online community.

8.IC.SI.02 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [**Practice 2.1** Collaborating Around Computing] [**Practice 7.3** Communicating About Computing]

The **Proficient** student:

- Identifies and reports inappropriate behavior while participating in an online community, when applicable.

8.IC.SLE 01 through 02 - PLDs other than Proficient are clustered as follows:

***8.IC.SLE.01** Using grade appropriate content and complexity, describe tradeoffs between allowing information to be public and keeping information private and secure. [**Practice 7.2** Communicating About Computing]

The **Proficient** student:

- describes tradeoffs between allowing information to be public and keeping information private and secure, **and**
- When considering both positive and malicious intent regarding software development and use, discusses the following:
  - legal impacts
  - social impacts
  - ethical impacts

8.IC.SLE.02 Using grade level appropriate content and complexity, discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent. [**Practice 1.1** Fostering an Inclusive Computing Culture] [**Practice 7.2** Communicating About Computing]

The **Proficient** student:

- describes tradeoffs between allowing information to be public and keeping information private and secure, **and**
- with regard to positive and malicious intent:
  - discusses the legal impacts associated with software development and use,
  - discusses the social impacts associated with software development and use,
  - discusses the ethical impacts associated with software development and use.

9-12 Computer Science Content Standards & PLDs

*denotes a content standard with a connected performance standard

Introduction: In high school, students will continue to develop their knowledge of computing systems, their components, and how systems interact. Students will use their understanding about the basic principles of computation, that algorithms describe a step-by-step solution to a problem, that programs are algorithms written in a language that a computer can understand, and that the solution to many problems can be described as a program. A solid foundation of algebraic concepts is important for success in high school computer science courses. Students will expand their ability to identify patterns and create algorithms that can model the observed patterns.

By the end of 12th grade, students can:

- Create a computer program using sequencing, selection, and iteration.
- Decompose complex problems into smaller, more manageable sections.
- Use tools of coding to create, debug, and document the evolution of an artifact.
- Compare and contrast trade-offs in programming techniques.
- Develop complex computer program individually and as part of a group.
- Recognize how various components of a complex computing system work together.
- Use tools to analyze data and know how data is stored.
- Explain how cybersecurity issues affect networks and the internet.
- Justify how proliferation of computing affects privacy, rights, opportunities, and responsibility.

The high school standards are organized into 2 levels. Mostly, Level 1 is intended to be at the introductory level, and Level 2 reaches at a deeper level.

COMPUTING SYSTEMS - Devices (D), Hardware & Software (HS), and Troubleshooting (T)

L1.CS.D.01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student explains how abstractions hide the underlying implementation details of computing systems embedded in everyday objects.

The **Basic** student identifies abstractions that hide the underlying implementation details of computing systems embedded in everyday objects.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.CS.HS.01 Explain the interactions between application software, system software, and hardware layers. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., student demonstrates knowledge of specific, advanced terms for computer architecture, such as BIOS, kernel, or bus).

The **Proficient** student:

- Identifies the interactions between application software, system software, and hardware layers.
- Defines the interactions between application software, system software, and hardware layers.
- Explains the interactions between application software, system software, and hardware layers. For example, text editing software interacts with the operating system to receive input from the keyboard, convert the input to bits for storage, and interpret the bits as readable text to display on the monitor.

The **Basic** student:

- Identifies application software, system software, and hardware layers.
- Defines application software, system software, and hardware layers.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.CS.HS.01 Categorize the roles of operating system software. [**Practice 4.1** Developing and Using Abstractions] [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student categorizes the roles of the operating system software (e.g., roles could include memory management, data storage/retrieval, process management, and access control).

The **Basic** student categorizes some of the roles of operating system software.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.CS.T.01 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and resolve errors. [**Practice 6.1 & 6.2** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., someone with limited experience or knowledge could follow student developed guidelines).

The **Proficient** student develops guidelines independently that convey systematic troubleshooting strategies that others can use to identify and resolve errors (e.g., students could create a flow chart, a job aid for a help desk employee, or an expert system).

The **Basic** student develops guidelines with support that convey systematic troubleshooting strategies that others can use to identify and resolve errors.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.CS.T.01 Identify how hardware components facilitate logic, input, output, and storage in computing systems, and their common malfunctions. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student:

- Identifies how hardware components facilitate logic, input, output, and storage in computing systems.

- Identifies hardware components common malfunctions.

The **Basic** student identifies how some hardware components:

- facilitate logic, input, output, and storage in computing systems, **and/or**
- some of their common malfunctions.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

NETWORK & THE INTERNET - Network, Communication, & Organization (NCO) and Cybersecurity (C)

L1.NI.NCO.01 Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. [**Practice 4.1** Developing and Using Abstractions] [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., students can discuss different types of routers, switches, servers and/or topologies).

The **Proficient** student by describing the relationship between routers, switches, servers, topology, and addressing, evaluates the:

- scalability of networks.
- reliability of networks.

The **Basic** student:

- Identifies routers, switches, servers, topology, and addressing.
- Defines routers, switches, servers, topology, and addressing.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.NI.NCO.01** Describe the issues that impact network functionality (e.g., bandwidth, load, latency, topology). [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates an understanding of trade-offs between network functionality and design.

The **Proficient** student describes common issues that impact network functionality (e.g., bandwidth, load, latency, topology).

The **Basic** student describes a limited number of issues that impact network functionality (e.g., bandwidth, load, latency, topology).

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.NI.C.01 Give examples to illustrate how sensitive data can be affected by malware and other attacks. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student gives multiple detailed examples to illustrate how sensitive data can be affected by malware and other attacks.

The **Basic** student recalls examples to illustrate how sensitive data can be affected by malware and other attacks.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.NI.C.01 Compare ways software developers protect devices and information from unauthorized access. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., encryption strategies, authentication strategies).
The **Proficient** student compares ways software developers protect:

- devices from unauthorized access.
- information from unauthorized access.

The **Basic** student lists ways software developers protect:

- devices from unauthorized access.
- information from unauthorized access.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.NI.C.02 Recommend cybersecurity measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts. [**Practice 3.3** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student recommends cybersecurity measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts.
The **Basic** student identifies cybersecurity measures to address various scenarios.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.NI.C.03** Compare various security measures, considering trade-offs between the usability and security of a computing system. [**Practice 6.3** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., discuss security policies that are in place that present a tradeoff between usability and security).
The **Proficient** student compares two or more security measures, considering trade-offs between the usability and security of a computing system.
The **Basic** student:

- Identifies various security measures.
- Defines various security measures.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.NI.C.04 Explain trade-offs when selecting and implementing cybersecurity recommendations. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., make a recommendation and justify).
The **Proficient** student explains trade-offs from multiple perspectives using appropriate terminology when selecting and implementing cybersecurity recommendations.
The **Basic** student when selecting and implementing cybersecurity recommendations, can give an example of trade-offs:

- from a single viewpoint, **and/or**
- with inappropriate terminology.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

DATA ANALYSIS – Storage (S), Collection, Visualization, & Transformation (CVT), and Inference & Models (IM)

L1.DA.S.01 Translate between different bit representations of real-world phenomena, such as characters, numbers, and images. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student translates between different bit representations of real-world phenomena, such as characters, numbers, and images.
The **Basic** student can translate between a bit representation of real-world phenomena, such as characters, numbers, or images.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.DA.S.02 Evaluate the trade-offs in how data elements are organized and where data is stored. [**Practice 3.3** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., research emerging technologies for data storage and evaluate trade-off with current technologies).
The **Proficient** student evaluates the trade-offs in how data elements are organized and where data is stored.
The **Basic** student:

- Identifies the trade-offs in how data elements are organized and where data is stored.
- Describes the trade-offs in how data elements are organized and where data is stored.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.DA.CVT.01** Create interactive data representations using software tools to help others better understand real-world phenomena (e.g., paper surveys and online data sets). [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., research emerging visualization techniques and use them to create new data representations).
The **Proficient** student creates appropriate interactive data representations using software tools to help others better understand real-world phenomena.
The **Basic** student creates, with errors, interactive data representations using software tools.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.DA.CVT.01 Use data analysis tools and techniques to identify patterns in data representing complex systems. [**Practice 4.1** Developing and Using Abstractions] [**Practice 7.1** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., make a plausible predication based on pattern).

The **Proficient** student uses data analysis tools and techniques to identify correct patterns in data representing complex systems.
The **Basic** student uses data analysis tools and techniques to identify patterns in data representing complex systems but draws incorrect conclusions.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.DA.CVT.02** Select data collection tools and techniques, and use them to generate data sets that support a claim or communicate information. [**Practice 7.1 & 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student:

- Selects data collection tools and techniques.
- Uses data collection tools to generate data sets that support a claim or communicate information.

The **Basic** student:

- Selects data collection tools and techniques.
- Uses data collection tools and techniques to generate data sets but are unable to support a claim or communicate information.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.DA.IM.01 Create computational models that represent the relationships among different elements of data collected from a phenomenon or process. [**Practice 4.4** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, provides little to no evidence in addressing the expectation(s).
The **Proficient** student creates accurate computational models that represent the relationships among different elements of data collected from a phenomenon or process.
The **Basic** student creates computational models that represent the relationships among different elements of data collected from a phenomenon or process.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.DA.IM.01 Formulate, refine, and test scientific hypotheses using models and simulations. [**Practice 4.4** Developing and Using Abstractions]
The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student:

- Formulates scientific hypotheses using models and simulations.
- Refines scientific hypotheses using models and simulations.
- Tests scientific hypotheses using models and simulations.

The **Basic** student formulates scientific hypotheses using models and simulations.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

ALGORITHMS & PROGRAMMING – Algorithms (A), Variables (V), Control (C), Modularity (M), and Program Development (PD)

***L1.AP.A.01** Create a prototype that uses algorithms (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., student generated problem).
The **Proficient** student creates a prototype that uses appropriate algorithms (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student.
The **Basic** student creates a prototype that uses an algorithm (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.A.01 Critically examine and trace classic algorithms. Use and adapt classic algorithms to solve computational problems (e.g., selection sort, insertion sort, binary search, linear search). [**Practice 4.2** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., use and justify why a given algorithm is more efficient than another).
The **Proficient** student:

- Critically examines and traces classic algorithms.
- Uses classic algorithms to solve computational problems.
- Adapts classic algorithms to solve computational problems.

The **Basic** student:

- Examines and traces classic algorithms with minor errors.
- Uses classic algorithms to solve computational problems.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.A.02 Describe how artificial intelligence algorithms drive many software and physical systems. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., student discusses different types of artificial intelligence algorithms).
The **Proficient** student describes how artificial intelligence algorithms drive many software and physical systems.
The **Basic** student describes how artificial intelligence algorithms drive a software system or physical system.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.A.02 Develop an artificial intelligence algorithm to play a game against a human opponent or solve a real-world problem. [**Practice 5.2 & 5.3** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., uses heuristics to select the moves of the computer).
The **Proficient** student:

- Develops an artificial intelligence algorithm to play a game against a human opponent or solve a real-world problem.
- Correctly implements all rules of the game.

The **Basic** student:

- Develops an artificial intelligence algorithm to play a game against a human opponent or solve a real-world problem.
- Incorrectly captures some rules of the game.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.AP.A.03** Evaluate algorithms (e.g., sorting, searching) in terms of their efficiency, correctness, and clarity. [**Practice 4.2** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student evaluates algorithms in terms of their:

- efficiency
- correctness
- clarity

The **Basic** student evaluates algorithms in terms of their:

- efficiency, **or**
- correctness, **or**
- clarity

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.V.01 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., uses standard list operations like filter, map, and reduce).

The **Proficient** student independently uses lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

The **Basic** student with guidance, uses lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.V.01 Compare and contrast simple data structures and their uses (e.g., lists, stacks, queues). [**Practice 4.2** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., trees).

The **Proficient** student compares and contrasts simple linear data structures and their uses.

The **Basic** student:

- Identifies simple linear data structures and their uses.
- Explains simple linear data structures and their uses.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.C.01 Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., exception handling).

The **Proficient** student:

- Justifies the selection of specific control structures when tradeoffs involve implementation, readability, and program performance.
- Explains the benefits and drawbacks of choices.

The **Basic** student justifies the selection of specific control structures when tradeoffs involve:

- implementation, **or**
- readability, **or**
- program performance.

The **Below Basic** student… provides little to no evidence in addressing the expectation(s).

L2.AP.C.01 Trace the execution of recursion, illustrating output and changes in values of named variables. [**Practice 3.2** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., Fibonacci).

The **Proficient** student independently:

- Traces the execution of linear recursion.
- Illustrates output and changes in values of name variables (e.g., factorial function).

The **Basic** student with guidance:

- Traces the execution of recursion.
- Illustrates output and changes in values of name variables.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.C.02 Trace the execution of loops and conditional statements, illustrating output and changes in values of named variables. [**Practice 3.2** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student traces the execution of:

- loops illustrating output and changes in values of named variables, **and**
- conditional statements illustrating output and changes in values of named variables.

The **Basic** student traces the execution of:

- loops illustrating output and changes in values of named variables, **or**
- conditional statements illustrating output and changes in values of named variables.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.AP.C.03** Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., using multiple user interface components).

The **Proficient** student:

- Designs and iteratively develops a computational artifact that uses events to initiate instructions for:
  - practical intent,
  - personal expression, **or**
  - societal issues.

The **Basic** student designs computational artifacts that uses events to initiate instructions.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.AP.M.01** Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects. [**Practice 3.2** Recognizing and Defining Computational]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., an appropriate class hierarchy).

The **Proficient** student decomposes problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.

The **Basic** student partially decomposes problems into smaller components.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.AP.M.01** Construct solutions to problems using student-created components, such as procedures, modules, and/or objects. [**Practice 4.3** Developing and Using Abstractions] [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student constructs solutions to problems using student-created components, such as procedures, modules, and/or objects.

The **Basic** student with guidance, constructs solutions to problems using student-created components, such as procedures, modules, and/or objects.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.M.02 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student independently, creates artifacts by using:

- procedures within a program, **or**
- combinations of data and procedures, **or**
- independent but interrelated programs.

The **Basic** student with guidance, creates artifacts by using:

- procedures within a program, **or**
- combinations of data and procedures, **or**
- independent but interrelated programs.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.M.02 Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. [**Practice 4.1** Developing and Using Abstractions]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student:

- Analyzes a large-scale computational problem.
- Independently identifies generalizable patterns that can be applied to a solution.

The **Basic** student:

- Analyzes a large-scale computational problem and with guidance.
- Identifies generalizable patterns that can be applied to a solution.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.M.03 Demonstrate code reuse by creating programming solutions using libraries and APIs. [**Practice 4.2** Developing and Using Abstractions] [**Practice 5.3** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student independently, demonstrates code reuse by creating programming solutions using libraries and APIs.
The **Basic** student with guidance, demonstrates code reuse by creating programming solutions using libraries and APIs.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.AP.PD.01** Plan and develop programs by analyzing a problem and/or process, developing and documenting a solution, testing outcomes, and adapting the program for a variety of users. [**Practice 5.1** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, **independently** plans and develops programs by:

- analyzing a problem and/or process.
- developing and documenting a solution.
- testing outcomes.
- adapting the program for a variety of users.

The **Proficient** student plans and develops programs by:

- analyzing a problem and/or process.
- developing and documenting a solution.
- testing outcomes.
- adapting the program for a variety of users.

The **Basic** student, with instructor support, plans and develops programs by:

- analyzing a problem and/or process.
- developing and documenting a solution.
- testing outcomes.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.01 Plan and develop programs that will provide solutions to a variety of users using a software life cycle process. [**Practice 5.1** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, independently:

- Plans a program that will provide solutions to a variety of users using a software life cycle process.
- Develops a program that will provide solutions to a variety of users using a software life cycle process.

The **Proficient** student:

- Plans a program that will provide solutions to a variety of users using a software life cycle process.
- Develops a program that will provide solutions to a variety of users using a software life cycle process.

The **Basic** student with instructor support:

- Plans a program that will provide solutions to a variety of users using a software life cycle process.
- Develops a program that will provide solutions to a variety of users using a software life cycle process.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.PD.02 Evaluate licenses that limit or restrict use of computational artifacts when using resources such as libraries. [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student evaluates licenses that limit or restrict use of computational artifacts when using resources such as libraries (e.g., students might consider two software libraries that address a similar need, justifying their choice based on the library that has the least restrictive license).
The **Basic** student:

- Identifies licenses that limit or restrict use of computational artifacts when using resources such as libraries.
- Defines licenses that limit or restrict use of computational artifacts when using resources such as libraries.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.02 Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (e.g., code documentation) in a group software project. [**Practice 2.4** Collaborating Around Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student:

- Uses version control systems in a group software project.
- Uses integrated development environments (IDEs) in a group software project.
- Uses collaborative tools and practices (e.g., code documentation) in a group software project.

The **Basic** student:

- Uses integrated development environments (IDEs) in a group software project.
- Uses collaborative tools or practices (e.g., code documentation) in a group software project.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.PD.03 Use debugging tools to identify and fix errors in a program. [**Practice 6.2** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student tests and debugs (identify and fix errors) a program or algorithm to ensure it runs as intended.
The **Basic** student identifies strategies to test and debug (identify and fix errors) a program or algorithm to ensure it runs.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.03 Develop programs for multiple computing platforms. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, develops programs for multiple cross-platform computing platforms (e.g., platforms could include: computer desktop, web, or mobile).
The **Proficient** student develops programs for multiple computing platforms (e.g., disparate programs for different platforms: computer desktop, web, or mobile).
The **Basic** student with instructor support, develops programs for multiple computing platforms.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.AP.PD.04** Design and develop computational artifacts, working in team roles, using collaborative tools. [**Practice 2.4** Collaborating Around Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard. As programs grow more complex, the choice of resources that aid program development becomes increasingly important and should be made by the students.
The **Proficient** student designs and develops computational artifacts while working in teams and using collaborative tools.
The **Basic** student:

- Designs computational artifacts using collaborative tools.
- Develops computational artifacts using collaborative tools.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.AP.PD.04** Evaluate key qualities of a program through a process such as a code review (e.g., qualities could include correctness, usability, readability, efficiency, portability, and scalability). [**Practice 6.3** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, evaluates key qualities of a program and makes recommendations to improve that program through a process such as a code review (e.g., correctness, usability, readability, efficiency, portability, and scalability).
The **Proficient** student evaluates key qualities of a program (e.g., correctness, usability, readability, efficiency, portability, and scalability) through a process such as a code review.
The **Basic** student:

- Identifies key qualities of a program.
- Defines key qualities of a program (e.g., correctness, usability, readability, efficiency, portability, and scalability).

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.PD.05 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student documents design decisions using:

- text, graphics, presentations, **and/or**
- demonstrations in the development of complex programs.

The **Basic** student partially documents design decisions using:

- text, graphics, presentations, **and/or**

- demonstrations in the development of complex programs.

The **Below Basic** student… provides little to no evidence in addressing the expectation(s).

***L2.AP.PD.05** Develop and use a series of test cases to verify that a program performs according to its design specifications. [**Practice 6.1** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student develops and uses a series of test cases to verify that a program performs according to its design specifications.
The **Basic** student uses a series of test cases to verify that a program performs according to its design specifications.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.AP.PD.06 Evaluate and refine computational artifacts to make them more usable and accessible. [**Practice 6.3** Testing and Refining Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, supports others as they:

- evaluate computational artifacts to make them more usable and accessible.
- refine computational artifacts to make them more usable and accessible.

The **Proficient** student:

- Evaluates computational artifacts to make them more usable and accessible.
- Refines computational artifacts to make them more usable and accessible.

The **Basic** student with support:

- Evaluates computational artifacts to make them more usable and accessible.
- Refines computational artifacts to make them more usable and accessible.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.06 Explain security issues that might lead to compromised computer programs. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, explains and provides potential solutions for security issues that might lead to compromised computer programs.
The **Proficient** student explains security issues that might lead to compromised computer programs (e.g., lack of bounds checking, poor input validation, and circular references).
The **Basic** student:

- Identifies security issues that might lead to compromised computer programs.
- Describes security issues that might lead to compromised computer programs.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.07 Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality). [**Practice 5.3** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student:

- Modifies an existing program to add additional functionality.
- Discusses intended and unintended implications (e.g., breaking other functionality).

The **Basic** student modifies an existing program to add additional functionality.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.AP.PD.08 Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student:

- Compares multiple programming languages.
- Discusses how their features make them suitable for solving different types of problems.

The **Basic** student:

- Identifies multiple programming languages.
- Explains multiple programming languages.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

IMPACTS OF COMPUTING – Culture (C) and Social Interactions (SI)

L1.IC.C.01 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student evaluates the ways computing impacts personal, ethical, social, economic, and cultural practices.

The **Basic** student:

- Identifies the ways computing impacts personal, ethical, social, economic, and cultural practices.
- Defines the ways computing impacts personal, ethical, social, economic, and cultural practices.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.IC.C.01 Evaluate the beneficial and harmful effects that computational artifacts and innovations have on society. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard

The **Proficient** student evaluates the beneficial and harmful effects that computational artifacts and innovations have on society.

The **Basic** student:
- Identifies the beneficial and harmful effects that computational artifacts and innovations have on society.
- Defines the beneficial and harmful effects that computational artifacts and innovations have on society.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.C.02 Test and refine computational artifacts to reduce bias and equity deficits. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., student creates a computational artifact that utilizes accepted accessibility standards).
The **Proficient** student:
- Tests computational artifacts to reduce bias and equity deficits.
- Refines computational artifacts to reduce bias and equity deficits.

The **Basic** student identifies how computational artifacts reduce bias and equity deficits.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L2.IC.C.02** Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society. [**Practice 1.2** Fostering an Inclusive Computing Culture]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student evaluates the impact of the following on the distribution of computing resources in a global society:
- equity
- access
- influence

The **Basic** student provides examples for how the following impacts the distribution of computing resources in a global society:
- equity
- access
- influence

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.C.03 Demonstrate how a given algorithm applies to problems across disciplines. [**Practice 3.1** Recognizing and Defining Computational Problems]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student demonstrates how a given algorithm applies to problems across disciplines.
The **Basic** student identifies several disciplines a given algorithm applies to.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.IC.C.03 Predict how computational innovations that have revolutionized aspects of our culture might evolve. [**Practice 5.2** Creating Computational Artifacts]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.
The **Proficient** student predicts how computational innovations, that have revolutionized aspects of our culture, might evolve.
The **Basic** student identifies computational innovations that have revolutionized aspects of our culture.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.SI.01 Use tools and methods for collaboration. [**Practice 2.4** Collaborating Around Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard (e.g., students could compare and recommend ways different tools could help a team become more cohesive).
The **Proficient** student uses a variety of tools and methods for collaboration.
The **Basic** student uses a variety of tools and methods for collaboration.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.IC.SI.01 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [**Practice 2.1** Collaborating Around Computing] [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, models grade-level appropriate behavior and responsibilities while participating in an online community.
The **Proficient** student:

- Practices grade-level appropriate behavior and responsibilities while participating in an online community.
- Identifies and reports inappropriate behavior.

The **Basic** student generally practices grade-level appropriate behavior and responsibilities while participating in an online community.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.SI.02 Practice grade-level appropriate behavior and responsibilities while participating in an online community. Identify and report inappropriate behavior. [**Practice 2.1** Collaborating Around Computing] [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, models grade-level appropriate behavior and responsibilities while participating in an online community.
The **Proficient** student:

- Practices grade-level appropriate behavior and responsibilities while participating in an online community.
- Identifies and reports inappropriate behavior.

The **Basic** student generally practices grade-level appropriate behavior and responsibilities while participating in an online community.
The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.SLE.01 Explain the beneficial and harmful effects that intellectual property laws can have on innovation. [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student:

- Identifies a beneficial effect intellectual property laws have had on innovation.
- Identifies a harmful effect intellectual property laws have had on innovation.

The **Basic** student:

- Identifies a beneficial effect intellectual property laws have had on innovation, **and/or**
- Identifies a harmful effect intellectual property laws have had on innovation.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.IC.SLE.01 Debate laws and regulations that impact the development and use of software and technology. [**Practice 1.1** Recognizing and Defining Computational Problems] [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student debates laws and regulations that impact the development and use of software and technology.

The **Basic** student:

- Identifies laws and regulations that impact the development and use of software and technology.
- Defines laws and regulations that impact the development and use of software and technology.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L1.IC.SLE.02 Explain the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users. [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student explains the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users.

The **Basic** student identifies the privacy concerns related to the collection and generation of data through automated processes that may not be evident to users.

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

L2.IC.SLE.02 Using grade level appropriate content and complexity, discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent. [**Practice 1.1** Fostering an Inclusive Computing Culture] [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student, when considering both positive and malicious intent regarding software development and use, discusses the:

- legal impacts
- social impacts

- ethical impacts

The **Basic** student, when considering both positive and malicious intent regarding software development and use, provides examples of the:

- legal impacts, **or**
- social impacts, **or**
- ethical impacts

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.IC.SLE.03** Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. [**Practice 7.3** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student evaluates the following implications of privacy in the context of safety, law, or ethics:

- social implications
- economic implications

The **Basic** student provides examples of the following implications of privacy in the context of safety, law, or ethics:

- social implications
- economic implications

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

***L1.IC.SLE.04** Using grade level appropriate content and complexity, discuss the legal, social, and ethical impacts associated with software development and use, including both positive and malicious intent. [**Practice 1.1** Fostering an Inclusive Computing Culture] [**Practice 7.2** Communicating About Computing]

The **Advanced** student, in addition to meeting the Proficient Level, demonstrates in-depth inferences and applications that go beyond the understanding or context of the standard.

The **Proficient** student, when considering both positive and malicious intent regarding software development and use, discusses the:

- legal impacts
- social impacts
- ethical impacts

The **Basic** student, when considering both positive and malicious intent regarding software development and use, provides examples of the:

- legal impacts, **or**
- social impacts, **or**
- ethical impacts

The **Below Basic** student provides little to no evidence in addressing the expectation(s).

This page and the next are a quick snapshot of the whole set of Performance Standards. They are embedded within the Content Standards in this document, in blue highlight.

GRADE K-2 PERFORMANCE STANDARDS

***2.CS.HS.01** - The Proficient student demonstrates and describes: 1) a variety of hardware components (e.g., input devices, printers) and 2) software applications (e.g., search engine, browsers, apps).
***2.NI.C.01** - The Proficient student: 1) Explains what authentication factors (e.g., login) are and why we use them and 2) Models use of authentication factors to appropriate apps and devices.
***2.AP.A.01** - The Proficient student, with guidance: 1) Follows algorithms to complete tasks and 2) Creates algorithms to complete tasks.
***2.AP.C.01** - The Proficient student, with guidance: 1) Individually creates programs that include sequencing, conditionals, and repetition and 2) Collaboratively creates programs that include sequencing, conditionals, and repetition.
***2.AP.M.01** - The Proficient student, independently: 1) Decomposes a problem and 2) Creates a sequence of instructions to solve that problem.
***2.AP.PD.03** - The Proficient student debugs using a programming language: 1) Independently and 2) Collaboratively.
***2.IC.SI.01** - The Proficient student, independently: 1) Makes appropriate choices when participating in an online community; 2) Interacts respectfully with others; and 3) Reports inappropriate behavior to an adult.

GRADE 3-5 PERFORMANCE STANDARDS

***5.CS.HS.01** - The Proficient student creates a simple example of how a computing device takes input, stores information, processes input and information, and provides output (e.g., click volume button and the sound level increases).
***5.NI.C.01** - The Proficient student: 1) Describes, with specificity, real-world cybersecurity problems; 2) Discusses with peers using a variety of methods (e.g., blog, discussion post, oral) the personal consequences of inappropriate use; and 3) Identifies and implements appropriate strategies for how personal information can be protected.
***5.DA.S.01** - The Proficient student explains/justifies why a particular location or format is appropriate for the data they are creating or using.
***5.DA.IM.01** - The Proficient student independently: 1) Makes a statement based on a data set to describe a relationship, and then 2) Predicts an outcome or communicates an idea.
***5.AP.A.01** - The Proficient student: 1) Compares and refines multiple algorithms for the same task and 2) Determines which algorithm is the most appropriate for the same task.
***5.AP.V.01** - The Proficient student creates programs that use variables to: 1) store data and 2) modify data.
***5.AP.C.01** - The Proficient student: 1) Independently, creates programs that include combinations of sequences, events, loops, and conditionals. and 2) Collaboratively, creates programs that include combinations of sequences, events, loops, and conditionals.
***5.AP.PD.03** - The Proficient student independently tests and debugs a program or algorithm to ensure it runs as intended.
***5.IC.SI.02** - The Proficient student practices grade-level appropriate behavior and responsibilities while participating in an online community.
***5.IC.SLE.01** - The Proficient student: 1) Appropriately uses public domain and creative commons media and 2) Discusses the social impact of violating intellectual property rights.

GRADE 6-8 PERFORMANCE STANDARDS

***8.CS.HS.01** - The Proficient student: 1) Designs a project that combines hardware and software components to collect and exchange data and 2) Refines a project that combines hardware and software components to collect and exchange data.
***8.NI.C.01** - The Proficient student critiques physical and digital procedures that could be implemented to protect electronic data/information.

***8.DA.CVT.01** - The Proficient student uses computational tools to: 1) Transform data to improve reliability by removing errors and 2) Highlight or expose relationships in the data.
***8.AP.V.01** - The Proficient student: 1) Creates clearly named variables; 2) Creates variables that represent different data types; and 3) Performs operations on the values of variables.
***8.AP.C.01** - The Proficient student designs and iteratively develops programs that include: 1) nested loops and 2) compound conditionals.
***8.AP.M.01** - The Proficient student decomposes problems and sub-problems into parts for the design, implementation, and review of programs.
***8.AP.PD.04** - The Proficient student documents programs in order to make them easier to follow, test, and debug.
***8.IC.SLE.01** - The Proficient student: 1) describes tradeoffs between allowing information to be public and keeping information private and secure.

GRADE 9-12 LEVEL 1 PERFORMANCE STANDARDS

***L1.NI.C.03** - The Proficient student compares two or more security measures, considering trade-offs between the usability and security of a computing system.
***L1.DA.CVT.01** - The Proficient student creates appropriate interactive data representations using software tools to help others better understand real-world phenomena.
***L1.AP.A.01** - The Proficient student creates a prototype that uses appropriate algorithms (e.g., searching, sorting, finding shortest distance) to provide a possible solution for a real-world problem relevant to the student.
***L1.AP.C.03** - The Proficient student designs and iteratively develops a computational artifact that uses events to initiate instructions for: 1) practical intent, 2) personal expression, or 3) societal issues.
***L1.AP.M.01** - The Proficient student decomposes problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
***L1.AP.PD.01** - The Proficient student plans and develops programs by: 1) analyzing a problem and/or process; 2) developing and documenting a solution; 3) testing outcomes; and 4) adapting the program for a variety of users.
***L1.AP.PD.04** - The Proficient student designs and develops computational artifacts while working in teams and using collaborative tools.
***L1.IC.SLE.03** - The Proficient student evaluates the following implications of privacy in the context of safety, law, or ethics: 1) social implications and 2) economic implications.
***L1.IC.SLE.04** - The Proficient student, when considering both positive and malicious intent regarding software development and use, discusses the: 1) legal impacts, 2) social impacts, and 3) ethical impacts.

GRADE 9-12 LEVEL 2 PERFORMANCE STANDARDS

***L2.NI.NCO.01** - The Proficient student describes common issues that impact network functionality (e.g., bandwidth, load, latency, topology).
***L2.DA.CVT.02** - The Proficient student: 1) Selects data collection tools and techniques and 2) Uses data collection tools to generate data sets that support a claim or communicate information.
***L2.AP.A.03** - The Proficient student evaluates algorithms in terms of their: 1) efficiency, 2) correctness, and 3) clarity.
***L2.AP.M.01** - The Proficient student constructs solutions to problems using student-created components, such as procedures, modules, and/or objects.
***L2.AP.PD.04** - The Proficient student evaluates key qualities of a program (e.g., correctness, usability, readability, efficiency, portability, and scalability) through a process such as a code review.
***L2.AP.PD.05** - The Proficient student develops and uses a series of test cases to verify that a program performs according to its design specifications.
***L2.IC.C.02** - The Proficient student evaluates the impact of the following on the distribution of computing resources in a global society: 1) equity, 2) access, and 3) influence.