Level 3B: GRADES 11-12		
Computing Systems (CS.3B) Conceptual understanding: People interact with a wide variety of computing devices that collect, store, analyze, and act upon information in ways that can affect human capabilities both positively and negatively. The physical components (hardware) and instructions (software) that make up a computing system communicate and process information in digital form. An understanding of hardware and software is useful when troubleshooting a computing system that does not work as intended.		
ID	**Standard**	**902148 GD2**
CS.3B.1	Categorize the roles of operating system software. Examples of roles could include memory management, data storage/retrieval, processes management, and access control. [HARDWARE & SOFTWARE] (P7.2)	
CS.3B.2	Illustrate ways computing systems implement logic, input, and output through hardware components. Examples of components could include logic gates and IO pins. [TROUBLESHOOTING] (P7.2)	
	Level 3B: GRADES 11-12 - Networks and the Internet	
Level 3B: GRADES 11-12 - Networks and the Internet		
ID	**Networks and the Internet (NI.3B)**	**902148 GD2**
	Conceptual understanding: Computing devices typically do not operate in isolation. Networks connect computing devices to share information and resources and are an increasingly integral part of computing. Networks and communication systems provide greater connectivity in the computing world by providing fast, secure communication and facilitating innovation.	
NI.3B.1	Describe the issues that impact network functionality (e.g., bandwidth, load, delay, topology). Recommend use of free online network simulators to explore how these issues impact network functionality. [NETWORK COMMUNICATION & ORGANIZATION] (P7.2)	
NI.3B.2	Compare ways software developers protect devices and information from unauthorized access. Examples of security concerns to consider: encryption and authentication strategies, secure coding, and safeguarding keys. [CYBERSECURITY] (P7.2)	
Level 3B: GRADES 11-12 - Data and Analysis		
ID	**Data and Analysis (DA.3B)**	**902148 GD2**
	Conceptual understanding: Computing systems exist to process data. The amount of digital data generated in the world is rapidly expanding, so the need to process data effectively is increasingly important. Data is collected and stored so that it can be analyzed to better understand the world and make more accurate predictions.	
DA.3B.1	Use data analysis tools and techniques to identify patterns in data representing complex systems. For example, identify trends in a dataset representing social media interactions, movie reviews, or shopping patterns. [COLLECTION, VISUALIZATION, & TRANSFORMATION] (P4.1)	
DA.3B.2	Select data collection tools and techniques to generate data sets that support a claim or communicate information. [COLLECTION, VISUALIZATION, & TRANSFORMATION] (P7.2)	
DA.3B.3	Evaluate the ability of models and simulations to test and support the refinement of hypotheses. [INFERENCE & MODELS] (P4.4)	
Level 3B: GRADES 11-12 - Algorithms and Programming		
ID	**Algorithms and Programming (AP.3B)**	**902148 GD2**
	Conceptual understanding: An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices. Algorithms and programming control all computing systems, empowering people to communicate with the world in new ways and solve compelling problems. The development process to create meaningful and efficient programs involves choosing which information to use and how to process and store it, breaking apart large problems into smaller ones, recombining existing solutions, and analyzing different solutions.	Unit 1, 2, 3, 5, 6, 7
AP.3B.1	Describe how artificial intelligence drives many software and physical systems. Examples include digital ad delivery, self-driving cars, and credit card fraud detection. [ALGORITHMS] (P7.2)	
AP.3B.2	Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem. Games do not have to be complex. Simple guessing games, Tic-Tac-Toe, or simple robot commands will be sufficient. [ALGORITHMS] (P5.3)	Unit 3, 5
AP.3B.3	Use and adapt classic algorithms to solve computational problems. Examples could include sorting and searching. [ALGORITHMS] (P4.2)	Unit 1, 2, 3, 5, 6, 7
AP.3B.4	Evaluate algorithms in terms of their efficiency, correctness, and clarity. Examples could include sorting and searching. [ALGORITHMS] (P4.2)	
AP.3B.5	Compare and contrast fundamental data structures and their uses. Examples could include strings, lists, arrays, stacks, and queues. [VARIABLES] (P4.2)	
AP.3B.6	Illustrate the flow of execution of a recursive algorithm. [CONTROL] (P3.2)	

AP.3B.7	Construct solutions to problems using student-created components, such as procedures, modules and/or objects. Object-oriented programming is optional at this level. Problems can be assigned or studentselected. [MODULARITY] (P5.2)	
AP.3B.8	Analyze a large-scale computational problem and identify generalizable patterns that can be applied to a solution. As students encounter complex, real-world problems that span multiple disciplines or social systems, they should decompose complex problems into manageable sub problems that could potentially be solved with programs or procedures that already exist. For example, students could create an app to solve a community problem by connecting to an online database through an application programming interface (API). [MODULARITY] (P4.1)	
AP.3B.9	Demonstrate code reuse by creating programming solutions using libraries and APIs. Libraries and APIs can be student-created or common graphics libraries or maps APIs, for example. [MODULARITY] (P5.3)	Unit 1, 2, 3, 5, 6, 7
AP.3B.10	Plan and develop programs for broad audiences using a software life cycle process. Processes could include agile, spiral, or waterfall. [PROGRAM DEVELOPMENT] (P5.1)	Unit 1, 2, 3, 5, 6, 7
AP.3B.11	Explain security issues that might lead to compromised computer programs. For example, common issues include lack of bounds checking, poor input validation, and circular references. [PROGRAM DEVELOPMENT] (P7.2)	
AP.3B.12	Develop programs for multiple computing platforms. Example platforms could include: computer desktop, web, or mobile. [PROGRAM DEVELOPMENT] (P5.2)	Unit 1, 2, 3, 5, 6, 7
AP.3B.13	Use version control systems, integrated development environments (IDEs), and collaborative tools and practices (code documentation) in a group software project. Group software projects can be assigned or student-selected. [PROGRAM DEVELOPMENT] (P2.4)	Unit 1, 2, 3, 5, 6, 7
AP.3B.14	Develop and use a series of test cases to verify that a program performs according to its design specifications. At this level, students are expected to select their own test cases. [PROGRAM DEVELOPMENT] (P6.1)	
AP.3B.15	Modify an existing program to add additional functionality and discuss intended and unintended implications (e.g., breaking other functionality). For instance, changes made to a method or function signature could break invocations of that method elsewhere in a system. [PROGRAM DEVELOPMENT] (P5.3)	Unit 1, 2, 3, 5, 6, 7
AP.3B.16	Evaluate key qualities of a program through a process such as a code review. Examples of qualities could include correctness, usability, readability, efficiency, portability and scalability. [PROGRAM DEVELOPMENT] (P6.3)	
AP.3B.17	Compare multiple programming languages and discuss how their features make them suitable for solving different types of problems. Examples of features include blocks versus text, indentation versus curly braces, and high-level versus low-level. [PROGRAM DEVELOPMENT] (P7.2)	
Level 3B: GRADES 11-12 - Impacts of Computing		
ID	**Impacts of Computing (IC.3B)**	**902148 GD2**
	Conceptual understanding: Computing affects many aspects of the world in both positive and negative ways at local, national, and global levels. Individuals and communities influence computing through their behaviors and cultural and social interactions, and in turn, computing influences new cultural practices. An informed and responsible person should understand the social implications of the digital world, including equity and access to computing.	
IC.3B.1	Evaluate computational artifacts to maximize their beneficial effects and minimize harmful effects on society. [CULTURE] (P6.1, 1.2)	
IC.3B.2	Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society. [CULTURE] (P1.2)	
IC.3B.3	Predict how computational innovations that have revolutionized aspects of our culture might evolve. Areas to consider might include education, healthcare, art/entertainment, and energy. [CULTURE] (P7.2)	
IC.3B.4	Debate laws and regulations that impact the development and use of software. Areas to consider might include education, healthcare, art/entertainment, energy, and artificial intelligence (for example ethical concerns around self-driving cars). [SAFETY, LAW, & ETHICS] (P7.3, 3.3)	