Foreword

This document serves as a guideline only. Teachers are encouraged to explore Project lesson plans, platform content and learning journals themselves to figure out how to integrate each Project into their curriculum units. Although Projects offer a complete experience with their own self-contained content, we recommend that teachers look for areas where they can incorporate additional content that makes use of their own unique experience, knowledge and classroom learning objectives. This document contains alignment information for the following curriculums:

If your curriculum is not yet included in this document, that does not mean that this Project will not be suitable for your class. We recommend all teachers explore the Project content themselves before deciding how to integrate it into their curriculum.

Table Key

The contents of this Project are well suited to delivering this aspect of the curriculum. As a teacher, you are free to skip over or de-emphasise content if this part of the curriculum is not something that you want to focus on.	**The contents of this Project can deliver part of this aspect of the curriculum.** This Project may require additional activities or discussions from the teacher to highlight key concepts of this aspect of the curriculum.	**The contents of this Project do not explicitly cover this aspect of the curriculum.** You can still use the simulations as a base for delivering your own content if you want to incorporate this aspect of the curriculum into the Project.

Australia - ACARA National Curriculum

Digital Technologies: Sequence of content F-10

Strand: Knowledge and understanding					
	F-2	3-4	5-6	7-8	9-10 (Elective subject)
Digital systems	Recognise and explore digital systems (hardware and software components) for a purpose (ACTDIK001)	Identify and explore a range of digital systems with peripheral devices for different purposes, and transmit different types of data (ACTDIK007)	Examine the main components of common digital systems and how they may connect together to form networks to transmit data (ACTDIK014)	Investigate how data is transmitted and secured in wired, wireless and mobile networks, and how the specifications affect performance (ACTDIK023)	Investigate the role of hardware and software in managing, controlling and securing the movement of and access to data in networked digital systems (ACTDIK034)
Representation of data	Recognise and explore patterns in data and represent data as pictures, symbols and diagrams (ACTDIK002)	Recognise different types of data and explore how the same data can be represented in different ways (ACTDIK008)	Examine how whole numbers are used to represent all data in digital systems (ACTDIK015)	Investigate how digital systems represent text, image and audio data in binary (ACTDIK024)	Analyse simple compression of data and how content data are separated from presentation (ACTDIK035)

Strand: Processes and production skills					
	F-2	3-4	5-6	7-8	9-10 (Elective subject)
Collecting, managing and analysing	Collect, explore and sort data, and use digital systems to present the data creatively	Collect, access and present different types of data using simple software to create information and	Acquire, store and validate different types of data, and use a range of software to interpret and visualise data	Acquire data from a range of sources and evaluate authenticity, accuracy and timeliness (ACTDIP025)	Develop techniques for acquiring, storing and validating quantitative and qualitative data from a

data	(ACTDIP003)	solve problems (ACTDIP009)	to create information (ACTDIP016)	Analyse and visualise data using a range of software to create information, and use structured data to model objects or events (ACTDIP026)	range of sources, considering privacy and security requirements (ACTDIP036) Analyse and visualise data to create information and address complex problems, and model processes, entities and their relationships using structured data (ACTDIP037)
Creating digital solutions by investigating and defining	Follow, describe and represent a sequence of steps and decisions (algorithms) needed to solve simple problems (ACTDIP004)	Define simple problems, and describe and follow a sequence of steps and decisions (algorithms) needed to solve them (ACTDIP010)	Define problems in terms of data and functional requirements drawing on previously solved problems (ACTDIP017)	Define and decompose realworld problems taking into account functional requirements and economic, environmental, social, technical and usability constraints (ACTDIP027)	Define and decompose realworld problems precisely, taking into account functional and non-functional requirements and including interviewing stakeholders to identify needs (ACTDIP038)
Creating digital solutions by generating and designing	n/a	n/a	Design a user interface for a digital system (ACTDIP018)	Design the user experience of a digital system, generating, evaluating and communicating alternative designs (ACTDIP028)	Design the user experience of a digital system by evaluating alternative designs against criteria including functionality, accessibility, usability, and aesthetics (ACTDIP039)
	n/a	n/a	Design, modify and follow simple algorithms involving sequences of steps, branching, and iteration (repetition) (ACTDIP019)	Design algorithms represented diagrammatically and in English, and trace algorithms to predict	Design algorithms represented diagrammatically and in structured English and validate algorithms and

				output for a given input and to identify errors (ACTDIP029)	programs through tracing and test cases (ACTDIP040)
Creating digital solutions by producing and implementing	n/a	Implement simple digital solutions as visual programs with algorithms involving branching (decisions) and user input (ACTDIP011)	Implement digital solutions as simple visual programs involving branching, iteration (repetition), and user input (ACTDIP020)	Implement and modify programs with user interfaces involving branching, iteration and functions in a general-purpose programming language (ACTDIP030)	Implement modular programs, applying selected algorithms and data structures including using an object-oriented programming language (ACTDIP041)
Creating digital solutions by evaluating	Explore how people safely use common information systems to meet information, communication and recreation needs (ACTDIP005)	Explain how student solutions and existing information systems meet common personal, school or community needs (ACTDIP012)	Explain how student solutions and existing information systems are sustainable and meet current and future local community needs (ACTDIP021)	Evaluate how student solutions and existing information systems meet needs, are innovative, and take account of future risks and sustainability (ACTDIP031)	Evaluate critically how student solutions and existing information systems and policies, take account of future risks and sustainability and provide opportunities for innovation and enterprise (ACTDIP042)
Creating digital solutions by collaborating and managing	Create and organise ideas and information using information systems independently and with others, and share these with known people in safe online environments (ACTDIP006)	Plan, create and communicate ideas and information independently and with others, applying agreed ethical and social protocols (ACTDIP013)	Plan, create and communicate ideas and information, including collaboratively online, applying agreed ethical, social and technical protocols (ACTDIP022)	Plan and manage projects that create and communicate ideas and information collaboratively online, taking safety and social contexts into account (ACTDIP032)	Create interactive solutions for sharing ideas and information online, taking into account safety, social contexts and legal responsibilities (ACTDIP043) Plan and manage projects using an iterative and collaborative approach, identifying risks and considering safety and sustainability (ACTDIP044)

England

National curriculum in England: computing programmes of study

Attainment targets			
Key stage 1	Key stage 2	Key stage 3	Key stage 4
Understand what algorithms are, how they are implemented as programs on digital devices, and that programs execute by following precise and unambiguous instructions	Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts	Design, use and evaluate computational abstractions that model the state and behaviour of real-world problems and physical systems	Develop their capability, creativity and knowledge in computer science, digital media and information technology
create and debug simple programs	Use sequence, selection, and repetition in programs; work with variables and various forms of input and output	Understand several key algorithms that reflect computational thinking [for example, ones for sorting and searching]; use logical reasoning to compare the utility of alternative algorithms for the same problem	Develop and apply their analytic, problem-solving, design, and computational thinking skills
Use logical reasoning to predict the behaviour of simple programs	Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs	Use 2 or more programming languages, at least one of which is textual, to solve a variety of computational problems; make appropriate use of data structures [for example, lists, tables or arrays]; design and develop modular programs that use procedures or functions	Understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to report a range of concerns
Use technology purposefully to create, organise, store, manipulate and retrieve digital content	Understand computer networks, including the internet; how they can provide multiple services, such as the World Wide Web, and the opportunities	Understand simple Boolean logic [for example, AND, OR and NOT] and some of its uses in circuits and programming; understand how numbers can be	

	they offer for communication and collaboration	represented in binary, and be able to carry out simple operations on binary numbers [for example, binary addition, and conversion between binary and decimal]	
Recognise common uses of information technology beyond school	Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content	Understand the hardware and software components that make up computer systems, and how they communicate with one another and with other systems	
Use technology safely and respectfully, keeping personal information private; identify where to go for help and support when they have concerns about content or contact on the internet or other online technologies	Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information	Understand how instructions are stored and executed within a computer system; understand how data of various types (including text, sounds and pictures) can be represented and manipulated digitally, in the form of binary digits	
	Use technology safely, respectfully and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact	Undertake creative projects that involve selecting, using, and combining multiple applications, preferably across a range of devices, to achieve challenging goals, including collecting and analysing data and meeting the needs of known users	
		Create, reuse, revise and repurpose digital artefacts for a given audience, with attention to trustworthiness, design and usability	
		Understand a range of ways to use technology safely, respectfully, responsibly and securely, including	

		protecting their online identity and privacy; recognise inappropriate content, contact and conduct, and know how to report concerns	

New Zealand

Technology Achievement Objectives & Progress Outcomes for levels 1 - 5

Achievement objectives (tasks)						
		Level 1	Level 2	Level 3	Level 4	Level 5
Technological practice	**Planning for practice**	Outline a general plan to support the development of an outcome, identifying appropriate steps and resources.	Develop a plan that identifies the key stages and the resources required to complete an outcome.	Undertake planning to identify the key stages and resources required to develop an outcome. Revisit planning to include reviews of progress and identify implications for subsequent decision making.	Undertake planning that includes reviewing the effectiveness of past actions and resourcing, exploring implications for future actions and accessing of resources, and consideration of stakeholder feedback, to enable the development of an outcome.	Analyse their own and others' planning practices to inform the selection and use of planning tools. Use these to support and justify planning decisions (including those relating to the management of resources) that will see the development of an outcome through to completion.
	Brief development	Describe the outcome they are developing and identify the attributes it should have, taking account of the need or opportunity and the resources available.	Explain the outcome they are developing and describe the attributes it should have, taking account of the need or opportunity and the resources available.	Describe the nature of an intended outcome, explaining how it addresses the need or opportunity. Describe the key attributes that enable development and evaluation of an outcome.	Justify the nature of an intended outcome in relation to the need or opportunity. Describe the key attributes identified in stakeholder feedback, which will inform the development of an outcome and its evaluation.	Justify the nature of an intended outcome in relation to the need or opportunity. Describe specifications that reflect key stakeholder feedback and that will inform the development of an outcome and its evaluation.

	Outcome development and evaluation	Investigate a context to communicate potential outcomes. Evaluate these against attributes; select and develop an outcome in keeping with the identified attributes.	Investigate a context to develop ideas for potential outcomes. Evaluate these against the identified attributes, select, and develop an outcome. Evaluate the outcome in terms of the need or opportunity.	Investigate a context to develop ideas for potential outcomes. Trial and evaluate these against key attributes to select and develop an outcome to address the need or opportunity. Evaluate this outcome against the key attributes and how it addresses the need or opportunity.	Investigate a context to develop ideas for feasible outcomes. Undertake functional modelling that takes account of stakeholder feedback in order to select and develop the outcome that best addresses the key attributes. Incorporating stakeholder feedback, evaluate the outcome's fitness for purpose in terms of how well it addresses the need or opportunity	Analyse their own and others' outcomes to inform the development of ideas for feasible outcomes. Undertake ongoing functional modelling and evaluation that takes account of key stakeholder feedback and trialling in the physical and social environments. Use the information gained to select and develop the outcome that best addresses the specifications. Evaluate the final outcome's fitness for purpose against the brief.
Technological knowledge	**Technological modelling**	Understand that functional models are used to represent reality and test design concepts and that prototypes are used to test technological outcomes.	Understand that functional models are used to explore, test, and evaluate design concepts for potential outcomes and that prototyping is used to test a technological outcome for fitness of purpose.	Understand that different forms of functional modelling are used to inform decision making in the development of technological possibilities and that prototypes can be used to evaluate the fitness of technological outcomes for further development.	Understand how different forms of functional modelling are used to explore possibilities and to justify decision making and how prototyping can be used to justify refinement of technological outcomes.	Understand how evidence, reasoning, and decision making in functional modelling contribute to the development of design concepts and how prototyping can be used to justify ongoing refinement of outcomes.
	Technological	Understand that	Understand that there	Understand the	Understand that	Understand how

	products	technological products are made from materials that have performance properties.	is a relationship between a material used and its performance properties in a technological product.	relationship between the materials used and their performance properties in technological products.	materials can be formed, manipulated, and/or transformed to enhance the fitness for the purpose of a technological product.	materials are selected, based on desired performance criteria.
	Technological systems	Understand that technological systems have inputs, controlled transformations, and outputs.	Understand that there are relationships between the inputs, controlled transformations, and outputs occurring within simple technological systems	Understand that technological systems are represented by symbolic language tools and understand the role played by the "black box" in technological systems.	Understand how technological systems employ control to allow for the transformation of inputs to outputs.	Understand the properties of subsystems within technological systems.
Nature of technology	Characteristics of technology	Understand that technology is purposeful intervention through design.	Understand that technology both reflects and changes society and the environment and increases people's capability.	Understand how society and environments impact on and are influenced by technology in historical and contemporary contexts and that technological knowledge is validated by successful function.	Understand how technological development expands human possibilities and how technology draws on knowledge from a wide range of disciplines.	Understand how people's perceptions and acceptance of technology impact on technological developments and how and why technological knowledge becomes codified.
	Characteristics of technological outcomes	Understand that technological outcomes are products or systems developed by people and have a physical nature and a functional nature.	Understand that technological outcomes are developed through technological practice and have related physical and functional natures.	Understand that technological outcomes are recognisable as fit for purpose by the relationship between their physical and functional natures.	Understand that technological outcomes can be interpreted in terms of how they might be used and by whom and that each has a proper function as well as possible alternative functions.	Understand that technological outcomes are fit for purpose in terms of time and context. Understand the concept of malfunction and how "failure" can inform future outcomes.

Progress outcomes (assessment)						
	Level 1	Level 2	Level 3	Level 4	Level 5	
Computational thinking for digital technologies	In authentic contexts and taking account of end users, students use their decomposition skills to break down simple non-computerised tasks into precise, unambiguous, step-by-step instructions (algorithmic thinking). They give these instructions, identify any errors in them as they are followed, and correct them (simple debugging).	In authentic contexts and taking account of end users, students give, follow and debug simple algorithms in computerised and non-computerised contexts. They use these algorithms to create simple programs involving outputs and sequencing (putting instructions one after the other) in age-appropriate programming environments.		In authentic contexts and taking account of end-users, students decompose problems into step-by-step instructions to create algorithms for computer programs. They use logical thinking to predict the behaviour of the programs, and they understand that there can be more than one algorithm for the same problem. They develop and debug simple programs that use inputs, outputs, sequence and iteration (repeating part of the algorithm with a loop).	In authentic contexts and taking account of end-users, students decompose problems to create simple algorithms using the three building blocks of programming: sequence, selection, and iteration. They implement these algorithms by creating programs that use inputs, outputs, sequence, basic selection using comparative operators, and iteration. They debug simple algorithms and programs by identifying when things go wrong with their instructions and correcting them, and they are able to explain why things went wrong and how they fixed them.	In authentic contexts and taking account of end users, students independently decompose problems into algorithms. They use these algorithms to create programs with inputs, outputs, sequence, selection using comparative and logical operators and variables of different data types, and iteration. They determine when to use different types of control structures. Students document their programs, using an organised approach for testing and debugging.

			Students understand that digital devices store data using just two states represented by binary digits (bits).	Students understand that digital devices represent data with binary digits and have ways of detecting errors in data storage and transmission. They evaluate the efficiency of algorithms, recognising that computers need to search and sort large amounts of data. They also evaluate user interfaces in relation to their efficiency and usability.	Students understand how computers store more complex types of data using binary digits, and they develop programs considering human-computer interaction (HCI) heuristics.
Designing and developing digital technologies	In authentic contexts and taking account of end users, students participate in teacher-led activities to develop, manipulate, store, retrieve and share digital content in order to meet technological challenges. In doing so, they identify digital devices and their purposes and understand that humans make them. They know how to use some applications, they can identify the inputs and outputs of a system, and they understand that digital devices store content, which can be retrieved later.		In authentic contexts and taking account of end-users, students make decisions about creating, manipulating, storing, retrieving, sharing and testing digital content for a specific purpose, given particular parameters, tools, and techniques. They understand that digital devices impact on humans and society and that both the devices and their impact change over time.	In authentic contexts, students follow a defined process to design, develop, store, test and evaluate digital content to address given contexts or issues, taking into account immediate social, ethical and end-user considerations. They identify the key features of selected software and choose the most appropriate software and file types to develop and combine digital content.	
			Students identify the specific role of components in a simple input-process-output system and how they work together, and they recognise the "control role" that humans have in the	Students understand the role of operating systems in managing digital devices, security, and application software and are able to apply file management conventions using a range of storage devices. They	

		system. They can select from an increasing range of applications and file types to develop outcomes for particular purposes.	understand that with storing data comes responsibility for ensuring security and privacy.

USA - California

Progression of California K–12 Computer Science Standards

	Grades K–2 Core	Grades 3–5 Core	Grades 6–8 Core	Grades 9–12 Core	Grades 9–12 Specialty
Computing Systems Devices	K-2.CS.1 Select and operate computing devices that perform a variety of tasks accurately and quickly based on user needs and preferences. (P1.1)	3-5.CS.1 Describe how computing devices connect to other components to form a system. (P7.2)	6-8.CS.1 Design modifications to computing devices in order to improve the ways users interact with the devices. (P1.2, P3.3)	9-12.CS.1 Describe ways in which abstractions hide the underlying implementation details of computing systems to simplify user experiences. (P4.1)	9-12S.CS.1 Illustrate ways computing systems implement logic through hardware components. (P4.4, P7.2)
Computing Systems Hardware & Software	K-2.CS.2 Explain the functions of common hardware and software components of computing systems. (P7.2)	3-5.CS.2 Demonstrate how computer hardware and software work together as a system to accomplish tasks. (P4.4)	6-8.CS.2 Design a project that combines hardware and software components to collect and exchange data. (P5.1)	9-12.CS.2 Compare levels of abstraction and interactions between application software, system software, and hardware. (P4.1)	9-12S.CS.2 Categorize and describe the different functions of operating system software. (P7.2)
Computing Systems Troubleshooting	K-2.CS.3 Describe basic hardware and software problems using accurate terminology. (P6.2, P7.2)	3-5.CS.3 Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies. (P6.2)	6-8.CS.3 Systematically apply troubleshooting strategies to identify and resolve hardware and software problems in computing systems. (P6.2)	9-12.CS.3 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. (P6.2)	n/a
Network & The Internet Network Communication & Organization	K-2.NI.4 Model and describe how people connect to other people, places, information and ideas through a network. (P4.4)	3-5.NI.4 Model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled	6-8.NI.4 Model the role of protocols in transmitting data across networks and the Internet. (P4.4)	9-12.NI.4 Describe issues that impact network functionality. (P4.1) 9-12.NI.5 Describe the design characteristics of the Internet. (P7.2)	9-12S.NI.3 Examine the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. (P4.4)

		at the destination. (P4.4)			9-12S.NI.4 Explain how the characteristics of the Internet influence the systems developed on it. (P7.2)
Network & The Internet Cybersecurity	K-2.NI.5 Explain why people use passwords. (P7.2)	3-5.NI.5 Describe physical and digital security measures for protecting personal information. (P3.1)	6-8.NI.5 Explain potential security threats and security measures to mitigate threats. (P3.1, P3.3)	9-12.NI.6 Compare and contrast security measures to address various security threats. (P7.2)	9-12S.NI.5 Develop solutions to security threats. (P5.3)
	K-2.NI.6 Create patterns to communicate a message. (P4.4)	3-5.NI.6 Create patterns to protect information from unauthorized access. (P4.4)	6-8.NI.6 Apply multiple methods of information protection to model the secure transmission of information. (P4.4)	9-12.NI.7 Compare and contrast cryptographic techniques to model the secure transmission of information. (P3.3, P4.4)	9-12S.NI.6 Analyze cryptographic techniques to model the secure transmission of information. (P3.3, P4.2)
Data & Analysis Storage	K-2.DA.7 Store, copy, search, retrieve, modify, and delete information using a computing device, and define the information stored as data. (P4.2)	3-5.DA.7 Explain that the amount of space required to store data differs based on the type of data and/or level of detail. (P4.2)	6-8.DA.7 Represent data in multiple ways. (P4.4)	9-12.DA.8 Translate between different representations of data abstractions of real-world phenomena, such as characters, numbers, and images. (P4.1) 9-12.DA.9 Describe tradeoffs associated with how data elements are organized and stored. (P3.3)	n/a
Data & Analysis Collection, Visualization, & Transformation	K-2.DA.8 Collect and present data in various visual formats. (P4.4, P7.1)	3-5.DA.8 Organize and present collected data visually to highlight relationships and support a claim. (P7.1)	6-8.DA.8 Collect data using computational tools and transform the data to make it more useful. (P7.1)	9-12.DA.10 Create data visualizations to help others better understand real-world phenomena. (P5.2)	9-12S.DA.7 Select and use data collection tools and techniques to generate data sets. (P7.1) 9-12S.DA.8 Use data analysis tools and techniques to identify patterns in data

					representing complex systems. (P4.1, P7.1)
Data & Analysis Inference & Models	K-2.DA.9 Identify and describe patterns in data visualizations, such as charts or graphs, to make predictions. (P4.1)	3-5.DA.9 Use data to highlight and/or propose relationships, predict outcomes, or communicate ideas. (P7.1)	6-8.DA.9 Test and analyze the effects of changing variables while using computational models. (P4.4, P6.1)	9-12.DA.11 Refine computational models to better represent the relationships among different elements of data collected from a phenomenon or process. (P4.4, P6.3)	9-12S.DA.9 Evaluate the ability of models and simulations to test and support the refinement of hypotheses. (P4.4)
Algorithms & Programming Algorithms	K-2.AP.10 Model daily processes by creating and following algorithms to complete tasks. (P3.2, P4.4)	3-5.AP.10 Compare and refine multiple algorithms for the same task and determine which is the most appropriate. (P3.3, P6.3)	6-8.AP.10 Use flowcharts and/or pseudocode to design and illustrate algorithms that solve complex problems. (P4.1, P4.4)	9-12.AP.12 Design algorithms to solve computational problems using a combination of original and existing algorithms. (P4.2, P5.1)	9-12S.AP.10 Describe how artificial intelligence drives many software and physical systems. (P3.1, P7.2) 9-12S.AP.11 Implement an algorithm that uses artificial intelligence to overcome a simple challenge. (P3.1, P5.3) 9-12S.AP.12 Implement searching and sorting algorithms to solve computational problems. (P4.2, P5.2) 9-12S.AP.13 Evaluate algorithms in terms of their efficiency. (P3.3)
Algorithms & Programming Variables	K-2.AP.11 Model the way programs store data. (P4.4)	3-5.AP.11 Create programs that use variables to store and modify data. (P5.2)	6-8.AP.11 Create clearly named variables that store data, and perform operations on their contents. (P5.1, P5.2)	9-12.AP.13 Create more generalized computational solutions using collections instead of repeatedly using simple variables. (P4.1)	9-12S.AP.14 Compare and contrast fundamental data structures and their uses. (P4.2)

Algorithms & Programming Control	K-2.AP.12 Create programs with sequences of commands and simple loops, to express ideas or address a problem. (P5.2)	3-5.AP.12 Create programs that include events, loops, and conditionals. (P5.2)	6-8.AP.12 Design and iteratively develop programs that combine control structures and use compound conditions. (P5.1, P5.2)	9-12.AP.15 Iteratively design and develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions. (P5.1, P5.2, P5.3) 9-12.AP.14 Justify the selection of specific control structures by identifying tradeoffs associated with implementation, readability, and performance. (P5.2)	9-12S.AP.15 Demonstrate the flow of execution of a recursive algorithm.(P3.2, P7.2)
Algorithms & Programming Modularity	K-2.AP.13 Decompose the steps needed to solve a problem into a sequence of instructions. (P3.2)	3-5.AP.13 Decompose problems into smaller, manageable tasks which may themselves be decomposed. (P3.2)	6-8.AP.13 Decompose problems and subproblems into parts to facilitate the design, implementation, and review of programs. (P3.2)	9-12.AP.16 Decompose problems into smaller subproblems through systematic analysis, using constructs such as procedures, modules, and/or classes. (P3.2)	9-12S.AP.16 Analyze a large-scale computational problem and identify generalizable patterns or problem components that can be applied to a solution. (P3.2, P4.2)
		3-5.AP.14 Create programs by incorporating smaller portions of existing programs, to develop something new or add more advanced features. (P4.2, P5.3)	6-8.AP.14 Create procedures with parameters to organize code and make it easier to reuse. (P4.1, P4.3)	9-12.AP.17 Create computational artifacts using modular design. (P4.3, P5.2)	9-12S.AP.17 Construct solutions to problems using student-created components, such as procedures, modules, and/or objects. (P4.3, P5.2) 9-12S.AP.18 Demonstrate code reuse by creating programming solutions using libraries and APIs. (P4.2, P5.3, P6.2)

Algorithms & Programming Program Development	K-2.AP.14 Develop plans that describe a program's sequence of events, goals, and expected outcomes. (P5.1, P7.2)	3-5.AP.15 Use an iterative process to plan and develop a program by considering the perspectives and preferences of others. (P1.1, P5.1)	6-8.AP.15 Seek and incorporate feedback from team members and users to refine a solution that meets user needs. (P1.1, P2.3)	9-12.AP.18 Systematically design programs for broad audiences by incorporating feedback from users. (P1.1, P5.1)	9-12S.AP.19 Plan and develop programs for broad audiences using a specific software life cycle process. (P2.2, P2.3, P5.2) 9-12S.AP.20 Develop programs for multiple computing platforms. (P5.2) 9-12S.AP.26 Compare multiple programming languages, and discuss how their features make them suitable for solving different types of problems. (P7.2)
	K-2.AP.15 Give attribution when using the ideas and creations of others while developing programs. (P7.3)	3-5.AP.16 Observe intellectual property rights and give appropriate attribution when creating, remixing, or combining programs. (P5.2, P7.3)	6-8.AP.16 Incorporate existing code, media, and libraries into original programs, and give attribution. (P4.2, P5.2, P7.3)	9-12.AP.19 Explain the limitations of licenses that restrict use of computational artifacts when using resources such as libraries. (P7.3)	9-12S.AP.23 Modify an existing program to add additional functionality and discuss intended and unintended implications. (P4.2, P5.3)
	K-2.AP.16 Debug errors in an algorithm or program that includes sequences and simple loops. (P6.2)	3-5.AP.17 Test and debug a program or algorithm to ensure it accomplishes the intended task. (P6.2)	6-8.AP.17 Systematically test and refine programs using a range of test cases. (P6.1)	9-12.AP.20 Iteratively evaluate and refine a computational artifact to enhance its performance, reliability, usability, and accessibility. (P6.3)	9-12S.AP.21 Identify and fix security issues that might compromise computer programs. (P6.2) 9-12S.AP.22 Develop and use a series of test cases to verify that a program performs according to its design specifications. (P6.1)
		3-5.AP.18 Perform different roles when collaborating	6-8.AP.18 Distribute tasks and maintain a project	9-12.AP.21 Design and develop computational	9-12S.AP.25 Use version control systems, integrated

		with peers during the design, implementation, and review stages of program development. (P2.2)	timeline when collaboratively developing computational artifacts. (P2.2, P5.1)	artifacts working in team roles using collaborative tools. (P2.4)	development environments (IDEs), and collaborative tools and practices (e.g., code documentation) while developing software within a group. (P2.4, P5.2)
	K-2.AP.17 Describe the steps taken and choices made during the iterative process of program development. (P7.2)	3-5.AP.19 Describe choices made during program development using code comments, presentations, and demonstrations. (P7.2)	6-8.AP.19 Document programs in order to make them easier to use, read, test, and debug. (P7.2)	9-12.AP.22 Document decisions made during the design process using text, graphics, presentations, and/or demonstrations in the development of complex programs. (P7.2)	9-12S.AP.24 Evaluate key qualities of a program through a process such as a code review. (P6.3)
Impacts of Computing Culture	K-2.IC.18 Compare how people lived and worked before and after the adoption of new computing technologies. (P3.1)	3-5.IC.20 Discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices. (P3.1)	6-8.IC.20 Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options. (P7.2)	9-12.IC.23 Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices. (P1.2, P3.1) 9-12.IC.26 Study, discuss, and think critically about the potential impacts and implications of emerging technologies on larger social, economic, and political structures, with evidence from credible sources. (P7.2) 9-12.IC.25 Demonstrate ways a given algorithm applies to problems across disciplines. (P3.1)	9-12S.IC.27 Evaluate computational artifacts with regard to improving their beneficial effects and reducing harmful effects on society. (P1.2, P6.1) 9-12S.IC.28 Evaluate how computational innovations that have revolutionized aspects of our culture might evolve. (P7.2)

		3-5.IC.21 Propose ways to improve the accessibility and usability of technology products for the diverse needs and wants of users. (P1.2)	6-8.IC.21 Discuss issues of bias and accessibility in the design of existing technologies. (P1.2)	9-12.IC.24 Identify impacts of bias and equity deficit on design and implementation of computational artifacts and apply appropriate processes for evaluating issues of bias. (P1.2)	9-12S.IC.29 Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society. (P1.2)
Impacts of Computing Social Interactions	K-2.IC.19 Work respectfully and responsibly with others when communicating electronically. (P2.1)	3-5.IC.22 Seek and explain the impact of diverse perspectives for the purpose of improving computational artifacts. (P1.1)	6-8.IC.22 Collaborate with many contributors when creating a computational artifact. (P2.4, P5.2)	9-12.IC.27 Use collaboration tools and methods to increase connectivity with people of different cultures and careers. (P2.4)	n/a
Impacts of Computing Safety, Law, & Ethics	K-2.IC.20 Describe approaches and rationales for keeping login information private, and for logging off of devices appropriately. (P3.1)	3-5.IC.23 Describe reasons creators might limit the use of their work. (P7.3)	6-8.IC.23 Compare tradeoffs associated with licenses for computational artifacts to balance the protection of the creators' rights and the ability for others to use and modify the artifacts. (P7.3) 6-8.IC.24 Compare tradeoffs between allowing information to be public and keeping information private and secure. (P7.2)	9-12.IC.28 Explain the beneficial and harmful effects that intellectual property laws can have on innovation. (P7.3) 9-12.IC.29 Explain the privacy concerns related to the collection and generation of data through automated processes. (P7.2) 9-12.IC.30 Evaluate the social and economic implications of privacy in the context of safety, law, or ethics. (P7.2)	9-12S.IC.30 Debate laws and regulations that impact the development and use of software. (P7.2)