Computer Science Competencies and Skills

Type	Code	Descriptor	Statement or Guiding Question	State Standards Alignment	CSTA K-12	Portrait of a learner Alignment
Area	CS	Computer Science				
Competency	CS.1	**Program Development**				
Skill	CS.1.1	Program Function	How can we design and implement effective program functions that achieve specific tasks efficiently and reliably?	11.0101.1		
Skill	CS.1.2	Program Design	How can we create a well-structured program design that efficiently addresses the requirements and objectives of a software project while promoting maintainability and scalability?	11.0101.1		
Skill	CS.1.3	Identifying and Correcting Errors	How do we systematically identify and rectify errors and bugs in software code to ensure the reliability and functionality of a program?	11.0101.1		
Skill	CS.1.4	Working in Teams	What strategies and communication methods are essential for successful collaboration and teamwork in software development, and how can they be applied to deliver high-quality products?	11.0101.1		
Skill	CS.1.5	Development Cycle	What are the key phases and methodologies involved in the software development life cycle, and how can they be adapted to manage and complete projects effectively and on schedule?	11.0101.1		
Skill	CS.1.6	Documentation	How can comprehensive and well-structured documentation play a crucial role in ensuring the clarity, maintainability, and long-term sustainability of software projects?	11.0101.1		
Skill	CS.1.7	User Testing	How can user testing be leveraged to gather valuable feedback and insights from end-users to improve the user experience and functionality of a software application?	11.0101.1		
Competency	CS.2	**Programming**				
Skill	CS.2.1	Variables and Assignments	How can we effectively use variables and assignments to store and manipulate data in a program, and what impact does variable scope have on program behavior?	11.0101.1		
Theory	CS.2.2	Data Abstraction	What are the principles of data abstraction, and how can it be used to manage complexity and improve the maintainability of code?	11.0101.1		
Skill	CS.2.3	Mathematical Expressions	How can mathematical expressions and operations be leveraged to perform calculations and solve problems in programming, and what considerations are important for precision and efficiency?	11.0101.1		
Skill	CS.2.4	Strings	How do strings represent and manipulate text data in programming, and what are the key methods for working with string data effectively?	11.0101.1		
Skill	CS.2.5	Boolean Expressions	How are Boolean expressions used to make decisions and control program flow, and what role do logical operators play in evaluating and manipulating Boolean values?	11.0101.1		
Skill	CS.2.6	Conditionals	How can conditional statements be utilized to create decision-making processes in a program, and what are the implications of different conditional structures on program behavior?	11.0101.1		
Skill	CS.2.7	Iteration	What is the significance of iteration (looping) in programming, and how can it be employed to repeat tasks and process data efficiently?	11.0101.1		
Skill	CS.2.8	Procedures	How can the creation and use of procedures or functions enhance code organization and reusability in programming?	11.0101.1		
Skill	CS.2.9	Libraries	What are libraries in programming, and how do they provide pre-built functions and resources that can simplify and extend the functionality of a program?	11.0101.1		
Skill	CS.2.10	Random Values	How can the generation and utilization of random values be applied to introduce variability and unpredictability in program behavior?	11.0101.1		
Skill	CS.2.11	Objects	What is object-oriented programming, and how does it facilitate the modeling of real-world entities and relationships within a program?	11.0101.1		
Skill	CS.2.12	Regular Expressions	How can regular expressions be employed to pattern match and manipulate text data, and what are their applications in data validation and transformation?	11.0101.1		
Competency	CS.3	**Algorithms**				
Skill	CS.3.1	Developing Algorithms	How do we design and refine algorithms to solve complex problems, and what are the key principles and methodologies involved in this process?	11.0101.1		
Skill	CS.3.2	Lists / arrays	How do lists or arrays facilitate the organization and manipulation of multiple data elements in a program, and what are the trade-offs between different data structures?	11.0101.1		
Skill	CS.3.3	Searches	What strategies and techniques can be used to efficiently locate specific data within a collection, and how does the choice of search algorithm impact performance?	11.0101.1		
Skill	CS.3.4	Sorts	How can sorting algorithms be applied to arrange data in a specific order, and what are the comparative advantages and disadvantages of different sorting techniques?	11.0101.1		
Skill	CS.3.5	Simulations	How can computer simulations model real-world phenomena, and what role do algorithms and data structures play in creating accurate and meaningful simulations?	11.0101.1		
Skill	CS.3.6	Algorithmic Efficiency	How can we evaluate and improve the efficiency of algorithms, and what measures and techniques can be used to analyze and compare their performance?	11.0101.1		

Computer Science Competencies and Skills

Type	Code	Descriptor	Statement or Guiding Question	State Standards Alignment	CSTA K-12	Portrait of a learner Alignment
Theory	CS.3.7	Undecidable Problems	What are undecidable problems in computer science, and how do they challenge our understanding of computation and the limits of algorithmic solutions?	11.0101.1		
Theory	CS.3.8	Parallel and Distributed Computing	How can parallel and distributed computing paradigms be leveraged to solve large-scale problems, and what are the key considerations and challenges in these approaches?	11.0101.1		
Competency	**CS.4**	**Data**				
Skill	CS.4.1	Notation Systems	How do different notation systems, such as binary, hexadecimal, and symbolic notations, represent and manipulate data, and what are their practical applications in computing and other fields?	11.0101.4		
Skill	CS.4.2	Data Compression	What techniques and algorithms are employed to reduce the size of data while preserving its essential information, and how do they impact data storage, transmission, and processing?	11.0101.4		
Skill	CS.4.3	Extracting Information from Data	How can data analysis and mining techniques be used to uncover meaningful insights and patterns from large datasets, and what tools and methods are essential in this process?	11.0101.4		
Skill	CS.4.4	Using Programs with Data	How can software programs be designed to efficiently work with and process various types of data, and what considerations are vital for data integrity and security?	11.0101.4		
Skill	CS.4.5	Using Programs with Data Visualization	How does data visualization aid in understanding and conveying information, and what are the best practices for creating effective data visualizations that communicate insights accurately?	11.0101.4		
Skill	CS.4.6	Database Fundamentals	What are the fundamental concepts and components of databases, and how do they enable efficient data storage, retrieval, and management for a wide range of applications?	11.0101.4		
Competency	**CS.5**	**Systems**				
Skill	CS.5.1	Computer Hardware	How do the components and architecture of computer hardware systems work together to perform tasks, and what factors influence the selection of hardware for specific applications?	11.0101.3		
Skill	CS.5.2	Networking Concepts	What are the fundamental principles of networking, and how do they underpin the creation and maintenance of communication systems that connect devices and share data?	11.0101.2		
Skill	CS.5.3	Software	What is the role of software in computing, and how is it designed, developed, and maintained to enable diverse applications and functionality?	11.0101.3		
Skill	CS.5.4	Computer Interfacing	How do devices and software interfaces enable users to interact with computers, and what are the principles of user-centered design in computer interfacing?	11.0101.3		
Skill	CS.5.5	Internet of Things	How does the Internet of Things transform the way devices and objects connect and exchange data, and what opportunities and challenges does it present in various domains?	11.0101.2, 11.0101.3		
Skill	CS.5.6	Troubleshooting	What strategies and methodologies are effective for diagnosing and resolving issues in computer systems and networks, and how can systematic troubleshooting be applied?	11.0101.3		
Skill	CS.5.7	Cybersecurity	How can cybersecurity measures and practices protect computer systems, networks, and data from threats, and what are the ethical and legal implications of cybersecurity in today's digital world?	11.0101.5		
Competency	**CS.6**	**Impact of Society**				
Theory	CS.6.1	Technology and the Environment	How does technology impact the environment, and what innovative solutions and sustainable practices can be employed to mitigate its negative effects on our planet?	11.0101.5		
Theory	CS.6.2	Technology and Society	How do advancements in technology influence social structures, culture, and everyday life, and what are the ethical and social implications of these changes?	11.0101.5		
Theory	CS.6.3	Ethics and Security	What ethical considerations are essential in the development and use of technology, particularly in the context of cybersecurity and personal data protection, and how can these considerations be integrated into security practices?	11.0101.5		
Theory	CS.6.4	Artificial Intelligence	What is the current state of artificial intelligence, and how does it challenge traditional notions of intelligence, decision-making, and the relationship between humans and machines?	11.0101.5		

CS.1 Creative Development

I can create and innovate using an iterative design process that is user focused, that incorporates implementation/feedback cycles, and leave ample room for experimentation and risk-taking.

	Level 9	Level 10	Level 11	Level 12
CS.1.1 Program Function	I can recall and recognize basic program function concepts, such as input and output, but may struggle to explain the purposes and capabilities of software programs.	I can describe common program functions, such as data processing, user interfaces, and output generation, and can create simple programs that perform basic tasks, demonstrating an understanding of program functionality.	I can analyze and design programs that provide more complex and specialized functions, including data analysis, automation, and interactive features, and can demonstrate an understanding of program architecture and advanced functionality.	I can innovate by developing advanced software applications that offer unique and powerful functions, creating custom software solutions, optimizing program efficiency, and addressing complex real-world functional requirements, showcasing expertise in program function, creative problem-solving, and software development.
CS.1.2 Program Design	I can recall and recognize basic program design and development concepts, such as writing code and defining variables, but may struggle to explain systematic software development methodologies and their applications.	I can describe common programming constructs, such as loops and conditional statements, and can write code to solve simple problems, demonstrating an understanding of the fundamentals of program design and development.	I can analyze and design software solutions for more complex problems, break down problems into smaller tasks, and develop well-structured and efficient code, demonstrating an understanding of software design and development principles.	I can innovate by developing advanced software solutions, creating custom algorithms, optimizing code performance, and addressing complex real-world programming challenges, showcasing expertise in program design and development, creative problem-solving, and software engineering.
CS.1.3 Identifying and Correcting Errors	I can recall and recognize basic error identification concepts, such as debugging and error messages, but may struggle to explain systematic error correction methodologies and their applications.	I can describe common error types, such as syntax errors and logic errors, and can use debugging tools and techniques to identify and correct basic programming errors, demonstrating an understanding of the fundamentals of error detection and correction.	I can analyze and design advanced debugging strategies, trace complex errors, and propose efficient solutions to programming problems, demonstrating an understanding of error analysis and advanced debugging techniques.	I can innovate by developing advanced error prevention measures, creating custom debugging tools, implementing automated testing, and addressing complex real-world programming challenges, showcasing expertise in error identification and correction, creative problem-solving, and software quality assurance.
CS.1.4 Working in Teams	I can recall and recognize basic teamwork concepts, such as collaboration and communication, but may struggle to explain key principles of effective teamwork and their applications.	I can describe common team roles and communication tools, such as project management software and version control systems, and can participate in basic teamwork activities, demonstrating an understanding of basic collaborative processes.	I can analyze and design effective team structures, assign roles, coordinate tasks, and facilitate communication within a team, and can identify and address teamwork challenges to enhance team performance.	I can innovate by developing advanced team collaboration strategies, creating custom workflows, optimizing team productivity, and addressing complex real-world teamwork challenges, showcasing expertise in teamwork, creative problem-solving, and leadership in collaborative projects.

CS.1 Creative Development

I can create and innovate using an iterative design process that is user focused, that incorporates implementation/feedback cycles, and leave ample room for experimentation and risk-taking.

	Level 9	Level 10	Level 11	Level 12
CS.1.5 Development Cycle	I can recall and recognize basic development cycle concepts, such as software development phases, but may struggle to explain how development cycles are structured and executed.	I can describe common development cycle phases, such as planning, design, implementation, testing, and maintenance, and can outline the sequence of activities in a typical development cycle.	I can analyze and design development cycles for specific software projects, identify appropriate methodologies (e.g., Agile, Waterfall), and create detailed development plans that consider project requirements, timelines, and resources.	I can innovate by developing advanced development cycle strategies, customizing development methodologies, optimizing development processes, and addressing complex real-world development challenges, showcasing expertise in the software development cycle, advanced problem-solving, and creative project management.
CS.1.6 Documentation		I can recall and recognize basic documentation concepts, such as the importance of documenting code and software projects, but may struggle to explain systematic documentation practices and their significance.	I can describe common documentation elements, including comments in code, README files, and user manuals, and can create basic documentation for software projects, demonstrating an understanding of the essentials of clear and concise documentation.	I can plan and produce more comprehensive documentation, including code documentation standards, user guides, and technical manuals, and can apply documentation best practices to ensure usability, maintainability, and clarity in software projects.
CS.1.7 User Testing	I can recall and recognize basic user testing concepts, such as usability testing and user feedback, but may struggle to explain systematic user testing methodologies and their importance.	I can describe common user testing methods, including usability testing, surveys, and user interviews, and can conduct basic user testing sessions to collect feedback, demonstrating an understanding of user testing fundamentals.	I can plan, design, and execute more complex user testing procedures, analyze user feedback, identify usability issues, and propose improvements based on user testing results, showcasing an understanding of user-centric design and evaluation.	I can innovate by developing advanced user testing strategies, creating custom testing scenarios, implementing A/B testing, and addressing complex real-world usability challenges, showcasing expertise in user testing, creative problem-solving, and user-driven design.

Last edited: 11-03-2023

CS.2 Programming

Students will understand that variables and assignments, data abstraction, mathematical expressions, strings, boolean expressions, conditionals, iteration, procedures, and libraries serve as fundamental building blocks in computer programming. Additionally, students will appreciate how the integration of random values and the use of objects allow for creative and effective problem-solving, leading to the development of robust and efficient software solutions.

	Level 9	Level 10	Level 11	Level 12
CS.2.1 Variables and Assignments	I can identify and define variables, but my use of them is basic, and I struggle with more complex assignments.	I can effectively declare and initialize variables in code, assign values to them, and use them in simple operations, like basic math or string concatenation, but I may have some difficulty with more advanced concepts.	I can use variables and assignments to solve moderately complex programming problems, work with different data types, and use variables in control structures, understanding their scope and data types' impact.	I can create complex programs that rely heavily on variables and assignments, use them in advanced programming techniques and data structures, optimize code, and demonstrate creativity in inventing new, efficient ways to use variables and assignments in code.
CS.2.2 Data Abstraction	I can recognize basic data types like integers and strings, but I struggle to explain or use more complex data abstraction concepts.	I can declare and initialize simple data structures and use basic data abstraction techniques like arrays and lists, but I may need guidance for more advanced data structures.	I can effectively apply data abstraction by creating and manipulating more complex data structures, such as linked lists or dictionaries, and use them in programming tasks, demonstrating a solid understanding of data abstraction concepts.	I can design and implement advanced data abstraction techniques and structures, optimize their use in code, and showcase creativity in inventing new, efficient ways to apply data abstraction in complex programming projects.
CS.2.3 Mathematical Expressions	I can identify basic mathematical operators (+, -, *, /) and use them for simple calculations, but I struggle with more complex expressions or order of operations.	I can use mathematical operators to perform calculations and create simple mathematical expressions, understanding the order of operations, but I may need guidance for more complex expressions.	I can create and manipulate mathematical expressions in code, incorporating variables and complex operators, and use them in practical programming tasks, demonstrating an understanding of mathematical concepts in coding.	I can create and optimize complex mathematical expressions, combining multiple operators and functions, and apply them to solve advanced programming problems, showcasing creativity and efficiency in mathematical expression usage within code.
CS.2.4 Strings	I can identify basic string operations like concatenation and length retrieval, but I struggle to perform more complex string manipulations or explain advanced concepts.	I can use basic string operations to manipulate and combine strings effectively, but I may need assistance with more advanced string handling tasks or regular expressions.	I can create and manipulate strings using advanced techniques, such as regular expressions or substring extraction, and apply them in practical programming scenarios, demonstrating a solid understanding of string manipulation concepts.	I can design and implement complex string handling solutions, optimize their use in code, and showcase creativity in inventing new, efficient ways to work with strings in challenging programming projects.
CS.2.5 Boolean Expressions	I can recognize basic Boolean operators (AND, OR, NOT) and use them for simple true/false evaluations, but I struggle with more complex Boolean expressions.	I can create and evaluate simple Boolean expressions, understanding the basic logic of combining conditions using AND, OR, and NOT, but I may need guidance for more complex expressions.	I can create and analyze complex Boolean expressions, combining multiple conditions and using parentheses effectively, and apply them in practical programming tasks, demonstrating a solid understanding of Boolean expression concepts.	I can design and optimize intricate Boolean expressions, applying advanced logic and combining conditions creatively to solve complex programming problems, showcasing creativity and efficiency in Boolean expression usage within code.

CS.2 Programming

Students will understand that variables and assignments, data abstraction, mathematical expressions, strings, boolean expressions, conditionals, iteration, procedures, and libraries serve as fundamental building blocks in computer programming. Additionally, students will appreciate how the integration of random values and the use of objects allow for creative and effective problem-solving, leading to the development of robust and efficient software solutions.

	Level 9	Level 10	Level 11	Level 12
CS.2.6 Conditionals	I can identify basic conditional structures (e.g., if statements), but I struggle with creating more complex conditions or nested if-else structures.	I can use basic conditional structures (e.g., if-else) to make simple decisions in code, but I may need assistance with more complex conditional logic or nested conditions.	I can create and evaluate complex conditional structures, incorporating logical operators and nested conditions effectively, and apply them in practical programming tasks, demonstrating a solid understanding of conditional concepts.	I can design and optimize intricate conditional structures, applying advanced logical reasoning and creative combinations of conditions to solve complex programming problems, showcasing creativity and efficiency in conditional usage within code.
CS.2.7 Iteration	I can identify basic iteration structures (e.g., loops), but I struggle with creating or understanding their practical use in code.	I can use basic iteration structures (e.g., for and while loops) to repeat simple tasks in code, but I may need guidance for more complex loops or understanding their practical applications.	I can create and use complex iteration structures, effectively incorporating loops with various conditions and iterations, and apply them in practical programming tasks, demonstrating a solid understanding of iteration concepts.	I can design and optimize intricate and creative iteration structures, using advanced logic and loops to solve complex programming problems efficiently, showcasing creativity and efficiency in iteration usage within code.
CS.2.8 Procedures	I can recognize the concept of calling procedures (functions or methods), but I struggle to create or explain their use in code.	I can call and use simple procedures in code, understanding their basic input and output, but I may need guidance for more complex functions or methods.	I can create and call complex procedures, incorporating multiple parameters and return values effectively, and apply them in practical programming tasks, demonstrating a solid understanding of calling procedure concepts.	I can design and optimize intricate and creative procedures, using advanced logic and functions/methods to solve complex programming problems efficiently, showcasing creativity and efficiency in calling procedures within code.
CS.2.9 Libraries	I can recognize the concept of libraries and their basic functions, but I struggle to explain or use them effectively in code.	I can use basic libraries and their functions in code, but I may need guidance for more advanced library usage or customization.	I can effectively use and customize libraries, incorporating their functions and features into practical programming tasks, demonstrating a solid understanding of library concepts.	I can design and implement complex library integrations, optimize their use in code, and showcase creativity in inventing new, efficient ways to leverage libraries in complex programming projects.
CS.2.10 Random Values	I can recognize the concept of generating random values, but I struggle to explain or effectively use random values in code.	I can use basic methods for generating random values in code, but I may need guidance for more advanced use cases or understanding their practical applications.	I can create and use advanced methods for generating random values, incorporating them into practical programming tasks, demonstrating a solid understanding of random value generation concepts.	I can design and optimize complex techniques for generating random values, applying advanced logic and creativity to solve complex programming problems efficiently, and showcasing creativity and efficiency in random value generation within code.

CS.2 Programming

Students will understand that variables and assignments, data abstraction, mathematical expressions, strings, boolean expressions, conditionals, iteration, procedures, and libraries serve as fundamental building blocks in computer programming. Additionally, students will appreciate how the integration of random values and the use of objects allow for creative and effective problem-solving, leading to the development of robust and efficient software solutions.

	Level 9	Level 10	Level 11	Level 12
CS.2.11 Objects	I can recognize the concept of objects in computer science and identify basic attributes or methods associated with objects.	I can create simple objects, define their attributes, and use basic methods, demonstrating an understanding of object-oriented programming fundamentals.	I can design and implement complex classes and objects, encapsulating data and behavior, applying inheritance, polymorphism, and other advanced object-oriented principles effectively.	I can innovate by designing and implementing intricate and creative object-oriented solutions, leveraging design patterns and advanced techniques, showcasing creativity and expertise in object-oriented programming for complex projects.
CS.2.12 Regular Expressions		I can recognize the concept of regular expressions and identify basic regular expressions, such as simple character matches and metacharacters like '*', '?', and '.'.	I can apply basic regular expressions to perform simple text matching and extraction tasks, using character classes, and quantifiers in regular expressions.	I can design and implement complex regular expressions for advanced text processing, effectively using lookaheads, lookbehinds, capturing groups, and backreferences.

Last edited: 11-02-2023

CS.3 Algorithms

Students will develop a deep understanding of computer science, enabling them to recognize that developing algorithms, utilizing lists, searches, sorts, simulations, optimizing algorithmic efficiency, and understanding undecidable problems provide the foundation for addressing complex computational challenges. Additionally, students will appreciate how parallel and distributed computing technologies extend computational capabilities, enabling the efficient and scalable solution of complex problems in modern computing environments.

	Level 9	Level 10	Level 11	Level 12
CS.3.1 Developing Algorithms	I can recognize the concept of developing algorithms, but I struggle to create or explain algorithms effectively.	I can design and implement simple algorithms to solve basic problems, but I may need guidance for more complex algorithm development or understanding their practical applications.	I can create and apply complex algorithms, incorporating multiple steps and logical reasoning, to solve practical programming tasks, demonstrating a solid understanding of algorithm development concepts.	I can design and optimize intricate and creative algorithms, using advanced logical reasoning and creative combinations of steps to solve complex programming problems efficiently, showcasing creativity and efficiency in algorithm development within code.
CS.3.2 Lists / arrays	I can recognize the concept of lists as a data structure, but I struggle to create or explain their use effectively in code.	I can declare and use simple lists in code to store and access data, but I may need guidance for more complex list operations or understanding their practical applications.	I can design and manipulate complex lists, effectively incorporating operations like insertion, deletion, and searching, to solve practical programming tasks, demonstrating a solid understanding of list concepts.	I can design and optimize intricate and creative list structures, using advanced list manipulation techniques to solve complex programming problems efficiently, showcasing creativity and efficiency in list usage within code.
CS.3.3 Searches	I can recognize the concept of searches, but I struggle to create or explain basic search algorithms effectively in code.	I can implement basic search algorithms (e.g., linear search) in code to find elements in a collection, but I may need guidance for more complex search algorithms or understanding their practical applications.	I can design and implement complex search algorithms (e.g., binary search) effectively, applying them to solve practical programming tasks and demonstrating a solid understanding of search concepts.	I can design and optimize intricate and creative search algorithms, using advanced search techniques and adapting them to solve complex programming problems efficiently, showcasing creativity and efficiency in search algorithm usage within code.
CS.3.4 Sorts	I can recognize the concept of sorting, but I struggle to create or explain basic sorting algorithms effectively in code.	I can implement basic sorting algorithms (e.g., bubble sort or selection sort) in code to arrange elements in a collection, but I may need guidance for more complex sorting algorithms or understanding their practical applications.	I can design and implement complex sorting algorithms (e.g., merge sort or quicksort) effectively, applying them to solve practical programming tasks, and demonstrating a solid understanding of sorting concepts.	I can design and optimize intricate and creative sorting algorithms, using advanced sorting techniques and adapting them to solve complex programming problems efficiently, showcasing creativity and efficiency in sort algorithm usage within code.
CS.3.5 Simulations	I can recognize the concept of simulations, but I struggle to create or explain basic simulations effectively in code.	I can implement simple simulations in code to model basic systems or processes, but I may need guidance for more complex simulations or understanding their practical applications.	I can design and develop complex simulations, incorporating multiple elements and logical rules, effectively modeling practical systems or processes in programming tasks, demonstrating a solid understanding of simulation concepts.	I can design and optimize intricate and creative simulations, using advanced logic and modeling techniques to create realistic and efficient simulations of complex systems, showcasing creativity and efficiency in simulation development within code.

CS.3 Algorithms

Students will develop a deep understanding of computer science, enabling them to recognize that developing algorithms, utilizing lists, searches, sorts, simulations, optimizing algorithmic efficiency, and understanding undecidable problems provide the foundation for addressing complex computational challenges. Additionally, students will appreciate how parallel and distributed computing technologies extend computational capabilities, enabling the efficient and scalable solution of complex problems in modern computing environments.

	Level 9	Level 10	Level 11	Level 12
CS.3.6 Algorithmic Efficiency	I can recognize the concept of algorithmic efficiency, but I struggle to explain or optimize algorithms effectively in code.	I can analyze and optimize basic algorithms for efficiency, understanding concepts like time complexity and space complexity, but I may need guidance for more complex algorithms.	I can design and develop complex algorithms with a focus on efficiency, demonstrating the ability to optimize time and space complexity in practical programming tasks, and showing a solid understanding of algorithmic efficiency concepts.	I can design and optimize intricate and creative algorithms, using advanced techniques for achieving the highest levels of efficiency in solving complex programming problems, showcasing creativity and expertise in algorithmic efficiency within code.
CS.3.7 Undecidable Problems	I can recognize the concept of undecidable problems, but I struggle to explain or identify them effectively in code or theoretical contexts.	I can identify and discuss basic undecidable problems in theory, but I may need guidance for more complex undecidable problems or understanding their practical applications.	I can analyze and demonstrate a solid understanding of complex undecidable problems in theory, effectively discussing their implications in various computational contexts and providing insights into their significance.	I can critically evaluate intricate and creative undecidable problems, using advanced logic to explore the boundaries of computational theory, showcasing creativity and deep understanding in dealing with undecidable problems in theoretical and practical contexts.
CS.3.8 Parallel and Distributed Computing	I can recognize the concept of parallel and distributed computing, but I struggle to explain or apply basic concepts effectively in code or theoretical contexts.	I can understand and use basic parallel and distributed computing concepts in theory, such as multi-threading or client-server models, but I may need guidance for more complex applications or understanding their practical implications.	I can design and implement complex parallel and distributed computing systems, effectively applying concepts like parallel processing, message passing, or distributed databases in practical programming tasks, demonstrating a solid understanding of parallel and distributed computing concepts.	I can design and optimize intricate and creative parallel and distributed computing solutions, using advanced techniques and algorithms to solve complex problems efficiently, showcasing creativity and expertise in parallel and distributed computing within code and system architecture.

Last edited: 11-02-2023

CS.4 Data

Students will demonstrate proficiency in computer science through a strong grasp of Notation Systems, Data Compression, Extracting Information from Data, Using Programs with Data, and Database Fundamentals. They will apply various notation systems for data representation, utilize data compression techniques for efficient data storage and transmission, extract valuable information from diverse data sources, manipulate data within programs for different applications, and exhibit a solid understanding of database fundamentals, including database design, querying, and data management.

	Level 9	Level 10	Level 11	Level 12
CS.4.1 Notation Systems	I can recognize and recall basic notation systems used in computer science, such as binary and hexadecimal, but may struggle to apply them effectively.	I can apply basic notation systems like binary, hexadecimal, and ASCII to perform simple conversions and basic calculations, demonstrating an understanding of their use in digital representation.	I can apply advanced notation systems, such as Unicode, and perform complex conversions, bitwise operations, and encoding/decoding tasks effectively, showcasing a deep understanding of their practical use in computer science.	I can innovate by creating or manipulating custom notation systems, leveraging advanced encoding schemes, encryption methods, or data compression techniques, and showcasing expertise in developing novel solutions based on notation systems.
CS.4.2 Data Compression		I can recall and recognize the basic concept of data compression and identify common compression methods but may struggle to apply them effectively.	I can apply basic data compression techniques like Run-Length Encoding (RLE) or Huffman coding to compress and decompress data, demonstrating an understanding of their use in reducing data size.	I can apply advanced compression techniques such as Lempel-Ziv-Welch (LZW) or Burrows-Wheeler Transform (BWT) to compress and decompress data effectively, and can analyze the trade-offs between compression ratios and computational complexity.
CS.4.3 Extracting Information from Data	I can recall and recognize the basic concepts of data extraction and identify common data extraction methods, but may struggle to apply them effectively.	I can apply basic data extraction techniques, such as text parsing and simple data filtering, to extract relevant information from structured and unstructured data sources, demonstrating an understanding of their use in data analysis.	I can apply advanced data extraction methods, like regular expressions, XPath, or data scraping, to extract complex and structured information from various data formats effectively and can design and implement strategies for transforming and processing the extracted data.	I can innovate by developing custom data extraction algorithms, applying machine learning or natural language processing techniques, and creating advanced data pipelines for extracting and analyzing complex and diverse data sources, showcasing expertise in extracting and processing information from data for complex real-world applications.
CS.4.4 Using Programs with Data	I can recall and recognize basic concepts of using programs with data, such as data input and output, but may struggle to apply them effectively.	I can apply basic data handling and program interaction concepts, including reading from and writing to files, simple data validation, and formatting output for effective data presentation.	can apply advanced data handling techniques, such as database connectivity, data transformation, and integration with external APIs, effectively managing and processing complex data within programs, and can make informed decisions on data storage and retrieval.	I can innovate by creating sophisticated data-driven applications, implementing data analytics, machine learning, or artificial intelligence solutions, and optimizing program data handling for high-performance and real-world applications, showcasing expertise in using programs with data for complex and creative problem-solving.

CS.4 Data

Students will demonstrate proficiency in computer science through a strong grasp of Notation Systems, Data Compression, Extracting Information from Data, Using Programs with Data, and Database Fundamentals. They will apply various notation systems for data representation, utilize data compression techniques for efficient data storage and transmission, extract valuable information from diverse data sources, manipulate data within programs for different applications, and exhibit a solid understanding of database fundamentals, including database design, querying, and data management.

	Level 9	Level 10	Level 11	Level 12
CS.4.5 Using Programs with Data Visualization	I can recall and recognize basic data visualization concepts, such as charts and graphs, but may struggle to explain how to create and interpret data visualizations using software tools.	I can describe common data visualization techniques, understand data types suitable for visualization, and use software tools to create simple data visualizations, demonstrating an understanding of basic charting and graphing.	I can analyze and design more complex data visualizations, selecting appropriate visualization types, customizing visualizations for effective communication, and using advanced features of data visualization software tools, showcasing an understanding of data representation and visualization best practices.	I can innovate by creating advanced data visualizations that convey insights and trends effectively, developing interactive and dynamic visualizations, implementing custom visualization scripts, and addressing real-world data visualization challenges, showcasing expertise in data visualization, creative data presentation, and a deep understanding of data analysis and communication.
CS.4.6 Database Fundamentals		I can recall and recognize basic database concepts, terminology, and components, such as tables, records, fields, and SQL statements, but may struggle to apply them effectively.	I can create and query simple databases, design basic table structures, and retrieve data using SQL commands, demonstrating an understanding of fundamental database operations.	I can design and implement complex databases, including normalization techniques, define relationships, and optimize database performance, and can write advanced SQL queries for data retrieval and manipulation effectively.
Last edited:	11-02-2023			

CS.5 Systems

I can effectively harness the powers of data to make sense of the world around me, as well as to better understand how my actions and activities generate data which can be used by others.

	Level 9	Level 10	Level 11	Level 12
CS.5.1 Computer Hardware	I can recall and recognize basic computer hardware components and their functions, such as the CPU, RAM, and storage devices, but may struggle to explain their interactions.	I can describe the key components of a computer system and their roles, including the central processing unit (CPU), memory (RAM), storage devices, input/output peripherals, and basic hardware connectivity.	I can can analyze and compare different types of hardware components, considering factors like performance, compatibility, and cost, and can explain how various hardware components work together to execute computer programs effectively.	I can innovate by designing and configuring complex computer hardware systems, including custom-built PCs, server setups, or network infrastructure, and can optimize hardware configurations for specific computing tasks, showcasing expertise in computer hardware engineering and advanced problem-solving.
CS.5.2 Networking Concepts	I can recall and recognize basic networking concepts, such as IP addresses, protocols, and the purpose of a network, but may struggle to explain how these concepts are applied.	I can describe key networking components, including routers, switches, and modems, and can explain fundamental concepts like IP addressing, data transmission, and common network protocols (e.g., HTTP, TCP/IP).	I can analyze and design simple network configurations, including LANs and WANs, and can troubleshoot basic network issues, demonstrating an understanding of network architecture, security, and the role of routers and switches.	I can innovate by designing and configuring complex network environments, implementing advanced security measures, optimizing network performance, and addressing real-world networking challenges, showcasing expertise in network design, administration, and advanced problem-solving.
CS.5.3 Software	I can recall and recognize basic software concepts, such as the difference between system software and application software, but may struggle to explain their practical use.	I can describe common types of software, including operating systems, word processors, and web browsers, and can explain their typical functionalities and uses in a computer system.	I can evaluate and compare software options, make informed choices for specific computing needs, and troubleshoot software-related issues, demonstrating an understanding of software installation, configuration, and maintenance.	I can innovate by developing and customizing software solutions, writing scripts or applications, or optimizing software performance, showcasing expertise in software development, advanced problem-solving, and creativity in software use and modification.
CS.5.4 Computer Interfacing	I can recall and recognize basic computer interfacing concepts, such as input and output devices, but may struggle to explain how these devices interact with a computer.	I can describe common input and output devices, such as keyboards, mice, monitors, and printers, and can explain their roles in interacting with a computer system.	I can analyze and design basic computer interfaces, considering factors like user experience, compatibility, and accessibility, and can troubleshoot hardware and software issues related to computer interfacing.	I can innovate by developing custom computer interfaces, integrating specialized hardware or software solutions, or optimizing user experiences for specific computing tasks, showcasing expertise in computer interfacing, advanced problem-solving, and creativity in designing interfaces.

CS.5 Systems

I can effectively harness the powers of data to make sense of the world around me, as well as to better understand how my actions and activities generate data which can be used by others.

	Level 9	Level 10	Level 11	Level 12
CS.5.5 Internet of Things	I can recall and recognize basic IoT concepts and terminology, such as connected devices and data communication, but may struggle to explain how IoT systems are designed and function.	I can describe the fundamental components of IoT systems, including sensors, actuators, communication protocols, and data processing, and can explain their roles in collecting and exchanging data in IoT environments.	I can analyze and design simple IoT applications, considering sensor types, data processing, communication options, and security aspects, and can troubleshoot basic IoT-related issues, demonstrating an understanding of IoT system architecture and implementation.	I can innovate by developing and implementing advanced IoT solutions, creating IoT prototypes, integrating cloud-based services, optimizing data analytics, and addressing real-world IoT challenges, showcasing expertise in IoT system design, advanced problem-solving, and creativity in IoT application development.
CS.5.6 Troubleshooting	I can recall and recognize basic troubleshooting concepts, such as identifying problems and reporting issues, but may struggle to explain systematic troubleshooting approaches.	I can describe common troubleshooting techniques, including identifying hardware and software issues, following error messages, and seeking online resources for solutions, and can apply them to resolve simple computer problems.	I can analyze and solve more complex computer problems by employing systematic troubleshooting methodologies, identifying root causes, and proposing solutions, demonstrating an understanding of diagnostic tools, and refining their troubleshooting skills.	I can innovate by addressing challenging and multifaceted computer issues, developing custom tools or scripts for problem resolution, and optimizing system performance and security, showcasing expertise in advanced troubleshooting, creative problem-solving, and a deep understanding of computer systems.
CS.5.7 Cybersecurity	I can recall and recognize basic cybersecurity concepts, such as passwords, malware, and security risks, but may struggle to explain key cybersecurity principles and practices.	I can describe common cybersecurity threats and protective measures, including strong password creation, antivirus software, and firewall usage, and can apply basic cybersecurity practices to enhance digital security.	I can analyze and design cybersecurity strategies, identify advanced threats, evaluate network security configurations, and implement multifaceted security measures, demonstrating an understanding of cybersecurity risk management and incident response.	I can innovate by developing advanced security solutions, creating custom security tools, conducting penetration testing, and addressing complex real-world cybersecurity challenges, showcasing expertise in cybersecurity, advanced problem-solving, and creative security solutions.
Last edited:	11-03-2023			

CS.6 Impact of Society

The widespread use of computers and associated technologies has both positive and negative effects on the environment, society, and individuals. It is important to identify harmful effects on the environment and support agencies that reduce these effects. The effects on society include changes in the transport and agricultural sectors, as well as negative impacts such as the spread of misinformation and negative mental health problems. Ethical and security issues related to the use of computers must be understood and handled in an ethical and proper manner.

	Level 9	Level 10	Level 11	Level 12
CS.6.1 Technology and the Environment	I can demonstrate an awareness of the effects of various technologies on the environment.	I can identify harmful effects of the widespread use of computers and associated technologies on the environment, as well as agencies that reduce these effects.	I can describe environmental issues related to the widespread use of computers and associated technologies.	I can analyse environmental issues related to the widespread use of computers and associated technologies, and apply strategies to reduce environmental harm from computer use.
CS.6.2 Technology and Society	I can demonstrate an awareness of how various technologies affect society, as well as how society influences technological developments.	I can identify effects of the widespread use of computers and associated technologies on society.	I can describe societal issues related to the widespread use of computers and associated technologies.	I can analyse societal issues related to the widespread use of computers and associated technologies.
CS.6.3 Ethics and Security	I can identify ethical and security issues related to the use of computers.	I can demonstrate an understanding of ethical and security issues related to the use of computers.	I can describe ethical and security issues related to the use of computers.	I can describe ethical and security issues related to the use of computers and related technology.
CS.6.4 Artificial Intelligence		I can recall and recognize basic AI concepts, such as machine learning and neural networks, but may struggle to explain how AI technologies are designed and utilized.	I can describe common AI techniques, including machine learning algorithms and natural language processing, and can apply basic AI methods to solve simple problems, demonstrating an understanding of fundamental AI principles.	I can analyze and design AI solutions for more complex tasks, develop AI models, fine-tune parameters, and evaluate AI performance, demonstrating an understanding of AI model selection and optimization.

Last edited: 10-17-2023